F A R O S: Autonomous Multistrom Cluster of Satellites with Dynamic Rearrangement

Athanasia Nikolova

June 2026

1 Abstract

The design and visualization of an autonomous, independent satellite communication network for space objects identification will embrace cargo transfers and human missions in space. The study investigated the creation of a multistrom cluster of satellites with dynamic rearrangement and the establishment of bidirectional communication between satellites and the earth. Model simulations of satellite networks in several orbits were developed in Python. A 3D Earth interactive representation program to increase and decrease number of orbits, change altitude, and configure number of satellites was created. It incorporated Dijkstra's algorithm to find the shortest path for communication between satellites. Based on the simulations, it was found that the backbone is more effective in space because it reduces latency, offers more stability, and prevents natural phenomena and the physical environment from affecting it. As a result, satellite systems would be less exposed natural catastrophes, countries will be interconnected because of global access to the network of space, and space will be endorsed by a decentralized model of space governance.

2 Introduction

Traffic in space constitutes a serious and growing problem. Today, more than 10,000 satellites exist in space [17], and humanity has not even been a century since the first successful launch of satellite on October 4, 1957. In the next 20 years, with the technological pace continuing to advance, near space will become congested by a million satellites and human-created objects are pending being placed in orbit.

Space presents less natural obstacles in communication compared to Earth. This research utilizes distance as an important factor in calculations for automatic re-

arrangement aiming communication speed. In communications there are multiple information exchanges between data processing systems. For example, for the dispersion and assembly of data, response time plays an important role. If there is intense movement in a region of space during data assembly, this entails automated rearrangement of communication satellites (Access Points) and data satellites (Data Points). This cannot be applied on Earth because Datacenter cannot reduce the physical distance between them. That is, Backbones' development in Space consist of moving Datacenter.

Examining time response, this research showcased faster communication when all systems existed in space. CPU, memory, and storage are affected by response's speed in a decentralized network. The tests further unveiled that different systems such as Data Point could improve their time participating in the transmission. This brought the idea for the development of space objects - networks exchanging roles dynamically.

Python visualizations contribute to the idea's virtual implementation. Backbones' modules rearrangement is achieved using the Fibonacci sphere concept by providing evenly spaced modules without allowing updates or a different number of modules to disrupt the network. Simulating this idea using NumPy[9] and Matplotlib[10], it is ensured that such a network would be doable in space.

Research question

"How realistic is it to develop near-space infrastructure in all orbits connecting every hop - satellite - through radio-based network at the altitudes of 500 km - 36000 km totally independent from the Earth as basis of a decentralized model for space governance? Specifically, in terms of latency figure 5, redundancy, altitude, and least number of orbits, backbones, and satellites with ultimate purpose the shortest path for data migration using Dijkstra's algorithm."

3 Theory

Satellite motion relies on ideas of how orbits work, what forces shape them, and how satellites move between different orbital paths. Kepler's laws and Newton's universal gravitation define the basic behavior of orbital motion, while orbit classifications outline the environments satellites operate in. Additionally, transfer orbits explain how satellites shift between orbits, and broader gravitational ideas such as event horizons and repulsive forces help frame the limits and variations of gravitational influence in more extreme contexts. These concepts form a framework for analyzing and designing satellite trajectories.

3.1 Orbital Mechanics

Kepler's laws show that satellites follow elliptical paths, move faster when closer to Earth, and have orbital periods determined by the size of their orbit. These make it possible to predict satellite positions, and understand variations in orbital speed.

1. Law of Ellipses: Planets move in elliptical orbits with the Sun at one focus, not perfect circles.
2. Law of Equal Areas: A line from the Sun to a planet sweeps out equal areas in equal times, meaning planets speed up when closer to the Sun and slow down when farther away.
3. Law of Harmonies: The square of a planet's orbital period is proportional to the cube of its semi-major axis.

$$v = \frac{\Delta x}{\Delta t} \tag{1}$$

$$\dot{A} = \frac{dA}{dt} \tag{2}$$

$$T^2 \approx a^3 \tag{3}$$

$$T^2 \approx ka^3 \tag{4}$$

$$T \approx ka^{1.5} \tag{5}$$

$$k = \frac{yr^2}{au^3} \tag{6}$$

where: v = velocity, Δx = change in position, Δt = change in time, $\dot{A}$ = rate at which area is swept out, $\frac{dA}{dt}$ =instantaneous rate of change of area with respect to time, T = orbital period, a = semi-major axis of the orbit, k = proportionality constant that depends on the central body, $au = 150,000,000km$.

3.2 Newton's law of universal gravitation

Newton's law quantifies the gravitational attraction between Earth and a satellite, and provides the basis for calculating orbital velocity required for a moving object to stay in orbit.[14]

$$F_G = G\frac{Mm}{r^2}$$

where: F_G = Gravitational Force, G = Gravitational Constant $\approx 6.674 \times 10^{-11}$, M = Mass of the first object, m = Mass of the second object, r = Distance between the centers of the two masses

3.3 Kinds of orbits

Orbit classification defines the operational characteristics of different orbital regions. Factors such as coverage, and communication delay are tremendously important for allocation of satellites in different altitudes for the best performance.

In orbital mechanics, orbits are classified as either bound (closed), taking the shape of a circle or an ellipse, or unbound (open), which form a parabola or a hyperbola. The specific shape is dictated by the orbit's eccentricity (e).This motion depends on the total mechanical energy (E), which is the sum of kinetic energy (K) and potential energy (U). $E = K + U = \frac{1}{2}mv^2 - G\frac{Mm}{r} = 0$.
The escape velocity v_e is the minimum speed an object needs to break free from a celestial body's gravity without further propulsion. At this threshold, the total energy equals zero, meaning the kinetic energy perfectly balances the gravitational potential energy. $v_e^2 = \frac{2GM}{r} \Rightarrow v_e = \sqrt{\frac{2GM}{r}} = \sqrt{2}v$ where: v is the velocity an objects has in circular orbit.

Figure 1: e defines how "stretched" a shape is, ranging from 0 (circle) to 1 (parabola) and > 1 (hyperbola), with higher values representing greater elongation[11].

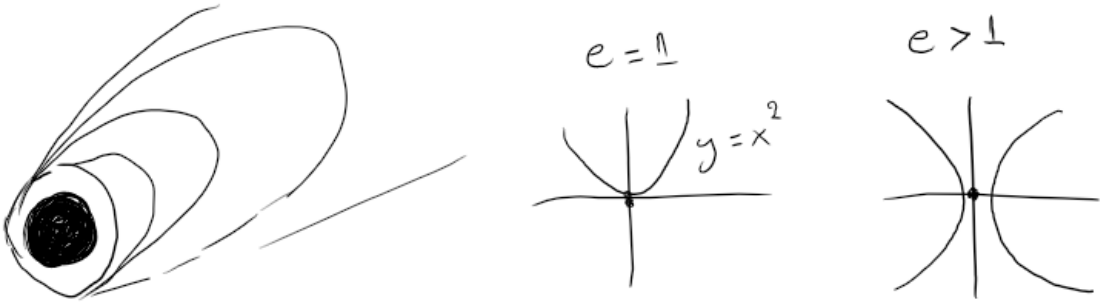

Figure 2: An escape orbit is followed when an object's velocity at a given distance from a central body meets or exceeds the local escape velocity, producing an open trajectory with eccentricity $e \geq 1$; because escape velocity depends on altitude and on the mass ratio $m < M$, smaller mass is treated as negligible in the calculation.

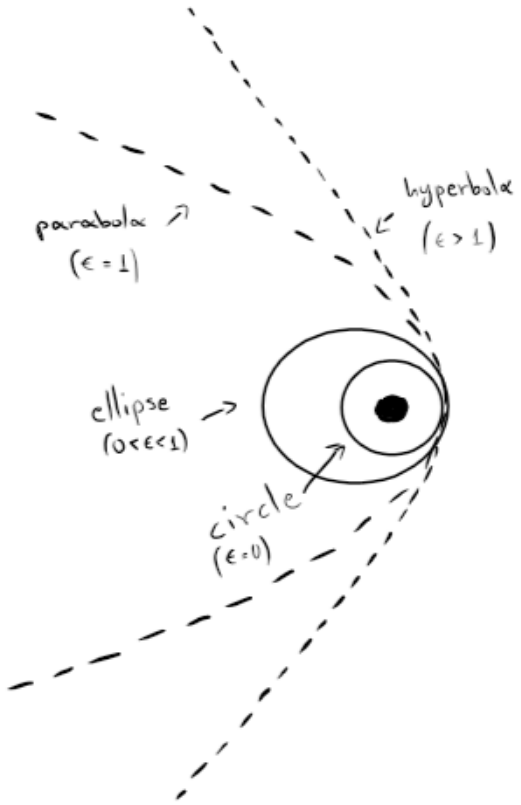

Figure 3: Bound orbits are closed, repeating paths (circles or ellipses) where an object is trapped by gravity, having negative total energy. Unbound orbits are open-ended, non-repeating paths (parabolas or hyperbolas) where an object has enough energy to escape the central body, having zero or positive total energy[18].

3.4 Transfer Orbits

After launch, a spacecraft or satellite is inserted into an initial injection orbit[4], typically a low-altitude circular or slightly elliptical orbit determined by the launch vehicle. From this initial orbit, the satellite must be transferred to its designated operational orbit. This transition is achieved through a transfer orbit, which is an intermediate elliptical trajectory connecting two circular orbits of radii r_1 and r_2[3]. Transfer Orbit Semi-Major Axis equation: $a_{trans} = \frac{r_1+r_2}{2}$. Total Time of Flight (Half Period): $t_{trans} = \pi\sqrt{\frac{a_{trans}^3}{\mu}}$. In these formulas, r_1 is the initial orbit radius, r_2 is the final orbit radius, and μ is the gravitational parameter (G x M) of the central body.

3.5 Horizon

Because Earth is a sphere, an observer at altitude h above the surface can only view a finite portion of the planet before the curvature blocks the line of sight. The boundary of this visible region is known as the horizon[2]. In order to cover the entire Earth without getting horizon's constraints, hops are placed around the globe filling all missing points. The higher the hop is placed, the more horizon it covers. The horizon angle is determined by the geometry between the Earth's radius R, the observer's altitude h, and the central angle ϕ between the observer and the tangent point on the Earth's surface. This relationship is expressed through the equation:

$$S = \phi R \times arc\cos\frac{R}{R+h}$$

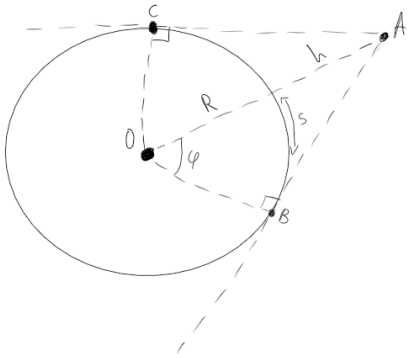

Figure 4: S (Arc Length): The distance you would have to travel along the Earth's surface to reach the horizon point. – R(Radius): The radius of the Earth (approx. 6,371 km or 3,959 miles). – h(Height): The observer's height above sea level. – arccos $\frac{R}{R+h}$: The distance from the center of the Earth to the observer's eyes.

3.6 Repulsive Force

In many spatial distribution and network formation problems, it is useful to describe the interaction between entities through a distance-dependent repulsive force. A formulation for such a force is

$$F = \frac{1}{d^N}$$

$F =$ Force magnitude

$d =$ Distance between two points

$N =$ Exponent controlling how fast force grows when points get close

Although this force does not correspond to a specific physical phenomenon in orbital mechanics, it provides a mathematically tractable way to model spacing in systems where proximity influences configuration.

The repulsive-force framework therefore serves as an analytical tool for studying how distributed elements arrange themselves in space under distance-dependent constraints.

This is a piece of the code I used to visualize the distance between a random number of different located points. The yellow framed line is how I used repulsive force formula using Python

```python
def compute_net_force(pos):
    F_net = np.zeros_like(pos)
    for i, posN in enumerate(pos):
        dist = pos - posN

        dist_left = dist[dist < 0]
        dist_right = dist[dist > 0]

        F_net[i] = np.sum(np.abs(1 / dist_left**N_dim)) - np.sum(
            np.abs(1 / dist_right**N_dim)
        )

    return F_net
```

3.7 Multistrom Cluster of Satellites

Multistrom is a term defining the many and different orbits, aiming to represent a situation where data, satellites, and backbones exist in all orbits, and can be transferred easily to any other orbit.

4 Methods

This research investigates satellite network connectivity using a custom Python simulation framework(https://github.com/nasianikolova/Cluster-of-satellites). The system generates orbital constellations by defining shells with configurable parameters, including altitude, inclination, number of planes, and satellites per plane. Satellite positions were computed in three dimensional space through geometric transformations that model inclination and Right Ascension of the Ascending Node (RAAN) offsets. Connecting nodes, the simulator built an inter-satellite link graph from pairwise distance calculations. Using Dijkstra's algorithm [5], the code calculates the shortest path between chosen nodes to assess routing performance. Paths were computed based on minimal distance requirements. The result interpretation has been conducted using 2D and 3D visualizations. Those visualizations displayed orbital rings, satellite positions, link structures, and highlighted routing paths. Besides the orbital shell model, the work also used an electrostatics-motivated technique for the problem of uniform satellite spacing. Satellites were considered as mutually repelling particles confined to the surface of a sphere in this method, and their coordinates were repeatedly changed so as to maximize the distances between them. The Fibonacci sphere distribution was used to obtain the starting locations for this procedure, which gives a nearly uniform first arrangement, and then the repulsion stages improve the spacing. In order to generate a 3D representation of Earth with LEO and Middle Earth Orbit (Altitudes: 2000 km - 35786 km [13]), and to ensure that the number of orbits and satellites can be configured and always the satellites will be equally spaced, the code plots graphs that show the relationship of 2 or more factors (altitude vs link budget vs latency), redundancy. More specifically, Matplotlib[10] was used for depiction of 3D and 2D graphs and visualizations, NumPy[9] was used for mathematical calculation that ultimately produce graphs and visualizations, tqdm[1] was used to display progress, and mpltoolkits was used for 3D renderings. The mathematical calculations were also dependent on the Friis equation (7) for link budget - A link budget sums all gains and losses from the transmitter to the receiver to determine whether the received signal is strong enough for reliable communication[8]. - calculation and Sound-to-Noise Ration (SNR) signal durability since SNR measures how strong signal is compared to background noise., and Fibonacci sphere for evenly spaced satellites within orbits.

$$R_r = P_t G_t G_r (\frac{\lambda}{4\pi d})^2 \tag{7}$$

where: P_r = Received power, P_t = Transmitted power, G_t = Transmitter antenna gain, G_r = Receiver antenna gain, λ = Wavelength of the signal, d = Distance between antennas

Every orbit crosses the equator twice:

1. Ascending node is where the satellite moves from south to north
2. Descending node is where it moves from north to south

$$RAAN_p = \frac{2\pi \cdot p}{N_{planes}} \tag{8}$$

where:

$RAAN$ = the angle to the ascending node.

p = the plane index

N_{planes} = the total number of orbital planes

One important part to continue this research was to visualize its purpose. Using Python and specifically the libraries NumPy, Matplotlib, tqdm, heapq, matplotlib.widgets, and matplotlib.pyplot, this research achieves to explain the idea of a new way of near space management.

The network topology is visualized using matplotlib by plotting nodes in a 2D Cartesian plane. The Earth is represented as a central circle, with MEO stations and satellites plotted at their respective ((x, y)) positions. Inter-satellite links (ISLs) and MEO-stations-to-satellite links are drawn as light gray lines. Shortest paths, calculated by Dijkstra's algorithm, are highlighted in red to visualize the routing path over the topology.

To calculate the delay by having the distance as controlled variable, the code takes into consideration the speed of light (299,792.458m/s) and divides the altitudes with the speed of light. As a result, the code outputs the number of seconds it would take to communicate with a satellite based on its height.

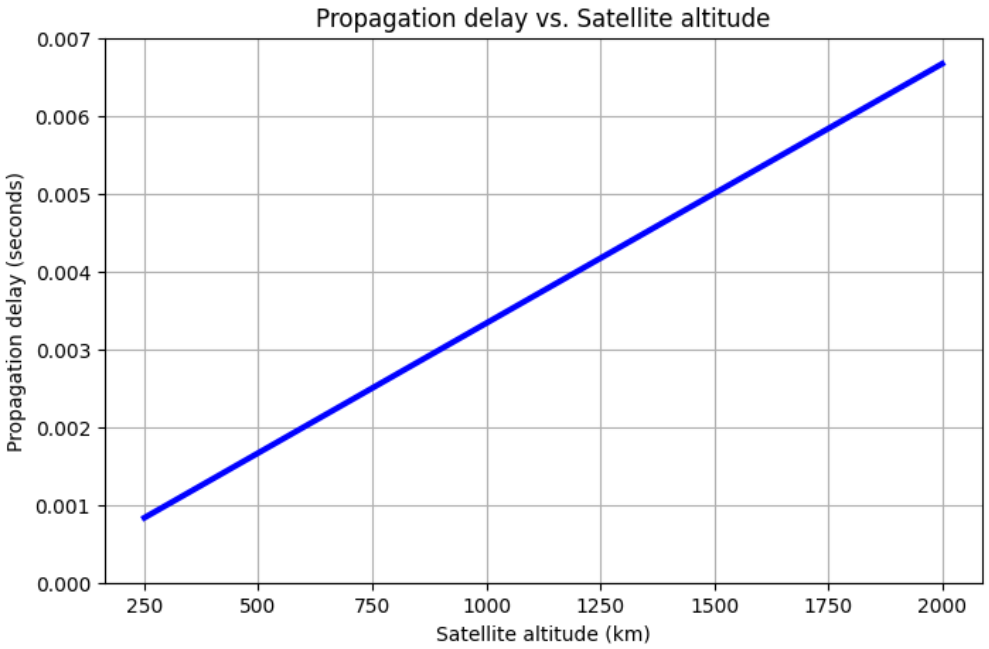

Figure 5: The graph illustrates the relationship between satellite altitude and signal propagation delay. Since electromagnetic waves travel at the speed of light, the delay increases linearly with distance. At the lowest altitude 250 km, the delay is very small – less than one millisecond. As the altitude increases to 2000 km, the delay grows to around 0.0067 seconds. Although these values are tiny, from the graph it is understood that higher orbits incorporate more latency.

This code visualizes how satellite altitude affects the number of hops required to route data between two ground stations. It iterates through a range of altitudes from 300 to 2000 km, running a custom network simulation for each to find the shortest path and count the resulting network hops. Because network hop counts typically decrease as altitude increases, the code fits a continuous exponential decay model to the discrete simulation data by applying a linear regression to the logarithm of shifted hop values. For example, at 250 km delay is approximately 0.00083 seconds, and at 2000 km is approximately 0.00667 seconds.

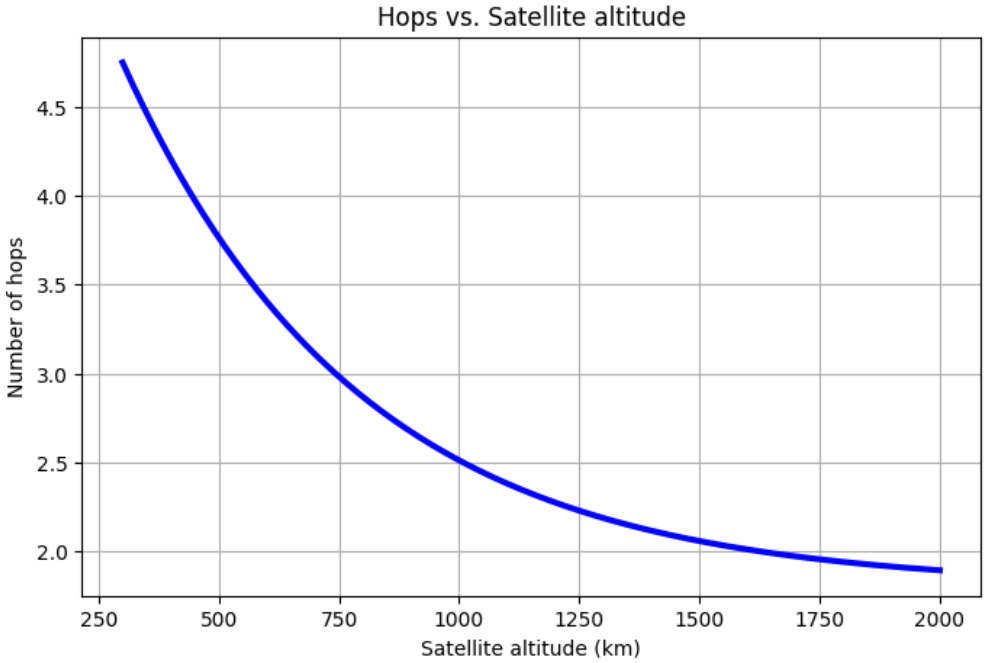

Figure 6: The figure illustrates how the number of communication hops within the satellite network changes as orbital altitude increases from 300 to 2000 km. A fitted exponential curve illustrates that higher altitudes reduce the required hops because each satellite covers a larger area. This visualization allows us to understand how constellation altitude influences routing complexity and helps explain why lower orbits need more satellites to complete long distance paths[15].

In order to investigate the effect of altitude on the Signal-to-Noise ratio, the SNR of a satellite communication link is computed and graphically plotted in terms of how its value varies as the satellite rises up in altitude. For the computation, we have to identify the physical constants which include the velocity of light, power of the transmitter, antenna gains and the carrier frequency which is assumed to be 12 GHz. Then, the code computes the signal strength of a satellite communication system for different altitudes from 300 to 2000 km when the satellite is directly above. In each step of increasing the altitude, the Friis transmission formula is used to compute the received power signal, and the constant thermal noise floor is computed using the system temperature and bandwidth of 20 MHz. Finally, the received power is divided by the noise power to find the linear SNR, and it plots the results, clearly illustrating how signal quality drops sharply at higher altitudes due to free-space path loss.

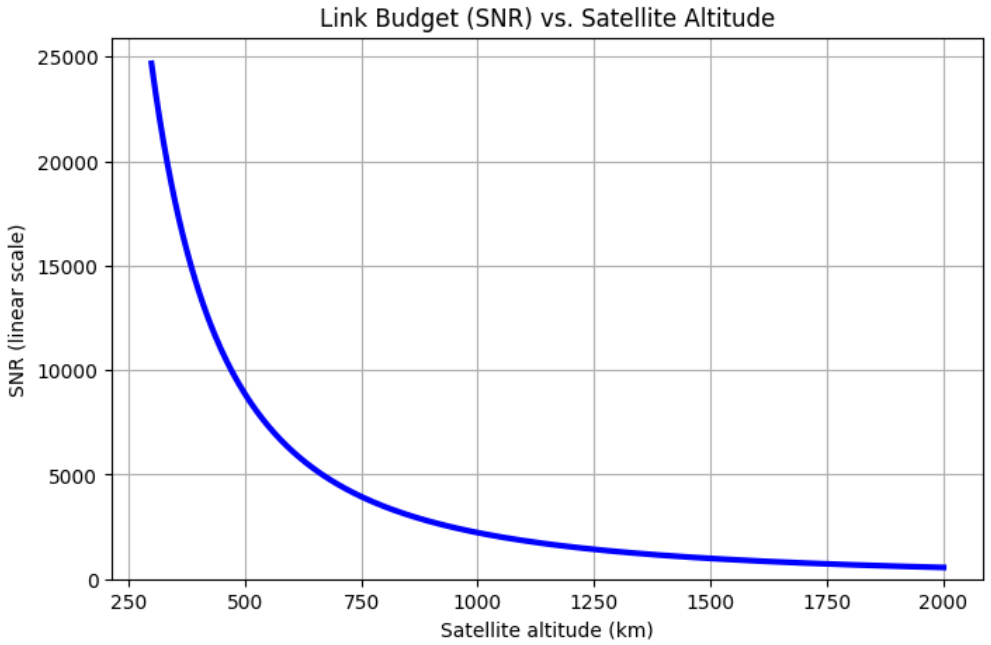

Figure 7: The figure illustrates how the signal to noise ratio (SNR) changes as satellite altitude increases from 300 to 2000 km. Using the Friis transmission model, the curve shows a steady decline in SNR with altitude because received power decreases with distance while noise remains constant. This visualization helps demonstrate how orbital height affects link quality and highlights the increasing power and antenna requirements for higher altitude communication[7].

In order to investigate the relationship among satellite height, propagation delay, and link budget accuracy, a simulation program was developed using Python and the NumPy and Matplotlib libraries. In this system, eight orbital altitudes were used starting from 250 km to 2000 km with an increment of 250 km at each level. Propagation delay was estimated by dividing height by the speed of light. Signal-to-Noise ratio (SNR) for all altitudes was estimated through the Friis transmission equation where received power is calculated. Received power was further divided by thermal noise power estimated based on Boltzmann constant. Finally, the interdependent mathematical relationship between the above three continuous variables was illustrated graphically through a 3D parametric spatial diagram.

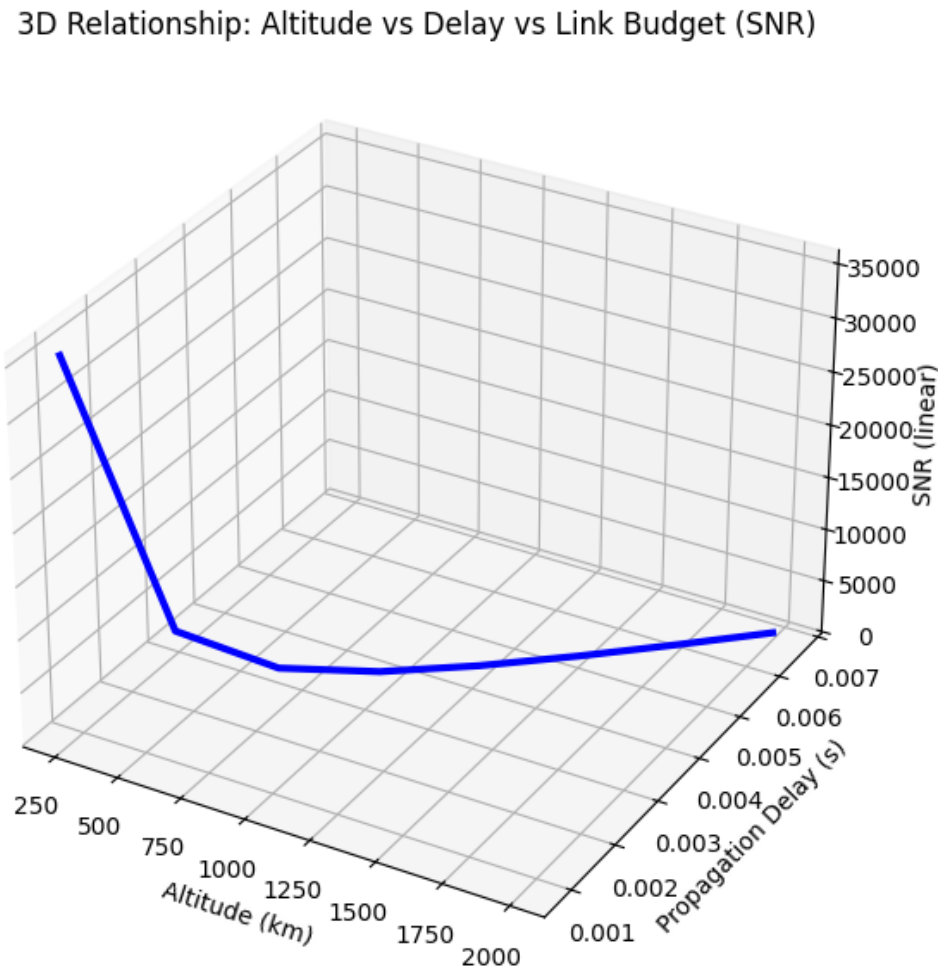

Figure 8: The figure presents a 3D plot showing how satellite altitude simultaneously affects propagation delay and link quality. Delay increases linearly with altitude, while SNR decreases because received power weakens with distance. Plotting these variables together notes the functionality of satellite communication. Higher orbits provide broader coverage but experience longer delays and inferior SNR[7]. This visualization helps illustrate how altitude influences overall system performance.

A 3D earth is created to support a multistrom satellite constellation, and display its topology in space. To do that, a certain number of orbital shells will be generated. Each of them is characterized by the number of planes, the number of satellites in each plane, the altitude, and the inclination of the orbit. The satellites' positions are obtained from the parametric model based on circular orbits. Appropriate rotations provide required inclination and right ascension of the ascending node. The connectivity between satellites is established based on a maximum inter-satellite link (ISL) distance, creating a graph representation of

the network. The shortest path between two specified nodes is computed using Dijkstra's algorithm, which is implemented with a priority queue for efficiency. The resulting path and its total distance are then visualized in a 3D plot that includes the Earth, orbital rings representing the shells, and the satellites as points in space. The plot also highlights the computed shortest path.

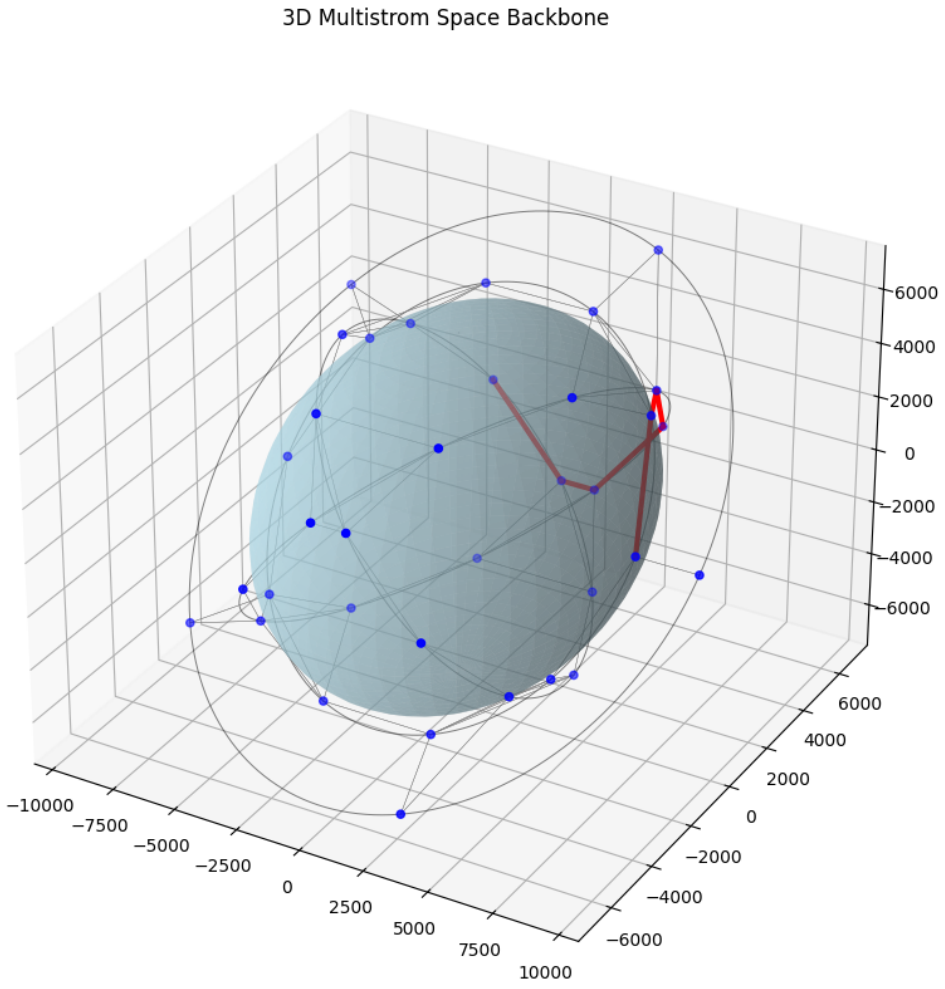

Figure 9: 3D representation of Earth. Above Earth there are 3 LEO orbits consisted of 10 satellites each, at the altitude of 550km. There is also one MEO orbit consisted of 5 satellites, at the altitude of 3000km. Satellites in MEO orbit represent Backbones. Bacakbones from MEO orbit directly communicate with satellites from LEO orbit. This way, latency between satellites is reduced. LEO satellites can send information to MEO, and MEO will transfer the information without needing to pass through many hops. This Multistrom system of communication works faster compared to communication within one orbit only, because as we go up, the optic horizon gets bigger.

The code is based on the simulation of distributing random number of satellites around the planet Earth through positioning them within a highly symmetrical grid on a sphere through the use of Fibonacci sequence with the use of the golden ratio to position them in an equal vertical spiral to ensure no stacking occurs. The process involves running the simulation through an iteration model whereby the satellites are positioned at an equal distance between each other while applying a repulsive force among them that is inversely proportional to their squared distance from one another. Once they move away from each other through the repulsive forces, they are forced back to their positions on a perfect orbital radius that ensures they form a spherical position. The process is rendered in a visualized format of an earth with the satellite dots being plotted with the graphical grid.

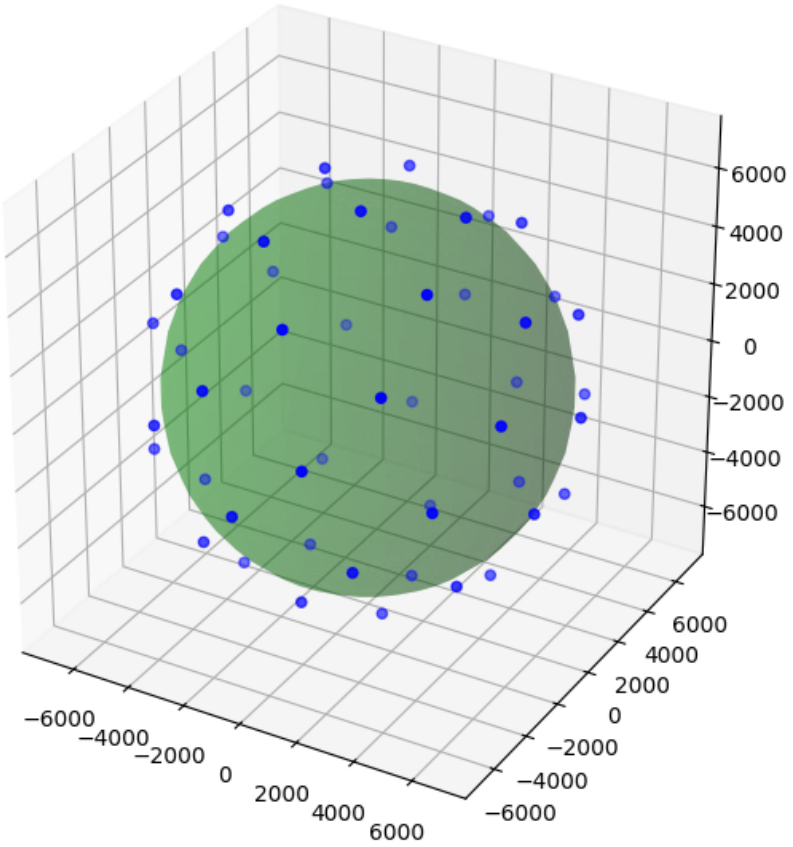

Figure 10: The figure presents a 3D representation of earth and surrounding satellites which have a uniform distance between each other. There are 50 satellites evenly distributed satellites to the Earth. This figure is a step before include orbits that will form the final network of evenly spaced satellites.

This research code implements a Walker Delta satellite constellation through the computation of precise three-dimensional coordinates for satellites in several orbital planes and altitudes. First, it evenly distributes the orbits on a sphere, assigning satellites within each orbit based on a phase parameter that prevents collision between satellites. It also uses rotational equations to shift the orbits away from the equator. In addition, the software computes the Euclidean distance between pairs of satellites, building up a network of connections between satellites when they have distance smaller than a kilometer threshold. Finally, the code generates a visualization of a 3D plot with a green spherical Earth and the orbital planes drawn in black and blue dots representing the satellites. An equal aspect ratio was used in order to maintain accuracy of the plot and top-down camera view.

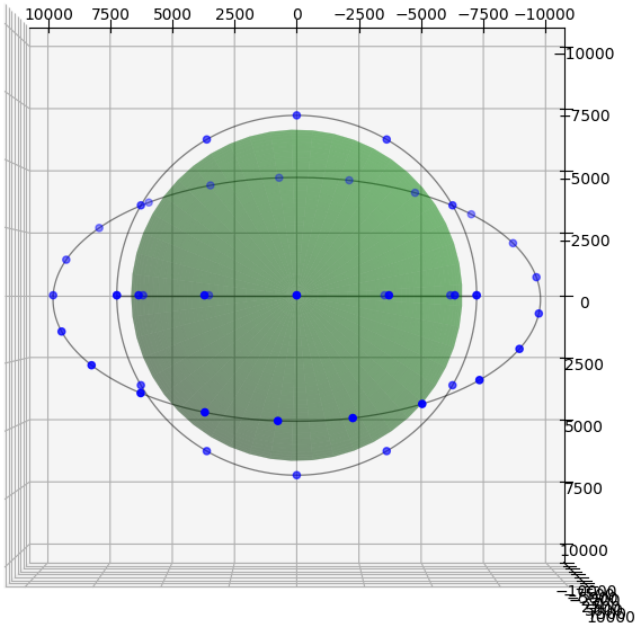

Figure 11: 3D representation of Earth with evenly spaced satellites. The are three orbits depicted in this figure. One MEO orbit that consists of 21 satellites, and two basic LEO orbits that consist of 12 satellites each, the Polar orbit with inclination at 90°, and Equator orbit with inclination at 0°.

5 Results

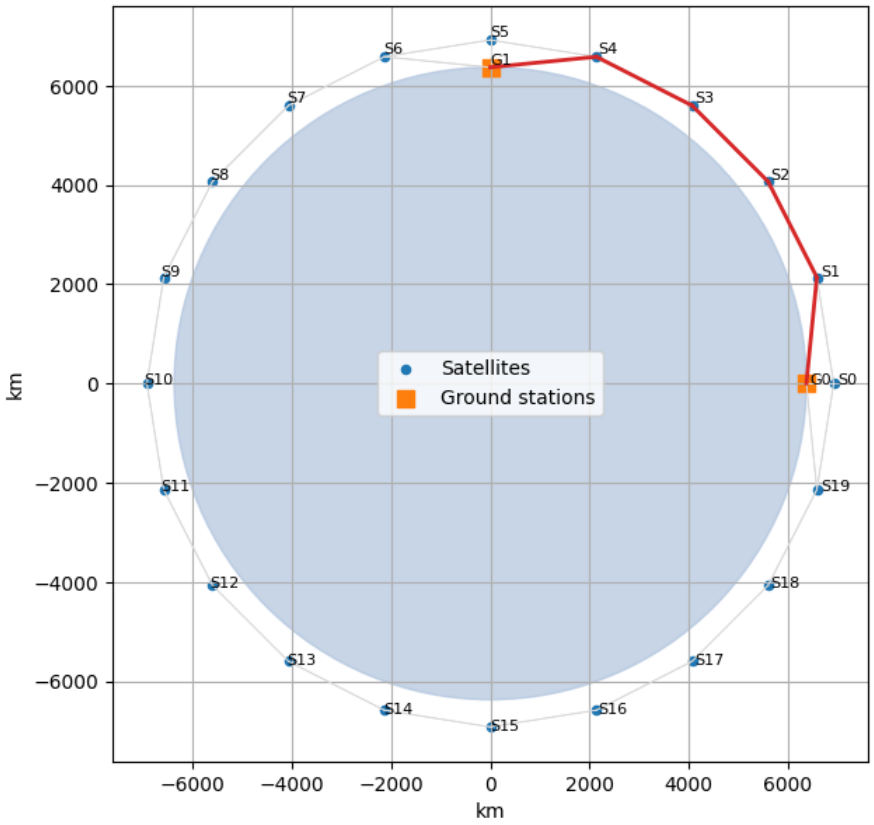

Figure 12: 3D representation of Earth with evenly spaced satellites. The are three orbits depicted in this figure. One MEO orbit that consists of 21 satellites, and two basic LEO orbits that consist of 12 satellites each, the Polar orbit with inclination at 90°, and Equator orbit with inclination at 0°.

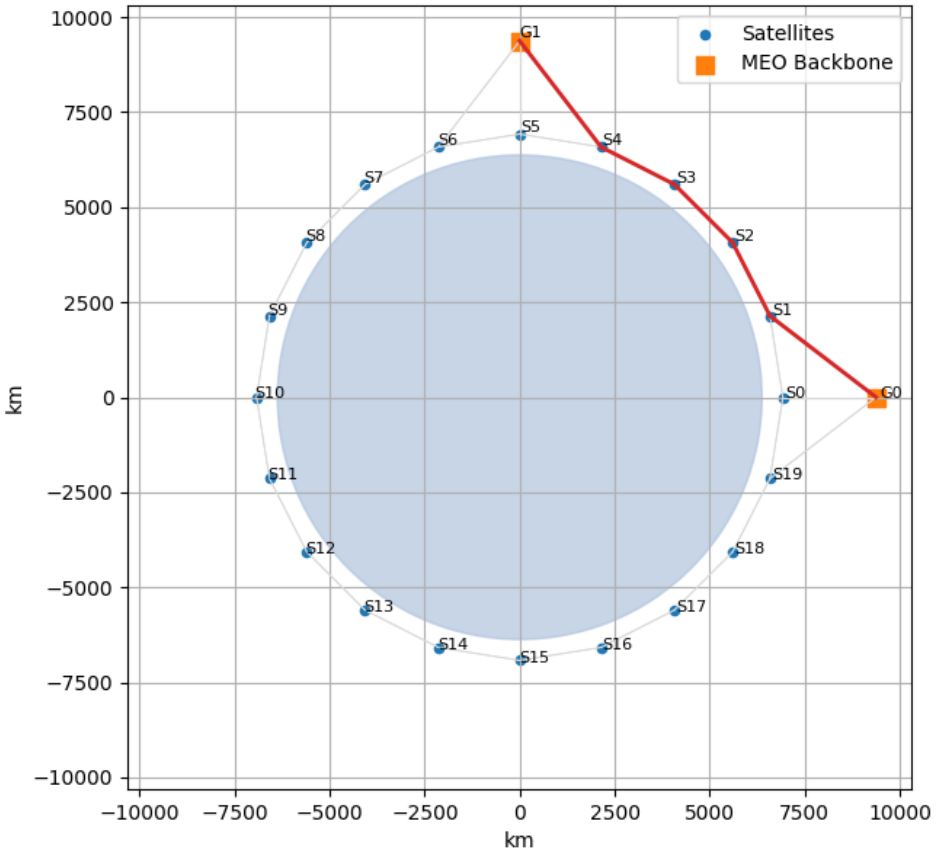

Figure 13: The figure represents earth and surrounding satellites. It showcases the interconnectivity of satellites through the network structure. In LEO orbit, satellites can be found. In MEO orbit Backbones can be found. The idea is that LEO satellites will communicate directly to MEO satellites forming a multistrom way of communication. In this way, time is saved and every data transaction takes place faster following a multistrom path.

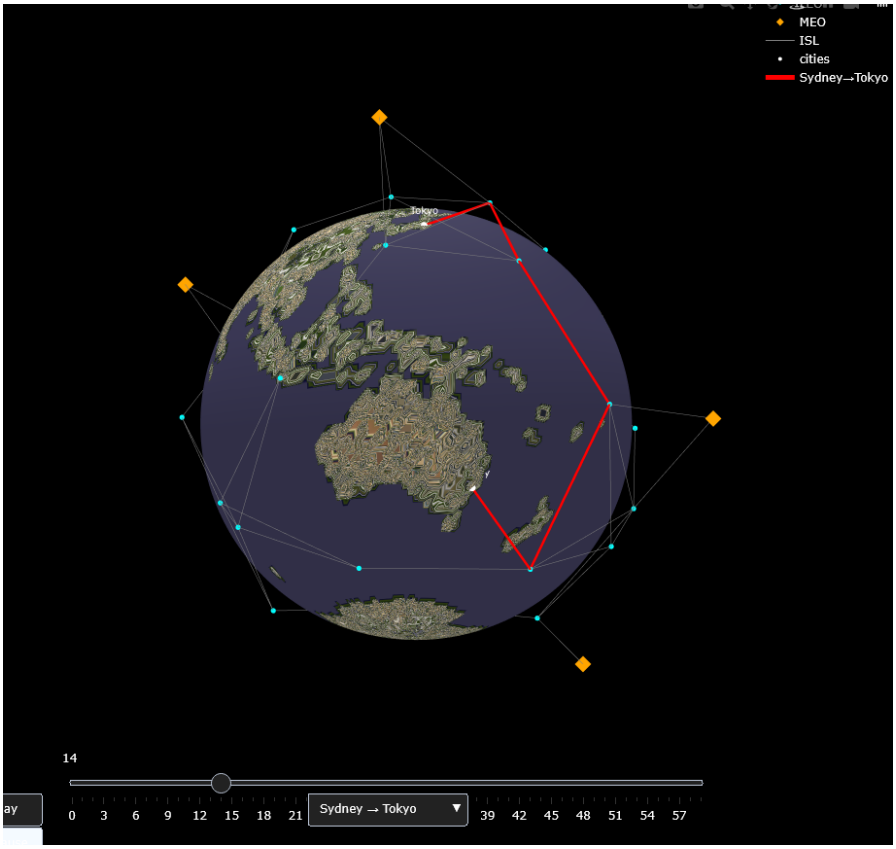

Figure 14: This figure is a screenshot from an experimental video with a timeline, representing the shortest path -using Dijkstra's algorithm- data would follow if no backbone available for communication in MEO or in any higher orbit.

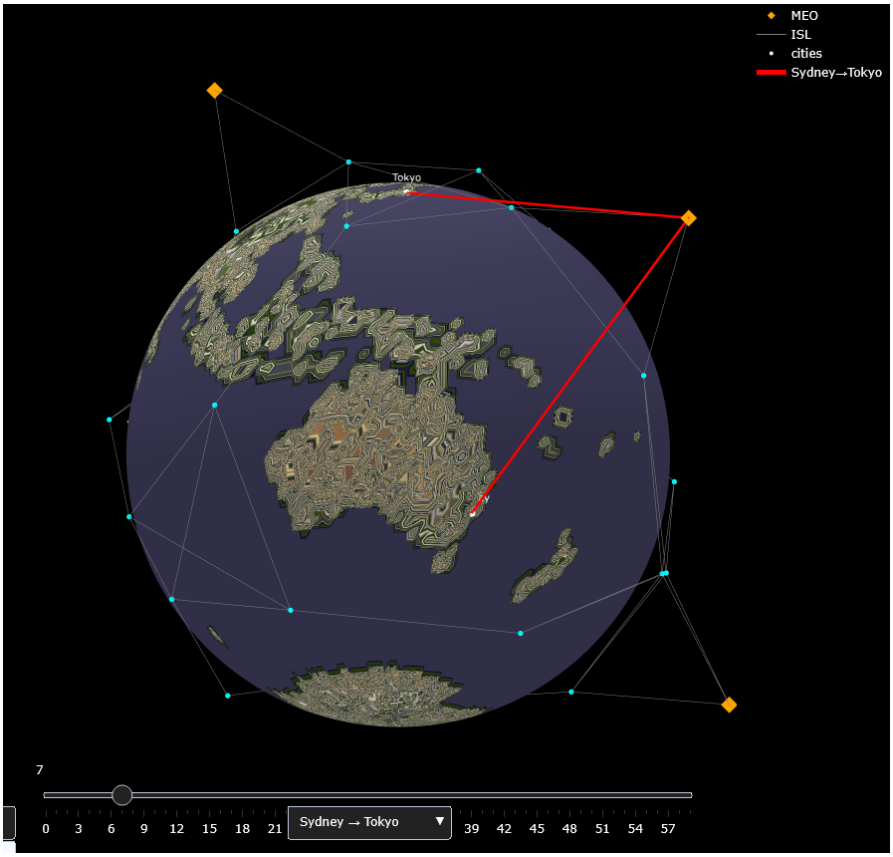

Figure 15: This figure is a screenshot from an experimental video with a timeline, representing the shortest path -using Dijkstra's algorithm- data would follow if a backbone available for communication in MEO or in any higher orbit existed.

6 Discussion

This research begun by suggesting backbone's existence in orbit along with satellites. The purpose of placing backbone in Space is to reduce trip-delay and find the shortest path between nodes while taking into consideration hops existing in higher orbits.

As shown in Figure 14 and 15, for the same route (Sydney - Tokyo in this case) the trip-delay was shorter when a backbone existed in higher orbit. On the contrary, when following a route that does not involve backbone in higher orbit, data must travel through many hops-satellites whose event horizon is shorter because of their altitude, and as a result it takes longer to reach the final destination.

In addition, the research shows that such a network must use technologies that do

not require human presence. That is because humans are vulnerable to conditions outside Earth. This can be microgravity[12], lethal radiation[6], etc. Space is a hostile environment, making robotic exploration both safer and vastly more cost effective[19].

Also, technologies based on quantum science appear to be the most optimal both for the energy supply and for data manipulation and storage[16].

All data found in space are stored into the backbones. Multiple requests lead to the demand of no-time response among data points found in the backbone. Functionalities such as automatic data assembly will take place at very high speed due to the low trip-delay between satellites and data points. Automatic rearrangement will have instant response because all recalculations will take place in Space. This entire system is automated, and includes its processing power as well as its data in an ultra low trip-delay environment.

7 Conclusion

This research starts with the ultimate purpose to visualize a new multistrom environment where satellites and backbones will communicate regardless of the orbit to which they belong. That is, data exist in all orbits; however, this research tested only from 250km up to 5000km - covering the entire LEO and a small part of MEO. It concludes that higher number of orbits results on better performance, latency, and redundancy of this network. An exemplary situation is the distance from 250km up to 2000km altitude, where the trip delay at 250km is 0.83ms and at 2000km is 6.67ms. This research showed that communication between earth and space is problematic due to the huge distance needed to be covered for data confined in a straightforward path. This research illustrates that all parts of a space network can exist in orbit and even in different altitudes. This research establishes the multistrom network, launching data and backbones all in space. A topic not expanded on this research is modules that surround all backbones. These modules support the backbone by exchanging information and provide it with all the changes scientists wish to include in future works. The question is how they will orbit the backbone, return to the Earth and be replaced with a revised version of the backbone.

References

[1] tqdm: A fast, extensible progress meter for python and cli. *Journal of Open Source Software*, 4(37):1277, 2019.

[2] John Christian. A tutorial on horizon-based optical navigation and attitude determination with space imaging systems. *IEEE Access*, 9:19819–19853, 01 2021.

[3] Howard D. Curtis. Chapter 6 - orbital maneuvers. In Howard D. Curtis, editor, *Orbital Mechanics for Engineering Students (Fourth Edition)*, Aerospace Engineering, pages 287–350. Butterworth-Heinemann, fourth edition edition, 2020.

[4] Giuseppe Di Pasquale, Manuel Sanjurjo-Rivo, and Daniel Pérez Grande. Optimization of constellation deployment using on-board propulsion and earth nodal regression. *Advances in Space Research*, 70(11):3281–3300, 2022.

[5] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1:269–271, 1959.

[6] William Dobney, Louise Mols, Dhruti Mistry, Kevin Tabury, Bjorn Baselet, and Sarah Baatout. Evaluation of deep space exploration risks and mitigations against radiation and microgravity. *Frontiers in Nuclear Medicine*, 3:1225034, 2023.

[7] Sergi Gallardo Marquina. Study and simulation of communication links in a leo satellite constellation based on link budget calculations. 2022.

[8] Juan Misael Gongora-Torres, Cesar Vargas-Rosales, Alejandro Aragón-Zavala, and Rafaela Villalpando-Hernandez. Link budget analysis for leo satellites based on the statistics of the elevation angle. *IEEE Access*, 10:14518–14528, 2022.

[9] Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, September 2020.

[10] J. D. Hunter. Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, 9(3):90–95, 2007.

[11] George A Jennings. Conics. In *Modern Geometry with Applications*, pages 83–113. Springer, 1994.

[12] Ashish Vijay Khune et al. Physiological responses to space travel: A systematic review. *European Journal of Cardiovascular Medicine*, 14:450–454, 2024.

[13] Tian Qiu, Jun Hong, Yu Wang, Kaichu Xing, Shaoyan Du, and Yang Qi. Meo-sar in-orbit elevation antenna pattern determination using nano calibration satellite. In *2022 Photonics & Electromagnetics Research Symposium (PIERS)*, pages 278–282, 2022.

[14] Ahmed Ragab Mahmoud. Law of universal gravitation: Force of attraction between masses, 09 2025.

[15] Chirag Rao and Eytan Modiano. Minimum-hop constellation design for low earth orbit satellite networks. In *IEEE INFOCOM 2025-IEEE Conference on Computer Communications*, pages 1–10. IEEE, 2025.

[16] Quansheng Ren, Tamara Sarac, Sung Yim, and Koen Bertels. Dna-based quantum computing, data storage and quantum applications needed for earth as well as for the mars exploration, 04 2026.

[17] Jorgelina M. Talamoni and Andrea J. C. Borlaff. About 15,000 satellites are circling earth — and they're disrupting the sky. *Discover Magazine*, January 2026.

[18] Michael Turner. Gravitational radiation from point-masses in unbound orbits-newtonian results. *Astrophysical Journal, Part 1, vol. 216, Sept. 1, 1977, p. 610-619.*, 216:610–619, 1977.

[19] Rodrigo Ventura. Robotic systems for space exploration. In *Robotics and AI in Extreme Environments*, pages 11–36. Springer, 2026.