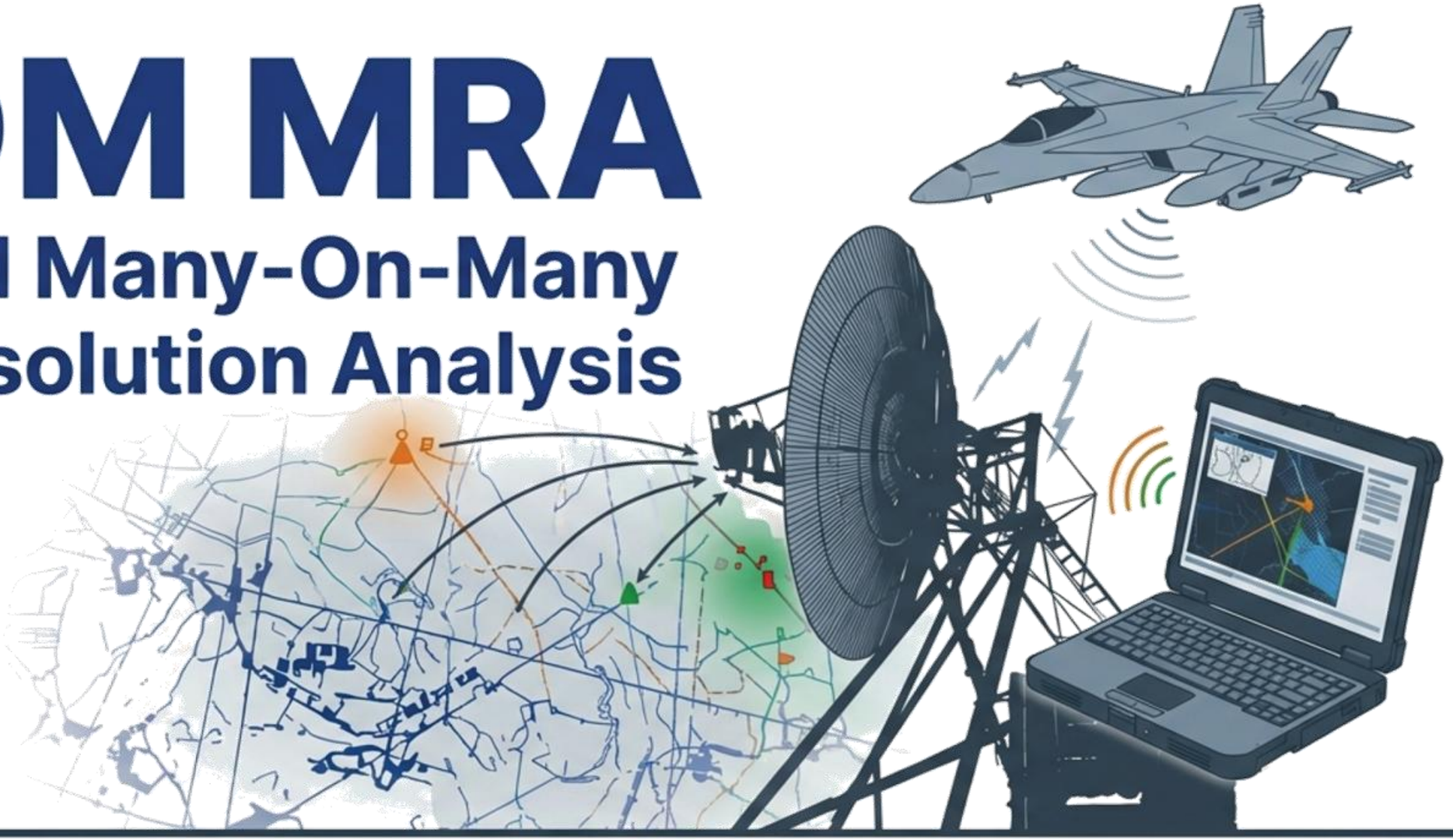
IMOM MRA

Improved Many-On-Many Multi-Resolution Analysis



Prototype Design & Development Test Using AI Assistance
GPU-Accelerated Architecture | Version 0.1
Technical Overview & Operational Workflow

TECHNICAL DOCUMENTATION //
Developed by Brian Morrison
CAO: 24 Jan 26

MY BACKGROUND

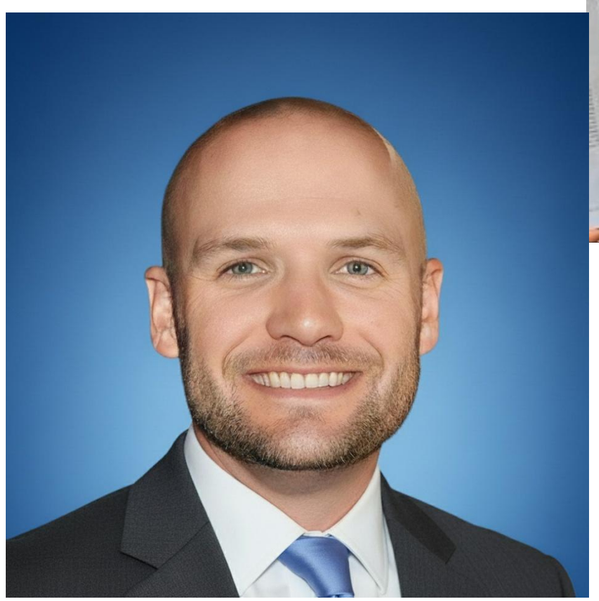
Brian Morrison

DoD systems architect and technical leader with 20+ years supporting joint and coalition operations across intelligence, secure communications, command and control, and unmanned systems.

Retired U.S. Air Force Chief Master Sergeant (Apr 2024) with 23 years of service in intelligence operations and associated systems development.

Career Highlights:

- Team Lead supporting SAF/CDMX Plans & Requirements; leads client delivery, risk management, and requirements execution for Mission Partner Environment capabilities.
- Selected as the Department of the Air Force's first Large Language Model (LLM) Specialist & Liaison in the DAF Chief Data & AI Office (CDAO); represented Air Force equities to Task Force Lima under OUSD CDAO and led service-wide AI adoption outreach.
- Led USAFE-AFAFRICA's 24-person innovation program delivering 100+ projects valued at $130M+, aligned to JCIDS and PPBE processes.
- Designed and executed open innovation challenge (ThunderDrone) through SOCOM's SOFWERX innovation organization engaging 400+ participants for prototyping and evaluation of Group 1-2 UAS capabilities.
- Founded/co-chaired the Joint Intelligence Google Earth Working Group for seven years (1,200+ members); built the first advanced Google Earth training course for the DoD/IC and trained 300+ RPA personnel across five units.



THE PROJECT

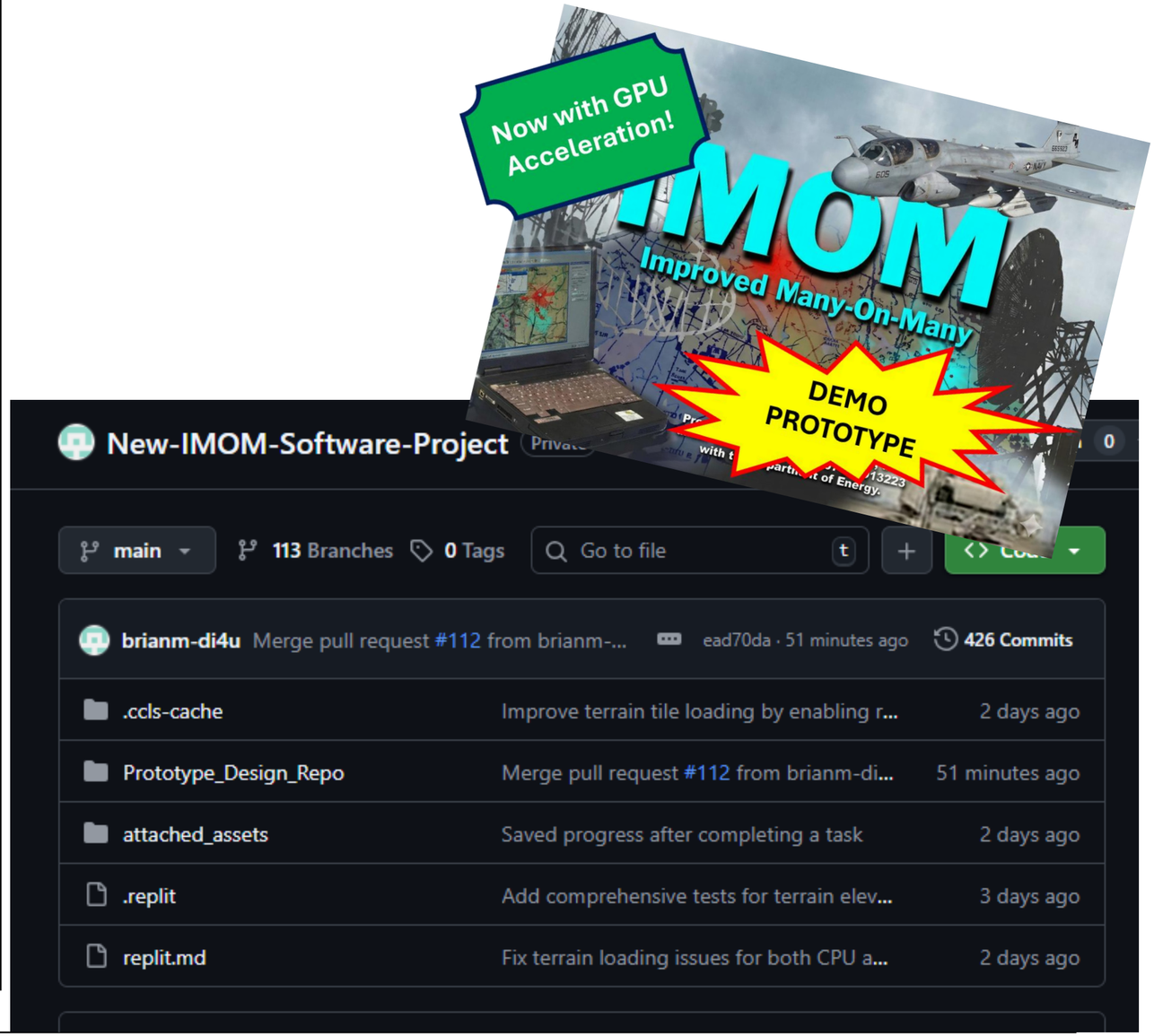
TEST HYPOTHESIS:

Modern AI coding tools **SHOULD** make it possible to reverse engineer the primary functions of highly complex GOTS software applications like IMOM Planner within reasonable time frames and have the same or better performance.

1. **Design:** Distill/Infer from public sources
2. **Build:** Utilize AI coding tools for speed
3. **Operate:** Windows PC software app
4. **Perform:** Faster than Planner???

This whole project was about trying to determine the art of the possible within a limited span of time with the data and tools available…there was only a hope not an expectation of success.

***Note:** Multiple features were added on later due to initial success



HOW I USED THE TOOLS

Research – Google Gemini

- Google Gemini's deep research functions are currently the fastest and very robust, so for speed alone they can be the best first pass of data identification and collection.
- It wasn't required for this project, but depending on the future development of available AI tools, or lack of access to various AI platforms, supplementing or replacing the deep research functions of Google's Gemini with OpenAI's ChatGPT or xAI's Grok would be functionally sufficient.
- The end state you are going for is to collect a repository of source documents for the design phase, so how well an individual AI "summarizes" it findings is mostly immaterial, as you are looking for links to the source documents themselves.
- Depending on what knowledge gaps and data deficiencies result from your initial research, you may end up needing to utilize any number of additional tools to ultimately address those issues.

Design & Documentation – OpenAI GPT 5.2 Pro & Codex

- This is where capability REALLY matters, so using the absolute best AI model you can to build the software development requirements and planning documentation is critical!
- At the time of this project OpenAI had the best general-purpose model with GPT 5.2 pro, and that model was used to generate both the software development requirements and the software development plan.
- As OpenAI provides the access to Codex with the necessary ChatGPT Pro account required to access GPT 5.2 pro model, it made the most sense to utilize Codex for the initial code development as well.

Coding – OpenAI Codex, Replit Agent 3, GitHub

- Github was used as the primary code repository to integrate the codebase across AI coding tools, and to sperate working from "developed" code bases
- Codex was used for all initial code drafting, then later major bug fixes
- Replit was used to construct, deploy, and test all but the final builds of the windows application on the local laptop pc.

TEST EQUIPMENT

Local Windows PC for final software build and testing

- Manufacturer: Dell
- Model: Inspiron 16 Plus 7640
- Processor: Intel(R) Core(TM) Ultra 7 155H, 3800 Mhz, 16 Core(s), 22 Logical Processor(s)
- RAM: 16 GB
- Hard Drive Size: 1 Terabyte SSD
- GPU: NVIDIA GeForce RTX 4060 Laptop GPU
- OS: Windows 11

DESIGN PROCESS

1. **Search for Technical Papers, Research Papers, and any Program Descriptions or Screenshots**
   - You are looking for the math, functional descriptions, and any images of the target application to reconstruct into a set of design requirements.
2. **Unless you have the original application's full suite of technical documents…there will be information gaps**
   - You aren't going to have and can't use any classified data in public domain software development, so you will need to construct your own replacements for that data to address mandatory gaps.
   - There will likely be more than a few gaps in either math or functional descriptions that will impede requirements generation based on the conglomeration of data from the research.
3. **Fill the Data Gaps with best approximations**
   - Here's were you go back to a combination of research and approximation.
   - Plenty of real-world data that can get you a defined enough boundary of knowledge, that the best AI models can infer adequate/usable data to fill the gap.
4. **Build Development Requirements**
   - Once you have filled all your original "known" data requirements, construct your initial set of fully detailed software development requirements.
   - Make any revisions to the development requirements necessary to address previous question or fulfill checklist.
5. **Build Software Development Requirements Checklist**
   - Review your requirements vs your original data set, focusing on cross checking the data from any user guides, screenshots, or other application descriptions (Will requirements result in an application that looks and works like this?).
   - This will serve as both the high-level list of initial requirements, and a functional check of the intended system design.
   - Repeat Step 3 and/or 4 until all checklist items are met.

PLANNING PROCESS

1. **Analyze requirements for potential code base, component, libraries, dependencies, and other development choices**
   - The requirements tell you what needs to be accomplished, but there are a LOT of ways to skin a cat.
   - This is also where having well defined functional end-state goals for the software is equally important.
   - Your functional goals and development requirements will ultimately limit you to or drive best choice for design aspects.
2. **Consolidate design choices into additional development requirements**
   - Once you have chosen all the associated ways of “how” the software architecture will be constructed, you need to add them into an additional document or list within your software development requirements.
3. **Build the first draft of the Software Development Plan**
   - Using all the software development requirements as the foundation and end-state validation criteria, construct a highly detailed multi-phase software development plan.
   - The software development plan should be broken down in a hierarchy of phases, steps, and details.
   - This plan should explicitly detail every step of the entire development process from the highest to lowest level.
   - Deconstruction of the development process into its smallest segments is key to ensuring that the AI model both builds what you intended, but also doesn’t leave anything out (assumptions are where corners are cut and bugs creep in).
4. **Review the Software Development Plan**
   - An obvious, but important step, is to manually (human) review the entire development plan.
   - There will almost always be something that you realize wasn’t either well defined, or needs to be added to fully flesh out some part of the overall process….like did you including test harnesses for each module?
   - This might even drive you all the way back to the requirements process for modification, depending on what is identified.
5. **Edit the Software Development Plan**
   - Make edits as necessary to the software development plan, until all requirements are met and the plan is sufficient.

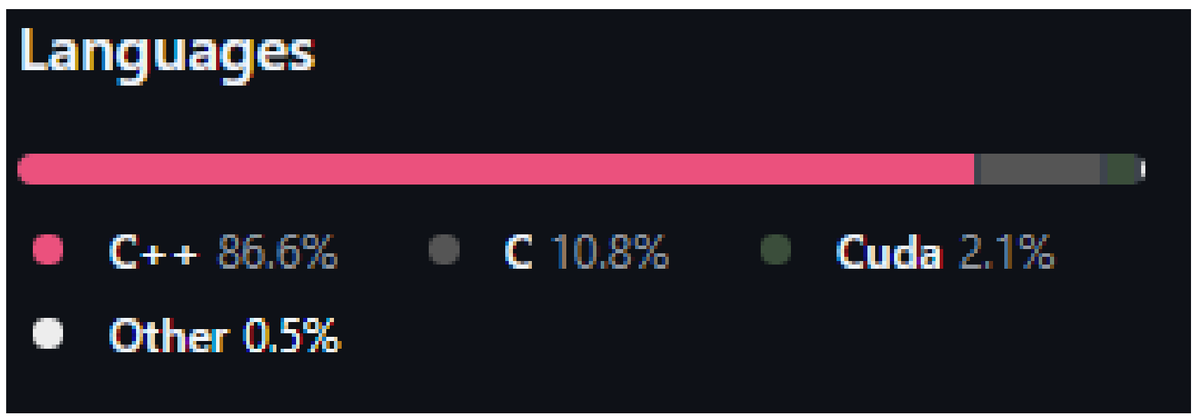
BUILD PROCESS

1. **Time to code**
   - Based on your tool utilization, start in best (or only) coding tool, and provide the software development requirements and plan for execution.
   - This is where prompt engineering comes in, as just saying “execute these docs” is just asking for trouble.
   - The model should execute the plan in its full extent to construct the initial draft of code for all or some of the modules, depending on the specifics of your plan.
2. **Monitor the process!**
   - No AI coding tool is prefect, and depending on the tool and its many characteristics, you’ll likely run into some kind of at least quirk of its coding process that will at a minimum “tweak” its execution of your intended plan.
   - This is where you really NEED TO KNOW something about coding specifically and software development in general to truly understand what you are looking at, and to be able to tell if/when the automation is going off track.
   - Generally, the simpler the plan the less deviation from the plan, but the more complicated the plan the more possibility for deviation…so watch out!
3. **Github is your friend, and separation of the codebase is important**
   - Most coding tool will enable you to keep your developed code base local to whatever the development environment happens to be, but it is invaluable to keep the work separate from the codebase, and to catalog the development.
   - Github is the perfect mechanism to accomplish both the separation, and the cataloging of the development, and is fully integrate in and with all AI coding tools and platforms.
   - 100% chance you will get to some point in testing and realize…the initial plan didn’t work…and you’ll need to back track.
4. **Tests are your early filter for and resolution of bugs.**
   - No code is “perfect” on first write, so built in and execute as many tests as possible all along the development process.

MODULAR SYSTEM ARCHITECTURE

Technology Stack:

- **C++20** project build with CMake.
- **YAML-CPP** for scenario parsing.
- **GDAL** for terrain ingest and KML/geo outputs.
- **GeographicLib** for geodesic calculations.
- **Arrow/Parquet** optional output format for tabular data.
- **CUDA (optional)** for GPU backend and terrain acceleration.



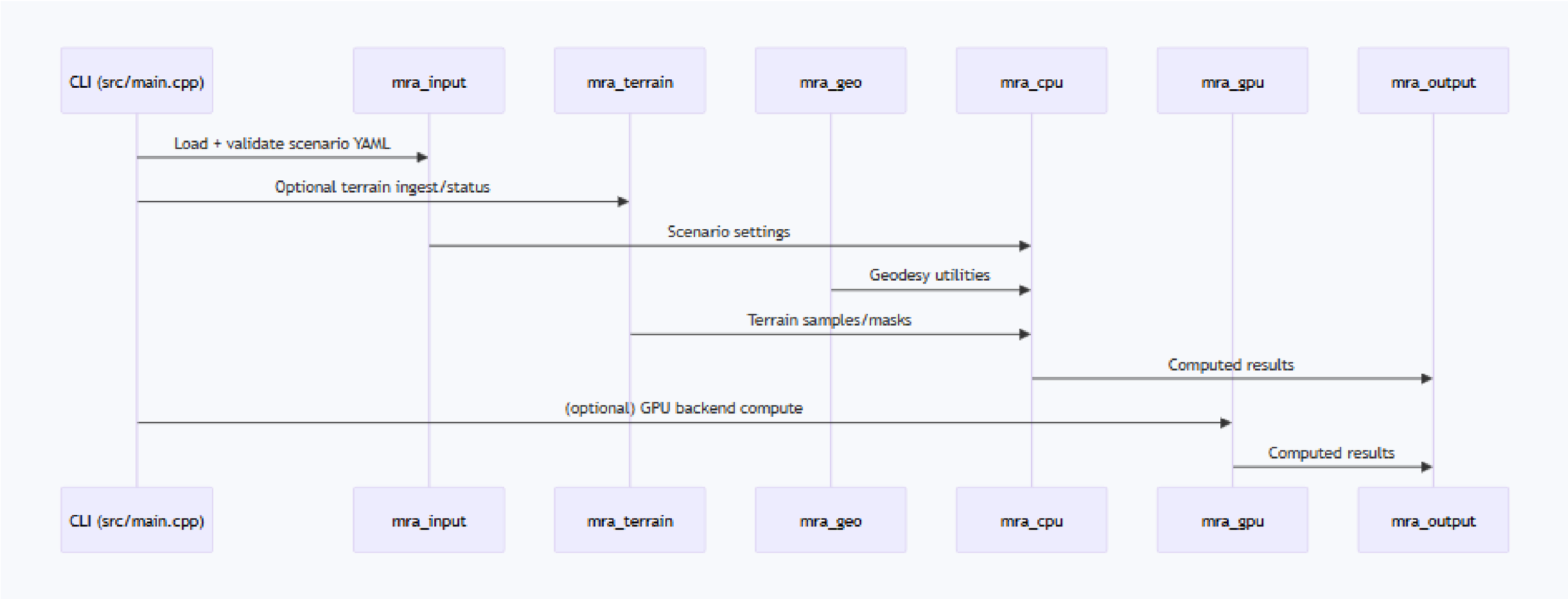
Component Decomposition:

- **mra_input:** Load scenario YAML, validate schema, parse radar/jammer parameters, and normalize units. It owns the scenario schema definition and loader routines.
- **mra_geo:** Geodesy utilities and earth curvature calculations used by CPU/GPU backends. Uses GeographicLib for geodesic calculations.
- **mra_terrain:** DTED ingest tooling and terrain lookups/masking backed by GDAL. Supports GDAL-based terrain access and ingest utilities.
- **mra_cpu:** CPU backend compute implementation that consumes inputs, geodesy, and terrain data.
- **mra_gpu:** CUDA-based backend (optional) for GPU-accelerated computations plus GPU terrain operations. Built only when CUDA is enabled.
- **mra_output:** Output writers for tabular results (CSV/Parquet) and KML artifacts. Links against GDAL and optionally Arrow/Parquet.
- **mra_core:** Core metadata and build info (version, build profile, compile-time feature flags) used by the CLI.

HIGH-LEVEL DATA FLOW

1. **Scenario YAML ingest**
    1. The CLI accepts a scenario YAML path and optional overrides (backend, precision, performance profile, terrain options, and output settings).
    2. mra_input loads YAML into the validated scenario schema (ScenarioSettings, MraConfig, OutputConfig, etc.).
2. **Scenario normalization & validation**
    1. Parsing includes radar/jammer parameter ingestion and unit normalization as part of mra_input loader utilities.
3. **Terrain ingest/lookup (optional)**
    1. When terrain masking is enabled, the CLI can ingest DTED tiles or run terrain status queries. Terrain data is consumed via GDAL-backed lookup in mra_terrain.
4. **Backend computation**
    1. Depending on backend mode and build profile, the CPU and/or GPU backends compute detections, coverage, max-range, or route analysis outputs. CPU depends on geodesy and terrain components; GPU is optional when CUDA is enabled.
5. **Outputs**
    1. mra_output writes tabular results (CSV or Parquet via Arrow) and KML artifacts for coverage visualization. Writers live in src/output/*.

DETAILED DATA FLOW



CPU MODULE DESIGN

Module Overview & Purpose

The CPU backend entry point is mra::cpu::RunMraCpu, which orchestrates geometry setup, terrain sampling, jammer preprocessing, and per-altitude evaluation. It returns a vector of PointResults, one entry per PE altitude. Each PointResults instance stores per-point fields (radial/step indices, location, ranges, SNR/JS values, detection status, and jamming effectiveness) used by downstream analysis or visualization. The per-altitude evaluation is delegated to RunMraCpuSingle, which operates on a single PE altitude, applying scan limits, terrain masking, detection logic, and jamming evaluation. The PointResults structure defines the output data schema for all computed results.

Dependencies & Libraries

The CPU backend combines internal math utilities with external libraries:

- **GeographicLib**: Uses GeographicLib::Geodesic::WGS84() for precise geodesic inverses (when not in PerfFast mode). This is used in jammer-to-point range and (optionally) PE-to-point ground range calculations.
- **GDAL terrain sampling**: Terrain sampling is performed using mra::terrain::TerrainGdal and mra::terrain::SampleTerrainGridMsl, configured via BuildTerrainConfig. The CPU backend uses sampled terrain to compute terrain masking and to set radar altitude above MSL.
- **OpenMP (optional)**: The per-radial/per-step evaluation loop is parallelized with OpenMP when enabled via CpuBackendOptions and compiled with _OPENMP.
- **Internal imom_math formulas**: Detection and jamming calculations use imom::math functions such as radar_signal_w, receiver_noise_w, jamming_signal_w, radar_signal_for_js_w, js_min_from_snrmp, mjv_from_jsmin, and linear_to_db.

CPU MODULE DESIGN

Mathematical Formulas

Radar Signal and Noise

- **Radar signal (Watts)**
  - Function: imom::math::radar_signal_w
  - Inputs: transmitter power Pt_w, gains (Gpc_lin, Gt_lin, Gr_lin), losses (Lr_lin), radar cross-section sigma_m2, range Rt_m, frequency fr_hz.
  - Output: received signal power in Watts used for SNR.
- **Receiver noise (Watts)**
  - Function: imom::math::receiver_noise_w
  - Inputs: receiver bandwidth Br_hz, noise factor Fn_lin, reference temperature (defaults to t0_k).
  - Output: noise power in Watts.
- **SNR (linear, dB)**
  - CPU backend computes snr_lin = signal_w / noise_w (same form as imom::math::snr_linear).
  - SNR dB is linear_to_db(snr_lin) (wrapped in SafeLinearToDb to handle non-positive values).

Detection logic: SNR is compared to snr_min_lin. If SNR is below threshold, status becomes kNotDetected; otherwise detection proceeds to jamming evaluation.

Geometry & Angle Calculations

- **Haversine range (meters)**
  - Function: HaversineRangeM
  - Inputs: earth radius, two lat/lon pairs (radians), precomputed sin/cos for both points.
  - Output: spherical surface range in meters. Used in PerfFast build profile for jammer-to-point range and PE-to-point ground range calculations.
- **Nadir angle cosine**
  - Function: CosNadirAngle
  - Inputs: earth radius, source altitude, target altitude, surface range.
  - Output: cosine of nadir angle (used for scan limit checks and terrain masking).

Use in detection flow:

•cos_alpha (radar-to-PE) compared against scan limits to flag in_scan_limits.
•cos_target (PE-to-radar) compared with min_cos_obstruction to determine terrain masking.
•cos_terrain updates min_cos_obstruction per radial for subsequent points.

Jamming Signal and JS Metrics

- **Jamming signal (Watts)**
  - Function: imom::math::jamming_signal_w
  - Inputs: jammer power Pj_w, bandwidth ratio BWratio, jammer/receiver losses Lc_lin, Lp_lin, Lr_lin, gains Gj_lin, Grj_lin, range Rj_m, and jammer frequency fj_hz.
  - Output: jammer power at receiver (Watts).
- **Radar signal used for J/S**
  - Function: imom::math::radar_signal_for_js_w
  - Inputs: similar to radar_signal_w but includes Gi_lin and uses Gi (interference gain).
  - Output: signal power used in J/S ratio.
- **J/S minimum from SNRmin**
  - Function: imom::math::js_min_from_snrmp
  - Output: 1 / snr_min_lin is the minimum J/S ratio needed to deny detection.
- **MJV from JSmin**
  - Function: imom::math::mjv_from_jsmin
  - Output: minimum jamming-to-victim power threshold after accounting for jammer loss and interference gain.

Jamming evaluation path:

1. CPU backend computes js_signal_w via radar_signal_for_js_w and jamming_w via jamming_signal_w for each active jammer.
2. js_lin = jamming_w / js_signal_w.
3. margin_db = linear_to_db(js_lin) - linear_to_db(mjv_lin).
4. EvaluateEffectivenessPct maps margin to a % effectiveness via jammer-specific breakpoints/mapping (continuous/step/linear).
5. JamStatusFromPct converts the % effectiveness into a StatusCode.

GPU MODULE DESIGN

Module Overview & Purpose

“mra_gpu” is the GPU entry point that executes the MRA pipeline on CUDA when a device is available. If no CUDA device is detected, it falls back to the CPU backend (mra::cpu::RunMraCpu) while preserving parity or performance intent based on GpuBackendOptions::parity_strict. The fallback sets CPU build_profile to parity-strict or perf-fast and forwards emit_debug_intermediates so output diagnostics remain consistent across backends. GpuWarmup performs a lightweight CUDA warmup, and ResetGpuWorkspace clears cached buffers and streams. GpuBackendOptions controls parity mode and whether debug intermediates (like dbg_cos_scan and dbg_cos_mask) are produced for debugging kernel results.

Dependencies & Libraries

- **CUDA Runtime**: Core GPU behavior relies on CUDA runtime APIs for memory allocation (cudaMalloc, cudaMallocHost), async copies (cudaMemcpyAsync), streams, and events. The kernels GpuTerrainSampleKernel, BuildTerrainPrefixKernel, and RunMraKernel are CUDA device code.

- **GeographicLib**: Used on CPU to compute geodesic distances for ground range generation when the center reference is not the PE. This ensures range-to-point calculations match CPU geometry logic.

- **GDAL Terrain Sampling**: The terrain pipeline uses mra::terrain::TerrainGdal to load tiles and then optionally uploads them into a GPU texture via TerrainTextureCache. When GPU textures are available, GpuTerrainSampleKernel performs terrain sampling; otherwise it falls back to CPU sampling using SampleTerrainGridMsl.

- **Internal imom_math**: The GPU kernels call shared math helpers (e.g., radar/jamming equations and dB conversions) compiled for device use via IMOM_HOST_DEVICE. This includes formulas for radar signal, noise power, jamming signal, JS ratio, and conversions to/from dB.

GPU MODULE DESIGN

GPU Memory Model

Device buffers

The GPU backend uses a suite of custom buffer wrappers:

- **DeviceBuffer<T>**: Owns device memory, resizable, and provides async H2D/D2H copies. It is used for geometry, terrain, jammers, and output buffers.
- **PinnedHostBuffer<T>**: Allocates pinned host memory for larger downloads. Output transfers use pinned staging for buffers above kPinnedThresholdBytes to improve copy performance.

Geometry, jammer, and output caches

The workspace maintains separate buffers for float32 and float64 paths:

- **Geometry caches**: DeviceGeometryBuffers hold latitude/longitude, trig cache (radar_lat_sin/cos), terrain elevation, and ground range data. Geometry hashes (HashGeometryInputs) avoid re-upload when scenario geometry is unchanged.
- **Jammer caches**: DeviceJammerBuffers store jammer parameters and breakpoint vectors with caching keyed on jammer data and js_min_lin. Active jammers are preprocessed, sorted, and uploaded in contiguous arrays.
- **Output buffers**: DeviceOutputBuffers contain terrain mask flags, detection flags, SNR/JS linear+db, effectiveness percent, status codes, radar altitude, and debug intermediates. A two-slot output buffer (kOutputBufferSlots = 2) enables overlap between kernel execution and host downloads.

Streams, events, and multi-slot output buffering

- **Streams**: A compute stream (compute_stream) runs kernels; a copy stream (copy_stream) performs D2H copies.
- **Events**: Per-slot events (kernel_done_events, copy_done_events) synchronize kernel completion and copy completion to enable overlapping work.
- **Double-buffered outputs**: Each altitude iteration uses a slot idx % kOutputBufferSlots so the next kernel can start while the prior altitude's results download in parallel. Finalization (FinalizePendingResult) waits on the copy event and performs any float-to-double conversion.

CUDA Kernels & Device Functions

Kernels

- **GpuTerrainSampleKernel**: Samples DTED tiles stored in a layered texture. It finds the correct tile, snaps to the nearest grid point, and writes elevation; missing tiles map to 0.0.
- **BuildTerrainPrefixKernel**: Computes min_cos_obstruction per radial/step by prefix-scanning terrain obstruction (cosine of nadir angle) along each radial. This is used for terrain masking in the main kernel.
- **RunMraKernel**: The core radar + jamming evaluation kernel. It calculates radar altitude, scan limits, terrain masking, SNR, detection, jamming effectiveness, and status codes. Debug outputs (dbg_cos_scan, dbg_cos_mask) are optionally emitted.

Device helper functions

- **DeviceCosNadirAngle**: Computes the cosine of the nadir angle given earth radius, altitudes, and surface range. Used for scan limits and terrain masking.
- **DeviceHaversineRangeM**: Haversine distance between radar point and jammer using precomputed sin/cos (if provided), used to derive jammer range.
- **DeviceEvaluateEffectivenessPct** and **DeviceJamStatusFromPct**: Evaluate jammer effectiveness from breakpoints and map to status codes.
- **DeviceSafeLinearToDb / DeviceNegInf**: Device-safe conversion that returns negative infinity when inputs are non-positive, used to match CPU semantics.

Float32 vs Float64 behavior & terrain mask epsilon

The GPU backend supports float32 and float64, chosen by scenario.scenario.precision. Float32 introduces precision differences in terrain masking, so the kernel uses a stricter epsilon in the comparison between target cosine and min_cos_obstruction to match CPU double-precision behavior.

COMPUTATIONAL PARITY

Mathematical Formulas (CPU/GPU parity)
The GPU backend uses device-side variants of
shared imom_math functions, providing parity with CPU formulas.
The same equations are called from kernels.

Radar equations

- **Signal power at receiver** (radar equation):
  - radar_signal_w(Pt_w, Gpc_lin, Lr_lin, Gt_lin, Gr_lin, sigma_m2, Rt_m, fr_hz)
- **Receiver noise**: receiver_noise_w(Br_hz, Fn_lin)
- **SNR**: snr_linear(S_w, N_w) and snr_db(S_w, N_w)
- **Max detection range**: max_detection_range_m(...)

These functions are available on device via IMOM_HOST_DEVICE, and the GPU kernel directly invokes radar_signal_w and receiver_noise_w.

Jamming equations

- **Jamming signal power**: jamming_signal_w(Pj_w, BWratio, Lc_lin, Lp_lin, Lr_lin, Gj_lin, Grj_lin, Rj_m, fj_hz)
- **JS ratio**: js_ratio(J_w, S_js_w) and dB variant js_ratio_db
- **Burnthrough range**: burnthrough_range_m(...)
- **Minimum JS**: js_min_from_snrmp(snrmp_lin)
- **MJv**: mjv_from_jsmin(js_min_lin, Lj_lin, Gij_lin)

The GPU kernel uses radar_signal_for_js_w, jamming_signal_w, and mjv_from_jsmin (during upload) to compute JS ratios and margin values.

dB conversions and parity
linear_to_db and db_to_linear provide consistent conversions across CPU/GPU. The GPU-side DeviceSafeLinearToDb wraps linear_to_db to return negative infinity when input is non-positive, mirroring CPU-side safe conversion semantics to avoid NaNs or inconsistent handling.

Processing Flow (End-to-End)
The CPU backend pipeline is executed in RunMraCpu, with RunMraCpuSingle performing the per-altitude scan/detection/jamming loop. The core pipeline is:

1.Geometry generation

- Build mra::geo::RadialsSpec from scenario inputs (center, angular increment, step size, radial length).
- Generate geometry via mra::geo::GenerateGeometry, producing per-point lat/lon grids and step/radial metadata.

2.Jammer preprocessing

- Convert enabled jammer configs to ActiveJammer records.
- Precompute breakpoints, lat/lon radian values, and MJV using mjv_from_jsmin.

3.Terrain sampling and elevation grid construction

- When terrain is enabled, build a GDAL terrain config and sample the terrain grid using TerrainGdal + SampleTerrainGridMsl.
- The sampled elevations become terrain_elevations used for masking and radar altitude.

4.Base PointResults initialization

- BuildBasePointResults creates the output structure, populating radial/step indices, azimuth, distances, and radar MSL altitude (terrain elevation + antenna height).
- PointResults::resize allocates all per-point arrays.

5.Ground range calculation (if center is not PE)

- If the scenario is centered elsewhere, ground ranges are computed either via Haversine (PerfFast) or GeographicLib WGS84 inverse (ParityStrict).

6.Per-altitude evaluation (RunMraCpuSingle)

- Validates constraints (RCs type, terrain reference requirements).
- For every radial/step:
  - Compute scan limits via CosNadirAngle and mark in_scan_limits.
  - Compute terrain mask against min_cos_obstruction.
  - Compute SNR and detection status (kNotDetected or kDetected).
  - If detected and jamming enabled, compute J/S and effectiveness % from the best jammer.
  - Update status_code and optional debug cosine arrays.

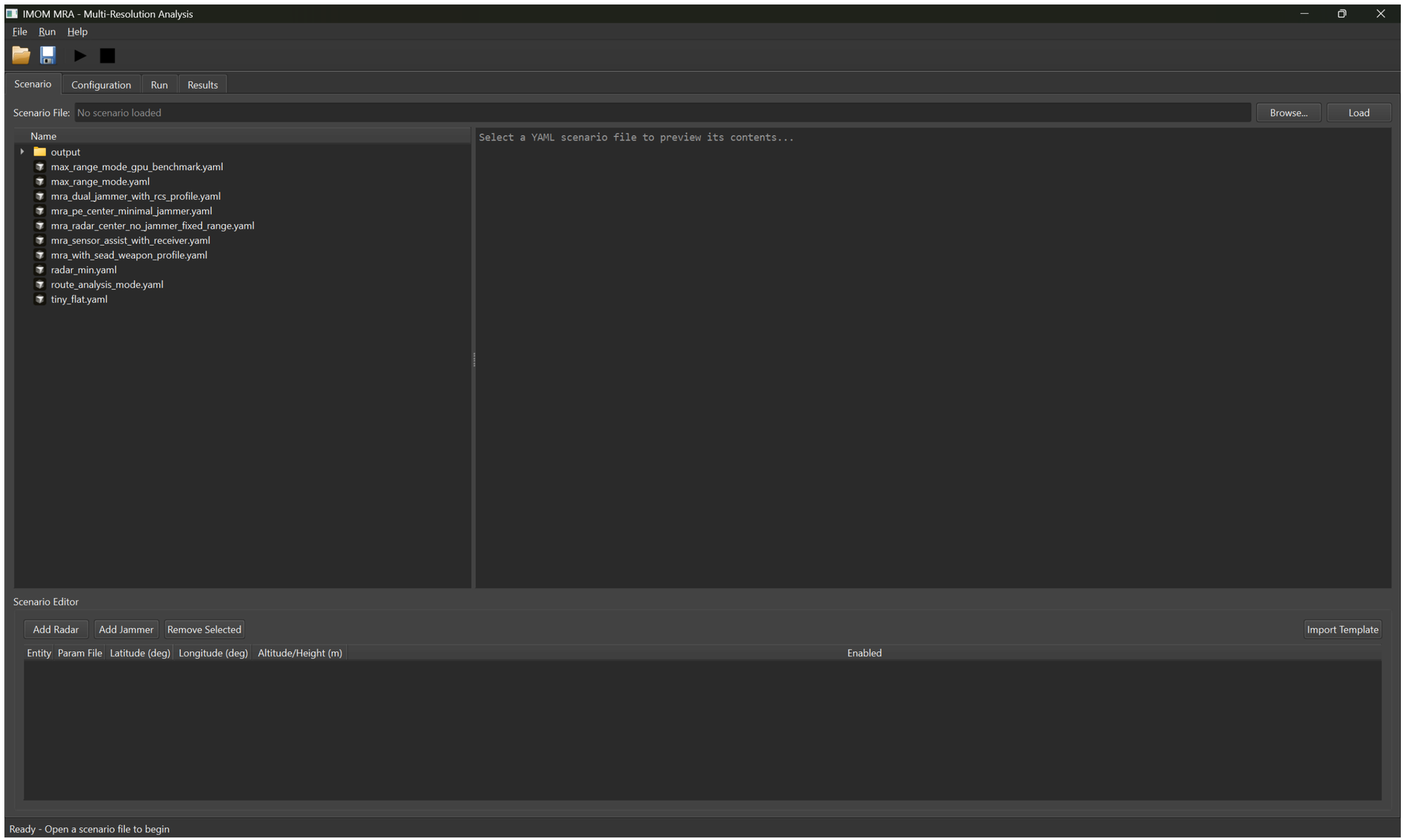
COMMAND LINE & GRAPHICAL USER INTERFACES

Command-Line Usage

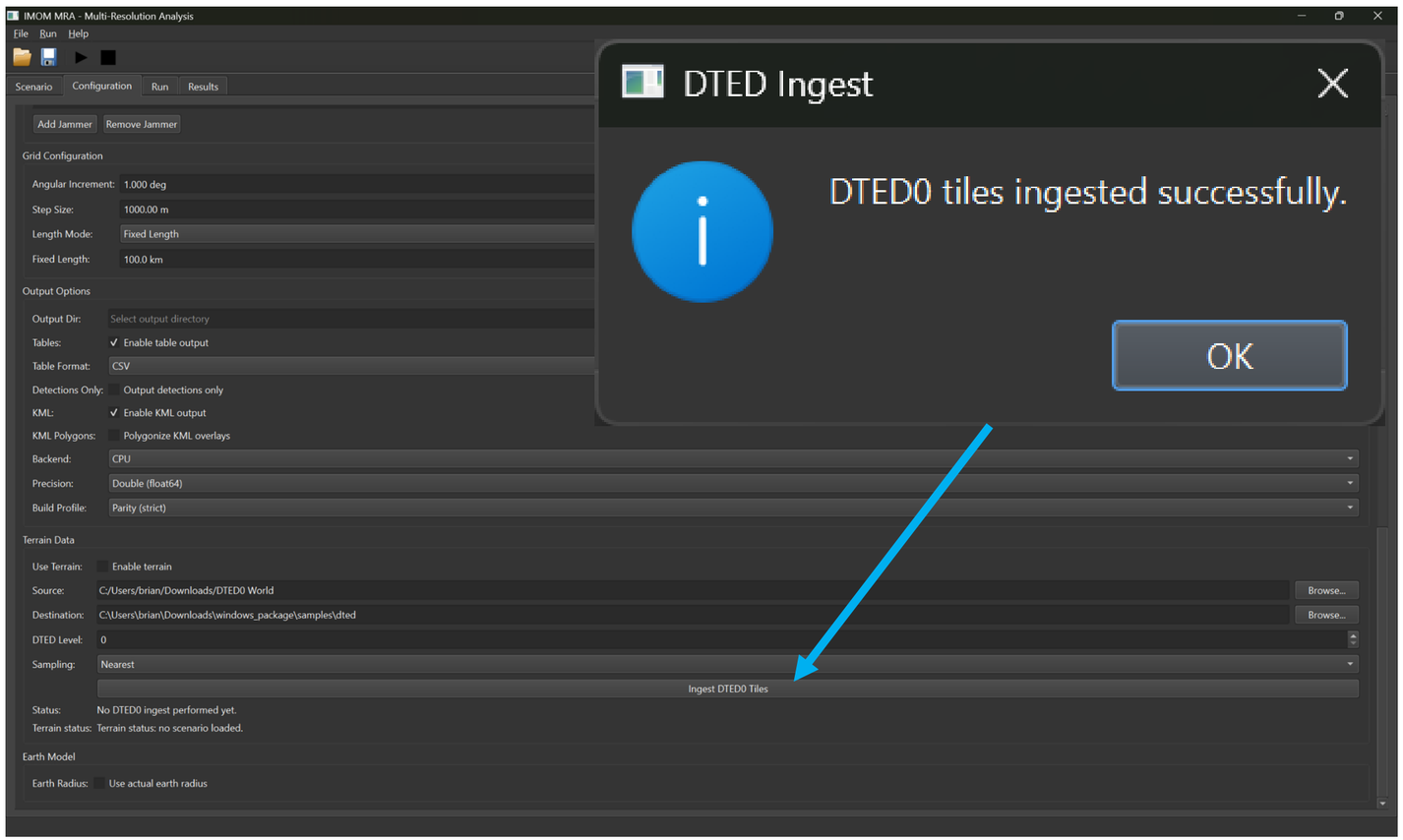
```
imom_mra <command> [options] <scenario.yaml>

Commands:
  run           Run a scenario and emit outputs (CSV, KML)
  verify        Run CPU/GPU parity verification
  benchmark     Run timing benchmarks
  version       Print build info
  help          Print detailed help

Options:
  --out-dir <path>         Override output directory
  --backend <cpu|gpu|both> Select compute backend (default: cpu)
  --precision <float64|float32>
  --tables <on|off>        Enable/disable CSV output
  --kml <on|off>           Enable/disable KML output
  --threads <N>            OpenMP thread count
```

Project Structure

```
windows_package/
├── CMakeLists.txt      # Build configuration
├── vcpkg.json          # Dependency manifest
├── build.ps1           # PowerShell build script
├── build.bat           # Batch file build script
├── README.md           # This file
├── src/                # Source code
│   ├── main.cpp        # CLI entry point
│   ├── cpu_backend.cpp # CPU computation
│   ├── gui/            # Qt GUI source files
│   │   ├── gui_main.cpp
│   │   ├── MainWindow.cpp
│   │   ├── ScenarioPanel.cpp
│   │   ├── ConfigPanel.cpp
│   │   ├── RunPanel.cpp
│   │   └── ResultsPanel.cpp
│   ├── input/          # Scenario/data loading
│   ├── output/         # CSV/KML writers
│   ├── terrain/        # GDAL terrain processing
│   └── geo/            # Geodesy calculations
├── include/            # Header files
│   └── mra/
│       └── gui/        # Qt GUI headers
├── cmake/              # CMake modules
├── config/             # Configuration schemas
└── samples/            # Test data
    ├── scenarios/      # Example YAML scenarios
    └── dted/           # Sample terrain data
```



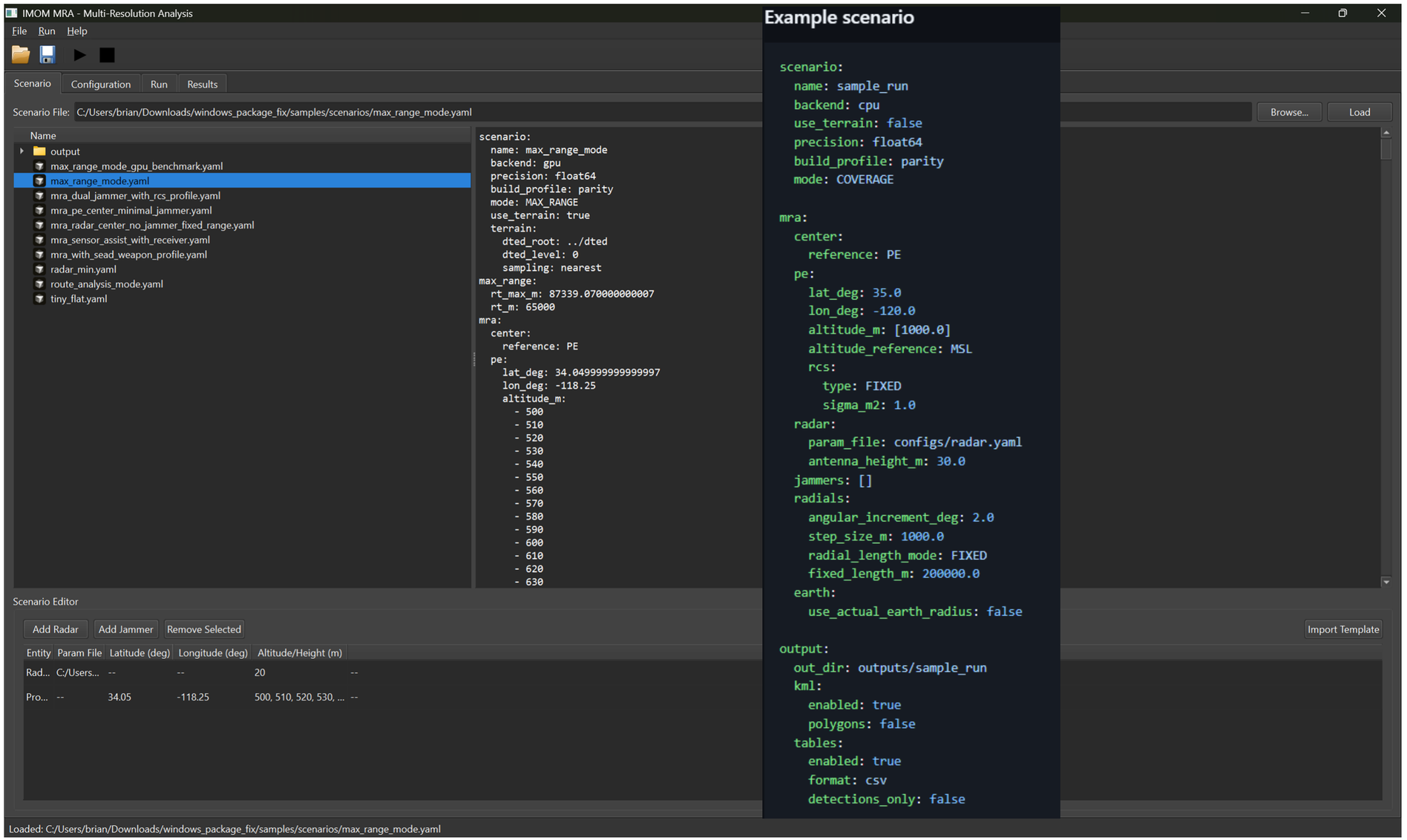
WORKFLOW STEP 1: TERRAIN INGESTION



1. **Select Source:** Directory containing raw DTED tiles.
2. **Define Destination:** Local application storage.
3. **Ingest:** System scans, validates uniqueness (checking lat/lon keys), and indexes files.

Note: DTED Level O supported. Automatic duplicate detection.

WORKFLOW STEP 2: SCENARIO DEFINITION



Example scenario

```yaml
scenario:
  name: sample_run
  backend: cpu
  use_terrain: false
  precision: float64
  build_profile: parity
  mode: COVERAGE

mra:
  center:
    reference: PE
  pe:
    lat_deg: 35.0
    lon_deg: -120.0
    altitude_m: [1000.0]
    altitude_reference: MSL
    rcs:
      type: FIXED
      sigma_m2: 1.0
  radar:
    param_file: configs/radar.yaml
    antenna_height_m: 30.0
  jammers: []
  radials:
    angular_increment_deg: 2.0
    step_size_m: 1000.0
    radial_length_mode: FIXED
    fixed_length_m: 200000.0
  earth:
    use_actual_earth_radius: false

output:
  out_dir: outputs/sample_run
  kml:
    enabled: true
    polygons: false
  tables:
    enabled: true
    format: csv
    detections_only: false
```

Scenario: Templated Analysis Settings

Can define some or all aspects of a preset number of configuration variables for a computational run of the software

WORKFLOW STEP 3: ASSET CONFIGURATION

```yaml
radar:
  tx:
    power_w: 25000
    freq_hz: 2700000000
  rx:
    noise_bandwidth_hz: 1000000
    noise_figure_lin: 2
    rx_loss_lin: 1.2
  gains:
    tx_gain_lin: 1000
    rx_gain_lin: 1000
  processing:
    pulse_compression_gain_lin: 10
    integration_gain_lin: 1
  detection:
    snr_multipulse_min_lin: 10
  scan:
    min_elevation_deg: -30
    max_elevation_deg: 30
```

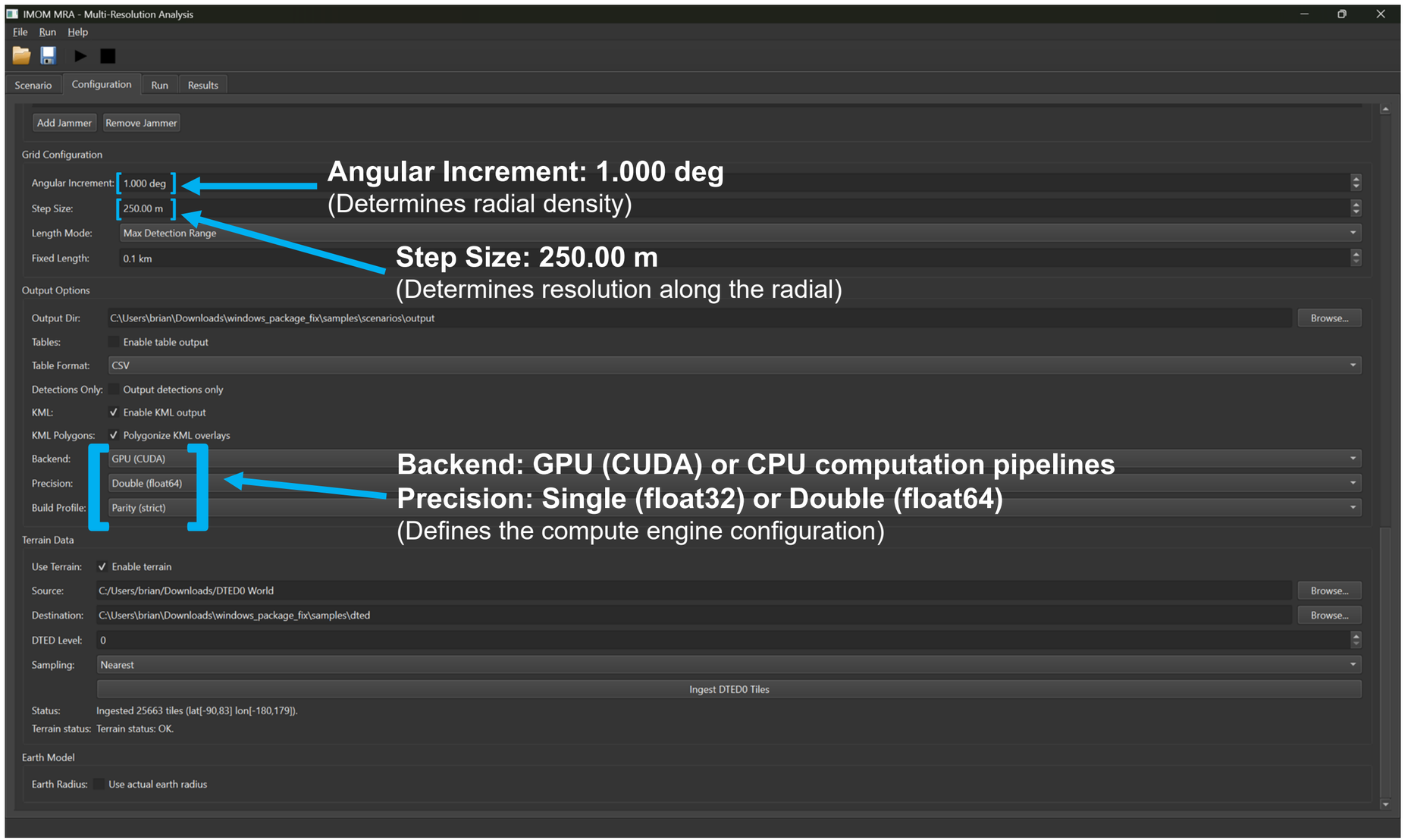
RADAR & Jammer Configuration Files

Contain associated equipment parametric data for use with an analysis scenario (Equipment configuration file specified in Scenario file)

GUI (planned) / CLI & File data (present)

Although the data and function of asset (equipment) configuration files fully exist in the working software to specify the specific variables for analysis runs, there was not time to build out the associated GUI functions for these files (image is of the data of a sample RADAR)

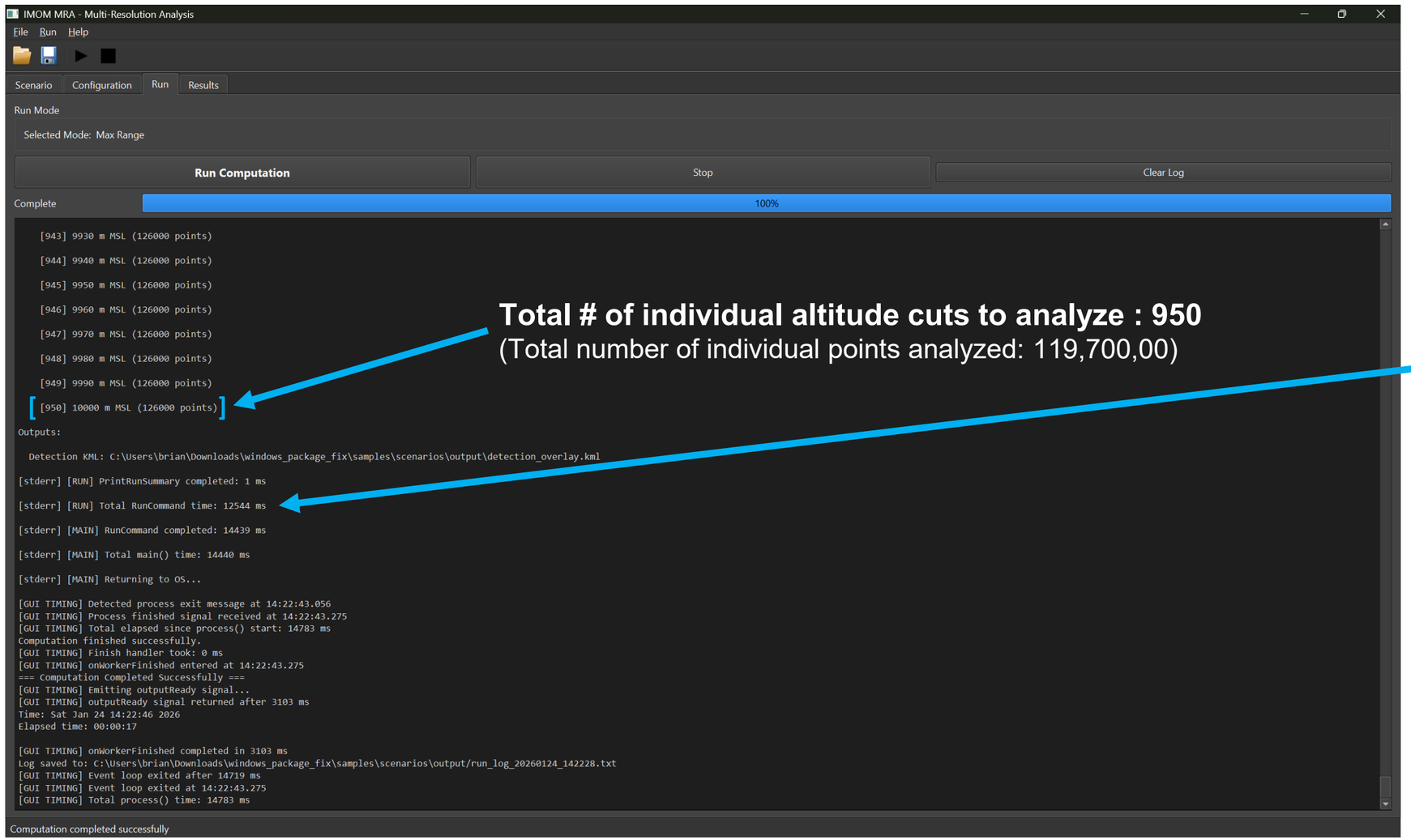
WORKFLOW STEP 4: GRID RESOLUTION & PRECISION



Multi-Resolution Control

Multi-Resolution control allows operators to balance fidelity against compute time. Critical missions utilize GPU acceleration with double-precision floating point math.

EXECUTION & REAL-TIME MONITORING



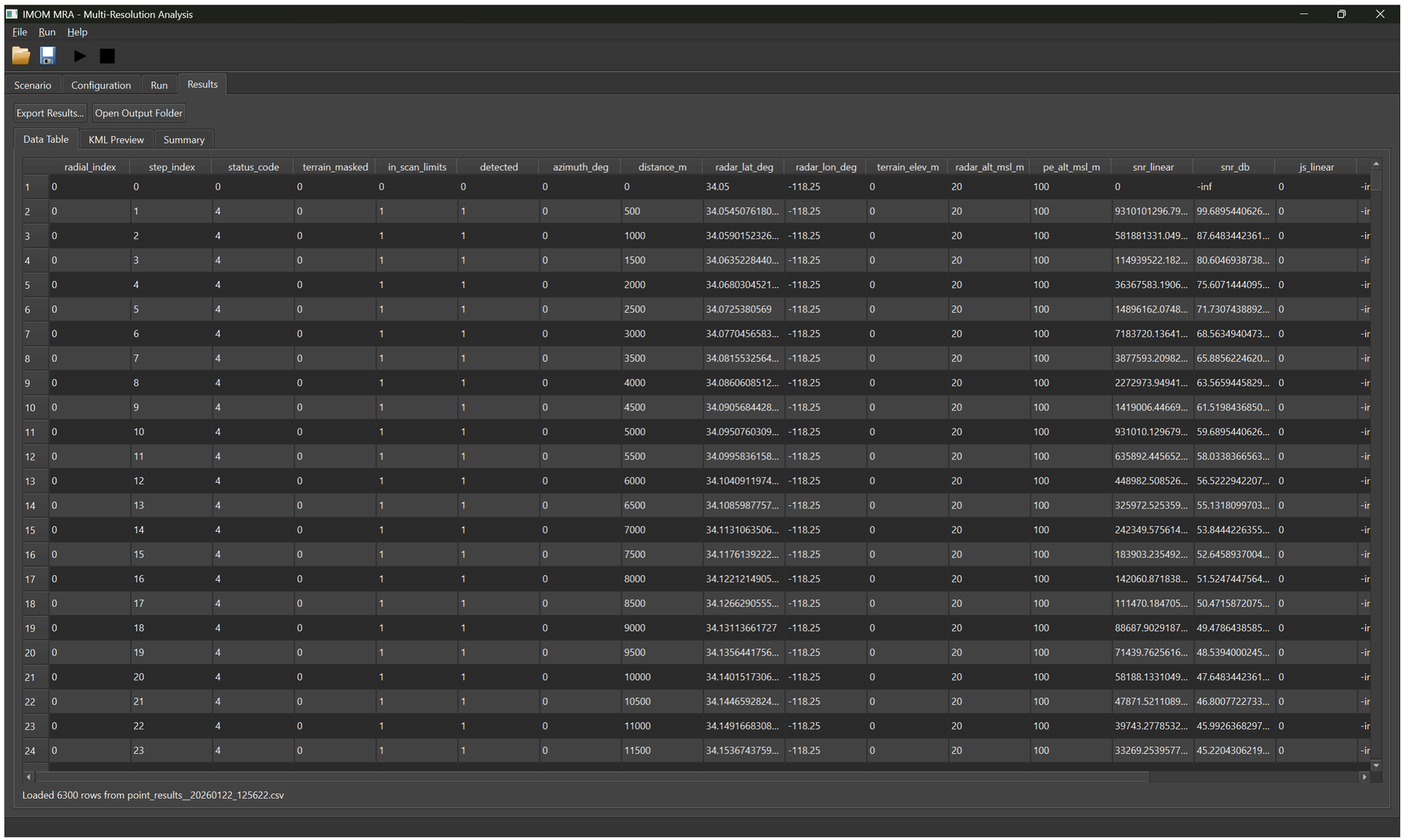
Total # of individual altitude cuts to analyze : 950
(Total number of individual points analyzed: 119,700,00)

Execution Metrics & Artifacts

- Performance Metrics: Total RunCommand time: **12544 ms**
- Artifact Confirmation: **Detection KML** .../output/detection_overlay.kml
- Computation Status: **Finished Successfully**

The integrated console captures standard error (stderr) and timing metrics for performance tuning and verification.

QUANTITATIVE OUTPUT: DATA TABLES



	radial_index	step_index	status_code	terrain_masked	in_scan_limits	detected	azimuth_deg	distance_m	radar_lat_deg	radar_lon_deg	terrain_elev_m	radar_alt_msl_m	pe_alt_msl_m	snr_linear	snr_db	js_linear	
1	0	0	0	0	0	0	0	0	34.05	-118.25	0	20	100	0	-inf	0	-ir
2	0	1	4	0	1	1	0	500	34.0545076180...	-118.25	0	20	100	9310101296.79...	99.6895440626...	0	-ir
3	0	2	4	0	1	1	0	1000	34.0590152326...	-118.25	0	20	100	581881331.049...	87.6483442361...	0	-ir
4	0	3	4	0	1	1	0	1500	34.0635228440...	-118.25	0	20	100	114939522.182...	80.6046938738...	0	-ir
5	0	4	4	0	1	1	0	2000	34.0680304521...	-118.25	0	20	100	36367583.1906...	75.6071444095...	0	-ir
6	0	5	4	0	1	1	0	2500	34.0725380569	-118.25	0	20	100	14896162.0748...	71.7307438892...	0	-ir
7	0	6	4	0	1	1	0	3000	34.0770456583...	-118.25	0	20	100	7183720.13641...	68.5634940473...	0	-ir
8	0	7	4	0	1	1	0	3500	34.0815532564...	-118.25	0	20	100	3877593.20982...	65.8856224620...	0	-ir
9	0	8	4	0	1	1	0	4000	34.0860608512...	-118.25	0	20	100	2272973.94941...	63.5659445829...	0	-ir
10	0	9	4	0	1	1	0	4500	34.0905684428...	-118.25	0	20	100	1419006.44669...	61.5198436850...	0	-ir
11	0	10	4	0	1	1	0	5000	34.0950760309...	-118.25	0	20	100	931010.129679...	59.6895440626...	0	-ir
12	0	11	4	0	1	1	0	5500	34.0995836158...	-118.25	0	20	100	635892.445652...	58.0338366563...	0	-ir
13	0	12	4	0	1	1	0	6000	34.1040911974...	-118.25	0	20	100	448982.508526...	56.5222942207...	0	-ir
14	0	13	4	0	1	1	0	6500	34.1085987757...	-118.25	0	20	100	325972.525359...	55.1318099703...	0	-ir
15	0	14	4	0	1	1	0	7000	34.1131063506...	-118.25	0	20	100	242349.575614...	53.8444226355...	0	-ir
16	0	15	4	0	1	1	0	7500	34.1176139222...	-118.25	0	20	100	183903.235492...	52.6458937004...	0	-ir
17	0	16	4	0	1	1	0	8000	34.1221214905...	-118.25	0	20	100	142060.871838...	51.5247447564...	0	-ir
18	0	17	4	0	1	1	0	8500	34.1266290555...	-118.25	0	20	100	111470.184705...	50.4715872075...	0	-ir
19	0	18	4	0	1	1	0	9000	34.13113661727	-118.25	0	20	100	88687.9029187...	49.4786438585...	0	-ir
20	0	19	4	0	1	1	0	9500	34.1356441756...	-118.25	0	20	100	71439.7625616...	48.5394000245...	0	-ir
21	0	20	4	0	1	1	0	10000	34.1401517306...	-118.25	0	20	100	58188.1331049...	47.6483442361...	0	-ir
22	0	21	4	0	1	1	0	10500	34.1446592824...	-118.25	0	20	100	47871.5211089...	46.8007722733...	0	-ir
23	0	22	4	0	1	1	0	11000	34.1491668308...	-118.25	0	20	100	39743.2778532...	45.9926368297...	0	-ir
24	0	23	4	0	1	1	0	11500	34.1536743759...	-118.25	0	20	100	33269.2539577...	45.2204306219...	0	-ir

terrain_masked: Boolean (0/1) indicating line-of-sight blockage.

snr _db: Signal-to-Noise Ratio (decibels).

detected: Final binary status based on Radar/RCS calculation.

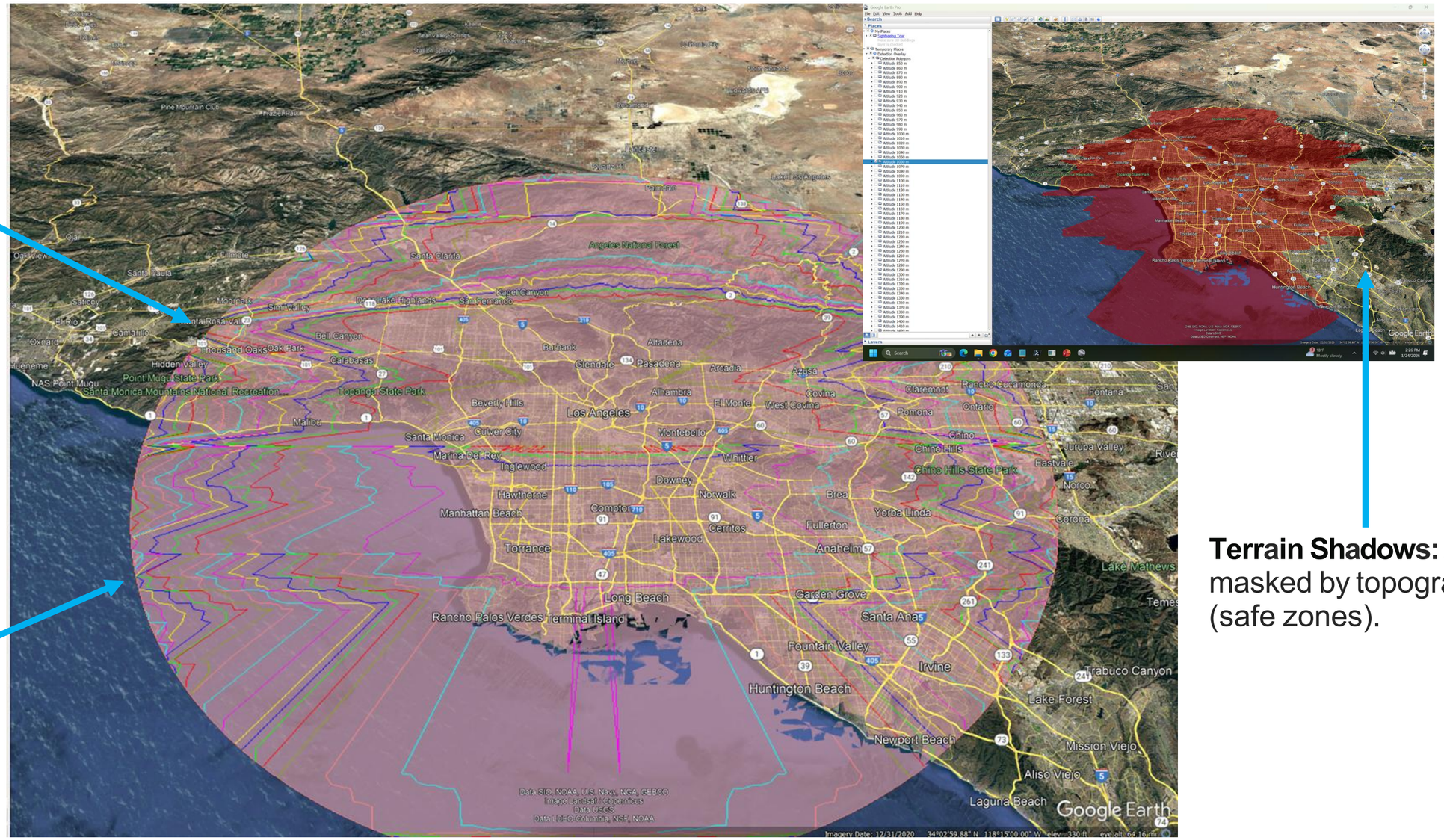
azimuth_deg / distance_m: Polar coordinates of the evaluation point.

Export Formats: CSV (Parquet & JSON possible).

GEOSPATIAL VISUALIZATION

Detection Rings: Visualize vulnerability zones.

KML Output: Native OGG-compliant files generated automatically. (JetBrains Mono)



Terrain Shadows: Areas masked by topography (safe zones).

LESSONS LEARNED

1. Reference documentation is invaluable, and the more the better!
2. You can’t spend too much time on design…and plan EVERY step of development.
3. You need the best frontier model quality for design and planning, but “2nd tier” models are more than adequate for coding tasks.
4. AI coding tools, like OpenAI Codex or Claude Code, are the best at complex coding of critical modules…especially the math.
5. AI Coding Platforms, like Replit or Loveable, are critical for the automation of module and functional capability testing.
6. Some issues are unavoidable, and require a manual approach, such as standalone workstation build and testing for local clients.

RECOMMENDATIONS FOR FUTURE

1. For near term opportunities, maximize the public release of system documentation.
2. Focus on the implementation of coding tools and platforms into GenAI.mil.
3. Should consider new ways to contract development, not just use new tools.
4. GOTS software presents a major opportunity for development process overhaul.
5. Research Labs & Service Academies valuable partner in new development process.
6. Faster/Cheaper development opens up new opportunities for lower priority projects.
7. All development must be accredited for operationalization, need plan for more/faster.

IMOM MRA DESIGN & DEVELOPMENT TEST

[✓] **Design: Math & functional descriptions in technical papers.**

[✓] **Build: Working prototype build within 3 days.**

[✓] **Operate: Windows client with CPU & GPU pipelines for ultra-fast processing.**

[✓] **Perform: Local Windows PC application w/ Intuitive GUI & CLI options.**

***Code Repo available upon request (includes full software documentation)

RESULT: SUCCESS

Email: brian.morrison01123@outlook.com

BACKUP SLIDES

IMOM MRA - Multi-Resolution Analysis

File Run Help

Scenario Configuration Run Results

Scenario File: No scenario loaded Browse... Load

Name

- output
- max_range_mode_gpu_benchmark.yaml
- max_range_mode.yaml
- mra_dual_jammer_with_rcs_profile.yaml
- mra_pe_center_minimal_jammer.yaml
- mra_radar_center_no_jammer_fixed_range.yaml
- mra_sensor_assist_with_receiver.yaml
- mra_with_sead_weapon_profile.yaml
- radar_min.yaml
- route_analysis_mode.yaml
- tiny_flat.yaml

Select a YAML scenario file to preview its contents...

Scenario Editor

Add Radar Add Jammer Remove Selected Import Template

Entity	Param File	Latitude (deg)	Longitude (deg)	Altitude/Height (m)	Enabled

Ready - Open a scenario file to begin

IMOM MRA - Multi-Resolution Analysis

File Run Help

Scenario | Configuration | Run | Results

Scenario Settings

Name:

Scenario Mode

Mode: Coverage

Center Reference

Reference: Protected Entity

Latitude: 0.000000 deg

Longitude: 0.000000 deg

Altitude (MSL): 0.0 m

Target/PE Configuration

Latitude: 0.000000 deg

Longitude: 0.000000 deg

Altitude (MSL): MSL

Altitude (m)
0.0 m

Add Altitude | Remove Altitude

Multiple altitudes create multiple rings in MAX_RANGE mode.

RCS Type: Fixed

RCS Sigma: 1.000 m^2

Radar Configuration

Param File: Select radar parameter file | Browse...

Antenna Height: 0.0 m

IMOM MRA - Multi-Resolution Analysis

File Run Help

Scenario | Configuration | Run | Results

Radar Configuration

Param File: Select radar parameter file — Browse...

Antenna Height: 0.0 m

Preset	Frequency (GHz)	Power (W)

Add Radar | Remove Radar

Jammer Configuration

Preset	Latitude (deg)	Longitude (deg)	Altitude (m)	Alt Ref	Power (W)	Frequency (GHz)	Enabled

Add Jammer | Remove Jammer

Grid Configuration

Angular Increment: 1.000 deg

Step Size: 1000.00 m

Length Mode: Fixed Length

Fixed Length: 100.0 km

Output Options

Output Dir: Select output directory — Browse...

Tables: ✓ Enable table output

IMOM MRA - Multi-Resolution Analysis

File Run Help

Scenario | Configuration | Run | Results

Add Jammer | Remove Jammer

Grid Configuration

Angular Increment: 1.000 deg

Step Size: 1000.00 m

Length Mode: Fixed Length

Fixed Length: 100.0 km

Output Options

Output Dir: Select output directory — Browse...

Tables: ✓ Enable table output

Table Format: CSV

Detections Only: Output detections only

KML: ✓ Enable KML output

KML Polygons: Polygonize KML overlays

Backend: CPU

Precision: Double (float64)

Build Profile: Parity (strict)

Terrain Data

Use Terrain: Enable terrain

Source: Select DTED0 source directory — Browse...

Destination: C:\Users\brian\Downloads\windows_package\samples\dted — Browse...

DTED Level: 0

Sampling: Nearest

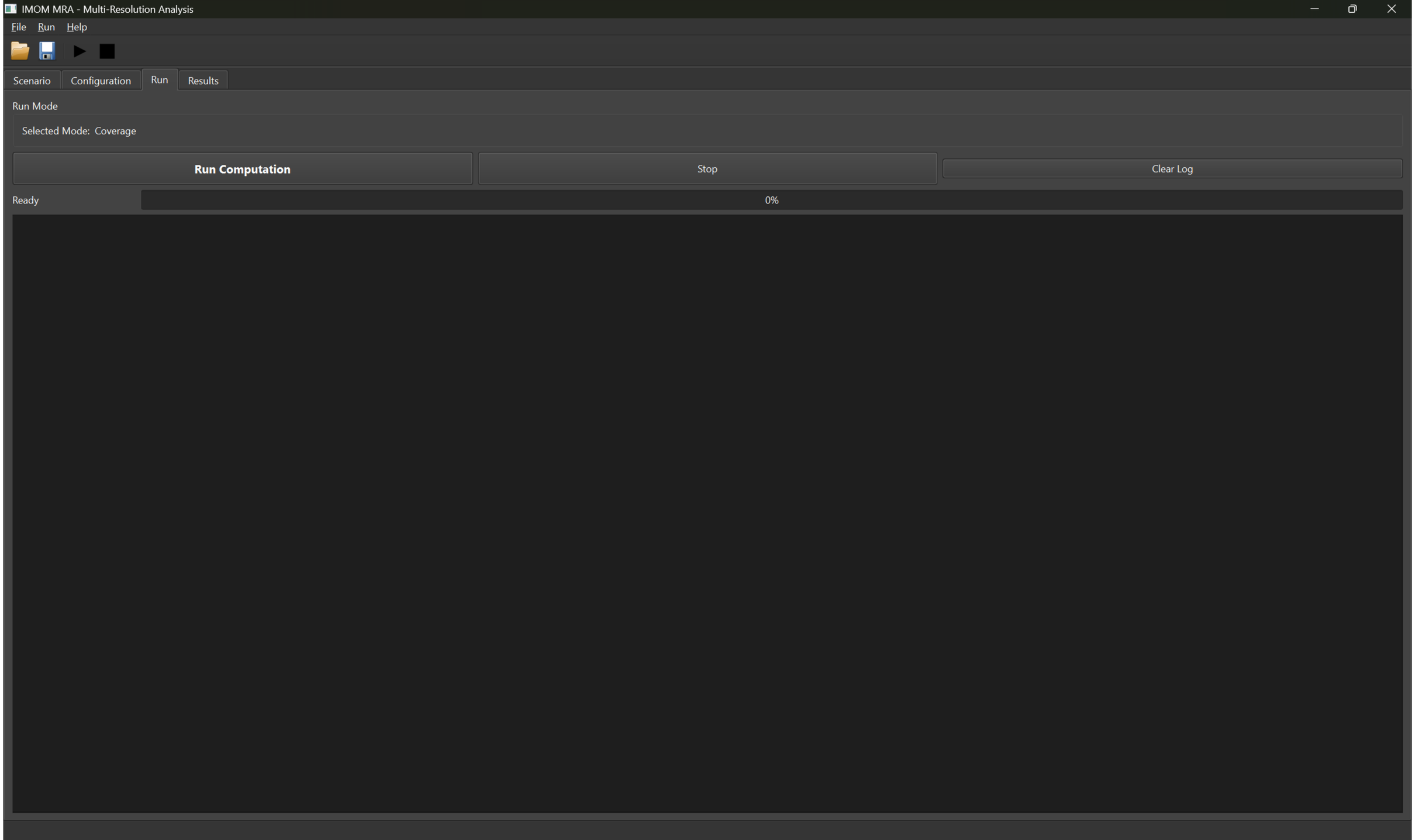
Ingest DTED0 Tiles

Status: No DTED0 ingest performed yet.

Terrain status: Terrain status: no scenario loaded.

Earth Model

Earth Radius: Use actual earth radius


IMOM MRA - Multi-Resolution Analysis
File
Run
Help
Scenario
Configuration
Run
Results
Run Mode
Selected Mode: Coverage
Run Computation
Stop
Clear Log
Ready
0%

Operating the Program

Load Terrain Data >> Load Scenario File >> Configure Settings >> Run Analysis

IMOM MRA - Multi-Resolution Analysis

File Run Help

Scenario | Configuration | Run | Results

Add Jammer | Remove Jammer

Grid Configuration

Angular Increment: 1.000 deg

Step Size: 1000.00 m

Length Mode: Fixed Length

Fixed Length: 100.0 km

Output Options

Output Dir: Select output directory — Browse...

Tables: ✓ Enable table output

Table Format: CSV

Detections Only: Output detections only

KML: ✓ Enable KML output

KML Polygons: Polygonize KML overlays

Backend: CPU

Precision: Double (float64)

Build Profile: Parity (strict)

Terrain Data

Use Terrain: Enable terrain

Source: C:/Users/brian/Downloads/DTED0 World — Browse...

Destination: C:\Users\brian\Downloads\windows_package\samples\dted — Browse...

DTED Level: 0

Sampling: Nearest

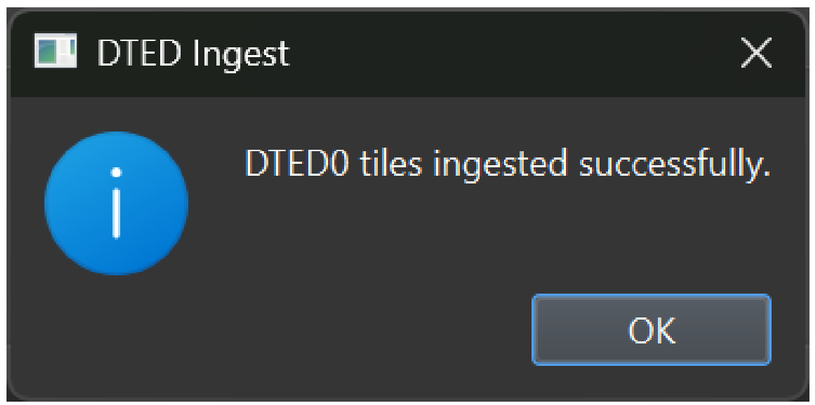
Ingest DTED0 Tiles

Status: No DTED0 ingest performed yet.

Terrain status: Terrain status: no scenario loaded.

Earth Model

Earth Radius: Use actual earth radius


DTED Ingest
DTED0 tiles ingested successfully.
OK

IMOM MRA - Multi-Resolution Analysis

File Run Help

Scenario | Configuration | Run | Results

Add Jammer | Remove Jammer

Grid Configuration

Angular Increment: 1.000 deg

Step Size: 1000.00 m

Length Mode: Fixed Length

Fixed Length: 100.0 km

Output Options

Output Dir: Select output directory — Browse...

Tables: ✓ Enable table output

Table Format: CSV

Detections Only: Output detections only

KML: ✓ Enable KML output

KML Polygons: Polygonize KML overlays

Backend: CPU

Precision: Double (float64)

Build Profile: Parity (strict)

Terrain Data

Use Terrain: Enable terrain

Source: C:/Users/brian/Downloads/DTED0 World — Browse...

Destination: C:\Users\brian\Downloads\windows_package\samples\dted — Browse...

DTED Level: 0

Sampling: Nearest

Ingest DTED0 Tiles

Status: Ingested 25663 tiles (lat[-90,83] lon[-180,179]).

Terrain status: Terrain status: no scenario loaded.

Earth Model

Earth Radius: Use actual earth radius

DTED root updated (no scenario file loaded).

IMOM MRA - Multi-Resolution Analysis

File Run Help

Scenario Configuration Run Results

Scenario File: C:/Users/brian/Downloads/windows_package_fix/samples/scenarios/max_range_mode.yaml Browse... Load

Name

- output
- max_range_mode_gpu_benchmark.yaml
- max_range_mode.yaml
- mra_dual_jammer_with_rcs_profile.yaml
- mra_pe_center_minimal_jammer.yaml
- mra_radar_center_no_jammer_fixed_range.yaml
- mra_sensor_assist_with_receiver.yaml
- mra_with_sead_weapon_profile.yaml
- radar_min.yaml
- route_analysis_mode.yaml
- tiny_flat.yaml

```
scenario:
  name: max_range_mode
  backend: gpu
  precision: float64
  build_profile: parity
  mode: MAX_RANGE
  use_terrain: true
  terrain:
    dted_root: ../dted
    dted_level: 0
    sampling: nearest
max_range:
  rt_max_m: 87339.070000000007
  rt_m: 65000
mra:
  center:
    reference: PE
  pe:
    lat_deg: 34.049999999999997
    lon_deg: -118.25
    altitude_m:
      - 500
      - 510
      - 520
      - 530
      - 540
      - 550
      - 560
      - 570
      - 580
      - 590
      - 600
      - 610
      - 620
      - 630
```

Scenario Editor

Add Radar Add Jammer Remove Selected Import Template

Entity	Param File	Latitude (deg)	Longitude (deg)	Altitude/Height (m)	Enabled

IMOM MRA - Multi-Resolution Analysis

File Run Help

Scenario | Configuration | Run | Results

Scenario File: C:/Users/brian/Downloads/windows_package_fix/samples/scenarios/max_range_mode.yaml Browse... Load

Name

- output
- max_range_mode_gpu_benchmark.yaml
- max_range_mode.yaml
- mra_dual_jammer_with_rcs_profile.yaml
- mra_pe_center_minimal_jammer.yaml
- mra_radar_center_no_jammer_fixed_range.yaml
- mra_sensor_assist_with_receiver.yaml
- mra_with_sead_weapon_profile.yaml
- radar_min.yaml
- route_analysis_mode.yaml
- tiny_flat.yaml

```
scenario:
  name: max_range_mode
  backend: gpu
  precision: float64
  build_profile: parity
  mode: MAX_RANGE
  use_terrain: true
  terrain:
    dted_root: ../dted
    dted_level: 0
    sampling: nearest
max_range:
  rt_max_m: 87339.070000000007
  rt_m: 65000
mra:
  center:
    reference: PE
  pe:
    lat_deg: 34.049999999999997
    lon_deg: -118.25
    altitude_m:
      - 500
      - 510
      - 520
      - 530
      - 540
      - 550
      - 560
      - 570
      - 580
      - 590
      - 600
      - 610
      - 620
      - 630
```

Scenario Editor

Add Radar | Add Jammer | Remove Selected | Import Template

Entity	Param File	Latitude (deg)	Longitude (deg)	Altitude/Height (m)	Enabled
Rad...	C:/Users...	--	--	20	--
Pro...	--	34.05	-118.25	500, 510, 520, 530, ...	--

Loaded: C:/Users/brian/Downloads/windows_package_fix/samples/scenarios/max_range_mode.yaml

IMOM MRA - Multi-Resolution Analysis

File Run Help

Scenario | Configuration | Run | Results

Scenario Settings

Name: max_range_mode

Scenario Mode

Mode: Max Range

Center Reference

Reference: Protected Entity

Latitude: 34.050000 deg

Longitude: -118.250000 deg

Altitude (MSL): 500.0 m

Target/PE Configuration

Latitude: 34.050000 deg

Longitude: -118.250000 deg

Altitude (MSL): MSL

Altitude (m)
500.0 m
510.0 m
520.0 m
530.0 m
540.0 m
550.0 m

Add Altitude | Remove Altitude

Multiple altitudes create multiple rings in MAX_RANGE mode.

RCS Type: Fixed

RCS Sigma: 1.000 m²

Radar Configuration

Param File: C:\Users\brian\Downloads\windows_package_fix\samples\params\max_range_radar.yaml Browse...

Antenna Height: 20.0 m

IMOM MRA - Multi-Resolution Analysis

File Run Help

Scenario | Configuration | Run | Results

Radar Configuration

Param File: C:\Users\brian\Downloads\windows_package_fix\samples\params\max_range_radar.yaml [Browse...]

Antenna Height: 20.0 m

Preset	Frequency (GHz)	Power (W)
ge_radar.yaml	2.700 GHz	25000.0 W

[Add Radar] [Remove Radar]

Jammer Configuration

Preset	Latitude (deg)	Longitude (deg)	Altitude (m)	Alt Ref	Power (W)	Frequency (GHz)	Enabled

[Add Jammer] [Remove Jammer]

Grid Configuration

Angular Increment: 1.000 deg

Step Size: 250.00 m

Length Mode: Max Detection Range

Fixed Length: 0.1 km

Output Options

Output Dir: C:\Users\brian\Downloads\windows_package_fix\samples\scenarios\output [Browse...]

Tables: Enable table output

IMOM MRA - Multi-Resolution Analysis

File Run Help

Scenario | Configuration | Run | Results

Add Jammer | Remove Jammer

Grid Configuration

Angular Increment: 1.000 deg

Step Size: 250.00 m

Length Mode: Max Detection Range

Fixed Length: 0.1 km

Output Options

Output Dir: C:\Users\brian\Downloads\windows_package_fix\samples\scenarios\output | Browse...

Tables: ☐ Enable table output

Table Format: CSV

Detections Only: ☐ Output detections only

KML: ☑ Enable KML output

KML Polygons: ☑ Polygonize KML overlays

Backend: GPU (CUDA)

Precision: Double (float64)

Build Profile: Parity (strict)

Terrain Data

Use Terrain: ☑ Enable terrain

Source: C:/Users/brian/Downloads/DTED0 World | Browse...

Destination: C:\Users\brian\Downloads\windows_package_fix\samples\dted | Browse...

DTED Level: 0

Sampling: Nearest

Ingest DTED0 Tiles

Status: Ingested 25663 tiles (lat[-90,83] lon[-180,179]).

Terrain status: Terrain status: OK.

Earth Model

Earth Radius: ☐ Use actual earth radius

IMOM MRA - Multi-Resolution Analysis

File Run Help

Scenario | Configuration | Run | Results

Run Mode

Selected Mode: Max Range

Run Computation | Stop | Clear Log

Complete 100%

```
    [943] 9930 m MSL (126000 points)

    [944] 9940 m MSL (126000 points)

    [945] 9950 m MSL (126000 points)

    [946] 9960 m MSL (126000 points)

    [947] 9970 m MSL (126000 points)

    [948] 9980 m MSL (126000 points)

    [949] 9990 m MSL (126000 points)

    [950] 10000 m MSL (126000 points)

Outputs:

  Detection KML: C:\Users\brian\Downloads\windows_package_fix\samples\scenarios\output\detection_overlay.kml

[stderr] [RUN] PrintRunSummary completed: 1 ms

[stderr] [RUN] Total RunCommand time: 12544 ms

[stderr] [MAIN] RunCommand completed: 14439 ms

[stderr] [MAIN] Total main() time: 14440 ms

[stderr] [MAIN] Returning to OS...

[GUI TIMING] Detected process exit message at 14:22:43.056
[GUI TIMING] Process finished signal received at 14:22:43.275
[GUI TIMING] Total elapsed since process() start: 14783 ms
Computation finished successfully.
[GUI TIMING] Finish handler took: 0 ms
[GUI TIMING] onWorkerFinished entered at 14:22:43.275
=== Computation Completed Successfully ===
[GUI TIMING] Emitting outputReady signal...
[GUI TIMING] outputReady signal returned after 3103 ms
Time: Sat Jan 24 14:22:46 2026
Elapsed time: 00:00:17

[GUI TIMING] onWorkerFinished completed in 3103 ms
Log saved to: C:\Users\brian\Downloads\windows_package_fix\samples\scenarios\output/run_log_20260124_142228.txt
[GUI TIMING] Event loop exited after 14719 ms
[GUI TIMING] Event loop exited at 14:22:43.275
[GUI TIMING] Total process() time: 14783 ms
```

Computation completed successfully

IMOM MRA - Multi-Resolution Analysis

File Run Help

Scenario | Configuration | Run | Results

Export Results... | Open Output Folder

Data Table | KML Preview | Summary

	radial_index	step_index	status_code	terrain_masked	in_scan_limits	detected	azimuth_deg	distance_m	radar_lat_deg	radar_lon_deg	terrain_elev_m	radar_alt_msl_m	pe_alt_msl_m	snr_linear	snr_db	js_linear	
1	0	0	0	0	0	0	0	0	34.05	-118.25	0	20	100	0	-inf	0	-ir
2	0	1	4	0	1	1	0	500	34.0545076180...	-118.25	0	20	100	9310101296.79...	99.6895440626...	0	-ir
3	0	2	4	0	1	1	0	1000	34.0590152326...	-118.25	0	20	100	581881331.049...	87.6483442361...	0	-ir
4	0	3	4	0	1	1	0	1500	34.0635228440...	-118.25	0	20	100	114939522.182...	80.6046938738...	0	-ir
5	0	4	4	0	1	1	0	2000	34.0680304521...	-118.25	0	20	100	36367583.1906...	75.6071444095...	0	-ir
6	0	5	4	0	1	1	0	2500	34.0725380569	-118.25	0	20	100	14896162.0748...	71.7307438892...	0	-ir
7	0	6	4	0	1	1	0	3000	34.0770456583...	-118.25	0	20	100	7183720.13641...	68.5634940473...	0	-ir
8	0	7	4	0	1	1	0	3500	34.0815532564...	-118.25	0	20	100	3877593.20982...	65.8856224620...	0	-ir
9	0	8	4	0	1	1	0	4000	34.0860608512...	-118.25	0	20	100	2272973.94941...	63.5659445829...	0	-ir
10	0	9	4	0	1	1	0	4500	34.0905684428...	-118.25	0	20	100	1419006.44669...	61.5198436850...	0	-ir
11	0	10	4	0	1	1	0	5000	34.0950760309...	-118.25	0	20	100	931010.129679...	59.6895440626...	0	-ir
12	0	11	4	0	1	1	0	5500	34.0995836158...	-118.25	0	20	100	635892.445652...	58.0338366563...	0	-ir
13	0	12	4	0	1	1	0	6000	34.1040911974...	-118.25	0	20	100	448982.508526...	56.5222942207...	0	-ir
14	0	13	4	0	1	1	0	6500	34.1085987757...	-118.25	0	20	100	325972.525359...	55.1318099703...	0	-ir
15	0	14	4	0	1	1	0	7000	34.1131063506...	-118.25	0	20	100	242349.575614...	53.8444226355...	0	-ir
16	0	15	4	0	1	1	0	7500	34.1176139222...	-118.25	0	20	100	183903.235492...	52.6458937004...	0	-ir
17	0	16	4	0	1	1	0	8000	34.1221214905...	-118.25	0	20	100	142060.871838...	51.5247447564...	0	-ir
18	0	17	4	0	1	1	0	8500	34.1266290555...	-118.25	0	20	100	111470.184705...	50.4715872075...	0	-ir
19	0	18	4	0	1	1	0	9000	34.13113661727	-118.25	0	20	100	88687.9029187...	49.4786438585...	0	-ir
20	0	19	4	0	1	1	0	9500	34.1356441756...	-118.25	0	20	100	71439.7625616...	48.5394000245...	0	-ir
21	0	20	4	0	1	1	0	10000	34.1401517306...	-118.25	0	20	100	58188.1331049...	47.6483442361...	0	-ir
22	0	21	4	0	1	1	0	10500	34.1446592824...	-118.25	0	20	100	47871.5211089...	46.8007722733...	0	-ir
23	0	22	4	0	1	1	0	11000	34.1491668308...	-118.25	0	20	100	39743.2778532...	45.9926368297...	0	-ir
24	0	23	4	0	1	1	0	11500	34.1536743759...	-118.25	0	20	100	33269.2539577...	45.2204306219...	0	-ir

Loaded 6300 rows from point_results__20260122_125622.csv

IMOM MRA - Multi-Resolution Analysis

File Run Help

Scenario | Configuration | Run | Results

Export Results... | Open Output Folder

Data Table | KML Preview | Summary

```
<!-- detection_overlay.kml -->
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://www.opengis.net/kml/2.2">
  <Document>
    <name>Detection Overlay</name>
    <Style id="debug">
      <LineStyle><color>ff00ffff</color><width>1</width></LineStyle>
    </Style>
    <Style id="detected-alt-0">
      <IconStyle><color>ff0000ff</color></IconStyle>
      <LineStyle><color>ff0000ff</color><width>1</width></LineStyle>
      <PolyStyle><color>7f0000ff</color></PolyStyle>
    </Style>
    <Style id="detected-alt-1">
      <IconStyle><color>ff00ff00</color></IconStyle>
      <LineStyle><color>ff00ff00</color><width>1</width></LineStyle>
      <PolyStyle><color>7f00ff00</color></PolyStyle>
    </Style>
    <Style id="detected-alt-2">
      <IconStyle><color>ffff0000</color></IconStyle>
      <LineStyle><color>ffff0000</color><width>1</width></LineStyle>
      <PolyStyle><color>7fff0000</color></PolyStyle>
    </Style>
    <Style id="detected-alt-3">
      <IconStyle><color>ff00ffff</color></IconStyle>
      <LineStyle><color>ff00ffff</color><width>1</width></LineStyle>
      <PolyStyle><color>7f00ffff</color></PolyStyle>
    </Style>
    <Style id="detected-alt-4">
      <IconStyle><color>ffff00ff</color></IconStyle>
      <LineStyle><color>ffff00ff</color><width>1</width></LineStyle>
      <PolyStyle><color>7fff00ff</color></PolyStyle>
    </Style>
    <Style id="detected-alt-5">
      <IconStyle><color>ffffff00</color></IconStyle>
      <LineStyle><color>ffffff00</color><width>1</width></LineStyle>
      <PolyStyle><color>7fffff00</color></PolyStyle>
    </Style>
    <Style id="detected-alt-6">
      <IconStyle><color>ff7f7fff</color></IconStyle>
      <LineStyle><color>ff7f7fff</color><width>1</width></LineStyle>
      <PolyStyle><color>7f7f7fff</color></PolyStyle>
    </Style>
    <Style id="detected-alt-7">
      <IconStyle><color>ff00aa7f</color></IconStyle>
      <LineStyle><color>ff00aa7f</color><width>1</width></LineStyle>
      <PolyStyle><color>7f00aa7f</color></PolyStyle>
    </Style>
```

Loaded: detection_overlay.kml

IMOM MRA - Multi-Resolution Analysis

File Run Help

Scenario | Configuration | Run | Results

Export Results... | Open Output Folder

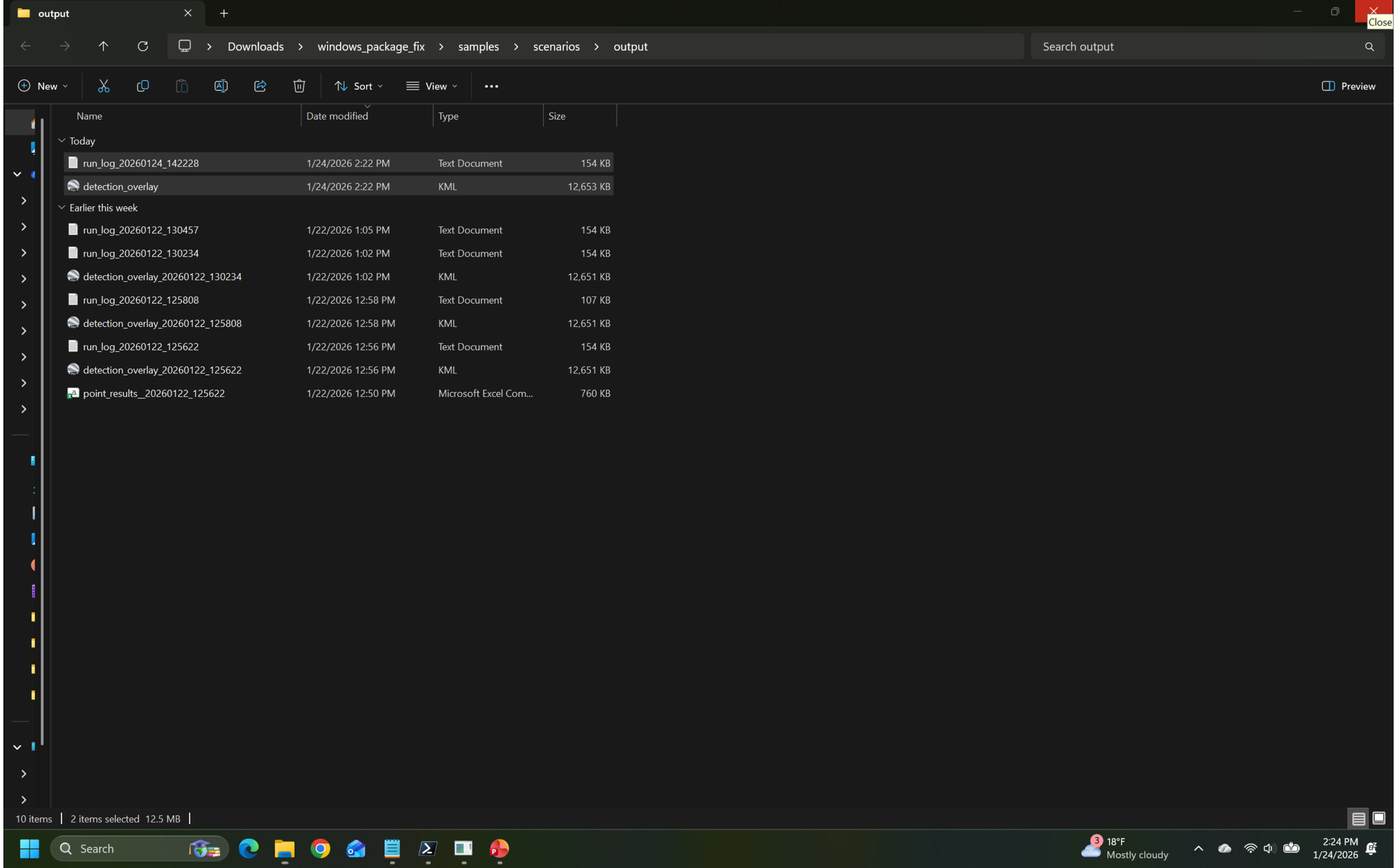
Data Table | KML Preview | Summary

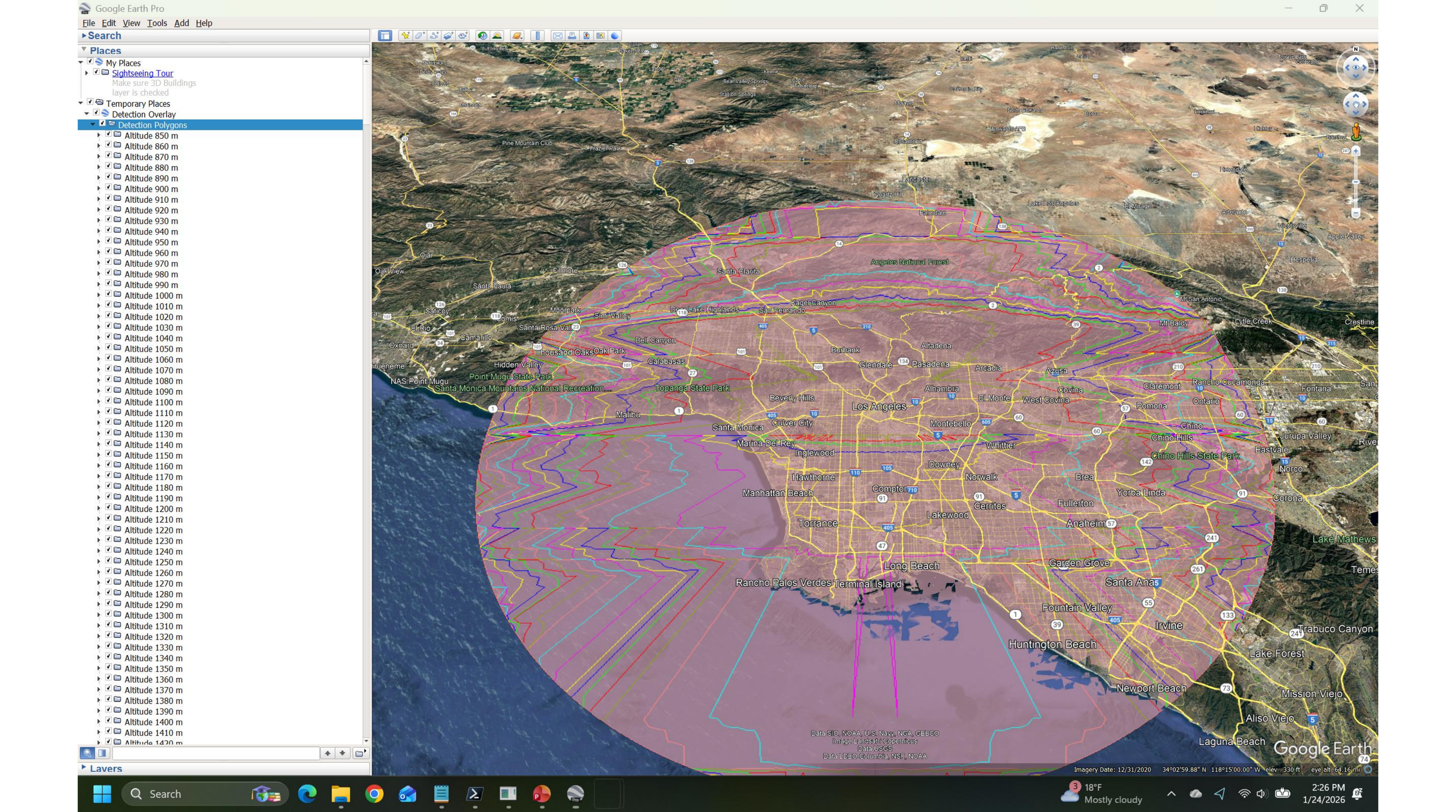
Computation Results

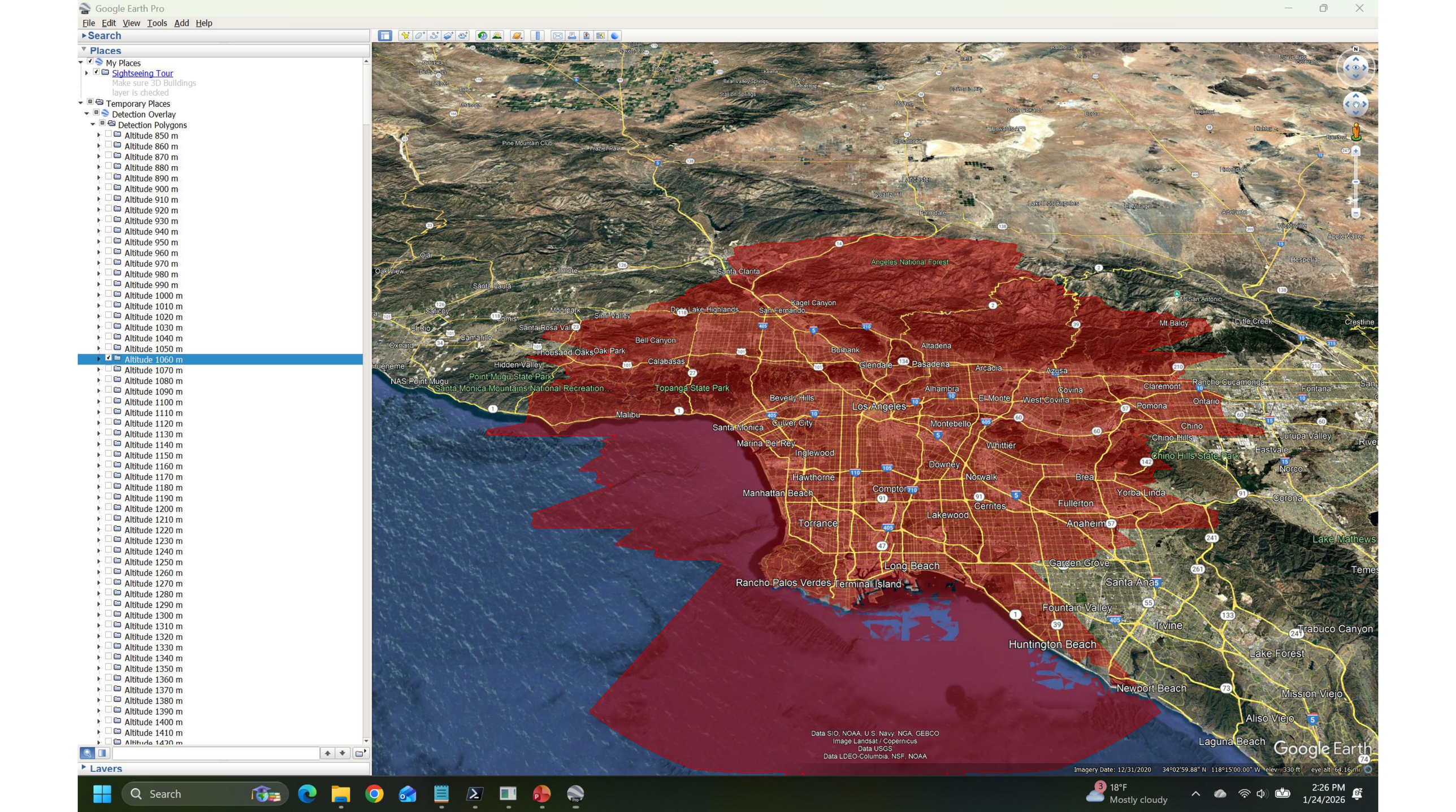
Output Directory: C:\Users\brian\Downloads\windows_package_fix\samples\scenarios\output

Generated Files:

- detection_overlay.kml (12652.6 KB)
- detection_overlay_20260122_125622.kml (12650.1 KB)
- detection_overlay_20260122_125808.kml (12650.1 KB)
- detection_overlay_20260122_130234.kml (12650.1 KB)
- point_results__20260122_125622.csv (760.0 KB)
- run_log_20260122_125622.txt (153.6 KB)
- run_log_20260122_125808.txt (106.9 KB)
- run_log_20260122_130234.txt (153.4 KB)
- run_log_20260122_130457.txt (153.4 KB)
- run_log_20260124_142228.txt (153.2 KB)


output
Close
Downloads
windows_package_fix
samples
scenarios
output
Search output
New
Sort
View
Preview
Name
Date modified
Type
Size
Today
run_log_20260124_142228 1/24/2026 2:22 PM Text Document 154 KB
detection_overlay 1/24/2026 2:22 PM KML 12,653 KB
Earlier this week
run_log_20260122_130457 1/22/2026 1:05 PM Text Document 154 KB
run_log_20260122_130234 1/22/2026 1:02 PM Text Document 154 KB
detection_overlay_20260122_130234 1/22/2026 1:02 PM KML 12,651 KB
run_log_20260122_125808 1/22/2026 12:58 PM Text Document 107 KB
detection_overlay_20260122_125808 1/22/2026 12:58 PM KML 12,651 KB
run_log_20260122_125622 1/22/2026 12:56 PM Text Document 154 KB
detection_overlay_20260122_125622 1/22/2026 12:56 PM KML 12,651 KB
point_results_20260122_125622 1/22/2026 12:50 PM Microsoft Excel Com... 760 KB
10 items | 2 items selected 12.5 MB
Search
18°F
Mostly cloudy
2:24 PM
1/24/2026


Google Earth Pro
File Edit View Tools Add Help
Search
Places
My Places
Sightseeing Tour
Make sure 3D Buildings layer is checked
Temporary Places
Detection Overlay
Detection Polygons
Altitude 850 m
Altitude 860 m
Altitude 870 m
Altitude 880 m
Altitude 890 m
Altitude 900 m
Altitude 910 m
Altitude 920 m
Altitude 930 m
Altitude 940 m
Altitude 950 m
Altitude 960 m
Altitude 970 m
Altitude 980 m
Altitude 990 m
Altitude 1000 m
Altitude 1010 m
Altitude 1020 m
Altitude 1030 m
Altitude 1040 m
Altitude 1050 m
Altitude 1060 m
Altitude 1070 m
Altitude 1080 m
Altitude 1090 m
Altitude 1100 m
Altitude 1110 m
Altitude 1120 m
Altitude 1130 m
Altitude 1140 m
Altitude 1150 m
Altitude 1160 m
Altitude 1170 m
Altitude 1180 m
Altitude 1190 m
Altitude 1200 m
Altitude 1210 m
Altitude 1220 m
Altitude 1230 m
Altitude 1240 m
Altitude 1250 m
Altitude 1260 m
Altitude 1270 m
Altitude 1280 m
Altitude 1290 m
Altitude 1300 m
Altitude 1310 m
Altitude 1320 m
Altitude 1330 m
Altitude 1340 m
Altitude 1350 m
Altitude 1360 m
Altitude 1370 m
Altitude 1380 m
Altitude 1390 m
Altitude 1400 m
Altitude 1410 m
Altitude 1420 m
Layers
Los Angeles
Long Beach
Anaheim
Santa Ana
Irvine
Huntington Beach
Newport Beach
Torrance
Pasadena
Glendale
Burbank
Santa Monica
Malibu
Beverly Hills
Hollywood
Culver City
Inglewood
Hawthorne
Manhattan Beach
Compton
Lakewood
Norwalk
Downey
Whittier
Montebello
El Monte
West Covina
Pomona
Ontario
Chino
Chino Hills
Fullerton
Brea
Yorba Linda
Corona
Garden Grove
Fountain Valley
Mission Viejo
Lake Forest
Aliso Viejo
Laguna Beach
Rancho Palos Verdes
Terminal Island
Santa Clarita
Angeles National Forest
Palmdale
Lancaster
Thousand Oaks
Simi Valley
Oxnard
Camarillo
Calabasas
Topanga State Park
Santa Monica Mountains National Recreation...
Point Mugu State Park
NAS Point Mugu
Hidden Valley
Bell Canyon
Oak Park
Moorpark
Santa Rosa Valley
Fillmore
Santa Paula
Ojai
Oak View
Malibu
Altadena
Arcadia
Azusa
Covina
Claremont
Rancho Cucamonga
Fontana
Mt San Antonio
Mt Baldy
Lytle Creek
Crestline
Hesperia
Apple Valley
Victorville
Lake Los Angeles
El Mirage
Pine Mountain Club
Frazier Park
Quartz Hill
Rosamond
Edwards AFB
Mojave
California City
Tehachapi
Lake Mathews
Corona
Norco
Eastvale
Jurupa Valley
Chino Hills State Park
Trabuco Canyon
Data SIO, NOAA, U.S. Navy, NGA, GEBCO
Image Landsat / Copernicus
Data USGS
Data LDEO-Columbia, NSF, NOAA
Google Earth
Imagery Date: 12/31/2020 34°02'59.88" N 118°15'00.00" W elev 330 ft eye alt 64.16 mi
Search
18°F
Mostly cloudy
2:26 PM
1/24/2026


Google Earth Pro
File Edit View Tools Add Help
Search
Places
My Places
Sightseeing Tour
Make sure 3D Buildings layer is checked
Temporary Places
Detection Overlay
Detection Polygons
Altitude 850 m
Altitude 860 m
Altitude 870 m
Altitude 880 m
Altitude 890 m
Altitude 900 m
Altitude 910 m
Altitude 920 m
Altitude 930 m
Altitude 940 m
Altitude 950 m
Altitude 960 m
Altitude 970 m
Altitude 980 m
Altitude 990 m
Altitude 1000 m
Altitude 1010 m
Altitude 1020 m
Altitude 1030 m
Altitude 1040 m
Altitude 1050 m
Altitude 1060 m
Altitude 1070 m
Altitude 1080 m
Altitude 1090 m
Altitude 1100 m
Altitude 1110 m
Altitude 1120 m
Altitude 1130 m
Altitude 1140 m
Altitude 1150 m
Altitude 1160 m
Altitude 1170 m
Altitude 1180 m
Altitude 1190 m
Altitude 1200 m
Altitude 1210 m
Altitude 1220 m
Altitude 1230 m
Altitude 1240 m
Altitude 1250 m
Altitude 1260 m
Altitude 1270 m
Altitude 1280 m
Altitude 1290 m
Altitude 1300 m
Altitude 1310 m
Altitude 1320 m
Altitude 1330 m
Altitude 1340 m
Altitude 1350 m
Altitude 1360 m
Altitude 1370 m
Altitude 1380 m
Altitude 1390 m
Altitude 1400 m
Altitude 1410 m
Layers
Angeles National Forest
Los Angeles
Santa Monica Mountains National Recreation...
Point Mugu State Park
Chino Hills State Park
Lake Mathews
Lake Los Angeles
Data SIO, NOAA, U.S. Navy, NGA, GEBCO
Image Landsat / Copernicus
Data USGS
Data LDEO-Columbia, NSF, NOAA
Google Earth
Imagery Date: 12/31/2020 34°02'59.88" N 118°15'00.00" W elev 330 ft eye alt 64.16 mi
Search
18°F
Mostly cloudy
2:26 PM
1/24/2026