Transformer Models for Efficient EO/IR Signature Generation

Nathan Highsmith
MTSI
Dr. Jorge O'Farrill
MTSI

ABSTRACT

In this paper we show that complex thermal physics of non-trivial geometries can be encoded into deep neural networks (DNNs). These networks can then be used to help characterize space objects given observations of their electro-optical signatures. We will detail: the physics-based models used to generate training data, the architecture of our DNN and the test and evaluation environment of the models.

1. INTRODUCTION

Physics-based models provide representative data from which concepts can be developed constructively or evaluated against realistic environments. These models typically have computation times that scale with fidelity. This computational overhead makes it hard to use such models as kernels for estimation engines, especially over large dimensional domains. Characterization of space object attributes can be used to predict future behavior and deviations from this expected behavior may involve estimating small changes in target attributes with limited data. Meaningful estimation of changing target attributes with paucity of data requires representative high-fidelity models. The computational burden of these models, however, is prohibitive for two reasons: proliferation assets in the near-earth space environment and lack of precise physical representation of targets of interest.

To bridge the gap between fidelity and run time we have developed narrow digital twins (NDTs) that encode the physical response, such as observed infrared signature of a specific target configuration, into deep neural networks. These machine learning (ML) models provide a high-fidelity response without the need for large volumes of training data. This reduction of data is also beneficial due to the large run-times of the physics-based models. To make our models robust, we use a reinforcement learning test environment called Harnessing Artificial Intelligence to Develop and Evaluate Systems (HADES) to perform adversarial attacks on our NDT. This testbed maps out the relationship between the input physical model and the outputs of the NDT, specifically where the NDT is failing or producing large errors. This paradigm of training limited but precise models and testing with adversarial reinforcement learning is extensible and can provide a path for the Space Community to perform precision estimation of changes which will facilitate Space Situational Awareness (SSA).

We create our NDT by training inferential deep neural networks to predict the temperatures on the faceted representation of two classes of satellites: cube satellite and cylindrical satellite with panels. The algorithm consists of a base model and an upsampler. The base model predicts the temperatures on a carefully selected subset of the facets, roughly 5% of the total facets. We then use convolutional neural networks to upsample these subset facet temperatures to the rest of the body. The complete thermal profile of the target is then used to generate an Electro-Optical Infra-Red (EO/IR) signature from the perspective of a user defined sensor. This model generates temperatures at rates of 1000-10000x of the physics-based models.

The algorithm is then used as the kernel of an optimization engine that ingests a trajectory estimate as well as a time series of EO/IR signatures of known wavebands and performs a least squares minimization on predicted and ingested radiometric signal. The optimizer searches target geometry space: size scaling, stretching, and segment removal for the case of the satellite with panels for the best fit. If the attitude of the target satellite is known this solution will determine the temperature profile and shape that best fits the incoming data set. This technique is shown to work for one sensor and improves when more sensors are added. If we have sufficient viewing diversity the attitude of the target can be estimates alongside the geometry and thermal profile. In this case the optimizer must now search a space that includes five motion parameters associated with the attitude of the body: angular momentum vector and precession rate and angle about this vector. We will provide examples of extracting geometric and attitude solutions and determine limits on precision as a function of sensor viewing diversity.

2. METHODOLOGY

2.1 OVERVIEW

The purpose of our analysis is twofold. First, we want to quantify the performance of our trained Artificial Intelligence (AI) models – the NDTs. Models are graded on accuracy as well as speed. We use our AI testbed HADES to characterize the performance of the NDTs. Fast accurate models have utility for SSA applications that involve edge processing and/or traversing large dimensional domains. We explore the utility of our trained models by using them to classify satellite classes and determine their sizes. We characterize the class and size of a small set of satellite targets as observed from satellite sensors. Our sensors collect light curves in the infrared: Mid Wave Infra-Red (MWIR, [3 5] micron) and Long Wave Infra-Red (LWIR, [7 9] micron). The sensors are in low earth orbit satellites in a Walker constellation as shown in the Fig. 1.

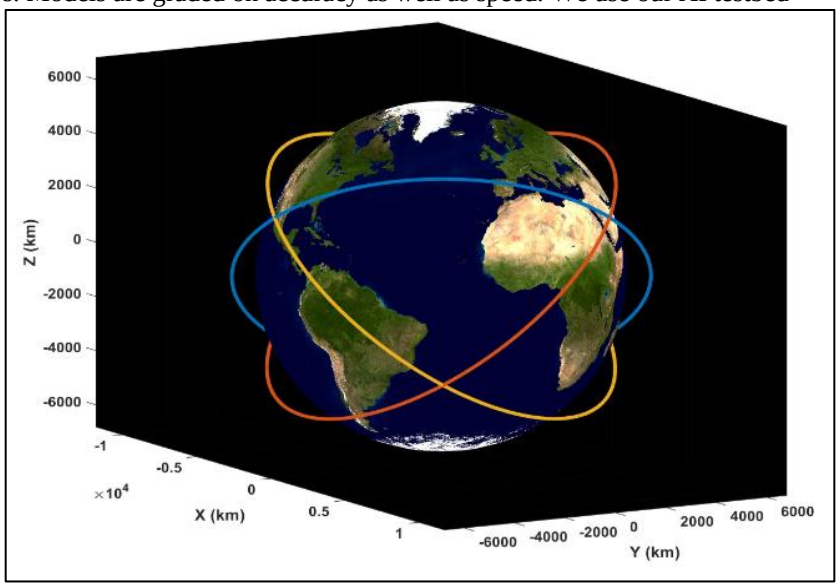

Fig 1: Sensor constellation

The constellation is three rings with four satellites per ring (3x4) with inclination 45^o and altitude of 800 km. Sensors detect targets when they are not obstructed by earth and the tangent height is greater than 80 km to avoid most of the earth limb. Sensor sensitivity is given be NEI (noise equivalent irradiance) of 5e-17 W/cm^2 in MWIR (medium wave infrared) and 1e-17 W/cm^2 in LWIR(long wave infrared). There are other sensor types such as Radar and Ladar that can be used to facilitate characterization, but we focus on passive EO/IR sensing. Every sensor that can detect our target provides a multiband signature to be used in characterizing the target. These signals are generated using out thermal model HiRTSS (High Rate Thermal and Signature Solver) and our AI models are the kernel for target characterization via simple EO/IR signal least squares fitting. First, we show that satellite target class can be determined using the AI model. We will then show that the AI model can be used to estimate the size of the satellite. This work is the first in a longer project that will create AI models able of simultaneously characterizing target shape as well as motion solution. Ultimately, we believe these models can be used to detect and characterize changes in these target attributes in support of SSA.

2.1.1 TARGET MODELS

We have three target classes: cube sat with panels, cylinder sat with panels and cube sat without panels. Our models are simple but meant to capture the relevant phenomenology in EO/IR signatures of space objects. The figures below show the mesh grids for our targets. Materials for the objects are Teflon, carbon phenolic, aluminum and silicon solar sail material.
Our satellite models are comprised of two components: panels and instrumented payload. In controlled flight, the panels attempt to maintain pointing to the sun and the body maintains pointing to the earth. In uncontrolled flight, the panels do not point to the sun.

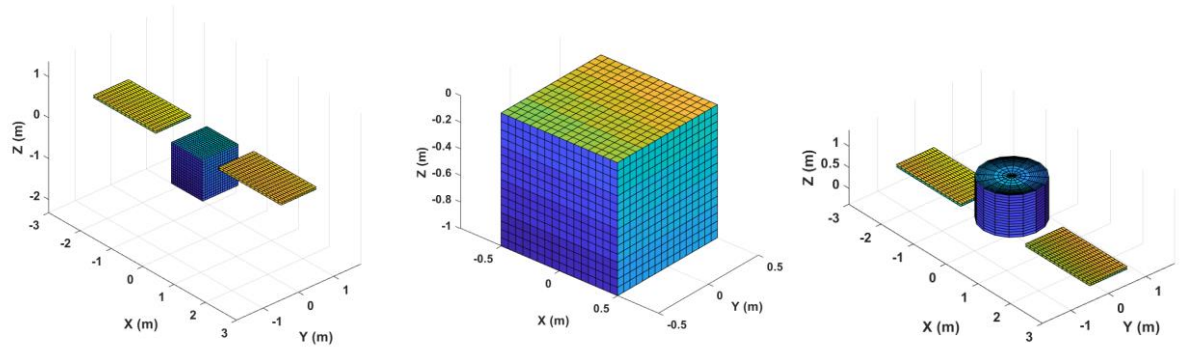

Fig. 2: Satellite Target Classes and Geometries

2.1.2 Target Classification

We will use our AI model to determine the class of the satellite given multisensory signature. First, we fly a target of a chosen class but randomized size on a trajectory and determine which satellite sensor has line of sight (LOS) to this target. We then generate EO/IR signals for all observing sensors in MWIR and LWIR. For each sensor collection we run our target classifier. This classifier ingests these multiband signals and makes predicted signals using the NDTs for a single target class. We use the Levenberg-Marquardt algorithm to solve for the best fit target dynamics as determined least squares residuals between predicted and ingested signals across all sensors. This process is repeated over the three classes and the target class with the smallest chi squared value is chosen as the class correct class. We parameterize this analysis across number of sensors observing and sensor viewing time and record results.

2.1.3 Target Size Estimation

We will use our AI model to determine the optimal size and attitude of a satellite given multi-senor signal data. The characterization will first involve identifying the class of the target to ensure no gross changes in target geometry have occurred since the last observation. Given a satellite class we will determine its orientation: motion solution assuming two different modes of motion: controlled and uncontrolled. We will quantify the accuracy of our motion solution. The final analysis will, and we will assume centralized processing so that multiple sensor data can be processed by our characterization algorithm. To address potential of edge processing, we will highlight single sensor performance when possible.

2.2 MODELS

In this section we discuss the physics-based model used to generate training data and the AI model we use to generate our NDT. Given that the NDT is only as good as the data it is trained on, we provide a detailed discussion of HiRTSS. We also discuss the architecture of our AI model and the methodology for training our AI model.

2.2.1 PHYSICS-BASED THERMAL MODEL

2.2.1.1 Introduction

Modern Technology Solutions, Inc. (MTSI) developed a High Rate Thermal and Signatures Solver (HiRTSS) in support of a Small Business Innovative Research program effort. This software was developed to meet the needs of in-line signature generation on complex and diverse objects. The baseline features are:

1) In-line rapid Thermal computations supporting emissive signature calculations for both thermally massive and thermally light objects
    a. Supports all dynamics types and shapes of objects
2) In-line Reflection computations for solar and earth reflections
3) In-line obscuration/shadowing computations

4) Ingesting of Missile Defense Agency Threat models into HiRTSS target model libraries to maintain rigor in object definitions
5) Ingesting of mesh files from external computer-aided design (CAD) sources: Satellites, Missiles, Aircraft, Ground and Naval Targets
6) High accuracy and high throughput, typically running at 50x real time in the current MATLAB prototype.
7) Full Monte Carlo capabilities over all major optical signature drivers including the ability to morph targets and create new objects without having to create new Target Model Libraries (TMLs)
8) The code can be run stand-alone through a graphical user interface (GUI) or command-line interface or it can be configured to run in-line within a larger simulation to provide on demand stimulation for sensor models or algorithms.
9) Morphing capabilities include:
    a. Target surface deformation (wrinkles)
    b. Shape and size changes (e.g. sphere to ellipse, radius increase or decrease, length increase or decrease etc.)
    c. Orientation changes of portions of the body including rotations of parts of the target, retractions of solar panels, modeling of transient events etc.

HiRTSS achieves this throughput, not by sacrificing accuracy but by a combination of dimensional reduction of key functions and precomputations based on full spectral characteristics of the target materials and geometry. HiRTSS has been used to train and test combat ID algorithms and to support AI-driven algorithm testing and reverse engineering of remotely sensed objects in space.

The HiRTSS signature solver uses a set of TMLs which contain geometric, material, environmental and reflection information. There are two paths to create these TMLs. Since HiRTSS was originally developed to complement the standard EO/IR threat signature generation tool used by the Missile Defense Agency (as well as various Intel centers), one path leverages the databases and input files that are part of the Optical Signatures Code (OSC) toolkit. This gives us access to the substantial databases of material properties and environmental models as well as direct traceability to the legacy and current approved threat models. The second path to creating TMLs will be discussed in a following section.

The pre-processor has the job of extracting the target model information from the Optical Signatures Code (OSC) input and geometry files and translating that into HiRTSS TMLs. There are currently 4 TMLs which contain not only the geometry and material information in the OSC input file but also the absorbed fluxes for each material, the target obscuration (shadowing) information and the full reflection computation for the object. Once these TMLs are created, then the solver can execute on any scenario without having to recompute any of these quantities. The process of creating TMLs is illustrated in Fig. 3.

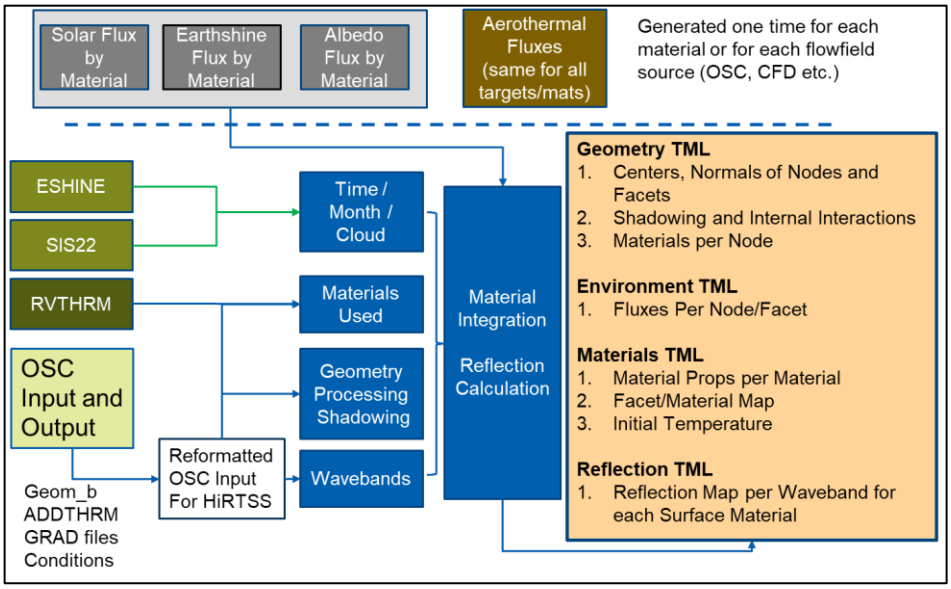

Fig. 3: TML formation for HiRTSS using the Optical Signatures Code databases and input files

To create the target model libraries there are several steps:

1) The target model geometry is read in from the OSC geometry file, geom_b

2) The materials designated in the input file are assigned to the appropriate parts of the target
3) The material thermos physical and optical properties are read in from the OSC databases
4) The spectra of the earth, albedo and sun and then convolved with the reflection properties of each material as a function of wavelength and angle to create an 'absorbed flux' for each material as a function of angle
5) The angle dependent solar absorptivity and temperature dependent emissivity of each material is derived from the optical data and stored
6) The target obscuration is computed as a function of viewing angle to the body, each facet gets a viewing factor from 0 to 1 indicating its relative visibility to the source or sensor
7) Finally, the reflected signature (including obscuration) is computed for 9 million combinations of sensor and illuminator viewing to allow for on-the-fly reflection calculation in the solver

A key benefit of HiRTSS is contained here: we leverage the highest fidelity data and methods so that no accuracy is sacrificed, and the precomputations of obscuration and reflection provide dramatic increases in throughput over traditional methods.

2.2.1.2 Overview of HiRTSS Solver

Once a TML is created, the HiRTSS solver can be run using that TML for any scenario needed. A scenario would include the time of day, day of year, trajectory, sensor location and initial temperature. There are essentially two sets of computations done by HiRTSS during a run.

First, the trajectory is used to generate an incident flux history for every thermal node on the target. This flux is summed at each time point for all the sources (Earth, Albedo, Solar, internal interactions and Aerothermal) and a 1-D thermal calculation is performed for each node (a 3-D conduction model for nosetips and leading edges has been prototyped but not yet implemented). From that temperature the emissive signature is directly computed from Plank's law using the measured reflectance properties of the materials.

The second set of calculations is done to interpolate the total reflected signature for the target for each time point. Since the target reflection depends on both the relative position of the sensor and observer as well as the orientation of the body, it is necessary to reduce the dimensionality of the computation to create a matrix which is compact enough for rapid interpolation and finely gridded enough for accuracy. Those time-consuming calculations are in the pre-processor so that only a one-dimensional interpolation is done within the solver proper.

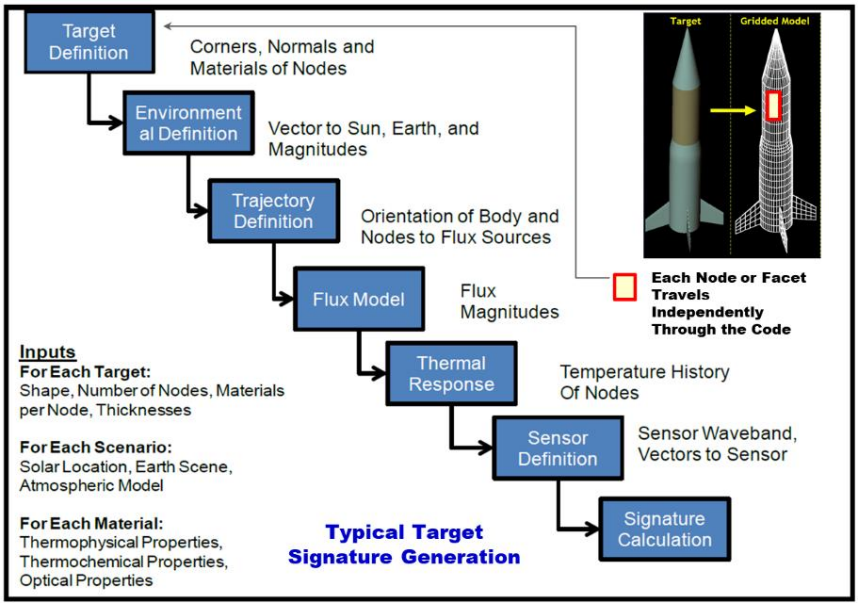

Fig. 4 Typical serial path for target signature generation as used in OSC and other legacy codes

This process contrasts with the standard serial process where each item is computed from geometry meshing to intensity summation. Fig. 4 and Fig. 5 show the differences in the flow through a standard code like the OSC (Fig. 4) and through the HiRTSS solver (Fig. 5).

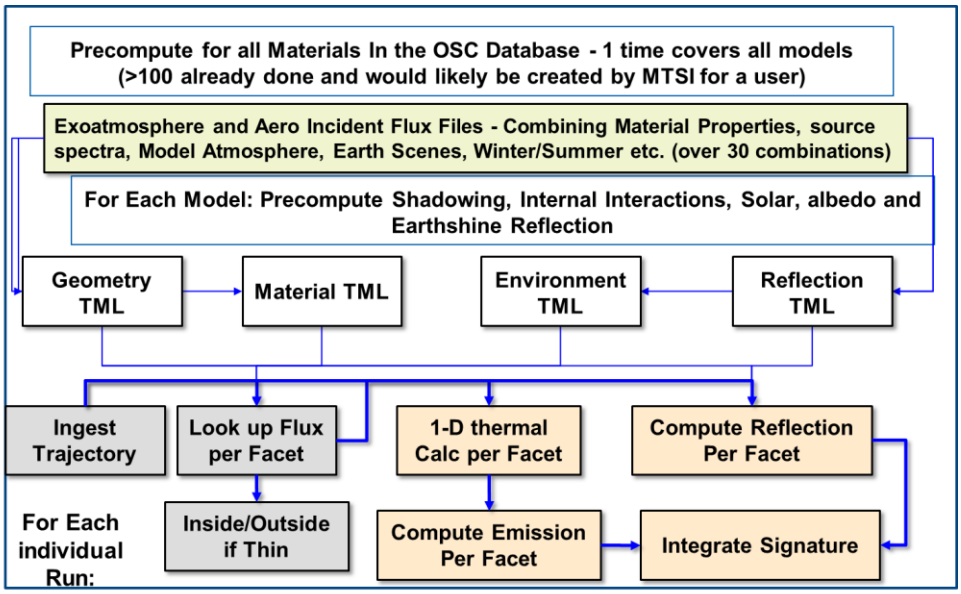

Fig. 5 HiRTSS process showing the minimal needed in-line calculations

2.2.1.3 Validation

Since HiRTSS is designed to provide a fast-running complement to the high-fidelity tools used by the Missile Defense Agency (MDA), the initial implementation leveraged the same input files to maintain a 1-to-1 relationship with the approved threat models. As a result, validating HiRTSS has taken the form of validating the output against the high-fidelity source (typically the OSC) from which the inputs were taken. The 'validation' is essentially inherited from historical and current validation of the OSC against measured data from flight and ground testing. There are currently well over 100 validation comparisons available between HiRTSS and OSC in wavebands ranging from visible to Infra-Red (IR). Many of these are classified and have been presented in various briefings across MDA. Currently, we are planning to collate all the validation examples in a separate report. Fig. 6-8 are unclassified examples showing both thermally massive and thermally light targets being compared between the OSC and HiRTSS. The differences seen are typical of those seen across the full test set (classified and unclassified).

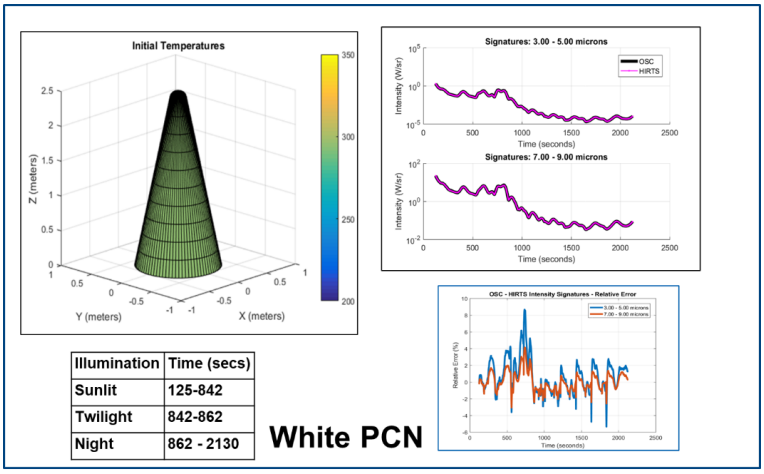

Fig. 6 Comparison of OSC and HiRTSS for a Thermally Light Cone

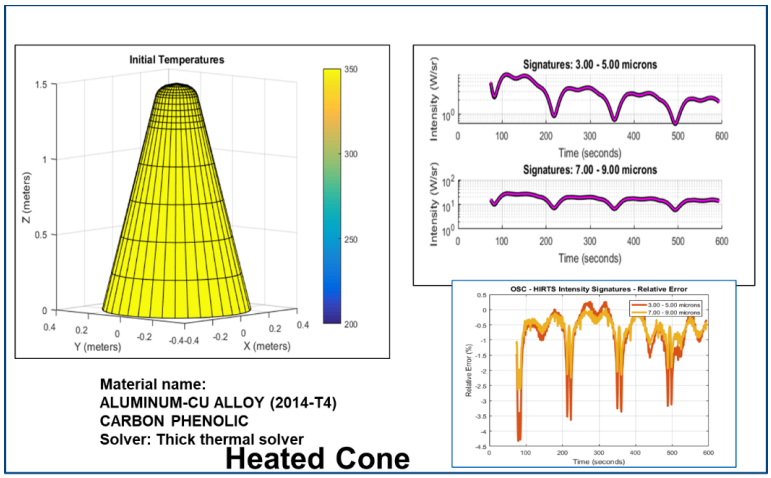

Fig. 7 Comparison of HiRTSS and OSC for a thermally massive cone

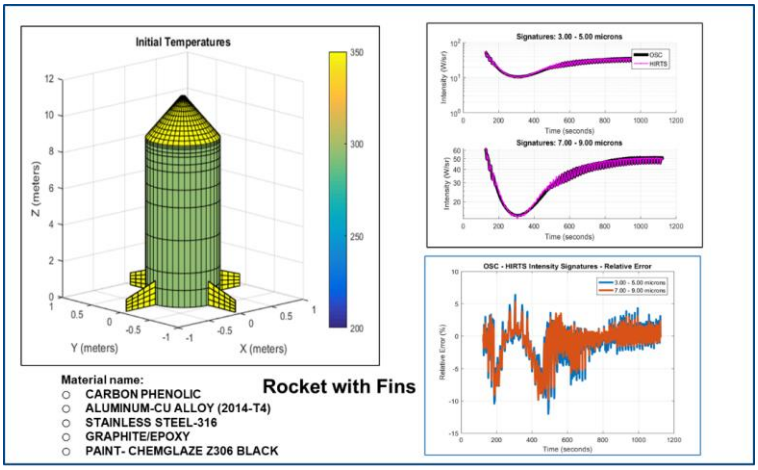

Fig. 8 Comparison of OSC and HiRTSS for a full Rocket Body with fins

2.2.1.4 Aerothermal Heating

One of the principal activities in our recent development efforts was to enhance the aerothermal modeling capabilities of the HiRTSS solver and to apply that heating to the challenge of predicting the temperatures and signatures of hypersonic glide vehicles (HGV) in the terminal phase of flight. In keeping with the structure of HiRTSS it was necessary to create a dimensionally reduced data set of incident aerothermal fluxes. There were several challenges involved in taking the information from OSC and creating such a flux table. Note that the use of the OSC to generate the flow field data was done out of expediency and that other sources (such as Computational Fluid Dynamics (CFD)) could also be used in future upgrades.

Fig. 9 is a plot of all the OSC aerothermal fluxes distributed on our test target body in our reduced coordinate system. The data spans 5 orders of magnitude with the highest fluxes occurring near the stagnation point and when the node points directly into the wind (0 degrees angle to the wind).

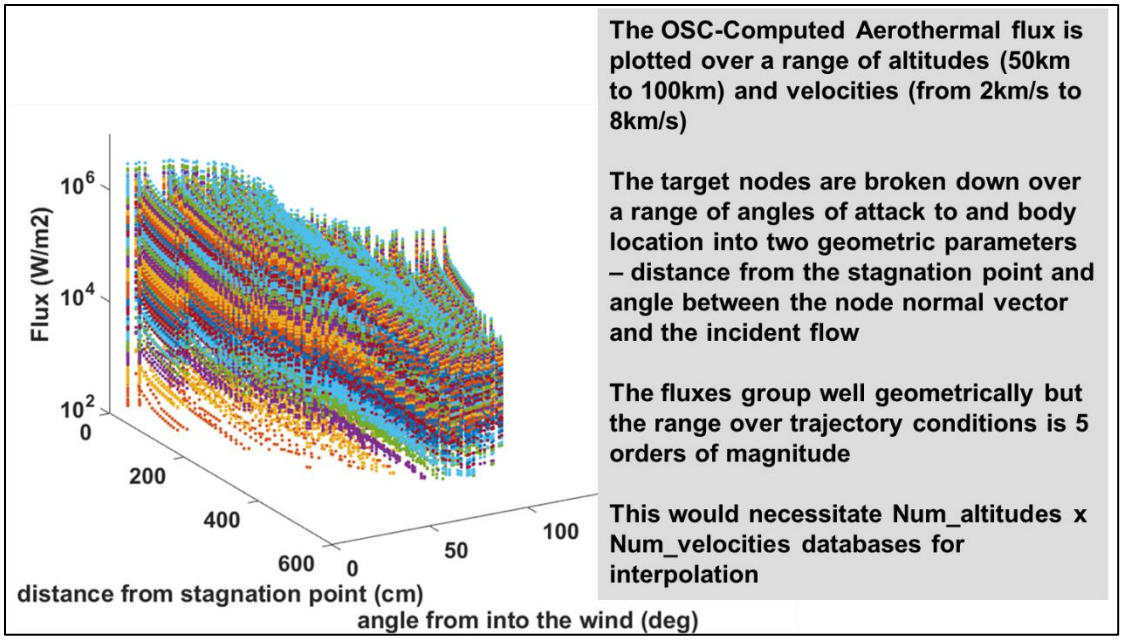

Fig. 9 Unscaled OSC Aerothermal Fluxes Used to create the HiRTSS aerothermal databases

To use this as a flux table the data was scaled to reduce the dependency on specific trajectories. The scaling variables were based on the dynamic pressure, and effective Mach number at each station. Since the dynamic pressure accounts for both density (effectively altitude) as well as velocity this scale greatly collapsed the flux magnitudes and created a more well defined 'surface' to interpolate the fluxes.
Fig. 10 shows the collapsed data as well as the surface grid which we fit through that data to create the flux table. Since the data still has some span in the z dimension there is a range of possible fits through the data (upper edge, center lower, edge of the distribution etc.). This provides us with 2 fitting coefficients we can use to optimize the temperatures when we compare to the OSC if needed.

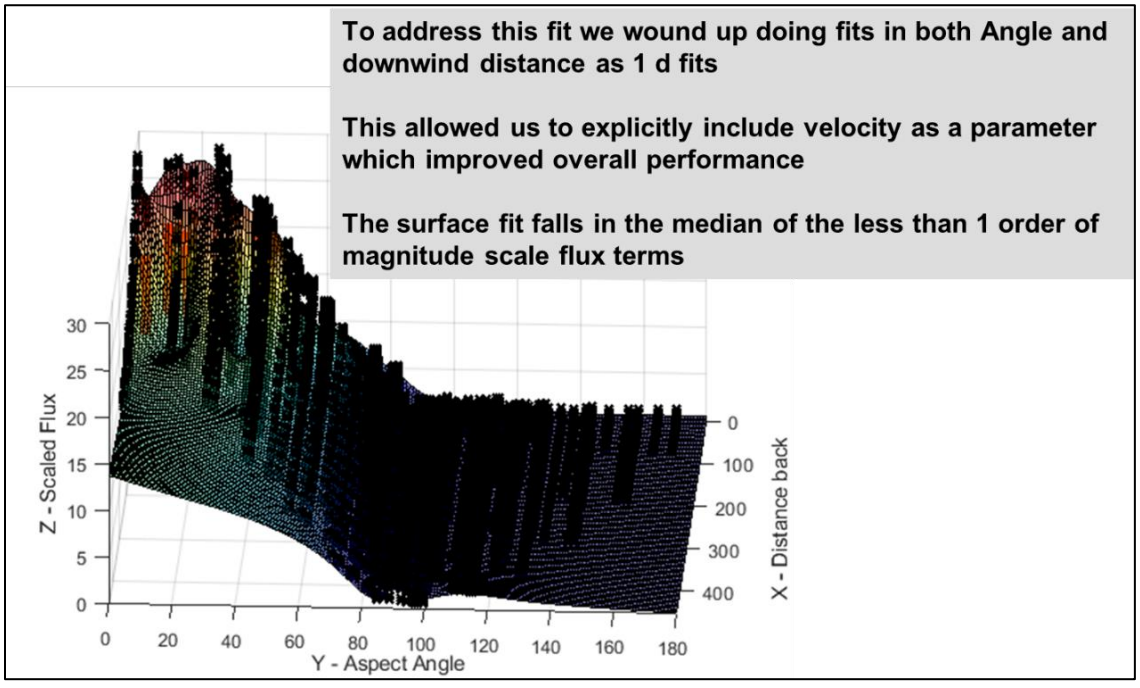

Fig. 10 Scaled Fluxes from OSC and the fitted surface used to create the flux terms within HiRTSS (the fluxes are then 'unscaled' to produce the proper incident values)

2.2.1.5 Ingestion of mesh models – the second path to TMLs in HiRTSS

To address winged bodies as well as allowing HiRTSS to operate on models where no OSC input file is available, we developed a process to translate a range of CAD-type mesh files and create HiRTSS TMLs. From the standpoint of the solver and pre-processor this requires a flag to be set in the system adjustable parameters (SAP) file and for the user to create two text files which define the material mapping to the body. Currently, we have implemented a few different conversions to allow for the ingestion of a wide range of target meshes. Again, we make use of the Blender toolset to help with this process and to assign the target material sections to the body. Fig. 11 is an example of a simple satellite model that was downloaded from Turbosquid. The colors on the model are material lay-up areas defined by the user in Blender and represent the material lay-ups that are defined in the target material files.

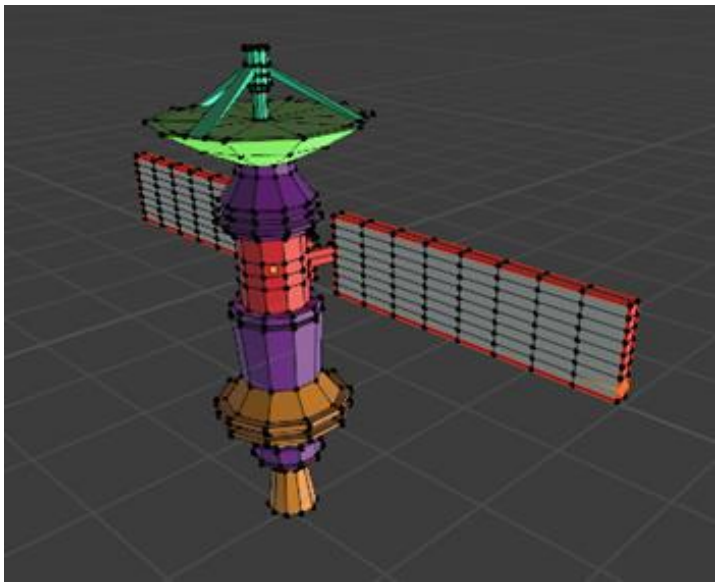

Fig. 11 HiRTSS satellite model

Fig. 12 is a mosaic of some other models that have been ingested and turned into HiRTSS models. This process allows us to rapidly create HiRTSS models out of complex geometries without having to worry about how to represent those shapes in the optical signatures code input files. One key script we have developed for this allows us to incorporate Radar Cross Section (RCS) facet models, like those used by the XPATCH RCS prediction code, and create optical models that are 1 to 1 with the geometry of the RCS models.

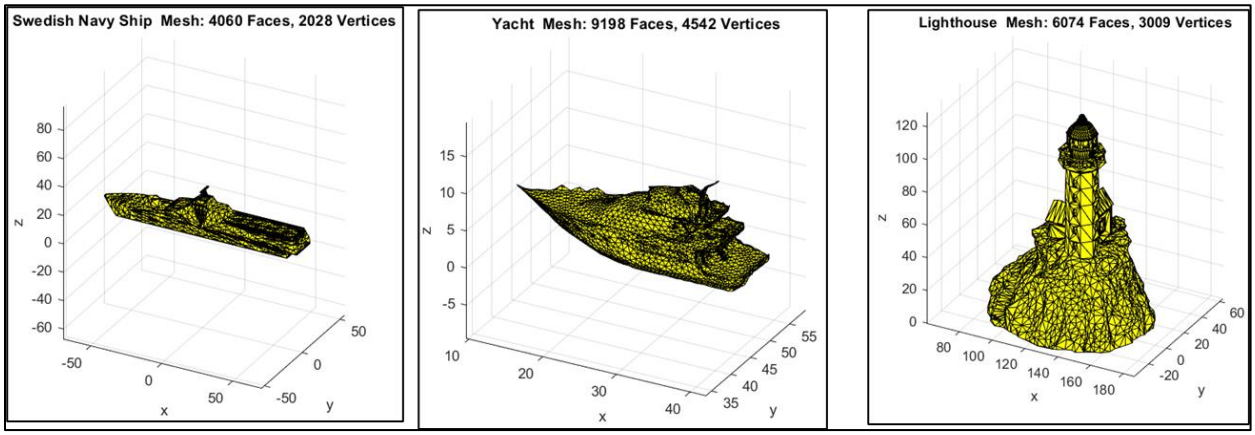

Fig. 12 Several Mesh Models that have been translated into HiRTSS mesh files

2.2.1.6 Hypersonic Glide Vehicles

This section presents an example of many of the developments over the course of this effort using a notional hypersonic glide vehicle target. To construct this model, we first downloaded a simple unclassified HGV model from the internet. The model was pulled into Blender where some of the facets were cleaned up and redundancies removed. While in Blender the material layup areas were identified and mapped to be body. A set of reasonable materials and thicknesses were created for each of the material layers and the model was then run through HiRTSS. The process models and results are shown in Fig. 13 and Fig. 14 below. At this time, we are developing a classified model to that we will compare to both the OSC predictions and collected data. The time from download to completion of the run was less than 1 workday. This stands in contrast to the current multi-stage processes that takes week to build up for an OSC-based model for this class of target.

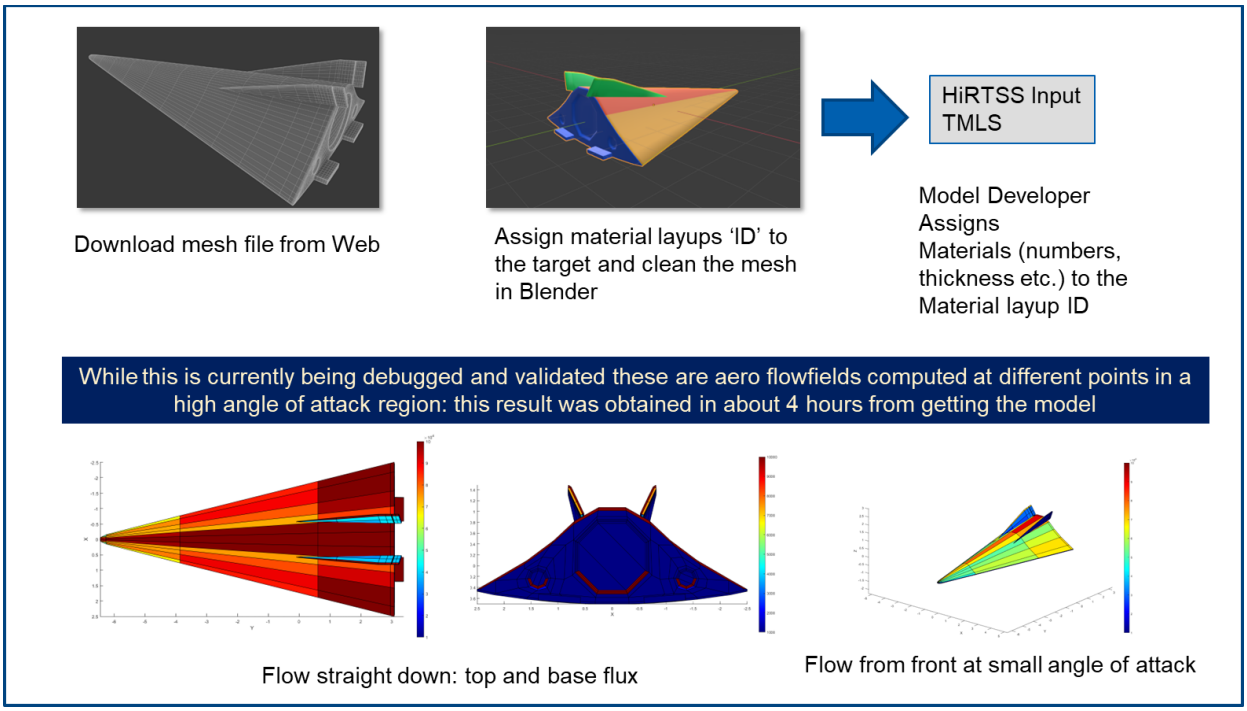

Fig. 13 Creating a Hypersonic Glide Vehicle Model in HiRTSS

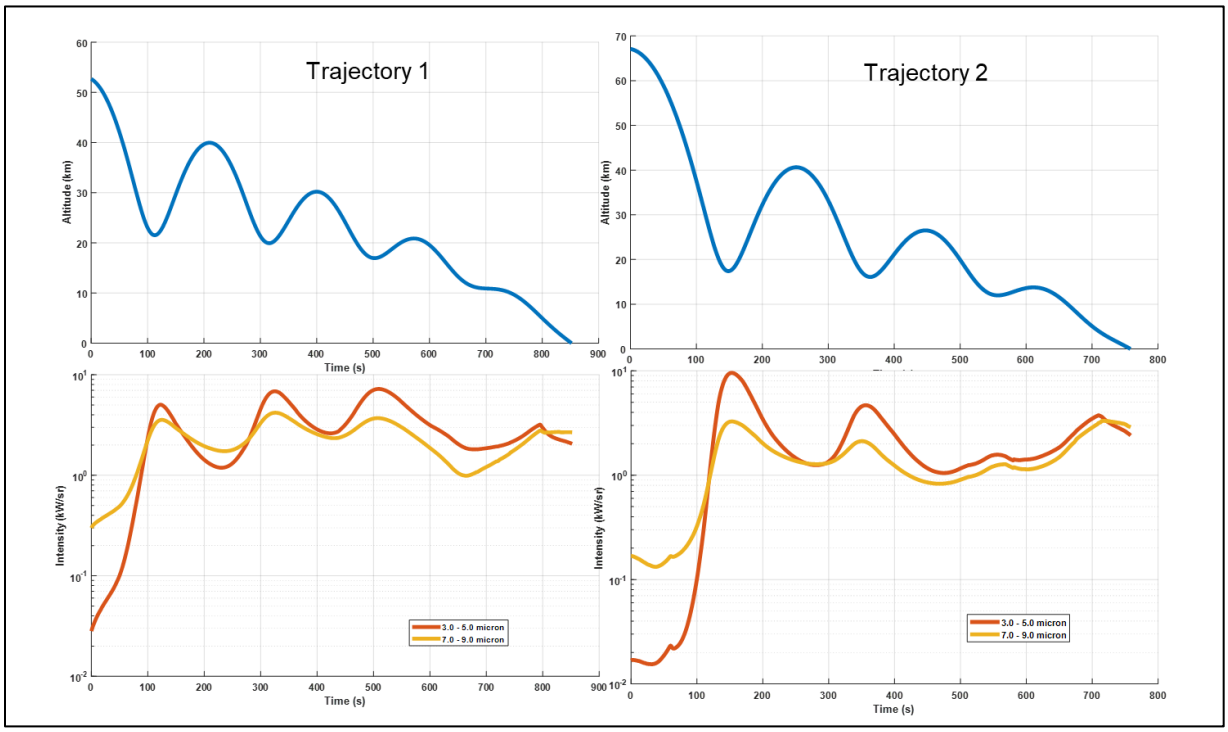

Fig. 14 Intensity outputs from the HiRTSS HGV model for two different flight profiles

2.2.2 AI MODEL

Our AI model architecture is composed of two neural networks. We have a base network that infers temperatures from a given satellite trajectory and an additional upscaling network that gives more resolution to the temperatures in relation to the target model. Both neural networks are transformers, a state-of-the-art network architecture that

excels in many machine learning tasks, including natural language processing and computer vision. We train an AI model for each target model shown in section 2.1.1.

2.2.2.1 BASE MODEL ARCHITECTURE

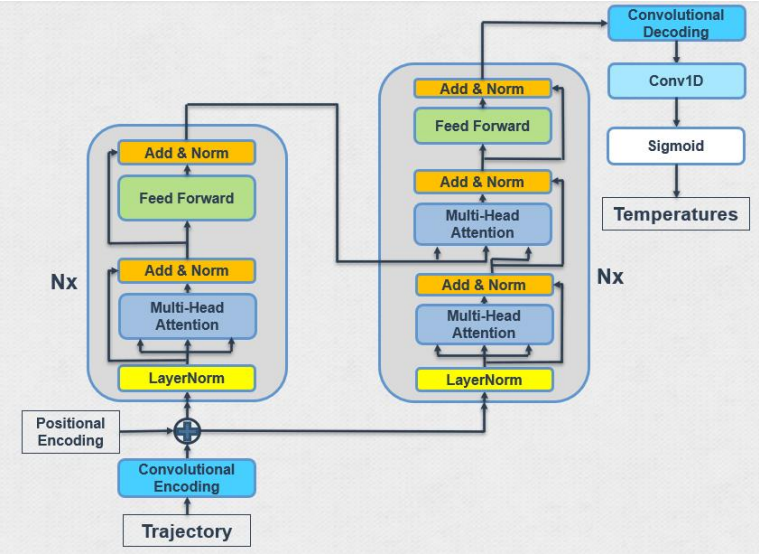

Fig. 15 Base Model Architecture

The base model architecture used is based on the original transformer proposed by Vaswani et al [1]. Slight modifications to the traditional transformer architecture were made to better fit this application such as the use of convolutional embedding for the inputs/outputs. Layer Normalization is also used at the start of the transformer encoder and decoder blocks to improve model stability and convergence. The outputs of the convolutional decoder are put into a 1-D Convolutional layer with a sigmoid activation function. Since the input data has been scaled to values between 0 and 1, this makes sure that the model can only output values within the training data's bounds. There are 4 encoder blocks and 4 decoder blocks, each with Multi Head Attention layers consisting of 1 head.

2.2.2.2 CONVOLUTIONAL EMBEDDING

Although transformers are extremely proficient at tasks that deal with sequential data, they are often limited by the computer hardware. Specifically, large transformers use vast amounts of memory to make inferences. To help offset this limitation, we use convolutional embedding on the inputs to shorten the input sequences and reduce the memory load during model training and inference. The use of convolutional embedding on the inputs also allows for the model to learn more meaningful patterns of the input data. To perform this convolutional embedding, we "encode" and "decode" input sequences with a series of convolutional layers and pooling/upsampling layers. Input sequences are put through 1-D Convolutional layers followed by 1-D Average Pooling layers until a desired shorter input sequence containing key features of the data is achieved.

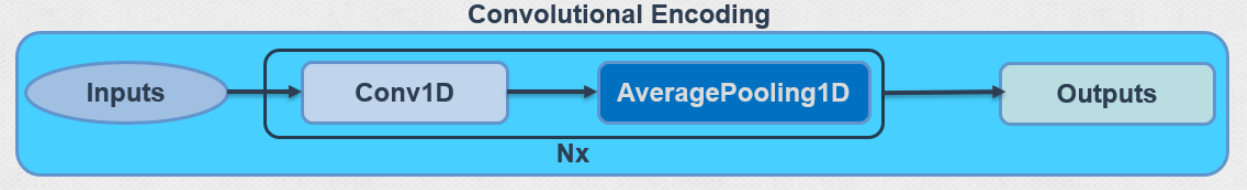

Fig. 16 Convolutional Embedding "Encoder" Architecture

Convolutional decoding works in a similar fashion: inputs of the decoder are put through 1-D transposed convolutional layers followed by 1-D upsampling layers until the original sequence length is achieved.

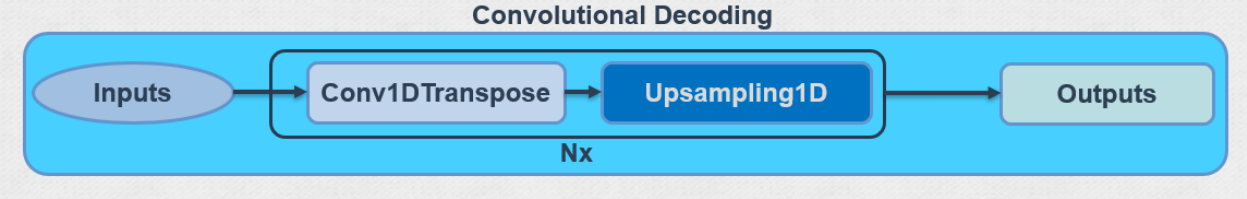

Fig. 17 Convolutional Embedding "Decoder" Architecture

2.2.2.3 TEMPERATURE UPSCALING

To further reduce model size and improve performance, only temperatures at selected facets are computed from trajectories using the adopted base model architecture. The remaining facet temperatures are interpolated using an upsampler model. This paradigm of base model outputs being given to an upsampler model is adopted from OpenAI's work on multi-modal generative models [3]. The upsampler model's architecture is derived from the Hybrid Attention Transformer (HAT) architecture proposed by Xiangyu Chen et al. [2]. To fit this application, the 2-D convolutional and Upsampling layers are replaced with their 1-D counterparts. Other aspects of the HAT such as Residual Hybrid Attention Group (RHAG) are foregone in our application. In our implementation, inputs are first put through a convolutional embedder to obtain shallow features from the data. Those shallow features are then fed into multiple transformer encoder blocks to extract deep features. The shallow features and deep features are then combined through an additive layer. The result of the additive layer is then fed into a convolutional decoder to retrieve the upsampled temperatures.

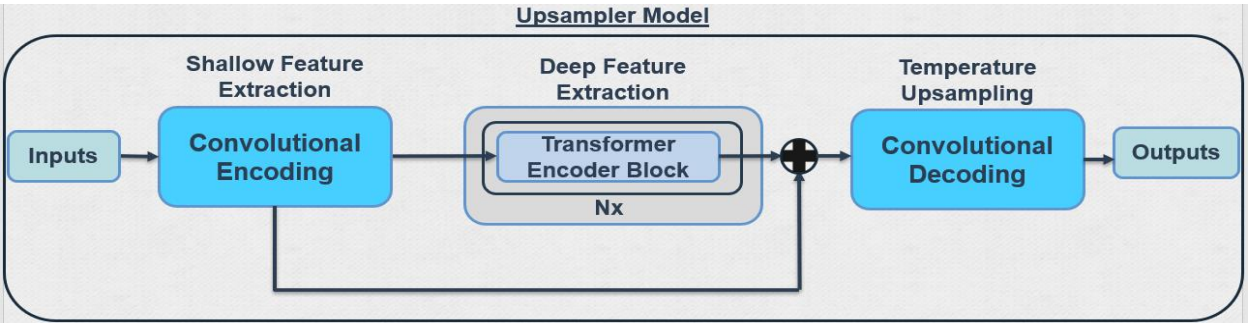

Fig. 18 Upsampler Model Architecture

Fig. 19 Example of Upsampler Usage

2.2.2.4 AI MODEL TRAINING

Base models and upsamplers for each satellite target were trained on the target trajectories and corresponding temperatures. The trajectories and temperatures were generated using the kinematic and target models discussed in the sections 2.1 and 2.2.1. Fig. 20 Shows a table of the trajectory state values and their corresponding units used to train the model. Of note, the satellite body's direction cosine matrix (DCM) and panels' DCM are included in the training data. Along with the sun vector, or the direction of the sun, these values should theoretically allow for the model to learn a target's thermal response to environmental heating from the sun. Also of note, morphing factors are included in the training data for just the cube satellite with panels. These morphing factors describe the scale of the model. This allows the AI model to learn the thermal response for a target model of varying sizes. For training, the mean absolute error between the predicted and truth temperatures was used as the objective function during training. Models were initially trained with an Adaptive Moment (ADAM) optimizer and, after sufficient convergence, finetuned with a Stochastic Gradient Descent (SGD) optimizer.

Value	Units	Size
Altitude	m	1
Latitude	rad	1
Longitude	rad	1
Orbital Inclination	deg	1
Sun Vector	unitless	3
Body DCM	unitless	9
Panel DCM	unitless	9
Time	sec	1
Morphing Factors	unitless	6

Fig. 20 Trajectory State Values Used for Training

2.4 HADES

2.4.1 INTRODUCTION

MTSI has developed a proven and robust enabling technology to address critical test and evaluation (T&E) engineering challenges which addresses a wide range of critical missions using a process and toolset we call HADES (Harnessing AI for the Development and Evaluation of Systems). *Our approach leverages the strengths of AI and machine learning by using them within the testing element where overtraining (that is learning the behavior surface of the test subject) is desired rather than an issue*. We have successfully adapted this technology to several key DoD challenges. This history of success provides evidence of robustness, confidence in the process, as well as a broad set of modeling capabilities which can be used on addressing key interest areas such as: Warfare Analysis, Test and Evaluation Engineering, AI/ML, providing fast analysis and testing of 'intelligent' systems.

2.4.2 HADES STRUCTURE

Fundamentally, HADES consists of an adversarial 'Agent' that drive the test, evaluation, or training/development process by leveraging modern ML and AI capabilities coupled with a flexible and fast generator to create new data specifically in response to the performance of the system under test (SUT). Regions of poor performance that may be opaque to humans are discovered naturally by the Agent's learning algorithms. In this way, the SUT itself drives the test/training data and regions of unexpected or poor algorithmic performance are mapped into the physical domain. Data from these regions can be used for retraining/hardening of the system or capturing as areas of concern. *This fundamental mapping of the high-dimensional and abstract space of the SUT into the physical domain or the real world is a key contribution of the HADES testbed*. Fig. 22 is an overview of the HADES approach.

The nature of the generator function is flexible, we have used several different tools to create the on-demand data. For our HGV thermal applications, the tool we leverage is called HiRTSS (High Rate Thermal and Signature Solver – see section 2.2.1 on HiRTSS for more details).

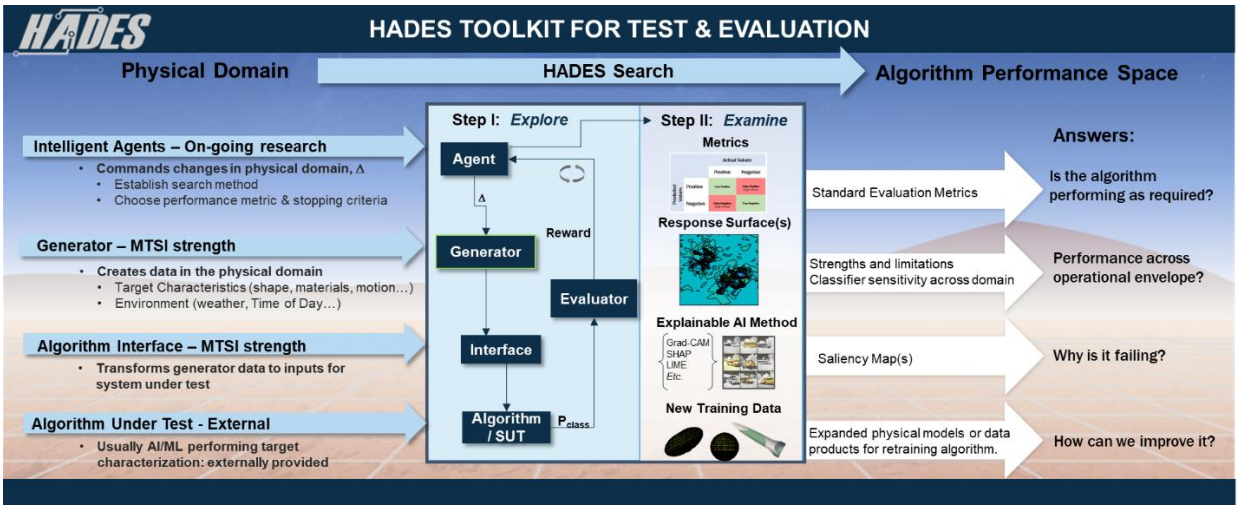

Fig. 21 Overview of the HADES testbed, functions, and benefits

As shown above, the overall HADES structure is modular and straightforward. An Agent drives a Generator to create data specifically to explore the SUT and toxic regions within the performance domain where the SUT behaves in an unexpected or unwanted manner. The ability to identify potential emergent behavior is a core value HADES provides.

HADES testbeds can be constructed for a wide range of applications if we can develop a Generator to create the high-throughput, high-quality data required to adaptively test the SUT. This points to another strength of our proposal, ***the generators themselves are powerful modeling and simulation tools which can be delivered as products.***
Fig. 22 shows prior successful applications for MDA, the Space Force (SF), the Missile and Space Intelligence Center (MSIC) and Hypersonic test and evaluation.

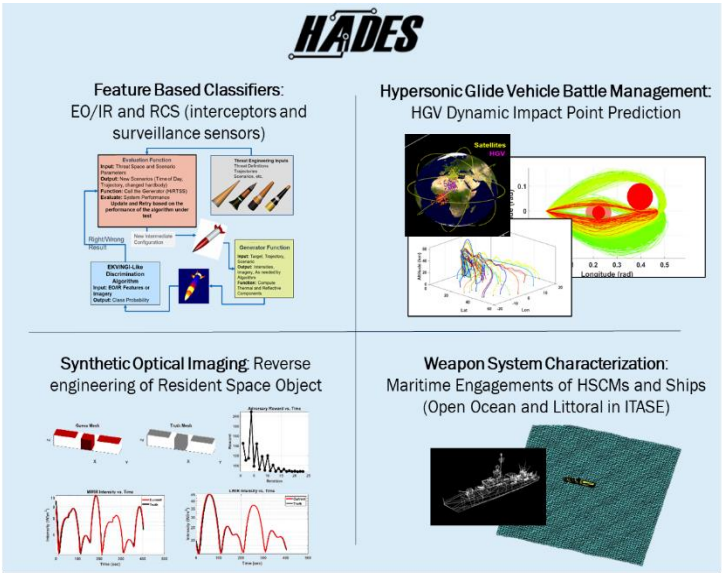

Fig. 22 Existing HADES applications for Defense

Fig. 23 is a listing of applications, and generators that can be leveraged or *provided as products*.

Domain	Generator/Possible Product	Description
Optical Sensing based systems (UAV, interceptors etc. anti-ship missiles etc.)	HiRTSS (High Rate Thermal and Signature Solver)	Full-fidelity, fast-running, EO/IR signature and scene generation for space or terrestrial applications
Hypersonics	PIA-HGV (Physics Informed AI for HGV)	An encoding of a full physics model for an HGV which provides high fidelity with extremely high throughput.

Fig. 23 Potential testbeds and simulation products

At the government's direction, the testbed can be redirected towards EO/IR sensors, enemy Radar typing, because the HADES testbed is highly flexible. The structure of the proposed HADES testbed for an electronic warfare (EW) application and its key benefits are provided in Fig. 24 below as an exemplar.

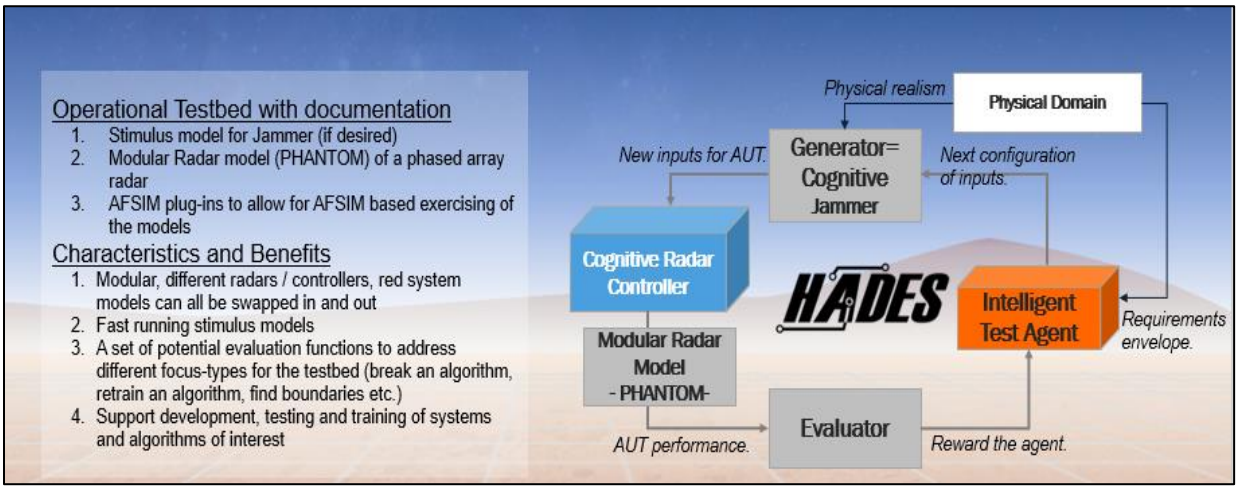

Fig. 24 Proposed Product and Benefits for EW

3. RESULTS

3.1 AI MODELS

After training and finetuning, the AI models were tested to properly gauge their performance on data within training data distribution. The AI models were evaluated on a set of 1000 trajectories and temperatures. HiRTSS was used to calculate temperatures and signatures for generated trajectories. The AI models were then run on the same trajectories, with HiRTSS using the AI models' outputted temperatures to generate signatures. The satellite trajectories were simulated with the methods detailed in section 2.1. For the trajectories, no rotational forces were applied to the main bodies of the satellite target models. The satellite target models with panels had a mixture of trajectory data in which the panels faced the sun and a random direction away from the sun. Mean Absolute Percentage Error (MAPE) was recorded as the test metric. MAPE is calculated by the following equation:

$$err_{MAPE} = \left|\frac{y - x}{y}\right| * 100$$

Fig 25 – 39 show the final test metrics for each AI model as well as example trajectories with model predictions. Fig. 40 shows the results from using our test and evaluation toolkit, HADES, to test the cylinder satellite model for robustness.

CUBE SATELLITE WITHOUT PANELS

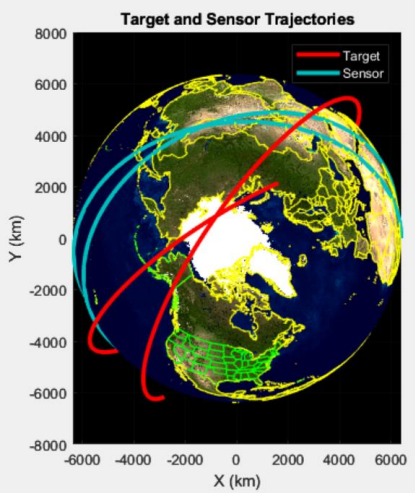

Fig. 25 Target and Sensor

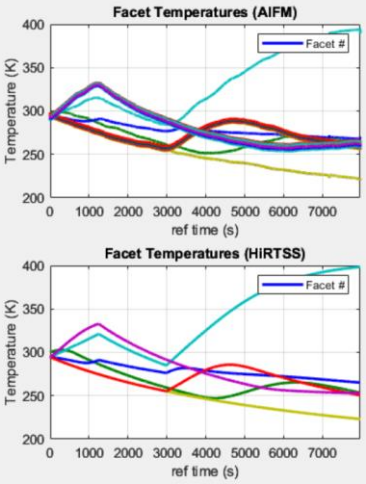

Fig. 28 Facet Temperatures

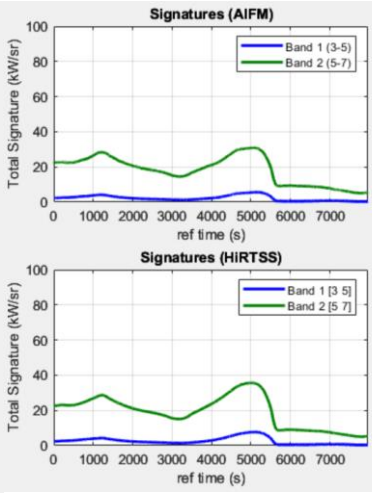

Fig. 31 Signatures

CYLINDER SATELLITE WITH PANELS

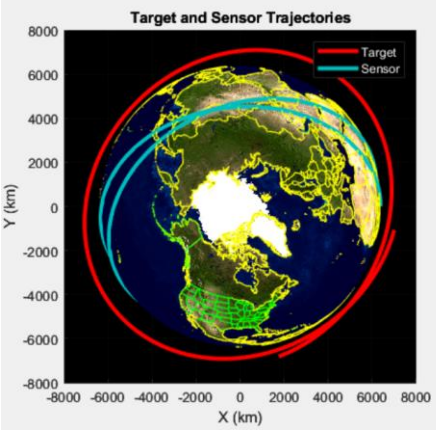

Fig. 26 Target and Sensor

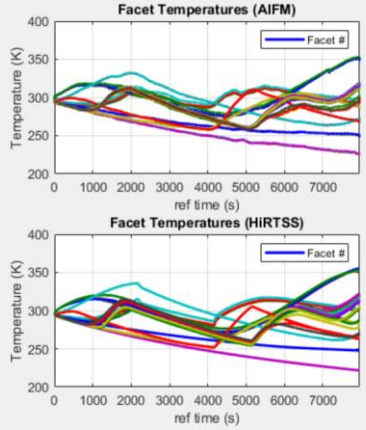

Fig. 29 Facet Temperatures

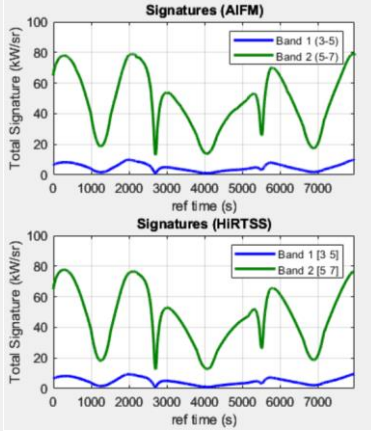

Fig. 32 Signatures

CUBE SATELLITE WITH PANELS

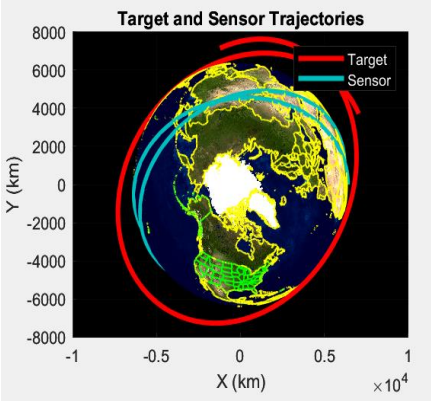

Fig. 27 Target and Sensor

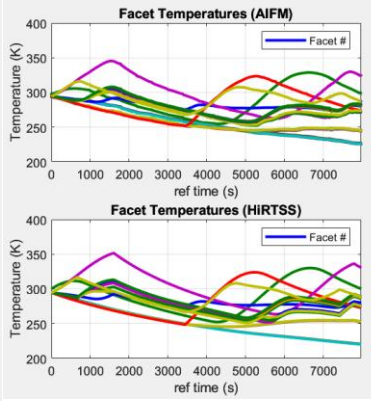

Fig. 30 Facet Temperatures

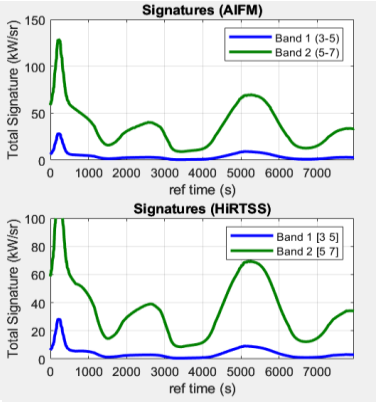

Fig. 33 Signatures

Fig. 25 – 27 shows an example target and sensor trajectory for the target models, both being satellites. Fig. 28 – 33, show the corresponding temperatures and signatures, respectively, for the target and sensor trajectories. As shown in Fig. 28 – 33, for a given target trajectory, the AI model's outputs for all target models align very closely with the HiRTSS outputs. This suggests that the model can properly learn the target model's thermal response given trajectory and constellation information.

CUBE SATELLITE WITHOUT PANELS

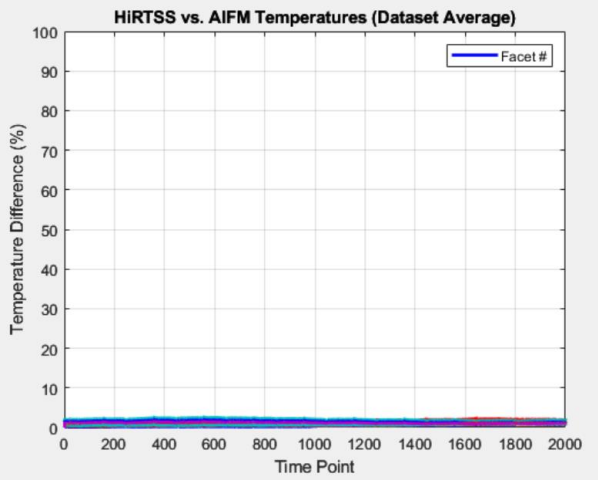

Fig. 34 AIFM Temperature Errors

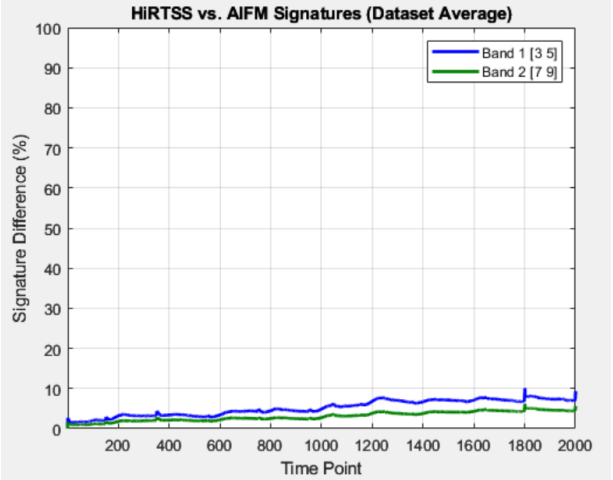

Fig. 37 AIFM Signature Errors

CYLINDER SATELLITE WITH PANELS

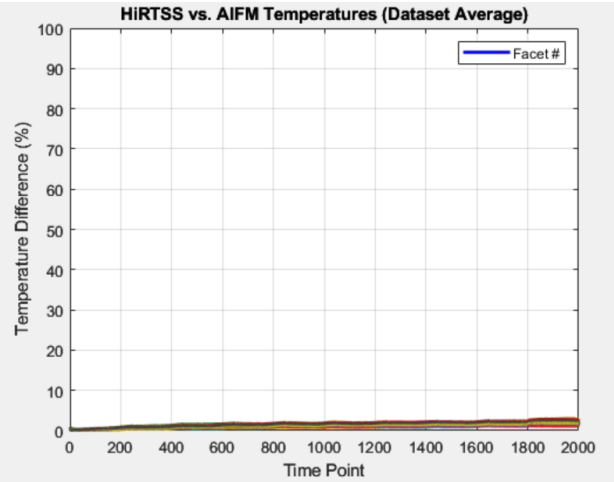

Fig. 35 AIFM Temperature Errors

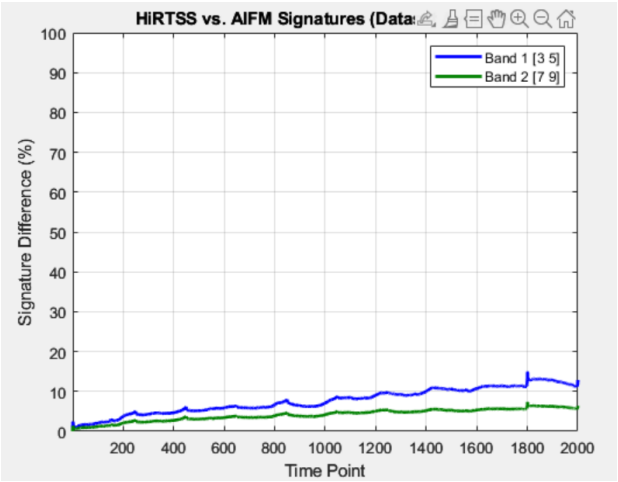

Fig. 38 AIFM Temperature Errors

CUBE SATELLITE WITH PANELS

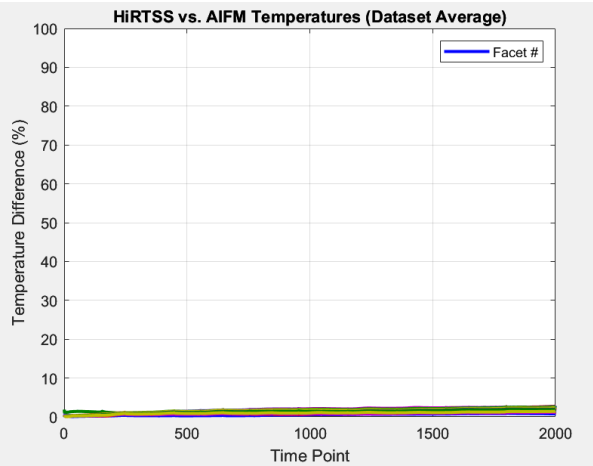

Fig. 36 AIFM Temperature Errors

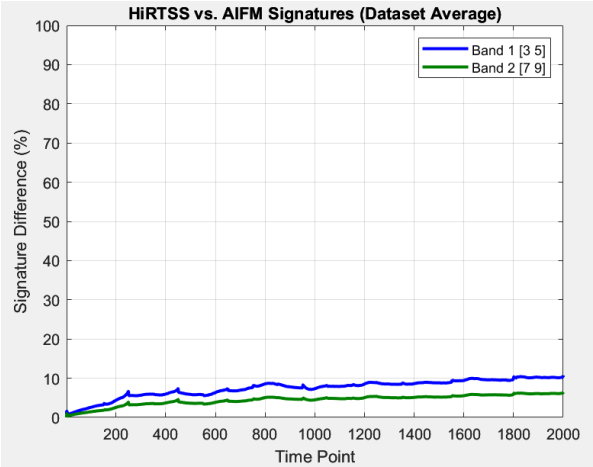

Fig. 39 AIFM Temperature Errors

Fig. 34 – 39 show the test metric averages for the entire dataset of 1000 trajectories. Trajectories and temperatures were interpolated to 2000 time points for analysis of errors in relation to the progression of trajectories. Fig. 34 – 36 show the AI models' temperature errors when compared to the temperatures outputted by HiRTSS. Fig. 37 – 39 show the AI models' signature errors when compared to the signatures outputted by full HiRTSS runs. As shown in Fig. 34 – 39, the AI model's outputs consistently align with the HiRTSS outputs in the thermal and signature domain throughout the trajectory, with the AI model maintaining < 20% in the signature domain for all target models. The AI model also seems to accrue more error over the course of the trajectory until reaching errors of 10-20%. This suggests that shorter trajectories would yield more accurate temperatures than longer trajectories.

HADES, our test and evaluation toolkit, was used to test the AI model of the cylinder satellite with panels. The AI model's signature MAPE for trajectories was used to calculate the HADES reward, with high rewards being an indicator of larger errors. Only two trajectory parameters were varied during trajectory generation: orbital inclination and orbital altitude. Also, panels were rotated to face the sun for all trajectories. As seen in Fig. 40, HADES result maintains low values for most of the performance. However, there are noticeable trends at constant values of orbital inclination. This suggests that the model's performance is more influenced by the orbital inclination than the orbital altitude. More specifically, it seems the model's performance suffers at orbital inclination values of approximately [50 90] deg. This trend could be due to several reasons such as lower frequency of trajectories with those characteristics in the training data. Nonetheless, with this performance space information gained from HADES, the model can be retrained to be more robust.

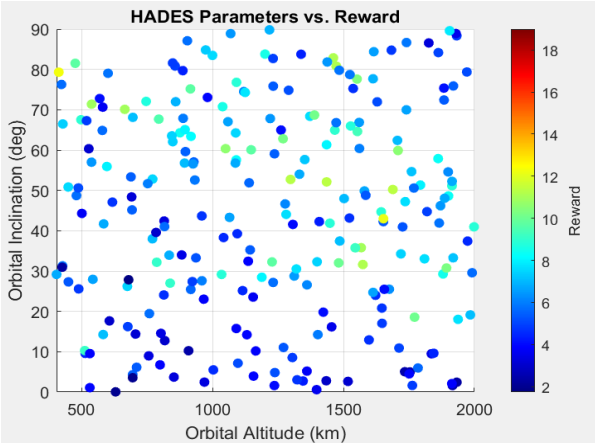

Fig. 40 HADES Results

3.2 APPLICATION OF AI MODELS

We used our AI models to characterize satellite targets. The satellite targets were placed in trajectories using the following orbital parameters for circular orbits: inclination ∈ [0,30,60,90] deg and altitude ∈ [400,600,800,1000] km and three different noise levels: NEI ∈ [1e-15, 1e-16, 1e-17] W/sr/cm^2 in MWIR with LWIR being 5x higher. The true class of the target was a cube satellite with panels. We observed with the constellation mentioned in section 2.1. We performed least squares fit to the truth signals using our three satellite models and selected the one with the smallest chi squared vale. Results are shown in Fig 41

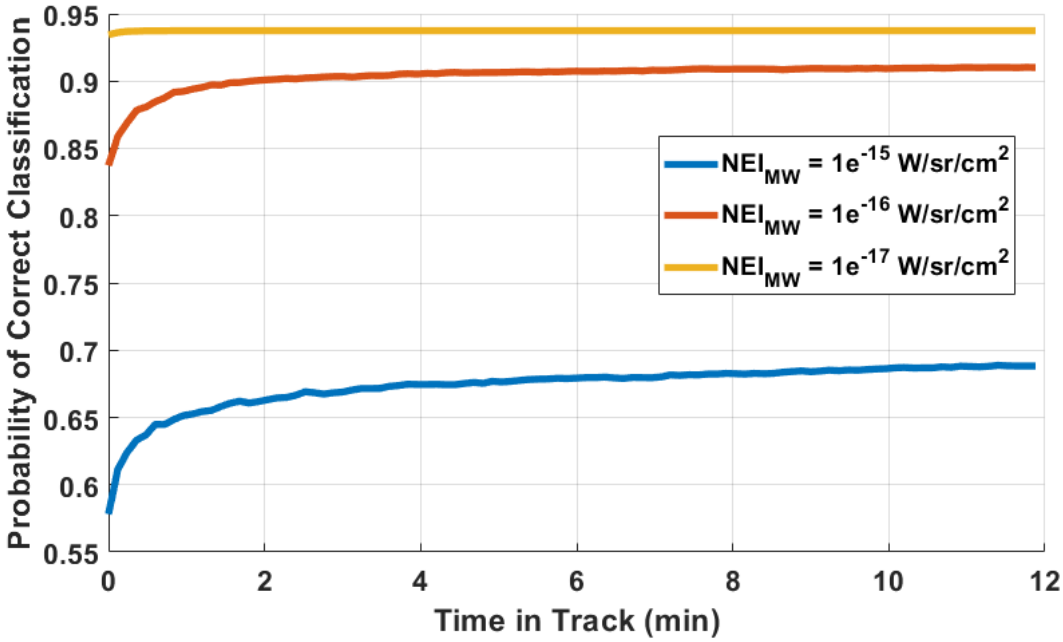

Fig. 41: Target classification results

The last result involves employing the AI model to find the size of the cube satellite with panels using the same infrared signals we used to characterize the satellite target class. Results are shown below. Both results show that an NEI < 1e-16 may be needed to correctly classify and characterize the satellite targets we have examined.

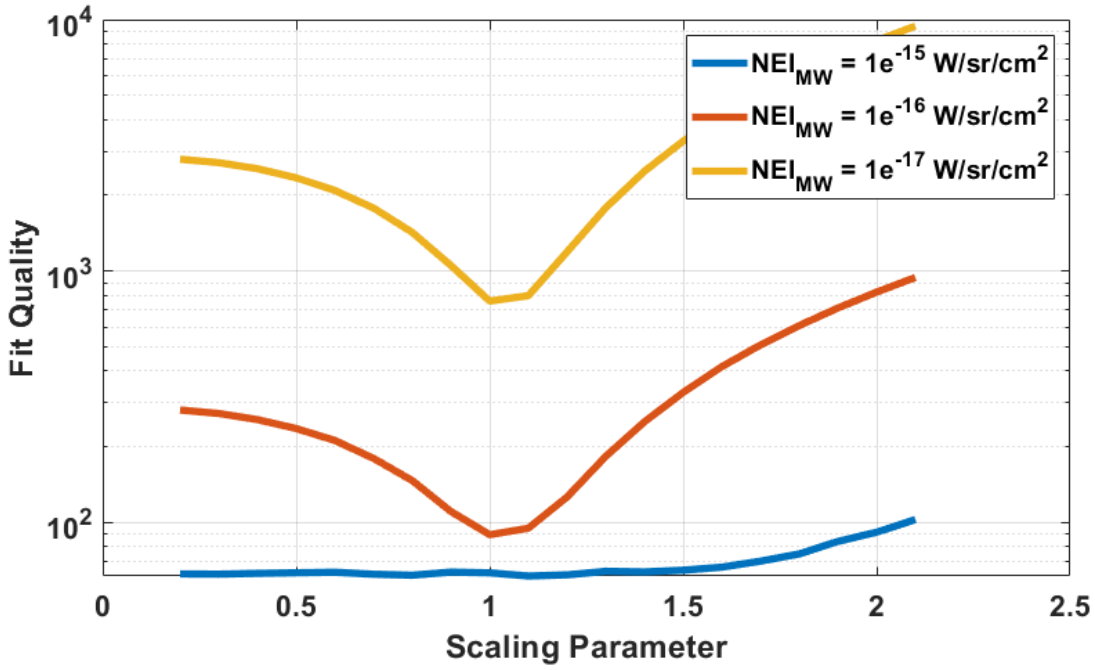

Fig. 42: Target size estimation

4. CONCLUSIONS

We have trained deep neural networks to generate a physical thermal response for satellite targets in space. These models were used to classify target using their infrared light curves. Our models are accurate enough to support simple size estimation as well. We will continue to improve our models so that we can detect changes in attitude as well as changes in signatures. We plan to train across a larger domain of target properties including materials and thicknesses.

REFERENCES

[1] Vaswani, Ashish et al. "Attention is All you Need." *Neural Information Processing Systems* (2017).

[2] Chen, Xiangyu et al. "Activating More Pixels in Image Super-Resolution Transformer." *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2022): 22367-22377.

[3] Ramesh, Aditya et al. "Hierarchical Text-Conditional Image Generation with CLIP Latents." *ArXiv* abs/2204.06125 (2022): n. pag.

[4] Slivkins, Aleksandrs. "Introduction to Multi-Armed Bandits." *ArXiv* abs/1904.07272 (2019): n. pag.