International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 10 Issue: 10 | Oct - 2026 SJIF Rating: 8.659 ISSN: 2582-3930

Space Debris Visualization and Collision Prediction Using Vectorized SGP4 Propagation, Two-Stage Conjunction Screening, and Random Forest Risk Classification

Siddhu Kamble[1], Ranveer Chandel[2], Takshil Choure[3], Shravan Naktode[4], Rutvik Pattewar[5], Shubhangi Lanjewar[6]

[1,2,3,4,5]B.Tech Student, Department of Computer Technology, Priyadarshini College of Engineering, Nagpur, India
[6]Assistant Professor (Project Guide), Department of Computer Technology, Priyadarshini College of Engineering, Nagpur, India
Email: {siddhukamble7385, ranveerchandel802, ameypattewar}@gmail.com

***

Abstract - Low Earth Orbit (LEO) is the most congested region of the near-Earth environment, with tens of thousands of catalogued objects and far more untracked fragments sharing a shell only a few hundred kilometres thick. Sustained operation in this regime demands conjunction assessment that is both physically faithful and computationally tractable. This paper presents a fully functional Python-based system that ingests NORAD Two-Line Element (TLE) sets, propagates them through a vectorized SGP4 implementation, screens all pairwise conjunctions in two stages, classifies encounter risk with a supervised learning model, and renders the resulting geometry as an interactive three-dimensional scene in a web browser. Propagation is delegated to a batched sgp4_array call over a shared split Julian-date grid, yielding TEME position and velocity histories for the whole catalogue in one vectorized O(N·T) pass. Conjunction detection uses a coarse 60-second sweep to shortlist candidate pairs, followed by a 2-second refinement inside a ±300-second window to recover the Time of Closest Approach (TCA) and miss distance, reducing fine-grid work from O(NZT) to O(KT′) with K ≪ NZ. A Random Forest of 150 trees maps a six-dimensional orbital-geometry feature vector onto three risk classes, yielding a soft decision boundary in place of a single distance cutoff. In a 24-object, 24-hour demonstration the system detected three CRITICAL and three WARNING conjunctions, with the classifier reaching 99.83% held-out accuracy.

Key Words: *Space Debris, SGP4 Propagation, Conjunction Screening, Random Forest, Space Situational Awareness, Plotly 3D.*

1. INTRODUCTION

The region of space below 2,000 km now contains more than 36,000 objects larger than 10 cm under routine surveillance by the United States Space Surveillance Network, alongside an estimated million-plus fragments too small to catalogue but energetic enough to destroy a spacecraft [4]. A 1 cm aluminium fragment at a typical LEO closing speed of 10–14 km/s carries the kinetic energy of a mid-size car at highway speed; shielding against such impact is impractical for most missions, so avoidance is the only viable mitigation. Kessler and Cour-Palais [3] showed that above a critical spatial density, fragmentation generates debris faster than atmospheric drag removes it, producing a self-sustaining collision cascade that can render orbital shells unusable for generations. The 2007 Fengyun-1C anti-satellite test and the 2009 Iridium-33/Kosmos-2251 collision each injected thousands of trackable fragments into the 700–900 km band, and large broadband constellations have increased conjunction alert volume by an order of magnitude.

Conjunction assessment reduces, in principle, to a simple question: for every pair of objects, what is the minimum separation over a look-ahead window? The naive formulation is brute force. For N objects propagated across T time steps, evaluating every pair at every step costs

$$C = N(N-1)/2 \cdot T \in O(N^2T) \quad (1)$$

Resolving a TCA to within a few hundred metres for objects closing at 14 km/s requires a time step of one or two seconds; a 24-hour window at a 2-second step is T = 43,200 samples. Even for a modest 25-object catalogue that is 300 pairs × 43,200 steps ≈ 1.30×10^7 distance evaluations, and at N = 20,000 the pair count exceeds 2×10^8, making the fine-grid formulation intractable on commodity hardware. The resolution of this tension is that fine temporal resolution is only needed where objects are actually close. This work implements a complete pipeline that exploits that observation. Its contributions are: (1) batched propagation of the whole catalogue through a single vectorized SGP4 call over a shared time grid; (2) two-stage conjunction screening replacing O(NZT) fine-grid work with O(NZTc) + O(KT′); (3) a learned multi-feature risk boundary in place of a single distance cutoff; and (4) browser-native 3D visualization served directly by Flask with no client-side installation.

2. LITERATURE REVIEW

Hoots and Roehrich [2] documented the SGP4/SDP4 analytical propagation models in Spacetrack Report No. 3, and Vallado et al. [1] consolidated the circulating code variants with verified test cases that form the basis of the open-source sgp4 library [10] used here. The literature consistently notes that TLEs and SGP4 form a matched pair whose output is expressed in the TEME frame, and that TLE-derived accuracy is of the order of a kilometre at epoch, degrading as the element set ages [1], [12]. Kelso's CelesTrak [5] established public TLE distribution, and the SOCRATES service [14] demonstrated catalogue-scale conjunction reporting of TCA and miss distance; classical screening literature further describes apogee/perigee sieves and coarse-to-fine temporal searches that avoid exhaustive fine-grained comparison. Operational avoidance decisions are made on collision probability, for which Foster and Estes [7] and Alfano [6] provide the standard formulations —

methods requiring covariance data that public TLEs do not carry. On the learning side, ESA's Spacecraft Collision Avoidance Challenge showed that ensemble tree methods perform strongly on real conjunction data, and Breiman's Random Forest [8], implemented in scikit-learn [9], is favoured for its robustness and interpretable Gini importances. No reviewed public system combines accurate propagation, economical screening, learned risk classification and interactive 3D visualization in one open, lightweight package; that gap defines this work.

3. METHODOLOGY

A. TLE Ingestion

Each TLE record comprises a name line and two fixed-column 69-character data lines encoding the epoch, BSTAR drag term, inclination i, right ascension of the ascending node fi, eccentricity e (implied leading decimal), argument of perigee ω, mean anomaly M and mean motion n in revolutions per day, each terminated by a modulo-10 checksum [5], [13]. The ingestion module (src/tle_parser.py) reads name/line-1/line-2 triplets, recomputes and repairs checksums — archival TLEs frequently carry corrupted final digits — and passes each pair to Satrec.twoline2rv. Derived quantities used downstream as learning features are computed explicitly; the semi-major axis follows from the mean motion through Kepler's third law:

$$a = (\mu / n^2)^{(1/3)}, \quad n = 2\pi\, n_rev / 86400 \ rad/s \quad (2)$$

with μ = 398,600.4418 km3/sZ, and mean altitude h = a − R⊕ relative to the WGS-84 equatorial radius R⊕ = 6378.137 km.

a. SGP4 in the TEME Frame

SGP4 is an analytical propagator: it applies closed-form secular and periodic corrections to the mean elements rather than numerically integrating equations of motion [1], [2]. The dominant secular effects arise from Earth's J2 zonal harmonic, which induces nodal regression and apsidal precession:

$$\dot{\Omega} < = -(3nJ_2R\oplus^2 / 2p^2) \cos i, \ \dot{\omega} < = (3nJ_2R\oplus^2 / 4p^2)(5\cos^2 i - 1) \quad (3)$$

where p = a(1−eZ) and $J_2 \approx 1.08263 \times 10^{-3}$. SGP4 additionally models drag through BSTAR and solves Kepler's equation E − e sin E = M iteratively to obtain the Cartesian state. Critically, SGP4 output is expressed in the True Equator, Mean Equinox (TEME) frame: elements fitted with SGP4 must be propagated with SGP4 and interpreted in TEME, or systematic error is introduced [1]. Because screening depends only on relative geometry at a common instant, all distances are computed directly in TEME. The propagator (src/propagator.py) builds a shared grid over the configured window (24 h) and invokes the batched Satrec.sgp4_array entry point, carrying Julian dates as a split integer/fractional pair to preserve timing precision. The call returns position and velocity tensors of shape (N, T, 3) with the inner loop in compiled code [10]; objects returning non-zero SGP4 error codes are excluded before screening.

C. Two-Stage Conjunction Screening

Stage 1 evaluates, for each of the N(N−1)/2 pairs, the separation history on a 60-second coarse grid with vectorized NumPy operations:

$$d(t) = \|r_A(t) - r_B(t)\|_2 \quad (4)$$

A pair is shortlisted when min d(t) ≤ 4D_warn with D_warn = 5.0 km. The factor of four guards against coarse-grid aliasing: objects closing at 14 km/s traverse 840 km between consecutive 60-second samples, so a true close approach can fall between grid points. Stage 2 re-propagates each of the K shortlisted pairs (K ≪ NZ) on a 2-second grid within ±300 s of the coarse minimum, recovering the TCA, the refined miss distance and the relative velocity at closest approach. Refined pairs are tagged CRITICAL (d_min ≤ 1.0 km), WARNING (≤ 5.0 km) or SAFE. Table 1 compares the formulations for a 25-object, 24-hour analysis: cost falls roughly 30×, from O(NZT) to O(NZTc) + O(KT′); the quadratic pair term survives, but the multiplicative constant — previously the binding constraint — is reduced by the coarse-to-fine step ratio.

Table -1: DISTANCE EVALUATION COST, N = 25, 24-HOUR WINDOW

Formulation	Steps T	Pairs	Evaluations
Single-stage fine (2 s)	43,200	300	1.30×10^7
(60 s) Two-stage: coarse	1,440	300	4.32×10^5
Two-stage: fine (±300 s)	301	5–15	1.5–4.5 × 103
Two-stage total	–	–	$\approx 4.4 \times 10^5$

D. Random Forest Risk Classification

The learning module trains a 150-tree Random Forest [8], [9] over the six-dimensional feature vector of Table 2. Each tree is grown on a bootstrap resample with a random feature subset at every split, minimising Gini impurity

$$G = 1 - \Sigma\, p_c^2, \ c \in \{CRITICAL, WARNING, SAFE\} \quad (5)$$

and the ensemble prediction is the majority vote with an associated class probability. The motivation is physical: a static cutoff treats a 3 km miss between near-coplanar objects identically to a 3 km miss between objects in near-perpendicular planes, despite an order-of-magnitude difference in closing speed and consequence severity. High Δi and Δfi imply crossing geometry and hypervelocity closure; low Δh with low Δi implies a slow, potentially repeating along-track encounter. A tree ensemble captures such conditional structure without hand-specified interaction terms. Since public ground-truth collision records do not exist in usable quantity, training labels are generated from the geometric thresholds — a choice whose implications Section 5 analyses explicitly. The fitted model is persisted via joblib (≈5 s on first start-up) so training never sits on the request path, and the system reports accuracy, a confusion matrix and Gini feature importances so the learned boundary can be inspected rather than trusted blindly.

Feature	Description	Unit
min_dist_km	Geometric miss distance at TCA	km
rel_velocity_kms	Relative speed at TCA	km/s
altitude_diff	Mean altitude difference	km
inclination_diff	Orbital plane tilt difference	deg
eccentricity_diff	Orbit shape difference	–
raan_diff	Node line angular difference	deg

Table -2: CLASSIFIER FEATURE VECTOR

B. System Architecture and Visualization

The application follows a model–view–controller separation: app.py exposes a landing route and an analysis route; the model comprises independently unit-tested src/ modules (parser, propagator, screener, classifier, visualizer, report assembler); Jinja2 templates form the view. All tunable constants are centralised in config.py (propagation window 24 h; coarse/fine steps 60 s / 2 s; refinement window ±300 s; CRITICAL/WARNING thresholds 1.0 / 5.0 km; shortlist gate 20 km; 150 trees), giving the screener, the label generator and the visualizer's colour mapping a single source of truth. Fig. 1 shows the end-to-end pipeline. The visualizer builds a Plotly figure [11] with three trace families: a parametric spherical Earth mesh, a Scatter3d trace per object through its propagated TEME positions, and severity-coded conjunction markers (red = CRITICAL, amber = WARNING) with hover text carrying identifiers, miss distance, TCA and relative velocity. Plotly delegates rendering to WebGL, so rotation, zoom and trace toggling run on the client GPU with no server round-trip; the figure is serialised into the Flask response so the user needs only a browser.

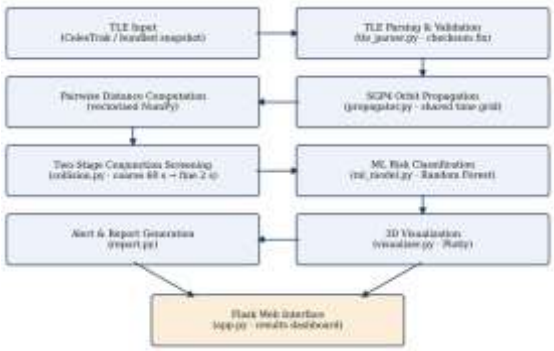


Fig -1: End-to-end processing pipeline

4. RESULTS

The system was evaluated on a bundled 24-object catalogue — operational satellites (ISS, Hubble, NOAA-19, Terra, Landsat-8, Sentinel-2A, Aqua, five Starlink spacecraft), fragments from the Cosmos-2251, Iridium-33 and Fengyun-1C events, and six synthetic near-miss objects included to guarantee demonstrable conjunction geometry — propagated over 24 hours. The two-stage screener detected six conjunctions: three CRITICAL (refined miss distances 0.4357 km, 0.7154 km and 0.7231 km) and three WARNING (1.1521 km, 1.6774 km and 2.4015 km), each with second-level TCA resolution. Fig. 2 shows the rendered scene with severity-coded markers along the synthetic debris orbits. The Random Forest, trained on 2,400 samples and evaluated on 600 held-out samples, reached 99.83% accuracy, and its predictions agreed with the geometric tags on all six events; Gini importances confirm that miss distance dominates (0.9056) with the five remaining features contributing ≈0.018–0.020 each. Wall-clock cost is dominated by propagation rather than screening: the 4.3×10^5 coarse evaluations complete within the propagation's own runtime, realising the ~30× reduction of Table 1 in practice. At larger catalogue sizes the binding constraint becomes response payload — each 3D trace must be serialised to the browser — which trace decimation decouples from screening resolution.

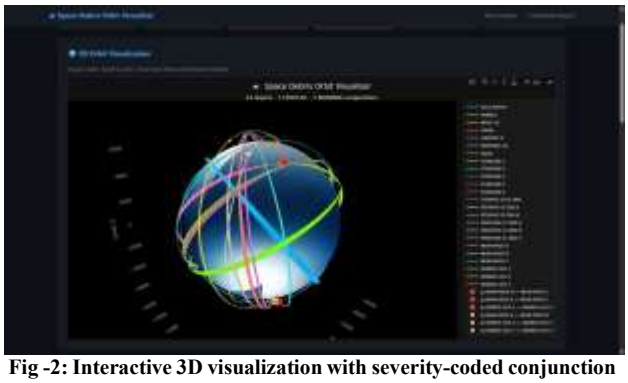


Fig -2: Interactive 3D visualization with severity-coded conjunction markers

5. DISCUSSION

Four limitations bound the current system. Ephemeris degradation: TLE-derived state carries in-track uncertainty of the order of a kilometre at epoch, growing to tens of kilometres within a week [1], [12]; since CRITICAL is declared at 1.0 km, a predicted miss distance can be smaller than its own uncertainty, so results are meaningful only for recently refreshed TLEs. Simplified perturbations: SGP4's drag treatment is deliberately approximate, solar radiation pressure is unmodelled, and atmospheric density does not respond to solar activity — significant below 500 km during elevated flux. Physics-derived labels: class labels are generated from the same thresholds the screener applies, so the classifier recovers a known rule; the high accuracy is a pipeline validation, not evidence of predictive skill, though the architecture transfers directly to operator-validated labels. No collision probability: the system reports deterministic miss distance, whereas operational decisions are made on Pc, which requires covariance absent from TLEs [6], [7]. Additionally, all $N(N-1)/2$ pairs enter Stage 1; apogee–perigee filtering [14] and spatial hashing would reduce this toward near-linear behaviour and are prerequisites for catalogue-scale operation.

6. CONCLUSIONS

This paper presented an end-to-end conjunction assessment system built entirely from open-source components. Its central contribution is the demonstration that a two-stage temporal decomposition — coarse screening to bound the search, fine refinement to resolve the answer — reduces distance-evaluation cost by roughly thirtyfold at demonstration scale while preserving second-level TCA accuracy, making

meaningful conjunction analysis achievable on commodity hardware. The vectorized propagation layer, the multi-feature learned risk boundary and the browser-native 3D visualization form a coherent pipeline from raw TLE text to actionable, inspectable alerts. The system's limitations are documented candidly, and the paths toward operational credibility — continuously refreshed elements via the authenticated Space-Track API, Monte Carlo collision probability in place of deterministic distance, a delta-V recommendation engine, and validated labels for the learned boundary — are clearly identified as future work.

ACKNOWLEDGEMENT

The authors sincerely thank their project guide, Prof. Shubhangi Lanjewar, Assistant Professor, Department of Computer Technology, Priyadarshini College of Engineering, Nagpur, for her continuous guidance and support throughout this work.

REFERENCES

[1] D. A. Vallado, P. Crawford, R. Hujsak, and T. S. Kelso, "Revisiting Spacetrack Report #3," in Proc. AIAA/AAS Astrodynamics Specialist Conf., Keystone, CO, USA, 2006, AIAA 2006-6753.

[2] F. R. Hoots and R. L. Roehrich, "Models for Propagation of NORAD Element Sets," Spacetrack Report No. 3, Aerospace Defense Command, Peterson AFB, CO, USA, Dec. 1980.

[3] D. J. Kessler and B. G. Cour-Palais, "Collision Frequency of Artificial Satellites: The Creation of a Debris Belt," J. Geophys. Res., vol. 83, no. A6, pp. 2637–2646, Jun. 1978.

[4] ESA Space Debris Office, "ESA's Annual Space Environment Report," European Space Agency, Darmstadt, Germany, 2025.

[5] T. S. Kelso, "CelesTrak." [Online]. Available: https://celestrak.org

[6] S. Alfano, "A Numerical Implementation of Spherical Object Collision Probability," J. Astronaut. Sci., vol. 53, no. 1, pp. 103–109, 2005.

[7] J. L. Foster and H. S. Estes, "A Parametric Analysis of Orbital Debris Collision Probability and Maneuver Rate for Space Vehicles," NASA JSC-25898, Aug. 1992.

[8] L. Breiman, "Random Forests," Mach. Learn., vol. 45, no. 1, pp. 5–32, 2001.

[9] F. Pedregosa et al., "Scikit-learn: Machine Learning in Python," J. Mach. Learn. Res., vol. 12, pp. 2825–2830, 2011.

[10] B. Rhodes, "sgp4: Track Earth Satellites Given TLE Data," Python Package Index. [Online]. Available: https://pypi.org/project/sgp4/

[11] Plotly Technologies Inc., "Collaborative Data Science," Montréal, QC, Canada, 2015. [Online]. Available: https://plot.ly

[12] D. A. Vallado, Fundamentals of Astrodynamics and Applications, 4th ed. Hawthorne, CA, USA: Microcosm Press, 2013.

[13] 18th Space Defense Squadron, "Space-Track.org." [Online]. Available: https://www.space-track.org

[14] F. R. Hoots, L. L. Crawford, and R. L. Roehrich, "An Analytic Method to Determine Future Close Approaches Between Satellites," Celestial Mechanics, vol. 33, no. 2, pp. 143–158, 1984.