An Innovative, Near-real-time High Fidelity Approach to SSA Data Simulation

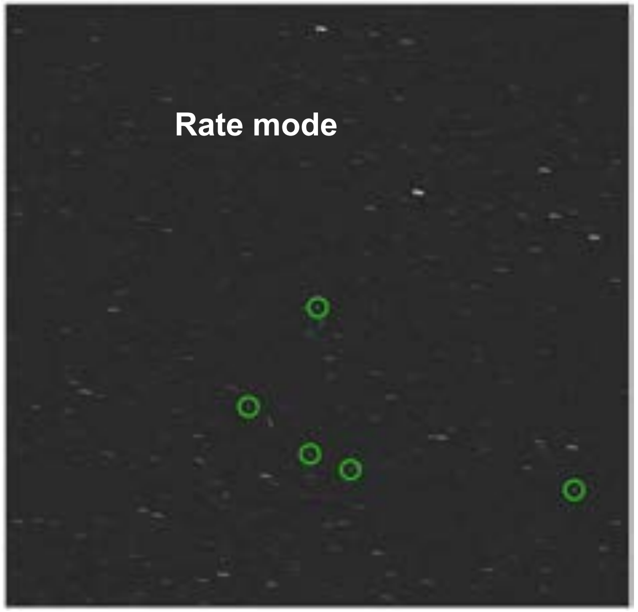
Susan P. Hagerty
Ball Aerospace
Boulder, CO
shagerty@ball.com

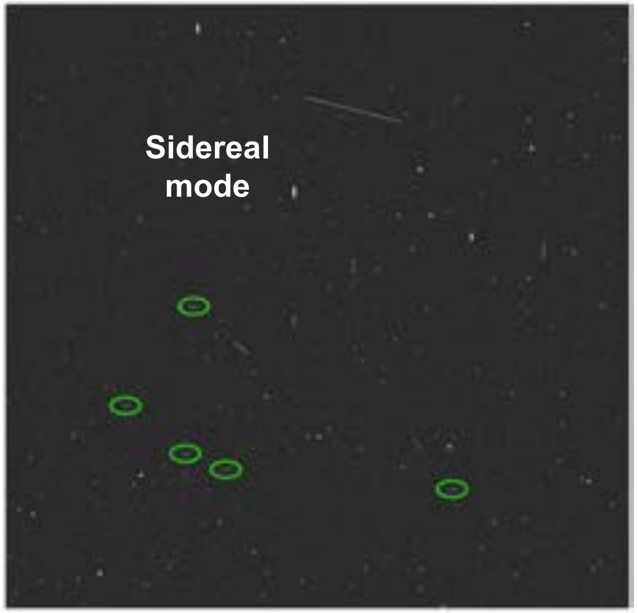
Khurrum Ansari
Ball Aerospace
Boulder, CO

4/18/2019

Agenda

- Introduction / Motivation
- Simulation description / block diagram
- Path to RT-PROXOR™
- RT-PROXOR™ processing
- Analysis and results
- EO model
- Environmental effects
- Software and HWIL simulation
- Future work and conclusions





Simulation examples

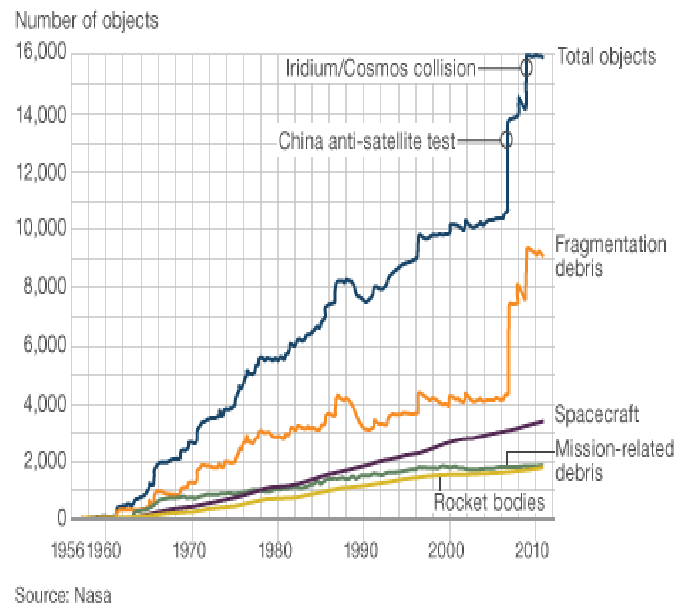
RSOs are circled in green

RSO = Resident Space Object

Introduction / Motivation

- Current challenges include[1] :
  - Increasingly prevalent threats from foreign nations
  - Large number of objects in space
- Challenges make it increasingly difficult to find new adversarial objects, detect changes in objects, and identify nefarious behavior
- As the need for enhanced SSA capabilities increases, the ability to evaluate different mission architectures and optical system designs, *test new algorithms*, and analyze/ predict mission capability performance becomes crucial
- Ball's RT-PROXOR™ provides realistic time-based mission and sensor data simulation that enables evaluating the performance of architecture and algorithms to:
  - Resolve conjunctions, detect new targets, discern changes in target characteristics

Growth of orbital space objects including debris[1]

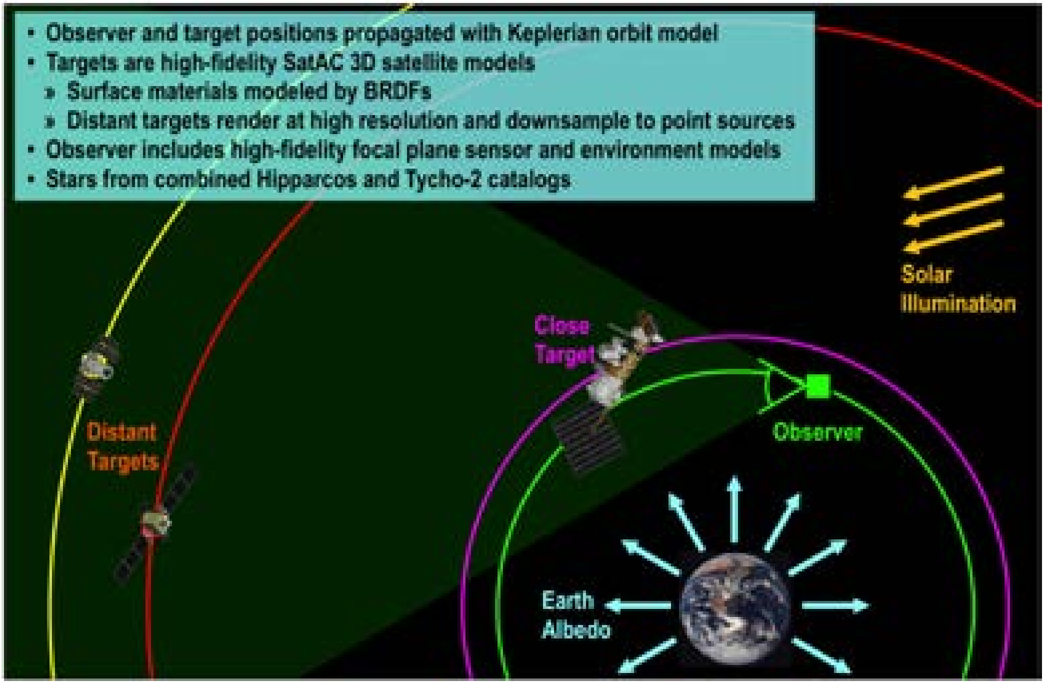


[1]http://www.bbc.com/news/science-environment-14763668

RT-PROXOR™ operates in near-real-time to faster-than-real-time

RT-PROXOR™ contains comprehensive aspects of SSA mission architecture needed for high fidelity simulation

- Observer(s) can be placed in any orbit, or can be fed ECI coordinates on a frame-by-frame basis
- Propagates observer(s) and all RSO positions
- Supports rate track and sidereal track
- Includes
  - Star field based on current frame pointing location
  - Solar illumination and earthshine
  - Intra-frame smear, optical PSF and jitter
  - Accurate radiometry
  - High fidelity EO sensor model
  - Environmental effects: radiation hits and background light

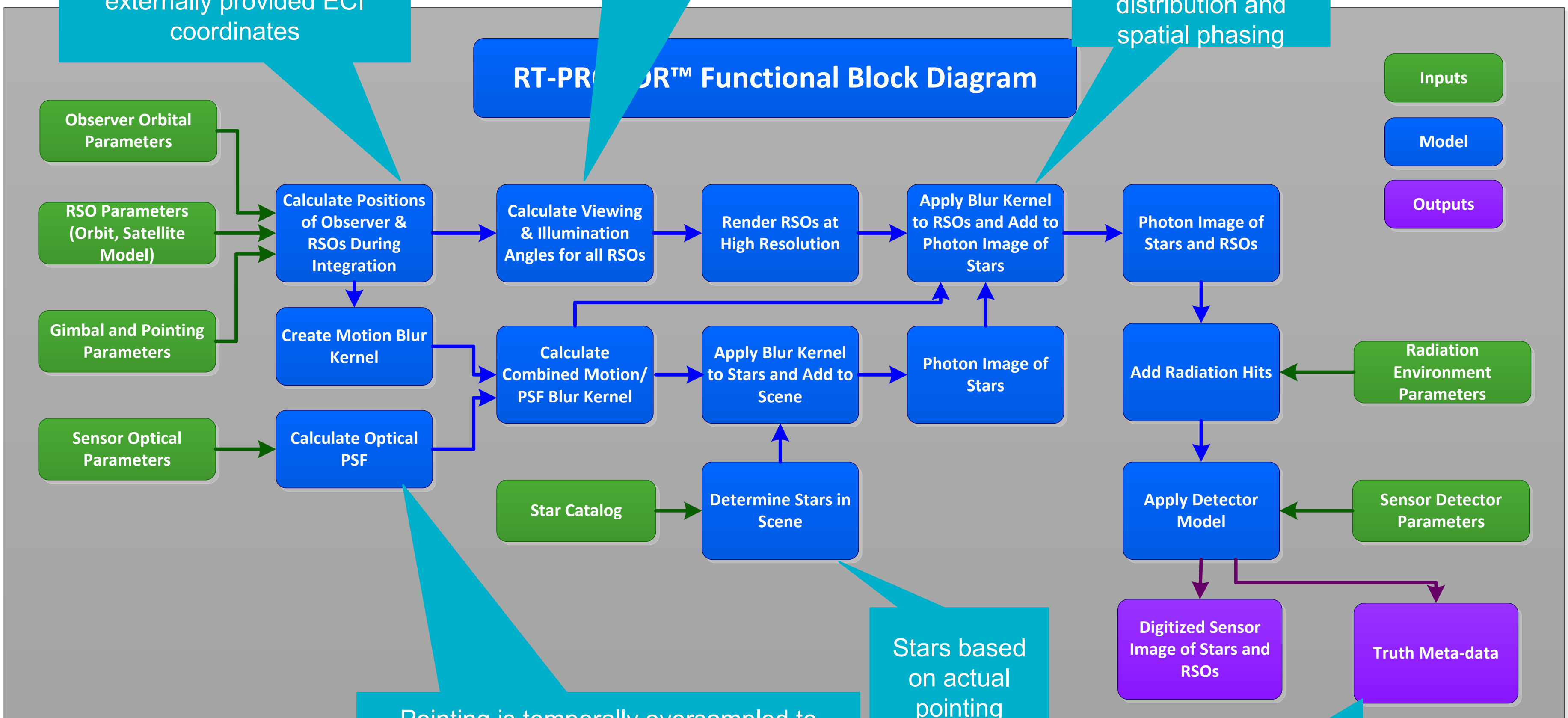


RT-PROXOR™ flow generates realistic sensor outputs

Positions are propagated using TLEs from the industry standard SGP4 or frame-by-frame using externally provided ECI coordinates

Viewing and illumination angles are computed as a function of time; pointing can be rate mode, sidereal or user specified

Spatial oversampling is used to properly capture signal distribution and spatial phasing



Stars based on actual pointing

Pointing is temporally oversampled to capture motions of stars and RSO's *including rotation around line of sight due to gimbal motion*

Truth is crucial for performance analysis and algorithm verification

SGP4 = Simplified General Perturbations 4

Path to RT-PROXOR™

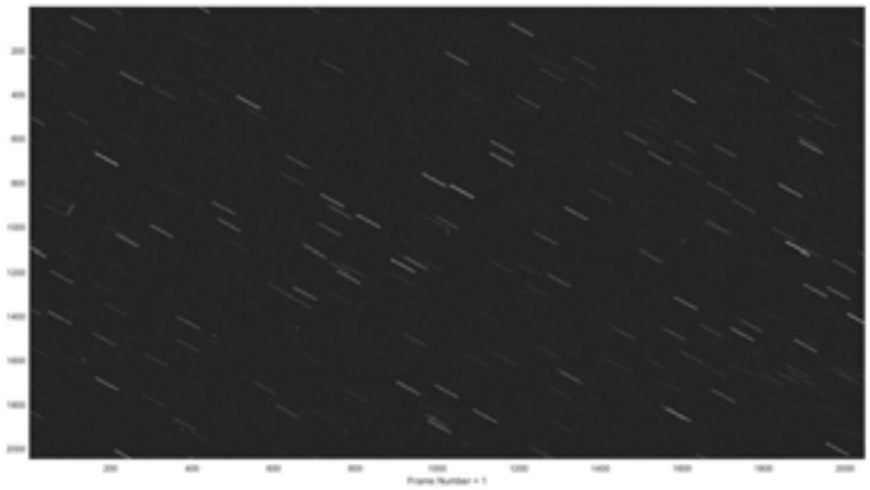
Implementation Step	Description
Software Code Base Conversion	Convert core Matlab functionality into C++ based code for Graphical Processor Unit (GPU) execution.
Updated Pointing Mechanisms	Internal pointing and attitude references were updated from DCM (direction cosine matrix) based expressions to purely 4 element quaternions.
Orbital Propagation	Upgraded the internal orbital propagator to use the industry standard SGP4. SGP4 describes all tracked space objects by the US government using a two-line element (TLE) vector database – facilitates easy modeling of satellites of interest for any date/time.
Hardware in the Loop (HWIL) Interface	-A packetized frame by frame interface for injecting target/acquisition satellites was formulated to allow scenarios/maneuvers in real-time that cannot be described by SGP4 TLEs. -This expands upon the capabilities of specialized use cases of rendezvous/acquisition scenarios. -Allows the updating of system parameters (focal plane/field of view/integration period) in real-time to switch a platform's operational mode. This interface can be injected in a purely software controlled interface as well, with the frame by frame updates ingested via a pre-generated JSON file.

Steps to RT-PROXOR™ lead to significantly faster execution speeds and an efficient validation path by leveraging the non-RT-PROXOR™ validation

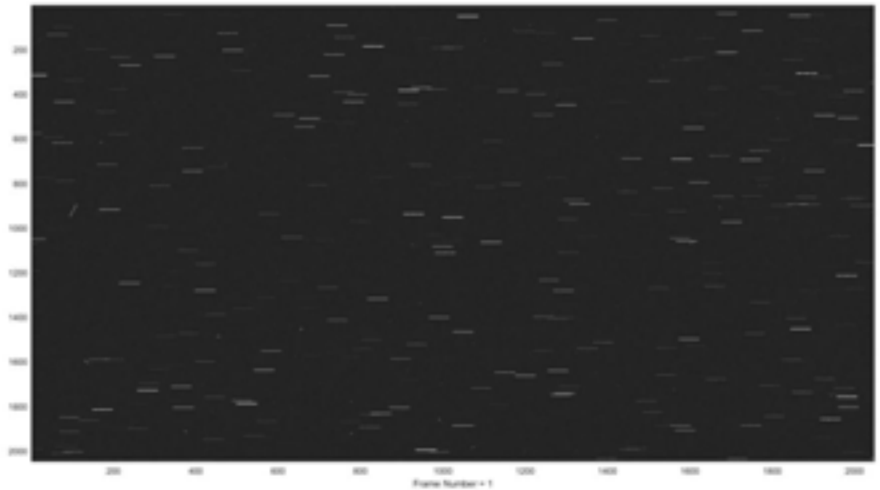
RT-PROXOR™ Processing Has Five Major Steps

1. Generate the position and attitude of the acquisition spacecraft/gimbal and all other satellites of interest
   a. The TLE data of all satellites are propagated many times during the integration period
2. Determine all stars that are visible on the focal plane
3. Create the background image of all the stars in the frame scene
   a. Convolve PSF with each star (variable or fixed kernel)
4. Render the target satellites
   a. Currently uses Lambertian spheres – will be expanded to render extended targets (PROXOR™ already does this)
   b. The image of each RSO is rendered at high resolution and then aggregated back to the sampling of the focal plane pixel
5. Add the effects of the high fidelity detector modeling

Sample RT-PROXOR™ simulated mission scenarios



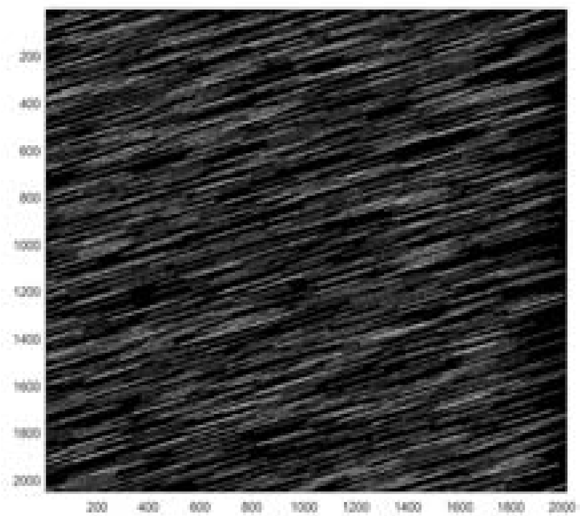
LEO track, medium slew (movie)



Circular slew (movie)

Analysis and Results (1 of 3)

– Sim image with worst case streaking is used for benchmarking



Streaks are about 200 pixels long

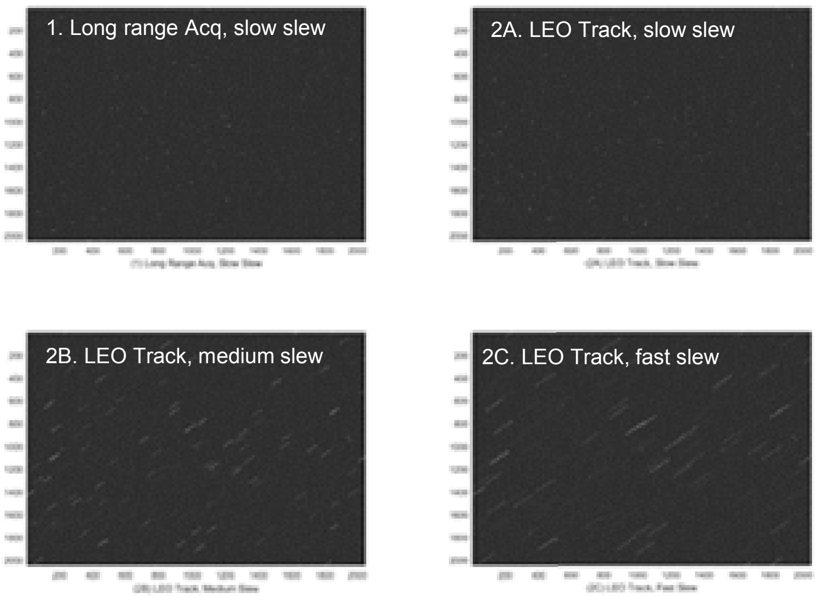
Code Baseline	Code Description	Average Single Frame Execution Time (sec)	Effective Frame Rate (Hz)
Iter 1	Original Matlab PROXOR™ 2017 (not designed for speed)	305	0.003278689
Iter 2	Optimized Matlab RT-PROXOR™ 2017 IR&D	115	0.008695652
Iter 3	Hybrid Matlab RT-PROXOR™ with C++ Star Rendering, 2017 IR&D	18.2	0.054945055
Iter 4	GPU Based C++ RT-PROXOR™, 2018 IR&D	4.4	0.227272727
Iter 5	GPU Based C++ RT-PROXOR™, 2019 Optimized Multi-Kernel Star Rendering Algorithm	0.744	1.3441

Execution times are for scenario with 2048x2048 FPA, 33x Spatial Oversample Factor, 501x Temporal Oversample Factor, and Multi-Kernel Star Rendering

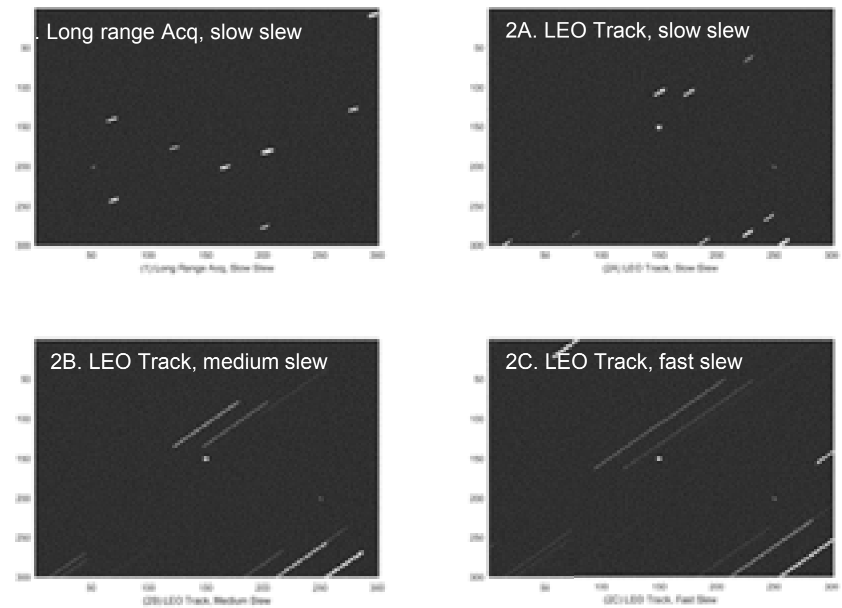
Latest iteration has produced a speed improvement of over 400x as compared to the original Matlab version

Analysis and Results (2 of 3)

– Benchmarking also done with four realistic test conditions



Four test scenarios

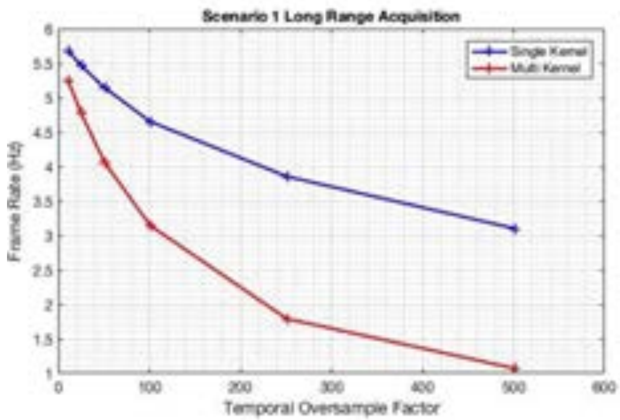


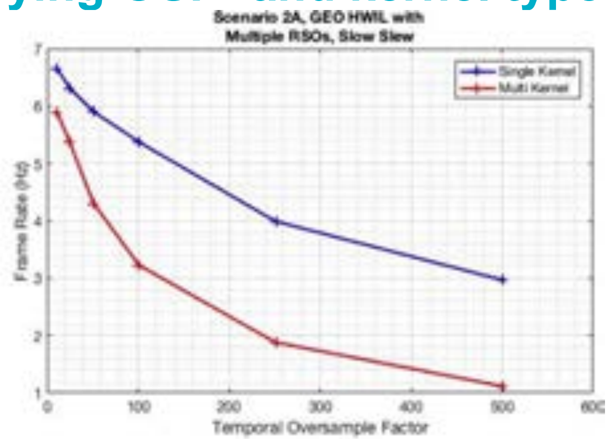
Zoom in of the four test scenarios

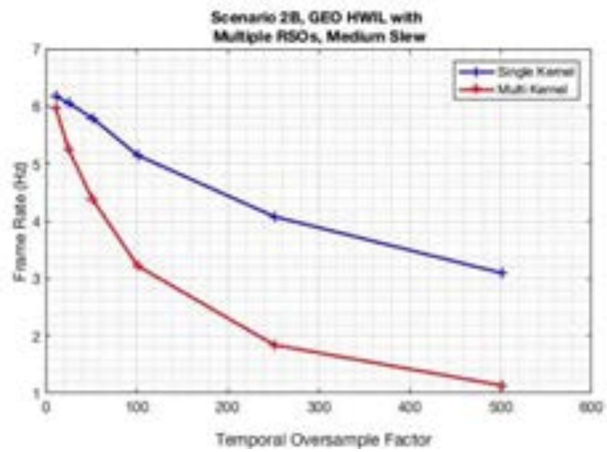
Four different simulations were setup for the benchmarking tests: one ‘typical’ long range acquisition setup (Scenario 1), and three short range acquisition setups with varying slew rates (Scenarios 2A, 2B, 2C)

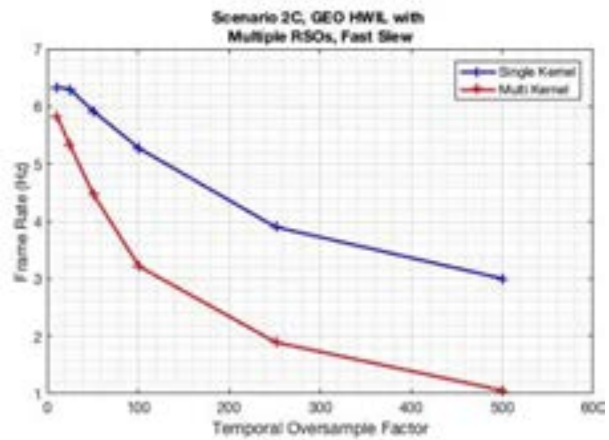
Analysis and Results (3 of 3)

– Benchmarking results shown over varying OSF* and kernel type









Benchmarking results show faster than real-time performance speeds (up to 6-7 Hz) for a 2048x2048 array with a 2 second integration time

*OSF = oversampling factor

PROXOR EO sensor model simulates CCD and CMOS type detectors sensitive to visible/near IR wavelengths

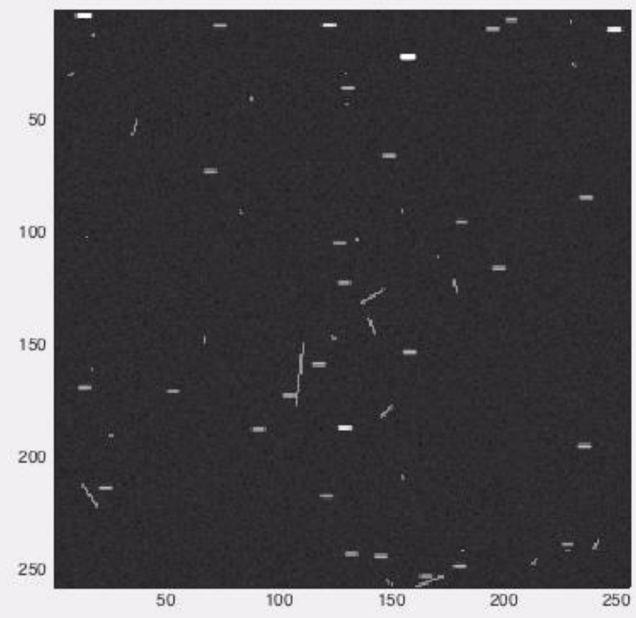
- Pixel-to-pixel fixed pattern noise simulated with Gaussian random pixel gain (QE) and dark current
- Random sample from Poisson distribution is drawn from sum of dark current and scene photon generated electrons
- Electron signal to be digitized for each pixel generated by combining result with Gaussian random sample of read noise
- A/D model quantizes electron signal with user selectable full well, bit depth, and offset
- Additional effects applied by EO sensor model include
  - Image persistence
  - Instrument background

High fidelity sensor model and approach is fundamental to realism of simulated data frames as it behaves close to actual sensor data both spatially and temporally

Radiation hits are simulated to allow testing algorithms in the presence of real environments

- RT-PROXOR™ includes radiation hit modeling with user-selectable spherical angular distribution and user specified parameters controlling the detector thickness, electron deposition rate, and mean number of hits per second

- Two angular distributions are currently defined (isotropic and side-shielded) and other distributions are easily defined



Animation of rate mode tracking shows frame to frame motion of star background and radiation hits from an isotropic angular distribution

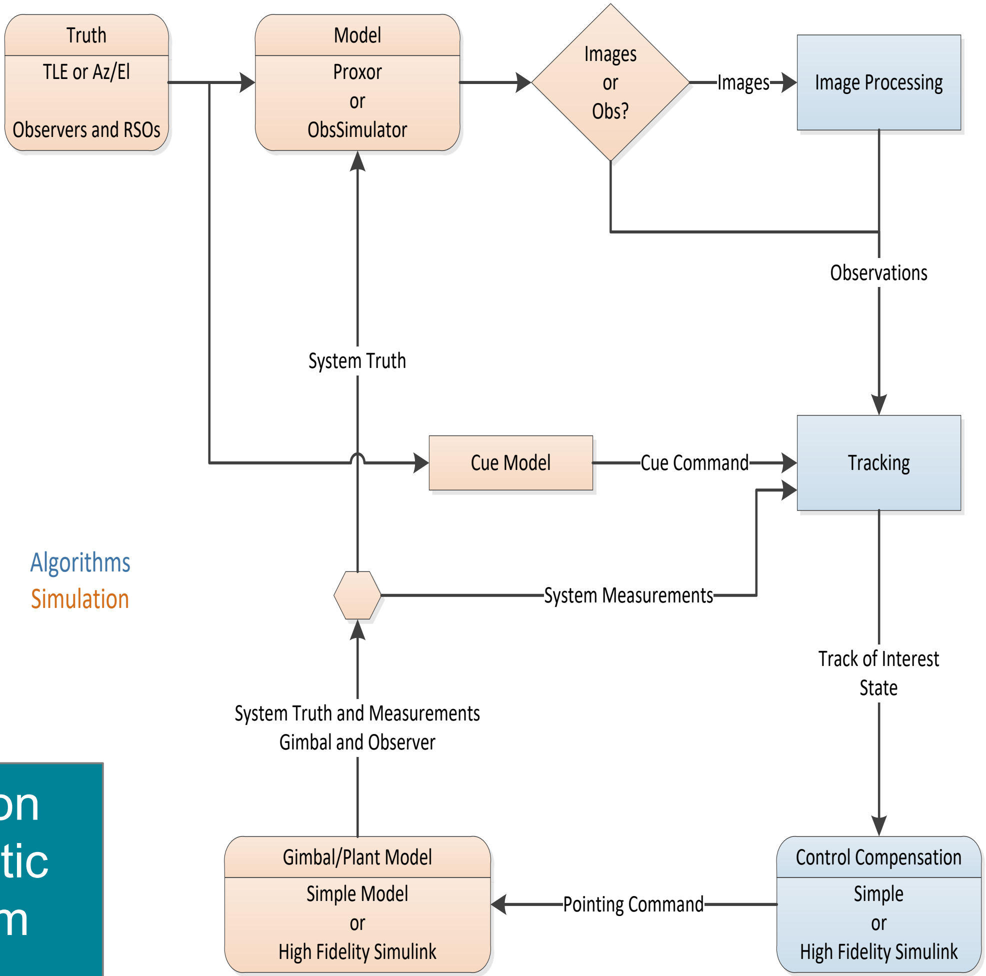
RT-PROXOR™ is critical part of SW Simulation and HWIL

- Both use RT- PROXOR™ to rapidly generate images with medium-to-high fidelity
  - –Can run many hours of testing over varied mission scenarios to increase algorithm quality and robustness
  - –Fully automated
  - –Reduces overall schedule and increases affordability by retiring risks early in the program
  - –Supports both open loop and closed loop testing
  - –Contains common scenario specification, top level interfaces, visualization and analysis tools
- Both approaches can operate via data cubes or a packetized frame by frame interface
- Packetized interface supports modeling of maneuvers in real-time to better model/address threats
- Allows updating of system parameters (focal plane, FOV, integration period, etc.) in real-time to switch a platform's operational mode

RT- PROXOR™ is instrumental in the development of robust algorithms

Ball's SHADE tool simulates open and closed loop systems for SSA missions

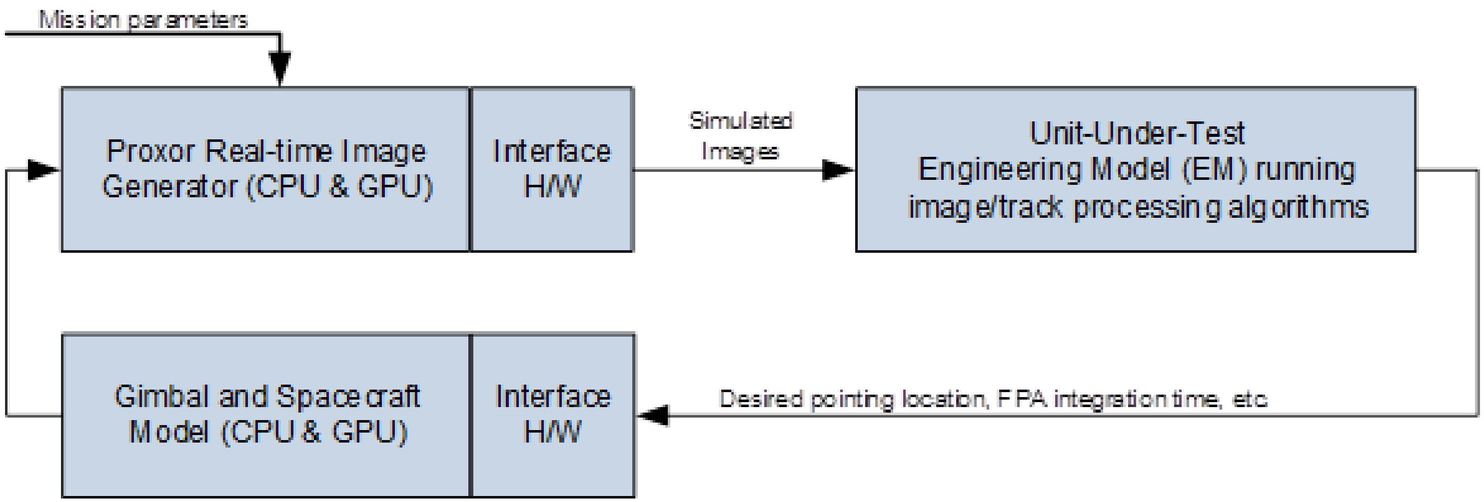
- SHADE includes
  - High fidelity models for image generation (via RT-PROXOR™)
  - Gimbal/plant, and spacecraft and target dynamics
  - Algorithms for image processing, tracking, and control
- Closed loop tracking can be initiated via a single cue or via autonomous acquisition
- Open loop tracking can also be simulated



SHADE provides a crucial simulation tool necessary for performing realistic mission data analysis and algorithm testing

*SHADE = Software-simulation High-performing Algorithm Development Environment

Hardware In The Loop (HWIL) Simulation



- The HWIL Sim uses a similar functional architecture to SHADE
  - Differences are:
    - Uses an Engineering Model (real hardware processor) for the processing algorithms
    - Uses actual hardware interfaces from:
      - RT-PROXOR™ Scene Generator/Scene Injector to the processing board
      - Processing board outputs to the Gimbal and Spacecraft Bus models

HWIL testing provides first opportunity to do thorough testing on real hardware, greatly reducing cost, schedule, and technical risk

Future Work and Conclusions

- Future work
  - Enhanced user interface
    - Provides mission designer with a flexible and visual front end to facilitate quick and effective scenario mockups
  - Add extended target rendering
  - Multi-GPU development
- Conclusions
  - RT-PROXOR™ addresses a critical need for supporting NRT/RT/faster-than-realtime scene generation for Software Simulation and HWIL simulations
    - RT-PROXOR™ supports high-fidelity, extended duration, mission scenario testing of mission data processing algorithms
    - Reduces program risk and increases mission performance
  - For worst case scenarios, Ball's development to date has shown a 400x improvement in speed as compared to our non-real-time PROXOR™ version