LEO Satellite Network Systems for Instantaneously Detecting Environmental Changes

by

Zian Wang

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Mathematics
in
Computational Mathematics

Waterloo, Ontario, Canada, 2026

Author's Declaration

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

The rapid deployment of Low Earth Orbit (LEO) constellations has introduced the concept of orbiting data centers—interconnected satellites with onboard computing and high-speed inter-satellite links (ISLs). This study tests the hypothesis that such architectures, equipped with a deep-learning computer vision pipeline, can achieve sub-minute information freshness suitable for real-time wildfire detection. We evaluate this through a simulation framework modeling orbital dynamics, distributed processing, and network routing, using Age of Information (AoI) as the performance metric. Over 720 trials spanning 12 real-world constellations (i.e., Starlink, Kuiper, Telesat, and OneWeb), we find that constellation design dramatically impacts AoI, with average AoI ranging from 66.5 s to over 6,300 s. The best-performing configurations achieve an average AoI below 70 s and a peak AoI under 100 s, confirming that orbital edge computing can deliver the such level of timeliness for near-instantaneous environmental monitoring.

Acknowledgements

I would like to thank Professor Peng Hu and Professor Grant Gunn for their continued support and for giving me this opportunity.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

AAoI Average Age of Information vi, viii, 6, 18, 25–30

AoI Age of Information 1, 5–7, 18, 23, 25, 32

ECEF Earth-Centered Earth-Fixed 11, 12

EO Earth Observation 7

GDAL Geospatial Data Abstraction Library 7

GS Ground Station 8, 9, 13, 20

ISL Inter-Satellite Link 2, 13, 15, 22

LEO Low Earth Orbit 2, 7, 8, 32

MTU Maximum Transmission Unit 15, 22

PAoI Peak Age of Information 6, 18

RAAN Right Ascension of the Ascending Node 8, 10

RoI Region of Interest 7, 9, 11, 12, 20, 29

SGL Satellite-to-Ground Link 13, 15, 22

Chapter 1

Introduction

The escalation of catastrophic wildfires globally has emerged as one of the most pressing environmental challenges of the 21st century. Forest fires impact twice the extent of tree cover compared to the early 2000s, with 2023 and 2024 marking some of the most destructive years recorded [1]. According to United Nations Environment Programme (UNEP), the frequency of extreme wildfire events is projected to increase by 30% by 2050 and 50% by the 2100 [2].

Historically, wildfire surveillance was fundamentally localized, relying on ground-based observation and human-centric monitoring. Traditional methods, such as direct visual assessment from fixed watchtowers and periodic airborne reconnaissance, provided high-fidelity in-situ data for specific, high-risk areas. However, traditional monitoring frameworks are inherently limited by their geographic scope[3]. While geostationary (GEO) platforms like the GOES-R series currently provide the frequent spaceborne updates for wildfire monitoring, their operational utility is fundamentally constrained by a critical resolution-frequency trade-off. As highlighted by [4], the GOES-R Advanced Baseline Imager acquires images every 10 to 15 min, (while superior to the multi-day revisit times of traditional sun-synchronous satellites) but still imposes a significant lower bound on the Age of Information (AoI). In the context of rapidly spreading wildfires, a 15-min sampling interval represents a significant temporal bottleneck that can allow an ignition to transition into an uncontrollable event before the first alert is generated. Furthermore, satellites in a high orbit (e.g., GEO) have limited thermal spatial resolution, approximately 2 km per pixel, so early-stage ignitions may appear as sub-pixel hot spots that are difficult to detect reliably.

To address the limitations of traditional remote sensing approach, recent research has

shifted toward Low Earth Orbit (LEO) architectures that may combine the high spatial resolution with the high temporal density of modern mega-constellations, although several challenges remain. Current LEO missions, such as Landsat 8, provide the 30-meter multispectral imagery required for precise fire segmentation but are limited by 16-day revisit cycles [5]. By contrast, emerging remote sensing constellations such as PlanetScope with hundreds of Dove satellites [6], or a LEO mega-constellation represented by Starlink can leverage large networks of interconnected nodes to potentially achieve much higher revisit rates. With high-speed Inter-Satellite Link (ISL) and significant onboard processing power, a LEO satellite constellation can function as a distributed orbiting computing cluster. This "Orbital Edge Computing" paradigm enables the local generation of fire masks, shifting the bottleneck from raw data transmission to distributed computational efficiency [7]. This study is based on the hypothesis that, if each node in a mega-constellation were equipped with remote sensing capabilities comparable to Landsat 8 and dedicated multi-threaded computational resources, then shifting the bottleneck from data transmission to parallelized local computation could reduce the AoI to a level suitable for real-time wildfire suppression, thereby transforming the constellation into a reliable, high-fidelity pipeline. In this research paper, we extend this paradigm through three main contributions:

- Real-world LEO constellation configurations are employed (e.g., Starlink, Kuiper, Telesat, OneWeb) rather than hypothetical Walker-delta topologies.
- A specialized sensing task for wildfire detection is investigated, using a pixel-wise U-Net model on multi-spectral Landsat-8 imagery over a considerably smaller region of interest that demands high coverage probability.
- Realistic processing latencies, derived from experiments on a high-performance computing (HPC) cluster, are incorporated into the analysis, grounding the simulations in observed computational delays.

Our contributions enable a rigorous evaluation of how physical architecture and orbital dynamics directly influence AoI in time-critical environmental monitoring. The remainder of the paper is structured as follows: Chapter 2 discusses the related work, Chapter 3 presents the system model; Chapter 4 discusses the experimental results; and Chapter 5 presents the concluding remarks.

Chapter 2

Related Work

The development of real-time environmental monitoring systems in LEO draws from several evolving research domains. We categorize the existing literature into theoretical foundations, scheduling optimization, and edge applications.

2.1 Theoretical Foundations of Information Freshness

Recent literature has established theoretical foundations for data freshness by investigating the joint impact of sensing, transmission, and computation. An *et al.* [8] derived closed-form expressions for the Peak AoI in mobile edge computing (MEC) systems, identifying a "U-shape" performance trend where increasing the sensing rate initially improves freshness but eventually leads to queueing saturation. Their work emphasizes that a partial computation strategy—distributing tasks between local and edge servers—offers superior stability and lower PAoI compared to purely local or edge-based processing. Theoretical analysis by Tang *et al.* [9] provides closed-form expressions for average AoI in multi-user settings, revealing that while edge computing minimizes AoI for small user groups, local computing becomes more effective as network congestion increases.

2.2 Optimization and Scheduling Frameworks

Beyond AoI models, research has been conducted toward optimizing the complex task flows of satellite networks. A distributed deep reinforcement learning (DRL) approach

[10] was proposed to minimize delay costs by predicting fluctuating satellite loads and optimizing multi-hop inter-satellite offloading. Addressing the integration of these tasks, Jaipuria *et al.* [11] proposed an integrated collect-communicate-compute framework for LEO constellations. By utilizing an integer linear programming (ILP) based solution, this approach generates optimal schedules that maximize the number of regions collected and analyzed.

2.3 Onboard Edge Intelligence and Applications

The practical feasibility of these paradigms is increasingly demonstrated through targeted applications and hardware benchmarking. The specific application of distributed edge computing architectures to LEO constellations was introduced for vessel detection using YOLOv8 on simplified Walker-delta shells [12]. The feasibility of such pipelines has been further strengthened by testing deep learning (DL) models, such as VFNet and RADCNet, for maritime identification on representative edge-AI hardware, including Raspberry Pi and VPU-based architectures. These studies demonstrate that end-to-end processing of raw imagery can be achieved within stringent power budgets and latency constraints [13]. Furthermore, transfer learning approaches are being utilized to enhance the performance of lightweight models for environmental tasks; for instance, Gain et al. demonstrated that a low-parameterized MobileNetV2 model can be fine-tuned using transfer learning to achieve near 100% accuracy in wildfire detection [14].

Based on the aforementioned works, we can see the theoretical foundations, scheduling optimization, and practical edge applications research domains are largely built on the assumption of hypothetical satellite networks with idealized computational power and transmission capability. The need exists to identify the capacity of current LEO network system models to reduce AoI for circumstances where immediate classification of imagery and transmission is needed. Our paper addresses this important gap in the current literature.

Chapter 3

System Model

3.1 Age of Information

Before introducing the simulation setting, the key concept, AoI, should be accurately defined. AoI is a quantitative metric that measures information freshness. Contrasting traditional metrics like end-to-end latency only measuring end to end delay, throughput which only measure data rate of the network, AoI is a combined measure of 2 main metrics, the frequency, or average time interval for of signal firing, and the end to end delay for a task (in the context of this article, it means from the time the satellite acquisition occurs, to the time when the finished symbolic result is fully received on ground station). We can define the age of information at time t as

$$AoI(t) = t - I(t) \tag{3.1}$$

where $I(t)$ is the timestamp recorded for the lastest received information [15]. Suppose we have a sequence of tasks initiated at times

$$t_0, t_1, t_2, \ldots \tag{3.2}$$

and completed at corresponding times

$$t'_0, t'_1, t'_2, \ldots \tag{3.3}$$

Between task completions, the age of information increases linearly with slope (as we are using a linear AoI. When a task t_n completes at time t'_n, the system has the opportunity

to update $I(t)$. Specifically, the new candidate timestamp is t_n, corresponding to the generation time of the newly finished task. If the current AoI right before t'_n is larger than $t'_n - t_n$, i.e.,

$$AoI(t'_n) > t'_n - t_n, \tag{3.4}$$

then the new information is fresher than all previously received updates, and the AoI resets according to

$$I(t'_n) = t_n. \tag{3.5}$$

Otherwise, if $t'_n - t_n$ is not smaller than the existing AoI, the update does not provide newer information, and thus $I(t)$ remains unchanged. This mechanism ensures that the AoI process evolves as a piecewise-linear, sawtooth-like function that only resets when genuinely fresher information becomes available.

To measure the overall system timeliness over a period of time T, we can introduce AAoI over a period of time T

$$AoI_{avg} = \lim_{T\to\infty} \frac{1}{T} \int_0^T AoI(t)\, dt \tag{3.6}$$

However, calculating continues AoI is quite challenging in term of simulation design, so we discretize the time into a series of timestamps with equal interval called time step Δt.

$$t_0, t_1, t_2, \ldots, t_n \tag{3.7}$$

where

$$\forall n \in \{1, 2, \ldots, n\}, \quad t_n - t_{n-1} = \Delta t \tag{3.8}$$

then we can write the average AOI in series

$$AoI_{avg} = \sum_{i=0}^{n} AoI(t_i) \times \Delta t \tag{3.9}$$

In addition to the AAoI, another important metric is the Peak Age of Information (PAoI). While AAoI captures the long-term timeliness of the system, PAoI characterizes the worst-case staleness that may occur. Formally, PAoI over a time horizon T is defined as

$$AoI_{\text{peak}} = \max_{0\le t\le T} AoI(t). \tag{3.10}$$

In the discretized setting, where the continuous process is sampled at intervals Δt, the Peak AoI can be written as

$$AoI_{\text{peak}} = \max_{0\le i\le n} AoI(t_i). \tag{3.11}$$

This metric is particularly relevant for applications where the maximum staleness cannot exceed a certain threshold, for example in real-time monitoring or control systems.

3.2 Operational Scenario: Distributed Wildfire Detection

To evaluate the proposed LEO network modeling pipeline, we define a time-critical mission scenario centered on real-time wildfire detection. This operational workflow bridges the high-level AoI requirements with the technical execution of the simulator, following a four-stage distributed computing sequence.

Event Triggering and Region Coverage

The simulation monitors a predefined geographic Region of Interest (RoI). In this framework, a satellite is designated as a Covered Satellite only when the vertices of the target RoI are fully within its swath [16] . This rigorous coverage constraint ensures the satellites ability to captured the scene encompasses the entire region for reliable inference. Once a satellite meets these criteria, an Earth Observation (EO) task is initiated at the discrete timestamp t_i.

Task Partitioning and Patch Generation

Upon capturing a high-resolution scene, the covered satellite acts as the *Primary Node* for the task. Given the substantial dimensions of raw satellite imagery and the input constraints of deep learning architectures—specifically U-Net-based active fire detection models [17]—the primary node performs spatial partitioning. Utilizing the Geospatial Data Abstraction Library (GDAL), the original scene is decomposed into a set of discrete, uncompressed image patches. This partitioning allows the computational load to be distributed across the constellation, preventing a processing bottleneck at the sensing node.

Parallel Processing and Distributed Computing

Each participating neighbor node functions as a worker, independently executing the active fire detection model on its assigned uncompressed image patches. The model performs

pixel-wise classification to generate binary inference masks, where specific pixel intensities represent potential fire risk anomalies. By leveraging the collective computational resources of the LEO constellation, the system can potentially achieve a significant reduction in processing latency through parallel execution across multiple orbital nodes.

Result Aggregation and Network Routing

Upon completion of the inference task, each worker node generates a symbolic result consisting of a binary classification mask for every processed patch. To facilitate efficient data delivery, these symbolic results are encapsulated into a series of discrete network data packets. We estimate the cumulative size of these generated masks to model the communication load accurately. These packets are then independently injected into the network and routed toward the predefined Ground Station (GS). The routing logic utilizes a Dijkstra-based shortest-path strategy over the time-varying network graph, prioritizing the most efficient delivery paths at the current simulation timestamp. A detection task is considered successfully concluded only when the full sequence of data packets—representing the complete set of classification masks for the original scene—has been received and reconstituted at the GS. At this point, the system records task end time.

3.2.1 Orbital Dynamics and Topology Generation

The simulation environment is modeled as a discrete-time evolution of a satellite network over a finite time horizon T_{sim}. We represent the system through a set of predefined static parameters and dynamic state variables.

Constellation The LEO constellation is defined as the parameter tuple

$$\mathcal{C} = \langle P, R, \iota, h, \theta_{\min}, F, \mathcal{W} \rangle \tag{3.12}$$

where P denotes the number of orbital planes and R denotes the number of satellites per plane. The pair (ι, h) specifies the orbital inclination (in degrees) and orbital altitude (i·n meters). The phasing parameter F, constrained by $0 \leq F < P$, controls the along-track angular offset between adjacent planes through $\Delta\phi = (360^\circ \cdot F)/(P \cdot R)$. The term $\theta_{\min}$ is the minimum elevation angle required to permit a satellite-to-ground link, and $\mathcal{W}$ denotes the Walker pattern class (e.g., Delta or Star), which determines the Right Ascension of the Ascending Node (RAAN) distribution across planes.

Ground Station (GS) The network includes a single fixed ground station represented by the terrestrial node g, with geodetic coordinates

$$g = (\phi_g, \lambda_g). \tag{3.13}$$

Here, ϕ_g and λ_g are the station latitude and longitude, respectively, and the station altitude is fixed at mean sea level, i.e., $h_g = 0$.

Region of Interest (RoI) The monitored wildfire area is modeled as a single predefined polygonal region r with four geodetic vertices, expressed as

$$r = \{\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3, \mathbf{v}_4\}, \quad \mathbf{v}_j = (\phi_j, \lambda_j). \tag{3.14}$$

This definition provides the geographic boundary used by the coverage and sensing logic throughout the simulation.

The simulation progresses through a sequence of discrete steps $i \in \{0, 1, \ldots, N\}$, where the total number of intervals is determined by $N = (T_{end} - T_{start})/\Delta t$. The system state at any specific step i is encapsulated by the tuple $\mathcal{T}_i$:

$$\mathcal{T}_i = \langle t_i, \mathcal{A}(t_i), G(t_i) \rangle \tag{3.15}$$

In this tuple, $t_i = t_{start} + i \cdot \Delta t$ denotes the discrete timestamp at step i. The term $\mathcal{A}(t_i) \subseteq \mathcal{S}$ denotes the active set of Covered Satellites that satisfy the spatial coverage condition for the region r at time t_i. The graph component is given by $G(t_i) = (\mathcal{V}, \mathcal{E}(t_i))$, which represents the directed communication topology at time t_i, with vertex set $\mathcal{V} = \mathcal{S} \cup \{g\}$ containing all satellites in $\mathcal{S}$ together with the single ground station node g.

The temporal evolution of the constellation is modeled by updating the spatial coordinates of each satellite $s_{p,r} \in \mathcal{S}$ at every discrete timestamp t_i. This process employs an analytical Keplerian propagator for circular orbits.

Each satellite $s_{p,r}$ is initialized as a point in a circular orbit ($e = 0$), where its position is defined relative to the Earth's center. The primary orbital elements are derived as follows:

Semi-major axis and inclination For a circular orbit at altitude h, the semi-major axis is given by $a = R_E + h$, where $R_E \approx 6371$ km denotes the mean Earth radius. The inclination ι specifies the angular tilt of the orbital plane relative to the equatorial plane and is measured in degrees.

RAAN distribution The RAAN, denoted by Ω_p, determines the orientation of orbital plane p in the inertial frame. Geometrically, it is measured eastward along the celestial equator from the *Vernal Equinox* (Υ) to the ascending node, where the satellite crosses the equator from south to north. Under a Walker pattern, the RAAN values are distributed uniformly across planes according to

$$\Omega_p = \left(\frac{\Psi \cdot p}{P}\right), \quad \Psi = \begin{cases} 360°, & \text{if } \mathcal{W} = \text{Delta} \\ 180°, & \text{if } \mathcal{W} = \text{Star} \end{cases} \tag{3.16}$$

for each $p \in \{0, \ldots, P-1\}$.

Mean anomaly at epoch The mean anomaly at epoch, denoted by $M_{p,r}$, specifies the along-track angular position of satellite index r in plane p at time t_{epoch}. This quantity incorporates both intra-plane spacing and inter-plane phasing through

$$M_{p,r} = \left(\frac{360° \cdot r}{R} + \frac{360° \cdot F \cdot p}{T}\right) \pmod{360}, \tag{3.17}$$

where $T = P \cdot R$ is the total number of satellites in the constellation.

The movement of the satellite is governed by its angular velocity, or mean motion n. Derived from Kepler's third law, n represents the rate of change of the satellite's position in rad/s. In the simulation, this is implemented using the standard gravitational parameter, which is the product of the universal gravitational constant G and the mass of the Earth M_{Earth}:

$$n = \sqrt{\frac{GM}{a^3}} \tag{3.18}$$

where $G \approx 6.6743 \times 10^{-11} \text{m}^3 \cdot \text{kg}^{-1} \cdot \text{s}^{-2}$ denotes the universal gravitational constant and $M_{Earth} \approx 5.972 \times 10^{24}$ kg denotes the mass of the Earth. By utilizing the semi-major axis a in km, the mean motion n is obtained in rad/s. At any simulation time t_i, the current state is updated by calculating the elapsed time $\Delta\tau$ in s since the initial epoch t_{epoch} (1 Julian Day = 86,400 s):

$$\Delta\tau = (JD(t_i) - JD(t_{epoch})) \times 86,400 \tag{3.19}$$

$$M(t_i) = (M_{p,r} + n \cdot \Delta\tau) \pmod{2\pi} \tag{3.20}$$

where $M_{p,r}$ is the mean anomaly at epoch in radians. Since the simulation assumes circular orbits (eccentricity $e = 0$), the true anomaly $\nu(t_i)$ is equivalent to the mean anomaly,

$\nu(t_i) = M(t_i)$. To map the satellite's position into three-dimensional space, we utilize the state vector defined by the primary Keplerian orbital elements: the semi-major axis a, inclination ι, Right Ascension of the Ascending Node Ω_p, and the instantaneous true anomaly $\nu(t_i)$. These parameters define the size, tilt, orientation, and position of the orbit within the Earth-Centered Inertial (ECI) frame, which is fixed relative to the celestial sphere.

The transformation process begins in the perifocal coordinate system, a two-dimensional plane where the x-axis points toward the perigee. For a circular orbit, the position vector $\mathbf{r}_{pqw}$ is defined as:

$$\mathbf{r}_{pqw} = \begin{bmatrix} a\cos(\nu(t_i)) \\ a\sin(\nu(t_i)) \\ 0 \end{bmatrix} \tag{3.21}$$

To express this position in the ECI frame, we apply a series of rotations to account for the orbital plane's orientation relative to the Vernal Equinox (Υ). The rotation sequence follows the 3-1-3 Euler angle transformation ($\Omega \to \iota \to \omega$). Since the argument of perigee ω is set to zero for circular orbits, the ECI position vector $\mathbf{p}_{ECI}$ is derived as:

$$\mathbf{p}_{ECI}(t_i) = \mathbf{R}_z(-\Omega_p) \cdot \mathbf{R}_x(-\iota) \cdot \mathbf{r}_{pqw} \tag{3.22}$$

Expanding this matrix multiplication yields the Cartesian coordinates (x, y, z) in the inertial frame:

$$\begin{aligned} x_{ECI} &= a\left(\cos\Omega_p \cos\nu(t_i) - \sin\Omega_p \cos\iota \sin\nu(t_i)\right) \\ y_{ECI} &= a\left(\sin\Omega_p \cos\nu(t_i) + \cos\Omega_p \cos\iota \sin\nu(t_i)\right) \\ z_{ECI} &= a\left(\sin\iota \sin\nu(t_i)\right) \end{aligned} \tag{3.23}$$

The rotation matrix $\mathbf{R}_z(-\Omega_p)$ orients the orbit relative to the Vernal Equinox along the equatorial plane, while $\mathbf{R}_x(-\iota)$ accounts for the tilt of the orbital plane. This inertial vector represents the satellite's coordinate in a frame that does not rotate with the Earth, serving as the prerequisite for the conversion to the Earth-fixed frame to determine geographic coverage. To evaluate terrestrial metrics, specifically for Ground Station visibility and coverage of the RoI, the inertial vector must be transformed into the Earth-Centered Earth-Fixed (ECEF) frame. This is achieved by accounting for the Earth's rotation through the Greenwich Mean Sidereal Time (GMST):

$$\mathbf{p}_{ECEF}(t_i) = \mathbf{R}_z(GMST(t_i)) \cdot \mathbf{p}_{ECI}(t_i) \tag{3.24}$$

where $\mathbf{R}_z(GMST)$ is the rotation matrix about the Earth's celestial axis. The resulting coordinates $\mathbf{p}_{ECEF}(t_i) = [x, y, z]^T$ are stored as the node positions for the graph $G(t_i)$ To

determine which satellites are capable of performing the fire detection task at any given time t_i, the simulation evaluates the geometric relationship between the satellite's sensor footprint and the targeted RoI. This process identifies the set of active sensing nodes $\mathcal{A}(t_i)$.

3.2.2 Active sensing nodes $\mathcal{A}(t_i)$ determination

First, the satellite's position in the ECEF frame $\mathbf{p}_{ECEF} = [x, y, z]^T$ is used to calculate its ground track, or sub-satellite point. This is the coordinate (ϕ_s, λ_s) on the Earth's surfxace directly beneath the satellite:

$$\lambda_s = \arctan 2(y, x), \quad \phi_s = \arctan 2(z, \sqrt{x^2 + y^2}) \tag{3.25}$$

where λ_s is the geodetic longitude and ϕ_s is the geodetic latitude.
To determine if the predefined RoI (r) is viewable by a specific satellite s at time t_i, the simulation performs a localized geometric containment check. This process translates geographic coordinates into a Euclidean plane to avoid the distortions inherent in spherical geometry.

Local projection strategy: For each satellite, an Azimuthal Equidistant projection is dynamically centered at the sub-satellite point (ϕ_s, λ_s). This projection f_{AEQD} [18] maps geodetic coordinates (WGS84 [19]) to a local tangent plane where the satellite's ground track is at the origin $(0, 0)$. This ensures that distances from the satellite's center are preserved accurately in m.

Sensor Swath Modeling: The satellite's sensor capability is represented as a *canonical swath* polygon $\mathcal{P}_{swath}$. In the local AEQD plane, this is modeled as a circular buffer of radius R_{swath} centered at the origin:

$$\mathcal{P}_{swath} = \{(x, y) \mid \sqrt{x^2 + y^2} \leq R_{swath}\} \tag{3.26}$$

Region Transformation and Containment: The geodetic vertices of the RoI $r = \{\mathbf{v}_1, \ldots, \mathbf{v}_4\}$ are transformed into the local coordinate system of the satellite to form the local polygon r_{local}. A satellite is designated as Covered and added to the set $\mathcal{A}(t_i)$ if and only if the swath polygon completely contains the region polygon:

$$s \in \mathcal{A}(t_i) \iff r_{local} \subseteq \mathcal{P}_{swath} \tag{3.27}$$

This vertex-strict containment check ensures that the satellite has a complete, unobstructed view of the entire fire-monitoring zone. In the implementation, this is achieved using the *pyproj* [18] library for the CRS transformation and the *Shapely* library for the geometric `contains` [20] operation. If the condition is met, the satellite is flagged as an active sensing node capable of initiating the distributed fire detection task.

3.2.3 Network Graph $\mathcal{G}(t_i)$ Construction

With the kinematic states and coverage sets established, the simulation constructs the time-varying network graph $G(t_i) = (\mathcal{V}, \mathcal{E}(t_i))$. The edge set $\mathcal{E}(t_i)$ is composed of two distinct link types: ISL and Satellite-to-Ground Link (SGL). Within each orbital plane, every satellite maintains bidirectional connectivity with its two adjacent neighbors. This intra-plane link structure is represented by

$$\mathcal{E}_{intra} = \{(s_{p,r}, s_{p,(r\pm1) \pmod R})\}. \tag{3.28}$$

In addition to same-plane connectivity, each satellite establishes a cross-plane link to the corresponding satellite in the adjacent plane $p+1$, thereby extending the local mesh across neighboring orbital planes.To preserve topological continuity across the orbital seam, where plane $P-1$ reconnects to plane 0, the phasing parameter F is incorporated to account for the along-track offset $\Delta\phi$. Accordingly, the seam-closing connectivity is written as

$$s_{P-1,r} \rightarrow s_{0,(r+F) \pmod R}. \tag{3.29}$$

This deterministic +Grid structure [21] ensures that the graph remains connected between satellites with 4 direct neighbors for each satellite 3.1, providing multiple redundant paths for the transmission of fire detection patches between the primary sensing node and its processing neighbors. The connectivity between the constellation and the single GS g is governed by a geometric visibility constraint. A bidirectional link (s, g) is added to $\mathcal{E}(t_i)$ if the satellite s is above the local horizon of the ground station's geodetic coordinates (ϕ_g, λ_g). Let $\mathbf{p}_s$ be the ECEF position of the satellite and $\mathbf{p}_g$ be the ECEF position of the ground station. The range vector is defined as $\mathbf{r} = \mathbf{p}_s - \mathbf{p}_g$. The elevation angle θ is calculated relative to the local zenith vector $\hat{\mathbf{z}}$ (the normal vector to the ellipsoid at the GS location):

$$\theta = 90^\circ - \arccos\left(\frac{\mathbf{r} \cdot \hat{\mathbf{z}}}{\|\mathbf{r}\|}\right) \tag{3.30}$$

A link is established in the graph if and only if the elevation angle satisfies the minimum threshold $\theta_{\min}$:

$$(s, g) \in \mathcal{E}(t_i) \iff \theta \geq \theta_{\min} \tag{3.31}$$

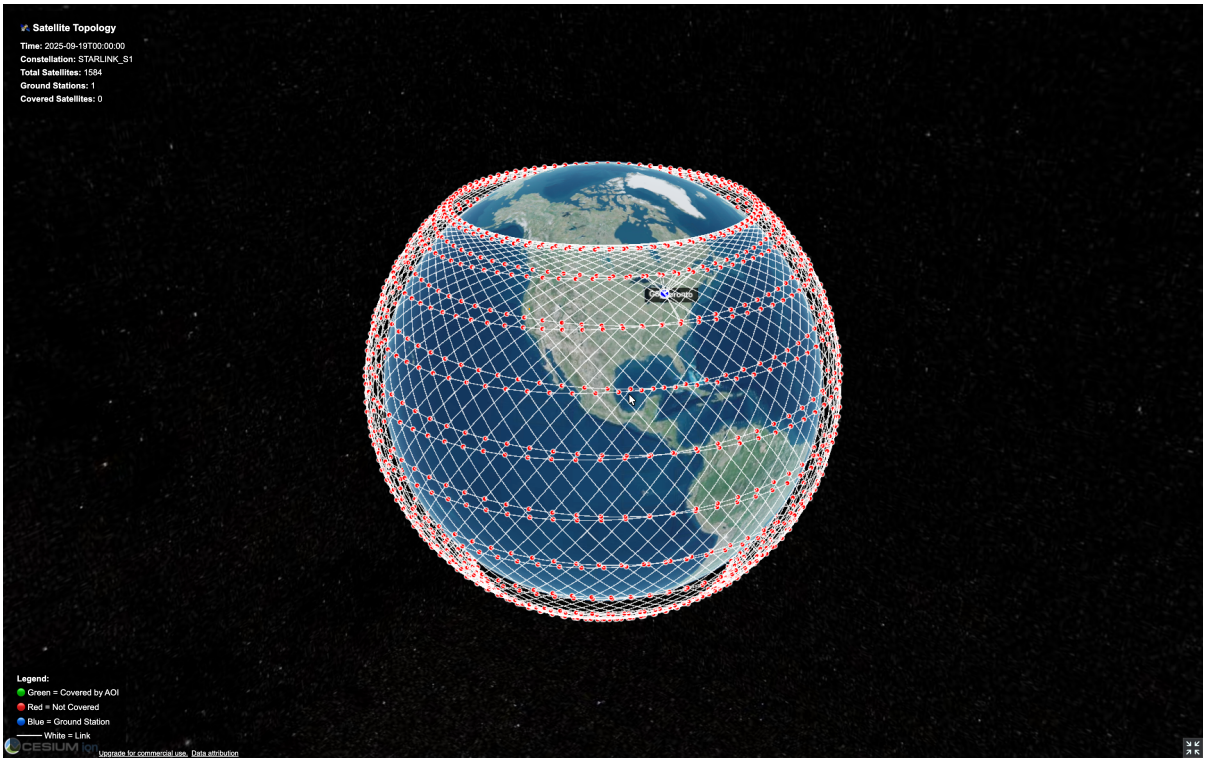

Figure 3.1: Topology visualization for one snapshot of Starlink s1

This SGL edge is critical for the final stage of the mission, where the aggregated fire detection results and compressed masks are downlinked from the satellite network to the ground station for analysis and response.

3.2.4 Network Simulation and Data Plane Modeling

The previously established temporal state $\mathcal{T}_i = \langle t_i, G(t_i), \mathcal{A}(t_i) \rangle$ provides the foundation for the functional data plane simulation. This layer of the framework transitions from geometric kinematics to a discrete-event network model, wherein $G(t_i)$ defines the available communication infrastructure and $\mathcal{A}(t_i)$ serves as the set of task generators. Within this model, each satellite $s \in \mathcal{S}$ is abstracted as an autonomous network entity equipped with dedicated resources for computation, storage, and communication.

The internal architecture of a node integrates autonomous forwarding capabilities with decoupled computational units. Routing decisions are executed via local lookups within

a forwarding table $\mathcal{F}_s \subset \mathcal{F}$, which is derived from the global topology state to determine the optimal egress interface for a packet p destined for a specific ground station g. Concurrently, to accommodate the on-board processing requirements for fire detection, each node possesses a dedicated processing unit. Computational tasks, including pre-processing and inference, are executed in a non-blocking manner relative to the routing engine. This architectural separation ensures that a node can continue forwarding transit traffic while simultaneously processing local RAW data fragments, a critical feature for scenarios where a satellite acts concurrently as a *Primary Node* for sensing and a *Relay Node* for neighboring traffic.

Physical transmission constraints are rigorously modeled to reflect the limitations of communication transceivers. Each outgoing link $(s, v) \in \mathcal{E}(t_i)$ is represented as a unique *SimPy Resource* [22] with a capacity of 1, ensuring that only a single packet may be transmitted over a specific link at any given time. Access to the communication medium is governed by a contention mechanism: upon identifying the next hop v, the node requests the resource associated with the edge (s, v). If the link is occupied, the routing process yields, incurring a queuing delay δ_{queue} until the resource is granted. Once acquired, the link is locked for a duration equal to the transmission delay ω/B_{sv}, effectively serializing physical transfer. This modular node design ensures that the simulation captures the subtle interplay between computational load and networking throughput.

This modular node design ensures that the simulation captures the subtle interplay between computational load and networking throughput, particularly in scenarios where a satellite is simultaneously acting as a *Primary Node* for a sensing task and a *Relay Node* for neighboring traffic.

To characterize the communication and computational constraints within the system, we formalize the key parameters as follows. A sensing mission initiated by a satellite $s \in \mathcal{A}(t_i)$ generates a raw image $\mathcal{M}_{raw}$ of size σ_{raw} bits; upon successful distributed processing, this is reduced to a compressed detection mask $\mathcal{M}_{proc}$ of size σ_{proc}. All data objects are discretized into a set of packets $\mathcal{P}$, where each packet $p \in \mathcal{P}$ carries at most ω bits, i.e., the Maximum Transmission Unit (MTU). Each edge $e \in \mathcal{E}(t_i)$ is assigned a capacity B_e, with three distinguishable classes: B_{intra} for intra-plane ISLs, B_{inter} for inter-plane ISL, and B_{SGL} for SGL. Two stochastic computational latencies complete the parameter set: δ_{pre}, the local pre-processing delay, and δ_{inf}, the inference pipeline delay, both modeled as random variables sampled from empirical distributions within the simulation environment.

When a satellite s enters the active sensing set $\mathcal{A}(t_i)$, it is designated as the *Primary Node* for a specific fire-detection flow and the task execution proceeds through three consecutive stages. First, the Primary Node incurs a delay δ_{pre} to segment the captured scene

and prepare fragments for distribution. Second, it identifies a set of available workers $\mathcal{W}_s \subseteq \text{Neighbors}(s, G(t_i))$; the raw data $\mathcal{M}_{raw}$ is partitioned equally among $\{s\} \cup \mathcal{W}_s$, with each fragment routed to its respective worker as a sequence of packets. Third, upon receiving all required raw fragments, each worker (including the primary) performs inference in parallel. The effective computation delay Δ_{comp} is scaled by the worker count to reflect the distributed U-Net speedup:

$$\Delta_{comp} = \frac{\delta_{inf}}{1 + |\mathcal{W}_s|} \tag{3.32}$$

The resulting processed masks $\mathcal{M}_{proc}$ are then generated and queued for transmission toward the Ground Station g.

The end-to-end latency L for a packet p traversing a time-varying path $\mathcal{K} = (v_0, v_1, \ldots, v_k)$ is the accumulation of hop-by-hop delays within the discrete-event scheduler. For any hop $(u, v) \in \mathcal{K}$, the hop latency is defined as:

$$L_{uv} = \delta_{queue}(u, v) + \frac{\omega}{B_{uv}} + \delta_{prop}(u, v) \tag{3.33}$$

where $\delta_{queue}(u, v)$ is the waiting time in the egress buffer of node u due to link contention, ω / B_{uv} is the deterministic transmission delay governed by packet size and edge capacity, and $\delta_{prop}(u, v) = d(u, v)/c$ is the propagation delay with $d(u, v)$ the Euclidean distance in ECEF space and c the speed of light.

As the constellation state transitions from $\mathcal{T}_i$ to $\mathcal{T}_{i+1}$, the forwarding table $\mathcal{F}$ is updated globally. For any packet p with a destination ground station g, the next hop is selected as:

$$\text{NextHop}(u, g) = \underset{v \in \text{Neighbors}(u, G(t_i))}{\arg\min} \text{dist}(v, g, G(t_i)) \tag{3.34}$$

where $\text{dist}(v, g, G(t_i))$ denotes the shortest-path distance from v to g in the current topology. If an edge (u, v) is removed from $\mathcal{E}(t_i)$ due to orbital dynamics while a packet is queued at node u, the simulator performs an immediate rerouting lookup in the updated graph $G(t_{i+1})$, ensuring the simulation accurately reflects the always-on nature of the Grid-Plus topology despite the high-velocity movement of the nodes.

3.2.5 Data Collection and Performance Metrics

The simulator implements a tiered logging architecture to monitor the lifecycle of every sensing task and the performance of the underlying data plane. Results are captured in structured Parquet format to allow for high-performance post-simulation analysis across three distinct levels of granularity.

Flow-Level Monitoring

At the highest level of abstraction, the system tracks the *End-to-End Flow Completion*. A flow represents the entire lifecycle of a fire-detection mission, starting from the moment a satellite enters the active set $\mathcal{A}(t_i)$ and ending when the final result is delivered to the ground station.

- **Mission Start Time (t_{start}):** The timestamp at which a satellite in $\mathcal{A}(t_i)$ triggers a sensing task and initiates the pre-processing delay δ_{pre}.
- **Mission Finish Time (t_{finish}):** The moment the ground station g receives the final packet p_n required to reconstruct the processed fire mask $\mathcal{M}_{proc}$ for that specific flow.
- **Flow Status:** A mission is marked as *Completed* only if all expected processed packets are received at the ground station.

The primary metric derived here are flow initialization time and flow end time, which will be used to calculate the end-to-end latency and the Age of Information (AoI) for each mission.

Packet-Level and Transmission Event Tracking

For a deeper analysis of packet routing and hop analysis, the simulator provides two additional levels of logging:

1. **Packet Flow Records:** This layer records the trajectory of individual packets. For every packet $p \in \mathcal{P}$, the simulator logs the source node, the final destination node, the total accumulated latency, and the full multi-hop path $\mathcal{K}$ traversed.
2. **Transmission Events:** The most granular level of logging captures every state change in the *SimPy* discrete-event scheduler. Each entry in the TX event log records:
    - **Queue Entry:** The time a packet requests a link resource.
    - **Transmission Start:** The time the resource is granted.
    - **Transmission End:** The time the packet has been fully transferred to the next hop's inbox.

With recorded timestamps of each mission's start and end times, we can calculate the AoI for each flow. Hence with the age of information definition introduced at the begining of this section, we can calculate the AAoI and PAoI for the entire simulation period for each parameter configuration. This allows us to evaluate the timeliness of the system under different constellation designs and operational scenarios.

Chapter 4

Environments Setup

To validate the proposed distributed sensing framework, we implement the simulation using the parameters detailed in this section.

4.1 Constellation and Orbital Parameters

The constellations are categorized by their specific orbital shells, which vary in altitude, inclination, and total satellite density. Each shell is modeled as a Walker Delta or Walker Star pattern, as defined in our kinematic model

Table 4.1: Tested Constellation Configurations $\mathcal{C} = \langle P, R, \iota, h, \theta_{\min}, \mathcal{W} \rangle$

Shell ID	**Planes** (P)	**Sats/Plane** (R)	**Inclination** (ι)	**Altitude** (h)	**Min. Elevation** ($\theta_{\min}$)	**Pattern** ($\mathcal{W}$)
Starlink Phase 1						
starlink_s1	72	22	53.0°	550 km	25.0°	Delta
starlink_s2	20	36	70.0°	570 km	25.0°	Delta
starlink_s3	6	58	97.6°	560 km	25.0°	Delta
starlink_s4	72	22	53.2°	540 km	25.0°	Delta
starlink_s5	4	43	97.6°	560 km	25.0°	Delta
Amazon Kuiper						
kuiper_k1	34	34	51.9°	630 km	35.0°	Delta
kuiper_k2	36	36	42.0°	610 km	35.0°	Delta
Telesat Lightspeed						
telesat_t1	27	13	98.98°	1015 km	10.0°	Delta
telesat_t2	40	33	50.88°	1325 km	10.0°	Delta
OneWeb						
oneweb_o1	36	49	87.9°	1200 km	25.0°	Star
oneweb_o2	32	72	55.0°	1200 km	25.0°	Star
oneweb_o3	32	72	40.0°	1200 km	25.0°	Star

Source: Adapted from [23, 24].

For all Walker Delta configurations, the phasing parameter F is set to 1. This selection is a deliberate design choice to maximize the separation distance between satellites, particularly as orbital planes converge in high-latitude and polar regions. By maintaining $F = 1$, we theoretically achieve no collision between satellites and congestion in the polar areas.

4.2 Terrestrial Setup: RoI and Ground Stations

To evaluate the the system's ability to deliver critical fire detection result from space to ground, 1 area of predefined high risk RoI and 4 global GS locations are included in the analysis.

4.2.1 Region of Interest

The monitoring target is modelled after the 2018 Camp Fire in California. The RoI is defined as a geodetic polygon with the following geographic bounds:
Camp Fire AoI: A rectangular region in Butte County, CA, bounded by latitudes $[39.636°\text{N}, 39.910°\text{N}]$ and longitudes $[121.642°\text{W}, 121.365°\text{W}]$.

4.2.2 Ground Stations

We tested the constellation against four globally distributed ground stations. These stations serve as the final sinks for all processed fire masks generated by the satellite network. Their coordinates are summarized in Table 4.2.

Table 4.2: Geodetic Coordinates of Evaluated Ground Stations

Station Name	**Latitude (ϕ_g)**	**Longitude (λ_g)**
Toronto	43.6532° N	79.3832° W
London	51.5074° N	0.1278° W
San Francisco	37.7749° N	122.4194° W
Singapore	1.3521° N	103.8198° E

4.3 Edge Computing Workload and Benchmark

In the system model, several parameters are defined to emulate the computational workload of fire detection algorithm and the communication workload of the transmission of the raw scene image and the processed fire masks. To ensure the realism of our simulation, a real scene image from the Landsat8 satellite is used as the unprocessed data for the algorithm.

4.3.1 Image Data

The primary data source for the fire detection pipeline is a scene picture from the Landsat 8 Operational Land Imager [25], the specific image used has the Landsat Product ID of `LC08_L1TP_046031_20200908_20200908_01_RT`. This image captures a particular region in Northern California on September 8, 2020, which is during the peak of the wildfire season. The scene consists of 10 spectral bands with a dimensions of 7811 * 7931 pixels.

4.3.2 GDAL Preprocessing and Runtime Environment

The preprocessing splitting the image into patches of dimension 256 * 256 was executed on Compute Canada Nibi Cluster on node `c465`, provisioned with 8 CPU cores and 32 GB RAM. The implementation environment consists of Bash and GDAL command-line utilities (`gdalinfo`, `gdal_translate`, and `gdal_retile.py`). The operation is performed 100 times to obtain a stable estimate of the gdal runtime, which serves as a heuristic for the parameter

δ_{pre} introduced before. The average runtime for the preprocessing step 49.62 s, with a standard deviation of 12.28 s. This runtime includes the time taken to read the original image, split it into patches, and save the processed patches to disk. The variability in runtime can be attributed to factors such as disk I/O performance and CPU load on the cluster node during execution.

4.3.3 U-NET Active Fire Detection Runtime Environment

After the GDAL preprocessing, aggregate size of the raw scene patches $\mathcal{M}_{raw}$ is 828 MB. This represents the total payload that must be distributed among the worker nodes $\mathcal{W}_s$ in the LEO constellation. Then the inference benchmarks were conducted on the same config on Compute Canada Nibi Cluster, with additional of a high-performance NVIDIA H100 (80GB HBM3) GPU. To model the multi-tenant or partitioned nature of satellite edge-computing resources, the experiments were executed on a Multi-Instance GPU (MIG) [26] slice, specifically the `3g.40gb` profile. The pipeline was implemented using the PyTorch framework, deploying a optimized U-Net architecture specifically tuned for multispectral Landsat 8 data. The inference operation was repeated 100 times, the same as the preprocessing step, yielding an average δ_{inf} of 43.27 s and a standard deviation of 9.00 s.

4.4 Network Configuration and Link Bandwidth

The network infrastructure is modeled after modern LEO constellation standards, focusing on high-speed laser ISL and SGL. Both intra-plane and inter-plane ISL bandwidth are set to a reasonable capacity 10 Gbps to reflect current optical wireless communication terminals [24]. The SGL bandwidth for downlinking to ground stations is modeled at a more constrained 100 Mbps [27]. To reduce the computation load for the simulator, 65,535 bytes, the size of the maximum payload of an IPv6 packet is adopted as the MTU. Hence making the simulator still resemble a real network.

Chapter 5

Experimental Results and Analysis

This section presents the performance evaluation of the proposed method across a wide range of orbital and network configurations. In total, 720 discrete simulation trails are performed, defined by the Cartesian product of the following sets of parameters:

- **12 Constellation Shells**
- **4 Ground Stations**
- **5 Swath Radius**
- **3 Simulation Time Intervals**

5.1 Overall AoI performance of different constellations

To evaluate the operational impact of orbital design on the AoI performance, Table 5.1. presents the comparison of the AoI performance of 12 constellation shells on the swath radius of 500 km and average across 4 ground stations and 3 simulation time intervals.

Table 5.1: Comparison of AoI Across Constellation Shells ($Swath = 500$ km). Results are averaged across 4 ground stations and 3 simulation timesteps (10 s, 20 s, 30 s) over one day window.

Constellation Shell	**Average AoI (s)**	**Peak AoI (s)**
Starlink		
starlink_s1	67.1	99.3
starlink_s2	268.9	1,419.3
starlink_s3	2,697.0	10,358.0
starlink_s4	66.8	99.3
starlink_s5	6,318.6	19,440.0
Amazon Kuiper		
kuiper_k1	66.5	76.0
kuiper_k2	3,914.6	12,971.0
Telesat		
telesat_t1	192.2	456.0
telesat_t2	66.5	76.0
OneWeb		
oneweb_o1	66.5	78.0
oneweb_o2	156.5	1,895.8
oneweb_o3	704.0	4,394.2

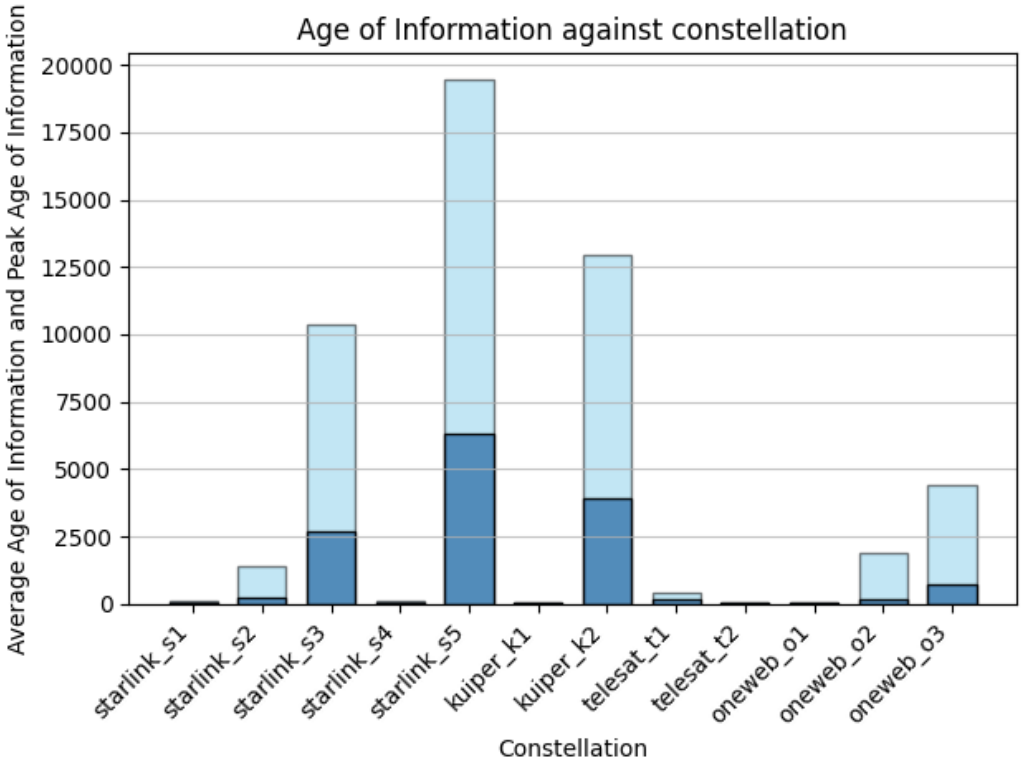

Figure 5.1: AoI performance comparison across constellation shells at swath radius of 500 km

We observe a substantial variation on the AoI performance across different constellation shells, with the average AoI ranging from 66.5 s to 6,318.6 s. The best performing constellation shells (starlink_s1, starlink_s4, kuiper_k1, telesat_t2, and oneweb_o1) all have an average AoI of around 66.5 s, while the worst performing constellation shell (starlink_s5) has an average AoI of 6,318.6 s, which is almost 100 times worse than the best performing shells.

To further understand the reasons different constellations affect AAoI, we analyzed the relationship between P, R and the resulting AAoI. By isolating the 10 unique configurations representing current major real-world constellation, we get a 2D topology map of AAoI performance as a function of P and R as shown in Figure 5.2.

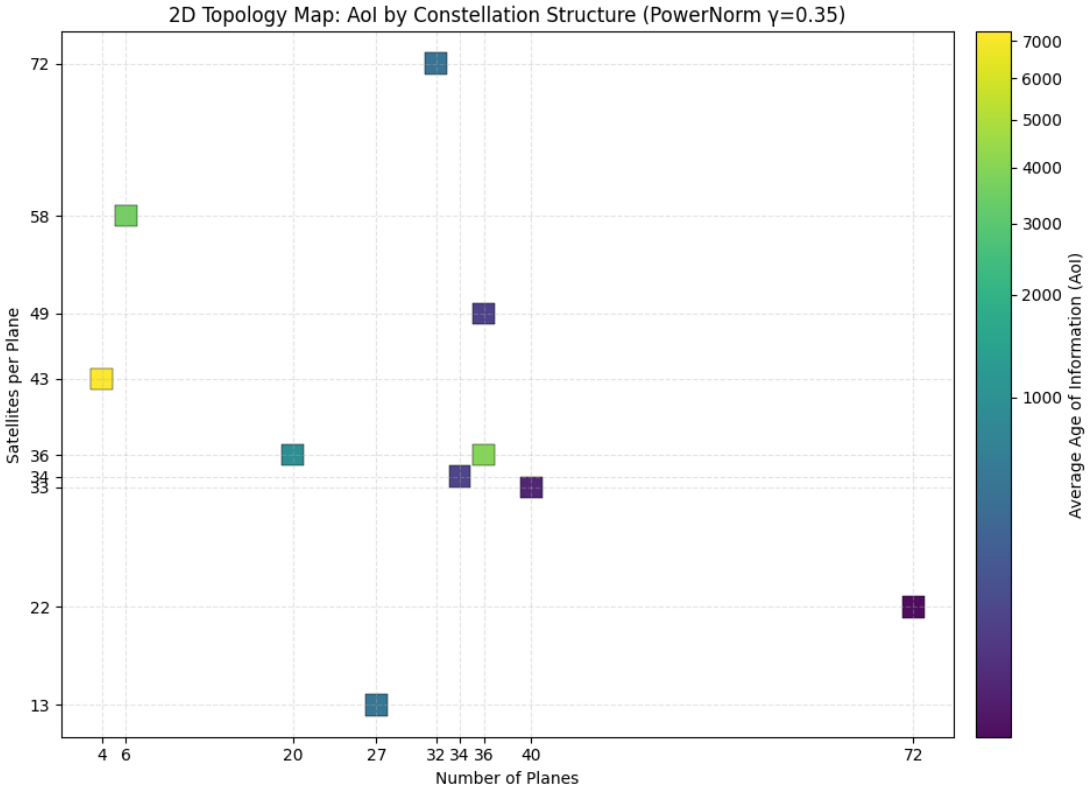

Figure 5.2: AoI performance topology as a function of constellation parameters P and R. Each point represents a unique constellation shell configuration, with color indicating the average AoI (darker colors indicate higher AoI).

Under the current simulation setting, the topology pattern indicates that increasing the number of orbital planes (P) has a stronger effect in reducing AAoI than simply increasing the number of satellites per plane (R). Configurations with larger P tend to cluster at lower AAoI levels even when R is moderate, suggesting that improving longitudinal/temporal revisit diversity contributes more to lowering AoI than adding satellites along the same plane.

5.2 AAoI performance against swath radius

To evaluate the impact of swath radius on AAoI performance, we compare the AAoI performance of the different constellation shells across 5 swath radius values (i.e., 100

km, 200 km, 300 km, 400 km, and 500 km) and average across 4 ground stations and 3 simulation time intervals. The selected constellation shells for this comparison all exhibit good AAoI performance at the swath radius of 500 km.

Table 5.2: Impact of Swath Radius on Average Age of Information across Representative Constellation Shells. Values are recorded in s.

Constellation Shell	**100 km**	**200 km**	**300 km**	**400 km**	**500 km**
starlink_s1	233.0	116.0	89.0	73.0	67.0
starlink_s4	221.0	112.0	86.0	72.0	66.0
kuiper_k1	473.0	124.0	83.0	68.0	66.0
telesat_t2	267.0	110.0	77.0	66.0	66.0
oneweb_o1	432.0	135.0	82.0	69.0	66.0

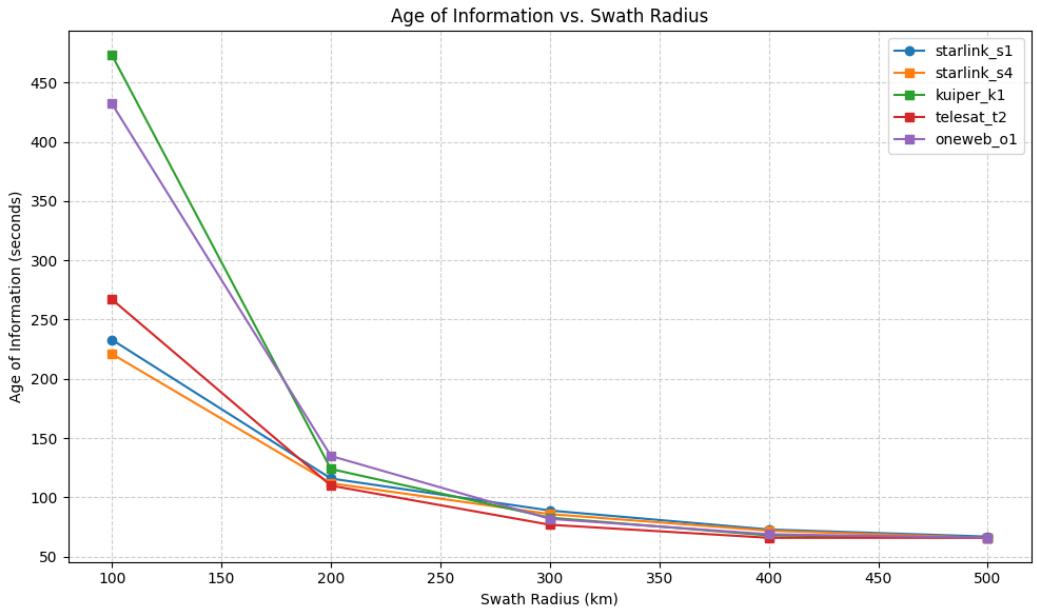

Figure 5.3: AoI against swath radius

For these selected configurations, the AAoI converges toward an asymptotic floor of approximately 66–67 s. This threshold corresponds closely to the cumulative latency of the image pre-processing, inference phases, and time interval overhead. This confirms

that for these specific orbital shells, the AAoI performance is no longer limited by the satellite revisit interval after 400 km or 500 km, but rather by the on-board computational and data-handling delay. Conversely, at a constrained swath radius of 100 km, the AAoI performance is significantly degraded even for those selected constellation shells, with an average ranging from 221 to 473 s across the selected shells. This disparity indicates that at a smaller swath radius, AoI is primarily dictated by orbital dynamics. Moreover, the relationship between swath radius and AAoI is characterized by a distinct non-linear decay. While the swath radius increases linearly, the corresponding reduction in AAoI exhibits diminishing marginal returns. The most substantial performance gains are realized between 100 km and 300 km; beyond this point, the curves begin to flatten significantly.

5.3 AoI performance against time intervals

A critical factor affecting the timeliness of fire detection data is the frequency at which sensing tasks are initiated within the simulation. This frequency, ultimately adjusted by the simulation time interval (Δt), determines how often the constellation identifies new sensing opportunities and updates its network routing state. Analyzing this parameter is essential to understand whether the current system performance is limited by the physical network capacity or by the granularity of the task scheduling itself.

As shown in Table 5.3, the simulation results across high-density shells demonstrate a nearly linear relationship between the reduction in task intervals and the improvement in AAoI.

Table 5.3: Sensitivity of AAoI (s) to Simulation Time Intervals ($R_{swath} = 500$ km).

Constellation Shell	**30 s Interval**	**20 s Interval**	**10 s Interval**
starlink_s1	74.0	67.3	60.1
starlink_s4	73.1	66.8	59.8
kuiper_k1	73.0	66.5	59.5
telesat_t2	72.9	66.5	59.5
oneweb_o1	72.8	66.5	59.4

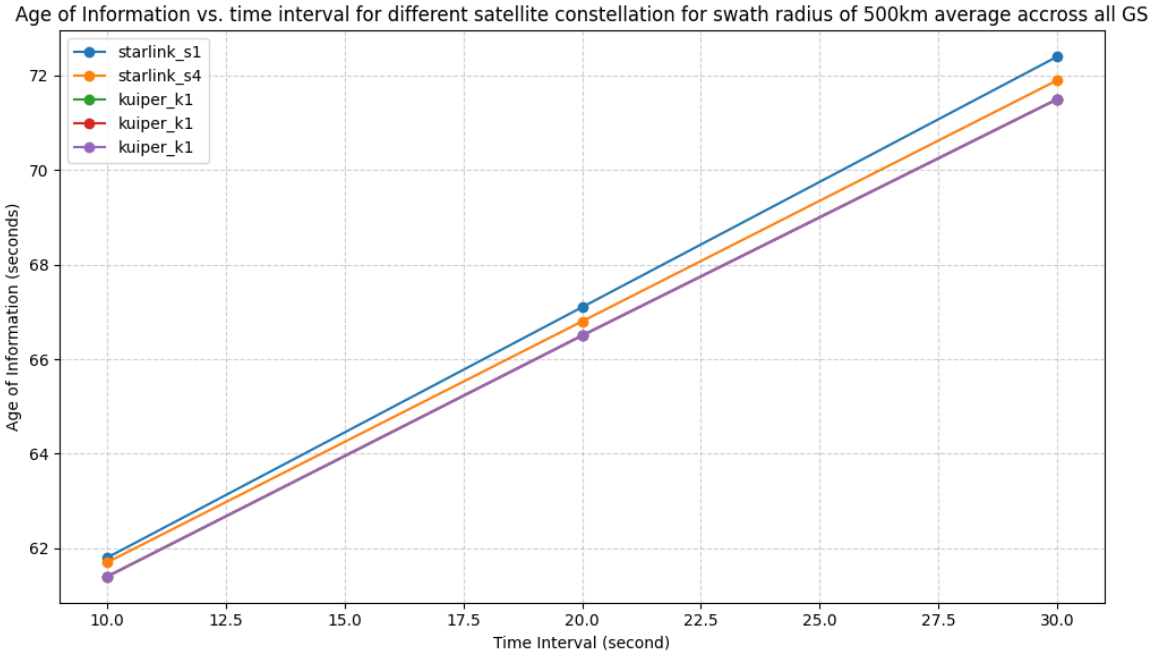

Figure 5.4: AoI performance against time intervals.

This linear trend suggests that at these intervals, the satellite network bandwidth is not the primary bottleneck. If the 10 Gbps inter-satellite links were reaching saturation, shortening the interval would no longer yield significant improvements in AoI. Instead, the consistent decrease in AAoI indicates that the system is highly responsive to more frequent task initiations and can efficiently handle the data volume across the available links.

5.4 Coverage Probability and Age of Information

While the internal network bandwidth is not a primary bottleneck, the system's performance will be heavily influenced by the temporal resolution of the sensing tasks. This suggests that the timeliness of a fire detection update is inextricably linked to how frequently the constellation can geographically observe the target area. To quantify this relationship, we move from analyzing network intervals to evaluating Coverage Probability—a metric representing the percentage of total simulation time intervals in which at least one satellite in the constellation has the RoI within its sensor swath. In our 24 h simulation window, the Coverage Probability is calculated as the ratio of "active" intervals (where the RoI is

covered) to the total number of simulation steps. To analyze the impact of this temporal visibility, we performed a cross-constellation comparison mapping this probability against the resulting AAoI for all 720 simulation trials.

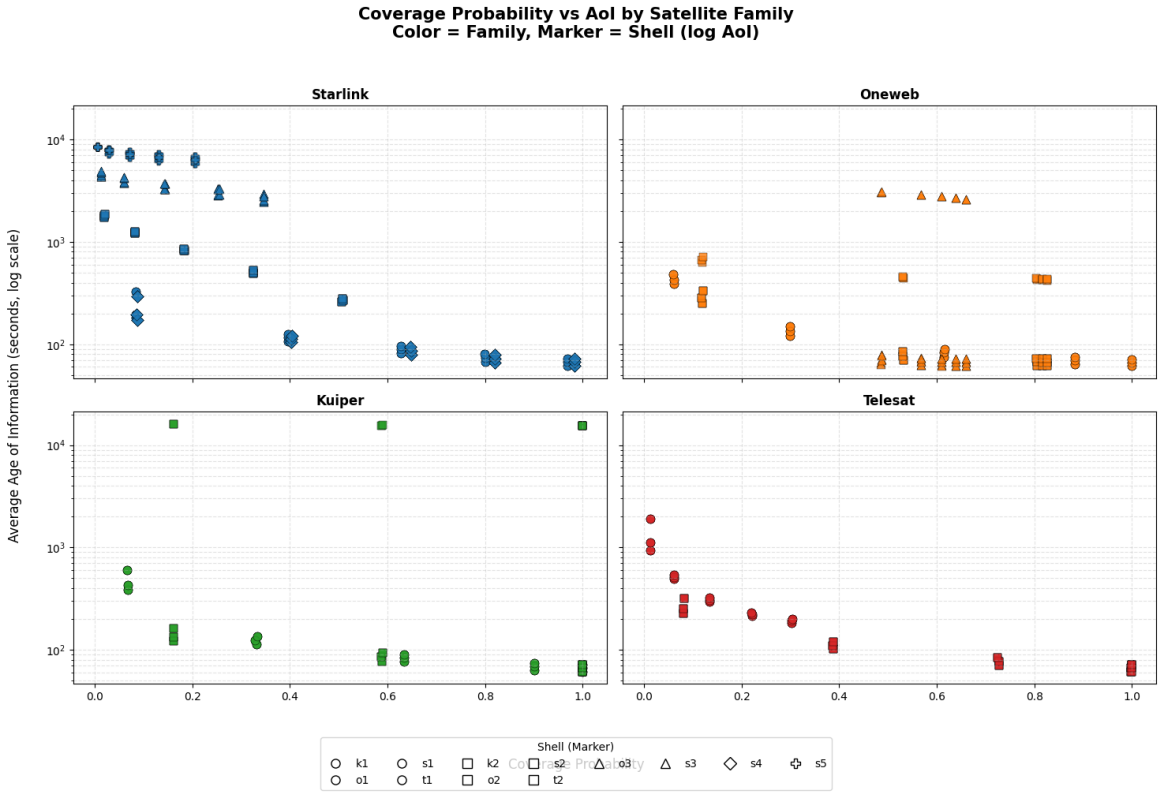

Figure 5.5: AoI performance against Coverage Probability.

The data, categorized by satellite family and individual orbital shells in Figure 5.5, reveals an inverse relationship between these two metrics. As the percentage of intervals with active coverage increases toward 100%, the AAoI drops across all constellation families. Another key observation from the scatter analysis is the behavior of the data as it reaches 100% coverage probability. For all data points where the target is visible in every simulation interval, there is a significantly smaller spread in the AAoI values compared to lower coverage levels. This reduction in variance marks a fundamental shift in system behavior as the elimination of revisit gaps removes the primary source of uncertainty. At this stage, the system transitions from a stochastic state to a more deterministic one, where the performance is no longer a game of chance based on satellite positions but is instead

governed by the fixed, predictable latencies of the hardware, network, and time interval.

Chapter 6

Conclusion

In this study a simulated orbital edge-computing architecture in real-world LEO constellations is demonstrated to capably provide information freshness suitable for time-critical wildfire monitoring. Using a simulation framework that integrates orbital dynamics, distributed U-Net inference, and time-varying ISL routing, and grounding computation delays with empirical HPC measurements, the constellation design strongly governs AoI performance: across 12 shells, average AoI ranges from 66.5 s to over 6,300 s, while the best configurations achieve sub-70 s average AoI and sub-100 s peak AoI. The results further indicate that increasing orbital-plane count and improving coverage probability are more decisive for timeliness than increasing satellites per plane, and that once near-continuous coverage is reached, performance becomes primarily limited by onboard processing and scheduling granularity rather than link bandwidth. In addition, the consistent AoI reduction observed when shortening the task interval (30 s to 10 s) suggests that, within the tested regime, inter-satellite links are not operating under saturation-limited conditions. These findings support the feasibility of LEO-based distributed sensing for near-instantaneous environmental alerting and motivate future work on multi-ground-station architectures, heterogeneous onboard compute allocation, and adaptive task scheduling under realistic operational constraints.

References

[1] James MacCarthy et al. “The Latest Data Confirms: Forest Fires Are Getting Worse”. en. In: (July 2025). (Visited on 03/27/2026).

[2] U. N. Environment. *Spreading like Wildfire: The Rising Threat of Extraordinary Landscape Fires — UNEP - UN Environment Programme*. en. Section: publications. Feb. 2022. URL: https://www.unep.org/resources/report/spreading-wildfire-rising-threat-extraordinary-landscape-fires (visited on 03/27/2026).

[3] Yang Chen, Douglas C. Morton, and James T. Randerson. “Remote sensing for wildfire monitoring: Insights into burned area, emissions, and fire dynamics”. In: *One Earth* 7.6 (June 2024), pp. 1022–1028. DOI: 10.1016/j.oneear.2024.05.014. (Visited on 03/27/2026).

[4] Yu Zhao and Yifang Ban. “GOES-R Time Series for Early Detection of Wildfires with Deep GRU-Network”. en. In: *Remote Sensing* 14.17 (Jan. 2022), p. 4347. DOI: 10.3390/rs14174347. (Visited on 03/27/2026).

[5] Michael A. Wulder et al. “Current status of Landsat program, science, and applications”. In: *Remote Sensing of Environment* 225 (May 2019), pp. 127–147. DOI: 10.1016/j.rse.2019.02.015. (Visited on 03/27/2026).

[6] Federico Frassy et al. “Optical Remote Sensing for Precursor Landslide Events: Two Case Studies”. In: *IGARSS 2024 - 2024 IEEE International Geoscience and Remote Sensing Symposium*. ISSN: 2153-7003. July 2024, pp. 3774–3778. DOI: 10.1109/IGARSS53475.2024.10641351. (Visited on 04/01/2026).

[7] Bradley Denby and Brandon Lucia. “Orbital Edge Computing: Nanosatellite Constellations as a New Class of Computer System”. In: *Proceedings of the Twenty-Fifth International Conference on Architectural Support for Programming Languages and Operating Systems*. ASPLOS ’20. New York, NY, USA: Association for Computing Machinery, Mar. 2020, pp. 939–954. DOI: 10.1145/3373376.3378473. (Visited on 03/27/2026).

[8] Baozhen An et al. “Age of Information for Joint Sensing, Transmission and Computation in Mobile Edge Computing”. In: *2024 IEEE 100th Vehicular Technology Conference (VTC2024-Fall)*. ISSN: 2577-2465. Oct. 2024, pp. 1–5. DOI: 10.1109/VTC2024-Fall63153.2024.10757801. (Visited on 04/14/2026).

[9] Zhifeng Tang et al. “Age of Information Analysis of Multi-user Mobile Edge Computing Systems”. In: *2021 IEEE Global Communications Conference (GLOBECOM)*. Dec. 2021, pp. 1–6. DOI: 10.1109/GLOBECOM46510.2021.9685769. (Visited on 04/14/2026).

[10] Changhao Li et al. “Delay-cost computation offloading for on-board emergency tasks in LEO Satellite Edge Computing networks”. In: *Future Generation Computer Systems* 169 (Aug. 2025), p. 107797. DOI: 10.1016/j.future.2025.107797. (Visited on 04/14/2026).

[11] Saumya Jaipuria, Ansuman Banerjee, and Himadri Sekhar Paul. “Distributed Sensing and Collaborative Edge Computing for LEO Constellations”. In: *2025 17th International Conference on COMmunication Systems and NETworks (COMSNETS)*. ISSN: 2155-2509. Jan. 2025, pp. 1037–1041. DOI: 10.1109/COMSNETS63942.2025.10885635. (Visited on 04/15/2026).

[12] Antonio Mercado-Martínez, Beatriz Soret, and Antonio Jurado-Navas. “Goal-Oriented Vessel Detection with Distributed Computing in a LEO Satellite Constellation”. In: *2024 6th International Conference on Communications, Signal Processing, and their Applications (ICCSPA)*. 2024, pp. 1–5. DOI: 10.1109/ICCSPA61559.2024.10794283.

[13] Roberto Del Prete et al. “Enhancing Maritime Situational Awareness Through End-to-End Onboard Raw Data Analysis”. In: *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 18 (2025), pp. 16997–17018. DOI: 10.1109/JSTARS.2025.3584999. (Visited on 04/15/2026).

[14] Mrityunjoy Gain et al. “LEO Satellite Oriented Wildfire Detection Model Using Deep Neural Networks: A Transfer Learning Based Approach”. In: *2024 6th International Conference on Electrical Engineering and Information & Communication Technology (ICEEICT)*. ISSN: 2769-5700. May 2024, pp. 214–219. DOI: 10.1109/ICEEICT62016.2024.10534509. (Visited on 04/15/2026).

[15] Yanwu Lu et al. “Analysis of Age of Information in Non-terrestrial Networks”. In: *2023 IEEE Globecom Workshops (GC Wkshps)*. 2023, pp. 239–244. DOI: 10.1109/GCWkshps58843.2023.10465028.

[16] Natural Resources Canada. *Satellite Characteristics: Orbits and Swaths*. eng. Last Modified: 2025-01-08. Jan. 2008. URL: https://natural-resources.canada.ca/maps-tools-publications/satellite-elevation-air-photos/satellite-characteristics-orbits-swaths (visited on 03/16/2026).

[17] Gabriel Henrique de Almeida Pereira et al. "Active fire detection in Landsat-8 imagery: A large-scale dataset and a deep-learning study". In: *ISPRS Journal of Photogrammetry and Remote Sensing* 178 (Aug. 2021), pp. 171–186. DOI: 10.1016/j.isprsjprs.2021.06.002. (Visited on 12/10/2025).

[18] *Transformer - pyproj 3.7.2 documentation*. URL: https://pyproj4.github.io/pyproj/stable/api/transformer.html (visited on 03/16/2026).

[19] *NGA Geomatics - WGS 84*. URL: https://earth-info.nga.mil/index.php?dir=wgs84&action=wgs84 (visited on 03/16/2026).

[20] *shapely.contains — Shapely 2.1.2 documentation*. URL: https://shapely.readthedocs.io/en/2.1.2/reference/shapely.contains.html (visited on 03/16/2026).

[21] Joseph Mclaughlin, Jee Choi, and Ramakrishnan Durairajan. "×Grid: A Location-oriented Topology Design for LEO Satellites". In: *Proceedings of the 1st ACM Workshop on LEO Networking and Communication*. LEO-NET '23. New York, NY, USA: Association for Computing Machinery, Oct. 2023, pp. 37–42. DOI: 10.1145/3614204.3616110. (Visited on 03/16/2026).

[22] *simpy.resources — Shared resource primitives — SimPy 4.1.2.dev8+g81c7218 documentation*. URL: https://simpy.readthedocs.io/en/latest/api_reference/simpy.resources.html (visited on 03/16/2026).

[23] Wenyi Zhang, Zihan Xu, and Sangeetha Abdu Jyothi. *An In-Depth Investigation of LEO Satellite Topology Design Parameters*. arXiv:2402.08988 [cs]. Sept. 2024. DOI: 10.48550/arXiv.2402.08988. URL: http://arxiv.org/abs/2402.08988 (visited on 02/10/2026).

[24] Nils Pachler et al. "An Updated Comparison of Four Low Earth Orbit Satellite Constellation Systems to Provide Global Broadband". In: *2021 IEEE International Conference on Communications Workshops (ICC Workshops)*. June 2021, pp. 1–7. DOI: 10.1109/ICCWorkshops50388.2021.9473799. (Visited on 11/03/2025).

[25] Earth Resources Observation and Science (EROS) Center. *Landsat 8 OLI/TIRS Collection 2 Level-1: LC08_L1TP_046031_20200908_20200908_01_RT*. EarthExplorer. Path 46, Row 31; Acquired 2020-09-08; Accessed 2026-03-18. Sept. 2020. URL: https://earthexplorer.usgs.gov/.

[26] *Multi-Instance GPU - Alliance Doc*. URL: https://docs.alliancecan.ca/wiki/Multi-Instance_GPU (visited on 03/18/2026).

[27] *Starlink*. en. URL: https://starlink.com (visited on 03/18/2026).