ORBITCAST: Exploiting Mega-Constellations for Low-Latency Earth Observation

Zeqi Lai, Qian Wu¶, Hewu Li, Mingyang Lv, Jianping Wu
Institute for Network Sciences and Cyberspace, BNRist, Tsinghua University, Beijing 100084, China
zeqilai@tsinghua.edu.cn, {wuqian, lihewu, jianping}@cernet.edu.cn, lvmy20@mails.tsinghua.edu.cn,

2021 IEEE 29th International Conference on Network Protocols (ICNP) | 978-1-6654-4131-5/21/$31.00 ©2021 IEEE | DOI: 10.1109/ICNP52444.2021.9651919

***Abstract*—Satellite-based Earth Observation (EO) systems are gaining popularity and widely used in many time-sensitive scenarios, including disaster monitoring, emergency response, forecasting and defense. Existing efforts for gathering EO data mainly rely on either ground station networks or geostationary (GEO) satellites. However, our quantitative analysis reveals that existing approaches are either limited as their achievable latency is far away from the desired value due to the insufficient coverage of ground stations, or hard to scale as the number of sensing satellites increases because of the high cost of GEO satellite relays.**

This paper explores the feasibility and performance of a novel approach that leverages emerging low Earth orbit (LEO) constellations to enable low-latency and scalable EO data delivery from space. We present ORBITCAST, a hybrid EO data delivery architecture upon LEO constellations and geo-distributed ground stations to forward EO data from the source remote sensing satellite to a collection of end users. To handle the network dynamicity caused by LEO satellite movements and achieve stable communication over the satellite network, we propose a geo-location driven scheme to forward and deliver data packets. To demonstrate the effectiveness of ORBITCAST, we build a testbed driven by public constellation information and implement the ORBITCAST prototype on top of the testbed. Extensive realistic-data-driven simulations demonstrate that ORBITCAST can significantly reduce the latency as compared to other state-of-the-art approaches, and complete the data delivery within five minutes for representative EO data traffic.

***Index Terms*—Satellite-based Earth Observation, Mega-Constellations, Integrated Satellite and Terrestrial Networks.**

I. INTRODUCTION

Satellite-based Earth Observation (EO) systems [4], [5], [34] that continuously gather physical, chemical, and biological information from orbits in space have become very popular in recent years, thanks to technical breakthroughs in the space industry [31], [39], [41]. The EO market size is poised to grow by USD 4.51 billion until 2024 [18]. About 44% of today's in-orbit satellites are for EO propose [24], and most of them are working in low Earth orbit (LEO) with altitude less than 2000km, collecting tens of Terabytes of data (*e.g.,* high-resolution imagery) per day during their orbit [59], [60].

Big EO data collected by remote sensing satellites needs to be downloaded to terrestrial EO data centers and then be distributed to a collection of customers who finally run their own applications to process the data for dedicated proposes. In many scenarios, EO systems require low-latency data delivery, *i.e.,* EO data collected by remote sensing satellites is expected to be delivered to users as soon as possible. Scenarios needing very low latency include disaster management and emergency response (*e.g.,* for floods, fires and earthquakes), forecasting for extreme weather conditions, remote monitoring and security (*e.g.,* maritime smuggling/rescue, illegal fishing) and defense, where the EO data is only useful if it is successfully delivered in a very short time period [56]. In addition, as many EO applications require low-latency, EO systems are also expected to be scalable to support various EO tasks simultaneously, as the number of sensing satellites increases.

In this paper, we perform a systematic study to tackle this problem facing the EO industry - *is it feasible to enable very low-latency data deliver in satellite-based EO systems?*

We carry out our study in three steps. Frist, we start our quest by exploring the attainable latency in existing EO systems and identifying the performance bottleneck. Typically, existing efforts for downloading EO data from space can be classified into two categories: *downloading data via ground station networks* [50], [66], or *downloading data via geostationary satellite relays* [64], [71]. In the former *"store first, and download later"* approach, the satellite operator has to deploy geo-distributed ground stations in advance. The sensing satellite downloads data to ground stations through satellite-ground communication links, if only the sensing satellite is in the transmission range of certain ground stations. While simple and practical, this approach may suffer from long data delivery latency due to the intermittent and unstable communication between satellites and ground stations. First, it may take minutes or even hours for the sensing satellite to move into the transmission range of an available ground station, as ground stations are hard to be deployed upon oceans which cover more than 70 percent of the surface of our planet. Second, the orbital velocity of EO satellites is very high, bringing a very short visible windows between satellites and ground stations. It is thus difficult to guarantee that EO data with large volume can be completely downloaded in one pass. Finally, the link quality between satellites and ground stations might also be significantly affected by poor weather conditions, resulting in reduced data rate and prolonged deliver latency.

The other approach for downloading EO data is exploiting satellite relays in geostationary (GEO) orbit to establish long-duration communication from the sensing satellite to terrestrial EO data centers. Many state-of-the-art EO systems like TDRS [71], EDRS [72] and EFFIS [7] are using this approach to achieve near-real-time Earth observation. While it enables stable, long-duration LEO→GEO→ground communication path, and free-space LEO→GEO laser links can achieve up to 1.8Gbps [72] datarate, the total delivery latency by this

¶Qian Wu is the corresponding author.

Authorized licensed use limited to: University of Regina IEL. Downloaded on November 20,2025 at 03:38:45 UTC from IEEE Xplore. Restrictions apply.

approach is still limited by the GEO→ground path with the speed less than 300Mbps. Moreover, the supported number of user-spacecrafts per GEO satellite relay is very limited (*e.g.,* only 2 per GEO relay in TDRS [64]). This indicates that although covering about one-third of Earth surface, one GEO satellite relay can only support two EO tasks simultaneously. Considering the high manufacturing and launch costs of GEO satellites (*e.g.,* ∼\$544m for a EDRS relay [6]), this approach can not scale well as the number of sensing satellites increases.

In the second step, to facilitate low-latency EO, in this paper we explore a novel approach to deliver big EO data collected from space: *leveraging a large number of interconnected satellites in LEO, together with geo-distributed ground stations to construct a hybrid data delivery network that enables long-duration, high-throughput and low-latency communication from the remote sensing satellite to terrestrial users.*

We present ORBITCAST, a low-latency framework for EO data delivery. Specifically, ORBITCAST adopts two key ideas to attain low-latency, scalable and bandwidth-efficient EO data delivery: (i) *Constructing a hybrid delivery network upon emerging LEO constellations and geo-distributed ground stations.* The hybrid network enables the opportunity of long-duration low-latency communication from the sensing satellite to users *whenever* the data is acquired. Intuitively, EO data is either delivered through ground station infrastructures on-demand, *e.g.,* Ground-Stations-as-a-Service (GSaaS) [3], [26], if ground communication is available underneath, or delivered directly to users (equipped with satellite terminals) through space routes constructed by high data-rate inter-satellite links (ISLs) and space-ground downlinks. (ii) *Exploiting a* ***location-driven*** *routing scheme in the hybrid network where the topology changes over time to deliver EO data.* ORBITCAST leverages a geo-location based, adaptive addressing scheme to index network nodes and drive the data delivery in the hybrid space-ground network. The location-based scheme enables low-latency, long-duration and bandwidth-efficient delivery through eliminating the address updating and route re-convergence overhead incurred by frequent connectivity change and supports multicast routing when the number of end users increases.

As the third step of our study, we build an experimental environment to simulate the space-ground network and implement the ORBITCAST prototype to evaluate the feasibility and network performance of our approach. To improve the fidelity of our testbed, we leverage public orbital information and EO dataset to drive the simulation of LEO constellation, and emulate network software stack and traffic based on Mininet [11]. Extensive evaluations shows that ORBITCAST can significantly reduce the EO latency and finish the delivery process within 5 minutes for representative EO data trace, even under situations where the constellation or user scale increases.

Collectively, the main contributions in this paper are as follows: (1) through a quantitative analysis on existing satellite-based EO systems, we identify that the insufficient coverage of ground station networks, limited forwarding data rate and high cost of geostationary satellites are the key challenges for obtaining low-latency and scalable EO data delivery; (ii)

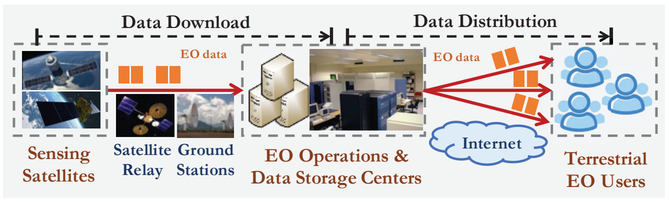


Fig. 1: The process of data delivery in typical EO systems.

proposing ORBITCAST, a low-latency framework for EO data delivery that exploits a hybrid space-ground network to enable low-latency, scalable and bandwidth-efficient EO data delivery; (iii) building a testbed to evaluate the feasibility and performance of ORBITCAST and quantify the end-to-end latency reduction using representative constellation patterns and EO traffic. To the best of our knowledge, we are the first to exploit the integration of emerging mega-constellations and ground station services to collaboratively achieve low-latency for EO applications at scale.

II. BACKGROUND AND MOTIVATION

A. Satellite-based Earth Observation (EO)

Big data collected from space. Satellite-based Earth observation is becoming very popular in recent years due to the evolved ability of sensing the Earth from space. The volume of EO data continues to increase exponentially with a unprecedented growth due to the launch of new spacecrafts, with more powerful, multi-spectral, and accurate sensors. In particular, the average daily traffic generated by NASA's EO system is about 27.9TB [60]. The present EO archive in European Space Agency (ESA) has exceeded 3 PB and it is foreseen that this volume will exceed 10 PB in few years [59].

Earth observation systems. The core functionality of today's EO systems is to download big EO data captured from space and distribute the data to customers/users who then run their own applications to process the data for specific proposes. As shown in Figure 1, a typically EO system follows two primary steps to complete the process of EO data delivery:

- (i) A ***data download*** step which downloads EO data from the *remote sensing satellites* to EO's operation and data storage centers. Remote sensing satellites typically carry dedicated instruments (*e.g.,* an optical sensor) designed for capturing useful data (*e.g.,* high-resolution images) during the orbit. Most existing EO sensing satellites are operated at a relatively low orbit (*e.g.,* 600-1200km altitude) [70]. For instance, the Proba-1, Proba-2 and SMOS spacecraft of ESA are observing the Earth from an altitude of about 700 km [70]. Sensing satellites continuously and periodically acquire and save EO data to the on-board storage. Data stored in sensing satellites can be downloaded to EO data centers via ground station networks [46], [50], [65], [66] or through geostationary satellite relays [34], [38], [69], [71] depending on the specific design in different EO systems.
- (ii) A ***data distribution*** step that distributes those downloaded EO data from the EO data centers to a collection of end users. This step can be done through typical distribution

Authorized licensed use limited to: University of Regina IEL. Downloaded on November 20,2025 at 03:38:45 UTC from IEEE Xplore. Restrictions apply.

Description	Data Set Volume(GB)
Moderate Resolution Imaging [12]	107.3
Multi-angle Imaging [13]	29.7
Atmosphere Infrared Sounder [2]	13.1

TABLE I: The public EO dataset used in our experiment.

methodologies in today's terrestrial Internet (*e.g.,* via Web servers or content distribution networks). Terrestrial end users finally run their own applications to process EO data for dedicated proposes, such as map and navigation services (*e.g.,* Google Maps), disaster monitoring and forecasting [7], or other research and education aspects [60].

Many EO applications require low latency data delivery. For many EO applications, the observed data is only useful if it is available in a very short time period since its data acquisition. These applications include disaster management and emergency response (*e.g.,* floods, fires, earthquakes), forecasting (*e.g.,* for extreme weather conditions), security (*e.g.,* maritime rescue) and defense *etc.*. Further, future EO systems are also expected to be extended to serve time-sensitive social scenarios (*e.g.,* satellite-based video streaming for observing space environments). We define the delivery latency of a EO task as the flow completion time of transferring data acquired from space to the corresponding terrestrial user. Therefore, ideally, a satellite-based EO system is expected to simultaneously be: (i) ***low-latency***, as the latency in above time-sensitive scenarios is expected to approach 1 minute or at least less than 5 minutes in particular [56]; and (ii) ***scalable***, which indicates that many kinds of EO applications/tasks can be concurrently supported when the number of sensing satellites and terrestrial users increases.

B. Limitations of Current EO Approaches

Since the approach of data distribution over terrestrial networks has been well studied and optimized, the data delivery latency in different EO systems is primarily affected by the data download step in the left Figure 1. Specifically, existing approaches for EO data download can be classified into two categories: ***download via ground station networks*** and ***download via geostationary satellite relays***. We next conduct two groups of experiment to introduce and analyze the attainable latency under above two primary approaches widely used in today's EO systems.

Experiment setup. We build a simulator based on public details of EO satellites to estimate the delivery latency by different downloading approaches. We use the two-line element (TLE) data of an EO sensing satellite LEMUR 2 ZO [10] published by CelesTrak [33] to drive the simulation of remote sensing satellites. TLE [32] is a data format encoding a list of orbital elements of an Earth-orbiting object for a given point in time. We use three public EO datasets [14] differed in size from NASA's EO system as the source content for delivery. Table I summaries the description of these data sets, which are originally collected for meeting the timely need of applications such as numerical weather and climate prediction, forecasting and monitoring natural hazards, *etc.*.

Data Set	Ground Stations		
	# GS = 5	# GS = 50	# GS = 173
MRI	401.7 mins - 928.4 mins	214.4 mins - 641.9 mins	22.0 mins - 226.6 mins
MI	12.7 mins - 448.6 mins	5.9 mins-369.1 mins	5.0 mins - 115.4 mins
AIS	2.2 mins - 313.8 mins	2.2 mins - 296.3 mins	2.2 mins - 86.5 mins

TABLE II: Attainable delivery latency by existing GS-based approaches. MRI: Moderate Resolution Imaging. MI: Multi-angle Imaging. AIS: Atmosphere Infrared Sounder.

1) Data download via ground station networks: As the sensing satellite moves in LEO, there is a *visible window* during which the sensing satellite is in the transmission range of a ground station. Therefore, one straightforward yet practical way to fetch EO data is to download data during the visible window of a single ground station. While practical, this *"store data first, and download later"* approach may suffer from high delivery latency due to a critical factor: the ground communication for LEO satellites is *intermittent* and *unstable*. More specifically, first, as the satellite moves in orbit and the Earth rotates underneath the satellite, it may take minutes to hours for the satellite carrying EO data to move into the transmission range of the ground station. Second, because sensing satellites move in very high speed, the visible window for a certain ground station is typically very short, *e.g.,* only ten minutes or less per day [65], and thus it may take multiple passes to completely download the entire EO data. Third, only relying on ground communication for data download is likely to be affected by poor weather conditions, as atmospheric attenuation may degrade the link quality between satellites and ground stations.

A recent effort [66] extends the amount of ground stations in above baseline approach, builds a geo-distributed ground station network to improve the availability of ground communication in one pass, and can reduce latency by collaboratively downloading EO data through a sequence of accessible ground stations. While using a ground station network extends the duration of visible window in one pass, it is still difficult to guarantee low latency for certain applications (*e.g.,* maritime readiness and response) due to a fundamental reason: *ground stations are difficult to be deployed upon oceans which cover more than 70 percent of the surface of our planet.* As the satellite moves and the Earth rotates simultaneously, it is hard to make sure that there are always available ground stations under the satellite for stable and long-duration data download.

Table II summarizes the delivery latency of using different numbers of ground stations to gather EO data. In our quantitative analysis, we set the location of ground stations and link capacity following the parameter settings in [66], and set the data rate of ground communication to 800Mbps. Although a number of distributed ground stations are used, the download latency still suffers from high variation ranging from minutes to hours as the source moves and Earth rotates, and the minimal latency can only be obtained if the observed area is just close to a nearby available ground station.

2) Data download through satellite relays: Another method for downloading EO data is using geostationary satellites as the relay for data transmission. For example, the European Data Relay System (EDRS) system [72] is a European constellation that relays information and data between satellites, spacecraft,

Authorized licensed use limited to: University of Regina IEL. Downloaded on November 20,2025 at 03:38:45 UTC from IEEE Xplore. Restrictions apply.

and ground stations. Similarly, the U.S. Tracking and Data Relay Satellite (TDRS) system is a network constructed by tracking and data relay satellites designed to satisfy the requirement for long-duration, highly-available space-to-ground communications [34], [64], [71]. With TDRS spacecrafts located in three geosynchronous regions around the Earth, customer spacecrafts (*e.g.,* sensing satellites in LEO) can attain global support for data relay and distribution. Using satellite relays, EO data is first transmitted from the sensing satellite to relay satellites working in geostationary orbit, and then be forwarded to ground stations and EO data centers later.

Quantitatively, the average download latency of delivering EO data via the TDRS system is 12.1/24.0/67.8 minutes for three EO datasets respectively. As exploiting GEO satellite relays enables long-duration and stable LEO$\rightarrow$GEO$\rightarrow$ground delivery paths, it achieves faster data download but still takes tens of minutes to hours to complete the downloading process, which might not be doable for time-sensitive applications. The root cause is that while state-of-the-art GEO satellite relays can be equipped with long-distance laser communication component with up to 1.8Gbps datarate [72], the end-to-end throughput is still constrained by the GEO$\rightarrow$ground path with a datarate up to 300Mbps [64], [72]. Another critical limitation of using GEO satellite relay is that it does not scale well for various geo-distributed EO tasks, as the number of sensing satellites increases. Specifically, the supported number of user-spacecrafts per satellite relay is very limited (*e.g.,* only 2 for each relay in TDRS [64]) due to the limited on-board payload weight available for high-speed laser communication components. This indicates that while covering about one-third of Earth surface, one GEO satellite relay can only support two EO tasks simultaneously. Finally, the manufacturing and launch costs of GEO satellites, together with the high-speed laser communication components are extremely high, *e.g.,* about $544 million for one EDRS relay [6]. Such high costs make it (likely) difficult to massively deploy GEO satellite relays to support many LEO sensing satellites concurrently.

C. Emerging Mega-Constellations: A Promising Enabler for Low-Latency EO?

Given that existing approaches are unlikely to guarantee very low latency EO data delivery (*i.e.,* in minutes), next we envision the opportunity and potential of a novel approach: *can emerging LEO mega-constellations help to enable low-latency EO data delivery?*

1) LEO mega-constellations: Over the past decade, the renaissance in the space industry [31], [39] declines the cost of access to space [41] (*e.g.,* the per-satellite cost of Starlink is below $500K [20]) and stimulates an exponential growth in constructing "New Space" mega-constellations. Many companies such as SpaceX, Amazon *etc.* are actively deploying their mega-constellations (*e.g.,* Starlink [22], Kuiper [1]) comprising thousands of satellites inter-connected by inter-satellite links (ISLs) [45], promising to offer broadband Internet service with potentially lower latency [43], [52] and high throughput [40], [68]. Unlike the first generation of satellite network which exploits satellites in geostationary orbit to forward data, emerging mega-constellations operate in low Earth orbit (LEO) with evolved computation and network capability [28], [29], [37], [42]. For example, in a recent beta test of Starlink shows the network performance with about 120Mbps downlink data rate and 20ms latency [23], while its predecessor HughesNet offers network access with about 19.8Mbps downlink data rate and about 800-900ms latency.

2) Opportunities and challenges of low-latency EO data delivery over emerging mega-constellations: We thus envision a new opportunity for low latency EO data delivery: *unlike existing "store first and download later" approaches, leveraging LEO constellations to pipeline the data download and distribution steps, and directly deliver EO data from the remote sensing satellite to users*. Intuitively, building a delivery network upon LEO constellation for EO has two potentials that may enable lower latency. First, emerging mega-constellations consist of thousands of LEO satellites and provide network access globally. Thus the sensing satellite can start to deliver data *wherever* it is, and does not have to wait for reaching the transmission range of a certain available ground station. Second, most planned constellations suggest the use of laser inter-satellite links that promise to achieve tens of Gbps or higher data rate for inter-satellite communication [45]. Hence emerging mega-constellations are likely to construct a satellite network which can potentially provide low-latency, high-throughput communication from the sensing satellite to terrestrial users. In addition, in terms of scalability, mega-constellations can also support multiple sensing satellites delivering EO data simultaneously. Each broadband satellite might equip with several ISLs (*e.g.,* five laser links might be doable for a Starlink satellite according to [43]), and thus in practice different sensing satellites can connect to a (different) nearby broadband satellite for data deliver. Note that as sensing satellites are typically working in LEO, with the orbital altitude and inclination similar to a subset of satellites in the mega-constellation. Therefore if the access satellite is selected properly, the relative velocity between the sensing and broadband satellites could be low, leading to stable connections between the sensing and broadband satellites. Collectively, if above potentials regarding low latency and scalability can be utilized, EO applications are likely to purchase the network services from constellation operators (*e.g.,* SpaceX), and exploit the delivery capability of emerging mega-constellations to distribute EO data efficiently.

Although promising, building such a delivery network in space is still very challenging, due to two specific characteristics in such emerging LEO satellite networks: **(i) Topology dynamicity in LEO satellite networks.** One unique aspect that differentiates LEO satellite network from other terrestrial networks is that: LEO satellites are moving fastly in orbits, with respect to the Earth and other satellites or ground stations. Since satellite links have limited range, the connectivity between satellites and ground stations/users changes over time. The dynamic network topology makes it difficult to establish and maintain a stable route for delivering EO data using existing IP-based routing protocols (*e.g.,* OSPF) [42], [74], [75]. Since

Authorized licensed use limited to: University of Regina IEL. Downloaded on November 20,2025 at 03:38:45 UTC from IEEE Xplore. Restrictions apply.

IP addresses are coupled with the network interface of a device, the frequent link connection and disconnection can trigger the re-calculation of routes, resulting in significant *routing convergence overhead* and *address updating overhead*, as routers have to inform neighborhoods the link state change. **(ii) Rare bandwidth resource in space.** Although the rapid evolution of on-board technology has led to the development of high data-rate ISLs which promise to provide tens and even hundreds of Gbps data rate [45] between satellites, in practice, maintaining high data rate requires more power allocation in each satellite [58], [73]. Unlimited use of energy in space will drain the lifetime of a satellite quickly, because there is a limitation on the maximum amount of complete recharge/discharge cycles of battery [73]. Therefore, EO data delivery is expected to be bandwidth-efficient and do not incur too much power consumption on satellites, especially when serving a large number of terrestrial users.

III. System Design

We present OrbitCast, a low-latency framework that collaboratively exploits emerging mega-constellations and distributed ground stations to deliver big EO data from remote sensing satellites to a collection of terrestrial users.

A. OrbitCast *Overview*

1) Design principles: The design of OrbitCast should satisfy three primary goals: (i) **Low data delivery latency.** As many time-sensitive EO applications require low latency, it is expected that the framework can deliver EO data to users timely, *e.g.,* within five or less minutes; (ii) **Scalability.** EO data might be delivered to a number of customers/users who are interested in using, analyzing and understanding the data. The system is expected to achieve good scalability, *e.g.,* being doable to guarantee low latency even if the number of source sensing satellites and end users significantly increases; (iii) **Bandwidth-efficiency.** As the resource in satellite such as power or link bandwidth is more scarce as compared to that in terrestrial networks, the data delivery process is not expected to impose too much traffic overhead (including both control and data flows) on inter-satellite links.

2) Key ideas of OrbitCast*:* OrbitCast adopts the following key ideas to deliver EO data from remote sensing satellites to terrestrial users: **(i) Constructing a hybrid delivery network upon emerging LEO constellations, with the supplement of geo-distributed ground stations offered by GSaaS for EO data delivery.** The sensing satellite delivers data through ground station infrastructure, *e.g.,* Ground-Stations-as-a-Service (GSaaS), if it is in the transmission range, because state-of-the-art ground stations (*e.g.,* [50]) typically can support much higher downlink data rate as compared to small dish-like satellite user terminals. If ground stations are unavailable (*e.g.,* out of transmission range), OrbitCast delivers data to users over the routes built upon LEO constellations. The hybrid network facilitates the opportunity of enabling long-duration and low-latency communication from the sensing satellite to users *whenever* the data is acquired. **(ii) In the hybrid network where the topology changes over time, exploiting a *location-driven* routing scheme to deliver EO data to users.** OrbitCast exploits a geo-location based, adaptive addressing scheme to index nodes and drive the data delivery in the hybrid network. The location-based scheme helps to significantly reduce the control overhead incurred by calculating and converging route in the dynamic topology, and also supports multicast in space to attain improved scalability and bandwidth-efficiency when the number of users increases.

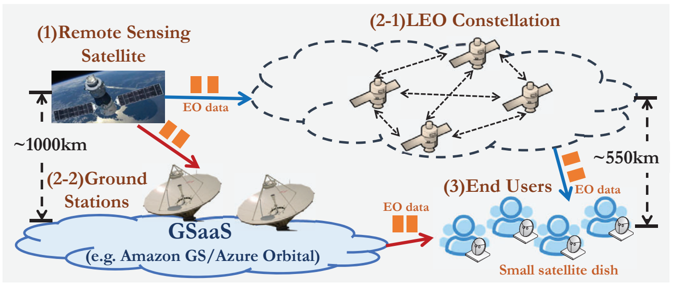


Fig. 2: OrbitCast system overview.

3) System overview and workflow: Figure 2 plots the system overview of OrbitCast. Collectively, the OrbitCast architecture consists of three key segments in the integrated satellite and terrestrial network: (1) the *sensing segment*, which includes one or more remote sensing/observation satellites working in (typically) LEO for collecting the raw EO data; (2) the *hybrid delivery network* which is built upon emerging constellations constructed by a considerable number of LEO satellites, together with geo-distributed ground stations offered by GSaaS operators (*e.g.,* Amazon Ground Stations [26] and Azure Orbital [3]); (3) the *terrestrial segment* that includes end users from different industries requesting EO data. The user can access GSaaS via today's terrestrial network, and connect to LEO satellites via very-small-aperture terminal (VSAT), *e.g.,* a SpaceX's Starlink satellite dish [25].

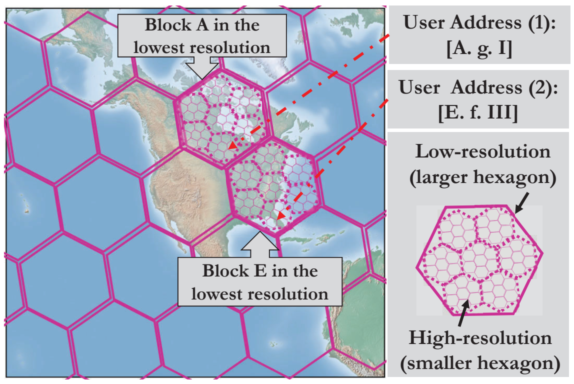
All nodes in the hybrid network, including LEO satellites, ground stations and users are indexed based on their current location (§III-B). Once the sensing satellite has acquired the data from space, it connects to a nearby available LEO broadband satellite or ground station to start the data delivery immediately, and the next hop is calculated based on the location. OrbitCast delivers EO data to one or a collection of distributed users over the route constructed upon LEO satellites and ground stations (§III-C).

B. *Geo-location-based Addressing*

1) Addressing scheme: Before building the EO data delivery network in space, OrbitCast first needs to index and identify each node (*e.g.,* LEO satellites, ground stations and terrestrial users) in the network for packet forwarding. Specifically, the OrbitCast framework follows a four-step process to address nodes in the integrated satellite and terrestrial network.

Step-I: building a grid system which divides the Earth surface into a number of *blocks*. Figure 3 plots a regional example of the grid system used in OrbitCast. Exploiting the grid system, OrbitCast classifies users and satellites

Authorized licensed use limited to: University of Regina IEL. Downloaded on November 20,2025 at 03:38:45 UTC from IEEE Xplore. Restrictions apply.


Block A in the lowest resolution
User Address (1): [A. g. I]
User Address (2): [E. f. III]
Low-resolution (larger hexagon)
High-resolution (smaller hexagon)
Block E in the lowest resolution
Fig. 3: The hierarchical grid system used in ORBITCAST.

into hexagonal blocks based on the division method in [8]. ORBITCAST discretizes the Earth surface into hexagonal blocks for several reasons. First, blocks are *static* related to the Earth surface and have their dedicated index (*e.g.,* the longitude and latitude pair of the center of a block), and the index of each block can be used as an *"anchor"* that guides the data forwarding. Second, blocks can cluster users in nearby locations. When delivering data over LEO satellites, data flows can be merged before entering a certain block that covers many users to save bandwidth overhead (*e.g.,* block-location-based multicast). Third, the hexagonal block shape can well fit the circle coverage of LEO satellites. Using the hierarchical grid system, ORBITCAST classifies a set of nearby entries into blocks, and maps a certain location to a length-varying block index for its current block. Moreover, we tune the resolution of the grid system to achieve different edge length of the block. A higher resolution indicates a smaller size hexagon block, while a lower resolution refers to larger hexagons.

Step-II: calculating the address of each node according to its current geo-location in the grid system. Under the grid system above, each block under a certain division resolution has a unique block index. Figure 3 also shows an example of a three-resolution grid division (*i.e.,* resolution set = $0, 1, 2$, where 0 is the lowest resolution with the largest hexagon block). Each block is further divided into seven smaller hexagons in a higher resolution. We then calculate the address of a node with location $(latitude, logitude)$ as the concatenation of all its block index under different resolutions. For example, in the case of Figure 3, two users are located in block A/g/I and E/f/III respectively under division of the resolution 0/1/2. Therefore their address are calculated as [A.g.I] and [E.f.III].

Step-III: conflict avoidance. Note that in some cases where two nodes are very close, after Step-II these two nearby nodes may have the same address, even though the most fine-grained division is applied. To avoid the address conflict, we add a suffix at the end of the address obtained from Step-II. Block indexes are cascaded from the lowest resolution to the highest resolution, followed by a suffix number for conflict avoidance.

Step-IV: address maintenance and update. Each LEO satellite in ORBITCAST maintains the address of itself, together with all addresses of its visible neighborhoods (*e.g.,* adjacent satellites, available ground stations or user terminals). In addition, because LEO satellites move in high speed and their

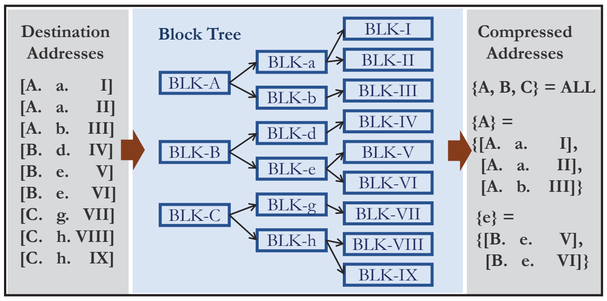


Fig. 4: Address compression by implicitly using the block tree.

geo-locations change over time, each satellite has to update its own address when it enters a new block, and then the satellite notifies its adjacent satellites for the update.

2) Packing the compressed address into packets: EO data might be delivered to a number of users. To indicate each forwarding satellite how to forward packets, the remote sensing satellite packs the addresses of all destinations into data packets (*e.g.,* after the IP header and before the payload). However, naively inserting all addresses into each packet is not doable as the number of end users increases. Leveraging the *location aggregation* property of ORBITCAST's address, we design a method for address compression/decompression which can significantly reduce the additional packing overhead.

Note that the geo-location based address used in ORBITCAST is location-aggregated, indicating that geographically close users may be covered by the same block, and are likely to have the same prefix. Such a property enables us to build a *block tree* that aggregates addresses that have equal prefix (*i.e.,* in the same block under a certain resolution). Figure 4 plots an example of the address compression with nine user addresses. Assume that each forwarding node in ORBITCAST framework knows all user addresses in advance. Thus each node can implicitly build and maintain a block tree containing all user addresses aggregated by prefix (as shown in the middle of Figure 4). Then, a set of addresses can be compressed and presented as a set of *rendezvous points* in the block tree. For example, these nine addresses shown in Figure 4 rendezvous at node A, B, and C. Therefore the addresses of all users in this case can be compressed as $\{A.B, C\}$, which means that a packet can pack $\{A.B, C\}$ to indicate that this packet should be forward to all nine users, instead of packing all nine addresses. Likewise, three addresses `[A,a,I]`, `[A,a,II]`, `[A,b,III]` in the same block A can be compressed as $\{A\}$, and two addresses `[B,e,V]` and `[B,e,VI]` can be compressed as $\{e\}$. The compression process is reversible and the original address can be recovered by searching the implicitly block tree.

C. Geo-location Driven Data Delivery

1) Distributed delivery algorithm: The proposed delivery method is a distributed algorithm running in each satellite, with no requirements on centralized computation at runtime. Specifically, each satellite makes the forwarding decision based on: (1) the current address of itself; (2) addresses of all its adjacent satellites and ground stations; and (3) addresses of all

Authorized licensed use limited to: University of Regina IEL. Downloaded on November 20,2025 at 03:38:45 UTC from IEEE Xplore. Restrictions apply.

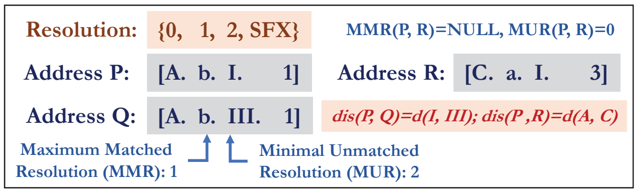

Resolution: {0, 1, 2, SFX}
MMR(P, R)=NULL, MUR(P, R)=0
Address P: [A. b. I. 1]
Address R: [C. a. I. 3]
Address Q: [A. b. III. 1]
dis(P, Q)=d(I, III); dis(P ,R)=d(A, C)
Maximum Matched Resolution (MMR): 1
Minimal Unmatched Resolution (MUR): 2

Fig. 5: An example of calculating MMR, MUR and the distance between two OrbitCast addresses.

destinations. Before we introduce the algorithm in details, we first clarify two important definitions.

D1: Maximum matched resolution (MMR) and minimum unmatched resolution (MUR). We define the *maximum matched resolution (MMR)* between two addresses as the resolution under which all block indexes are equal in the two addresses. MUR is the next resolution of MMR, *i.e.,* MUR=MMR+1. Figure 5 shows an example explaining the MMR and MUR between different addresses. Considering a three-level resolution division where 0 is the lowest resolution. The MMR is 1 for address P(`[A.b.I.1]`) and Q(`[A.b.III.1]`), and is `NULL` for address P(`[A.b.I.1]`) and R(`[C.a.I.3]`). Similarly, the MUR is 2 for address P(`[A.b.I.1]`) and Q(`[A.b.III.1]`), and is 0 for address P(`[A.b.I.1]`) and R(`[C.a.I.3]`).

D2: Distance of two addresses. We define the distance of two addresses as the great circle distance between the center of two blocks under the MUR. For example, in the case shown in Figure 5, the MUR between P and Q is 2, and thus the distance between P and Q is calculated as the great circle distance between the centers of block I and III (*e.g.,* $dis(P, Q) = d(I, III)$). Similarly, the distance between P and R is calculated as the great circle distance between the centers of block A and C (*e.g.,* $dis(P, R) = d(A, C)$). Assume $d(x, y)$ is the great circle distance between two points x and y.

Shortest distance match. For a forwarding satellite, the key operation is to resolve received packets, and decide which network interface (an/a inter-satellite or satellite-ground link) the packet should be forwarded to. OrbitCast follows the *shortest distance match* principle to make the forwarding decision. A packet is sent to the network interface which connects to a next node much more closer to the destination.

Algorithm 1 shows the details of our delivery algorithm. Once a forwarding satellite receives a packet, it decompresses the packet header to extract the destination address list (DAL) following the method introduced in Section III-B2 (line 4). The $forward\_tb$ is an array where the ith entry is a list of addresses, (*i.e.,* $forward\_tb[i]$). Packets to $forward\_tb[i]$ are forwarded by the ith network interface in the current satellite. Initially, $forward\_tb$ is set to $\{\varnothing\}$ (line 5). For each destination address, the next-hop is calculated as the one closest to the destination from all adjacent satellites/ground stations (line 6-13). Then the header is updated and packed into the packet, and finally the packet is forwarded according to $forward\_tb$ (line 15-18).

Algorithm 1 Geo-location-based data delivery algorithm.

```
1:  Input: (1) list of adjacent addresses (LAA); (2) compressed
    destination address (CDA).
2:  Output: forwarding decision.
3:  /* obtain the destination address list (DAL). */
4:  $dst\_list$=decompress(CDA)
5:  initialize $forward\_tb$ array = $\{\varnothing\}$
6:  for $addr_i$ in $dst\_list$:
7:      if $addr_i$ is visible: forward packet to $addr_i$
8:      else
9:      /* forward pkt to a node closer to destination. */
10:         $next\_hop \leftarrow arg_{n \in LAA} \min(dis(n, addr_i))$
11:         $forward\_tb[next\_hop]$.append($addr_i$)
12:     end if
13: end for
14: /* ith interface in $forward\_tb$ is responsible for destina-
    tions in $forward\_tb[i]$. */
15: for $addr\_list$ in $forward\_tb$:
16:     pack compress($addr\_list$) into the packet.
17:     forward the packet to corresponding interface.
18: end for
```

2) Putting it together: Collectively, the process of delivering EO data from space can be summarized as follows: **(1) Join or quit operation.** Users who are interested in EO data may join or quit the OrbitCast framework. The user broadcasts the join/quit message carrying the location information to all forwarding satellites in the constellation. In practice this can be done by registering/un-registering in a terrestrial control center. Then each forwarding satellite knows the entire destination address list; **(2) Node addressing.** Each node is addressed by the geo-location based addressing method; **(3) Data collection and packetization.** When the sensing satellite collects data (e.g. high-resolution EO images), it packetizes original content and packs the compressed destination address list into packets and starts data delivery via connecting to an available ground station or a forwarding satellite with the similar altitude, inclination and orbital direction in the mega-constellation; **(4) Packet forwarding.** LEO broadband satellites follow Algorithm 1 to route packets from the source to all destinations.

D. Congestion Avoidance and Fault Tolerance

Congestion avoidance. When the number of sensing satellites or end users increases, data traffic from different EO tasks might contend bandwidth resources in ISLs or ground communication link. To avoid overload and congestion collapse in the hybrid space-ground network, OrbitCast can exploit existing schemes like BBR [30] or TFMCC [62], MTCP [61] to achieve congestion avoidance in unicast or multicast mode.

Handling link failures. While emerging mega-constellations plan to launch thousands of small, short-lived satellites to provide low-latency high-throughput network services globally, the lifespan of those small satellites is relatively short (*e.g.,* around five years [48]) as compared with traditional high-cost communication satellites in GEO. According to a recent report [55], the failure rate of Starlink is about 2.5%, indicating

Authorized licensed use limited to: University of Regina IEL. Downloaded on November 20,2025 at 03:38:45 UTC from IEEE Xplore. Restrictions apply.

that about 1 in 40 of Starlink satellites may have failed. The geo-location-based delivery algorithm proposed in ORBITCAST can tolerate such link failures, as it calculates the forwarding decision locally, and does not require each node to maintain the entire network topology, which may cause route re-convergence in case of failures.

IV. Testbed and Prototype Implementation

Our testbed integrates three techniques described below to improve the fidelity of our experimental environment.

(i) Exploiting a number of Mininet [11] instances to emulate nodes in the hybrid network. Mininet is a network emulator which relies on `cgroups` and `network namespaces` to create a network of virtual hosts, routers, and links, and each Mininet instance can run standard Linux software. We build our testbed based on a number of inter-connected Mininet instances, and each instance emulates the network ability of a satellite/ground-station/user node, which can receive and parse realistic data traffic.

(ii) Using real orbital information to drive the simulation of satellite movements. Since we use Mininet instances to emulate satellites, the inter-connectivity and varying location of these network nodes are configured based on real orbital information. We collect the latest orbital information of mega-constellations from CeleTrak [33] and the open FCC filing. We then use orbit analysis tools [17] to calculate the accessibility and location of each satellite in different time slots.

(iii) Simulating the low-layer link quality. To improve the fidelity of the testbed, we simulate other low-layer factors that may affect the link quality in satellite networks. Specifically, we use ITU-Rpy [9] to simulate the impact of weather conditions on the atmospheric attenuation in slant and horizontal paths. Further, we use Linux `tc` to adjust the propagation delay among satellites according to their time-varying distance and the speed of light. The entire testbed is implemented and deployed on a DELL-R740 server with two Intel Xeon 5222 Processors (four-core, 3.8GHz) and 8*32G DDR4 RAM.

We have implemented the core functionality of ORBITCAST in around 1436 lines of C codes. Specifically, the hexagon grid system used in ORBITCAST is implemented based on H3 [8], which is originally used in terrestrial navigation systems. We set four levels of resolution in our implementation, and the average hexagon edge length in each resolution is about $1100km$, $420km$, $160km$ and $60km$ respectively. The length of a single block index ranges from 7 to 16 bits, according to the resolution. A lower resolution indicates a larger block size.

V. Performance Evaluation

Evaluations in this section aim at answering the following three questions. **Q1:** What is the achievable network performance of ORBITCAST, for example the ability of EO data transfer and corresponding delivery latency, as compared with other existing approaches under representative EO data traffic? **Q2:** How does ORBITCAST scale to the increasing number of EO data sources (*i.e.,* remote sensing satellite) and terrestrial users? **Q3:** What is the impact of various constellation designs on the attainable delivery latency?

A. Experiment Setup

Data source and user distribution. We use three real-world EO data sets collected by NASA's LANCE EO system as the EO traffic [2], [12], [13]. Orbital information of remote sensing satellites is obtained from [33]. In our experiment, we assume each end user has equipped a light-weight user terminal (*e.g.,* a satellite dish) [21] which can directly communicate to a LEO satellite. We use the public information of 173 ground stations provided by [63] to configure the distribution of available ground stations in our experiments. We select 113 organizations spread in four different continents as end users. These organizations are control and monitoring centers, companies or technology laboratories that use EO data for emergency readiness/response, disaster risk management, maritime search and rescue, and other research proposes (*e.g.,* [15]).

Performance comparison. We compare the effectiveness of ORBITCAST with other four existing approaches, two of which have been introduced in Section II: (i) Distributed Ground Station (DGS) [66], which leverages a hybrid ground station model to improve robustness and reduce downlink latency of data delivery. (ii) Tracking and Data Relay Satellite System (TDRSS) [71] that utilizes three groups of geostationary satellites to provide stable communication from sensing satellites to ground stations. The network capacity for LEO$\rightarrow$GEO and ground communication is configured according to [64]. We pipeline the data download and data distribution step in DGS and TDRSS to accelerate their data delivery. (iii) source-routing data from the source to every destinations following the shortest path calculated by Dijkstra (*i.e.,* denoted as SR-SP), as proposed in [43]. Note that this scheme requires to know the entire network topology in advance to calculate the shortest path periodically. In [43], gathering the time-varying topology is done on the ground (*e.g.,* in a terrestrial control center). However, in our scenario, it is unpractical for the data source (*i.e.,* a sensing satellite) to work as a control center and monitor the whole network topology over time. Thus, in our experiment we re-calculate the shortest path to each user on a ground node, and transfer the route results to the sensing satellite through geostationary satellites. (iv) On-Demand Multicast Routing Protocol (ODMRP) [16] which is an IP-based multicast routing protocol used in today's terrestrial network.

Constellations. For most of the experiments conducted in this section, we use SpaceX's Starlink as the LEO constellation used for data delivery, since Starlink is currently the largest commercial LEO constellation under heavy deployment. We use the complete Starlink phase I constellation which will deploy 1584 satellites in 72 orbits at 550km altitude as the primary constellation setting in our experiment. We also evaluate the performance under other constellations, *e.g.,* Amazon Project Kuiper [1] which plans to launch 3,236 satellites operating in 98 orbital planes in three orbital shells. For each constellation, we follow a well-known *+Grid* topology [29], [43], [51] to inter-connect each satellite, *i.e.,* each satellite has at least five laser ISLs, two of which are connected to the two neighborhoods in the same orbit, and other two links are connected to the

Authorized licensed use limited to: University of Regina IEL. Downloaded on November 20,2025 at 03:38:45 UTC from IEEE Xplore. Restrictions apply.

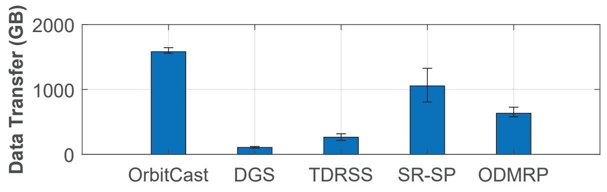


Fig. 6: Transferred data volume in one day.

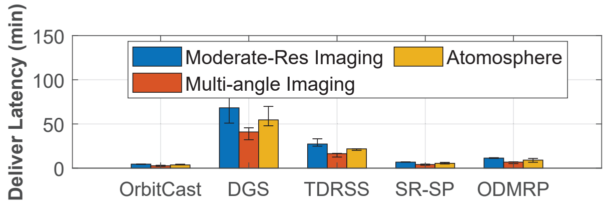


Fig. 7: Delivery latency for representative EO data traces.

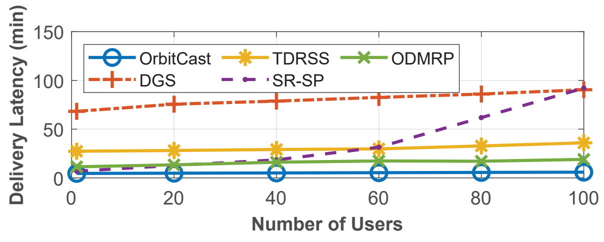
left/right satellite in the adjacent orbit. At least one ISL is available for connecting to a nearby sensing satellite.

B. Data Transfer

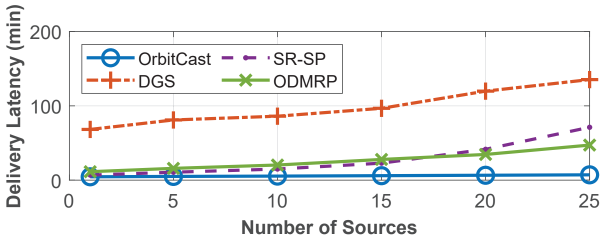
We first evaluate the ability of delivering EO data from the sensing satellite to terrestrial users via different delivery approaches. We randomly pick the initial phase of the sensing satellite and the location of terrestrial user from our data set in each run. Figure 6 plots the average data volume downloaded from remote sensing satellite via different approaches in one day. DGS downloads about 105GB data on average due to the intermittent ground communication during the orbit, and TDRSS downloads about 264GB data using the stable LEO→GEO→ground path. Compared to these existing approaches based on ground station networks or geostationary satellite, ORBITCAST increases the ability of data transfer over 14x and can gather about 1582GB data per day, since it leverages satellites and distributed ground stations to construct a delivery network, and data download can start immediately via high-throughput space route since the data acquisition.

C. Delivery Latency

We then evaluate the latency of delivering three types of representative EO data traffic to terrestrial users. EO data objects are sequentially acquired by the sensing satellite periodically and differ in size. Specifically, the delivery latency is measured as the time between data acquisition and its delivery to terrestrial users. Figure 7 shows the latency of transferring three types of EO traffic from one source to a single randomly picked terrestrial user under different delivery approaches in Starlink constellation. Through pipelining data download and data distribution, DGS and TDRS achieve 40-68 minutes and 16-27 minutes latency on average. Other three methods based on LEO constellations, *i.e.,*ORBITCAST, SR-SP and ODMRP obtain comparable performance results as they exploit the route built upon LEO satellites inter-connected by ISLs. ORBITCAST attains about 2.7-4.5 minutes for data delivery. The latency of SR-SP is about 10 minutes because the network topology changes over time, and the sensing satellite has to wait for the converged routing decision from the terrestrial control center, and the communication is interrupted during the routing convergence. The latency results under Kuiper constellation is similar to Starlink. But SR-SP obtains higher latency because it takes more time to re-calculate the path and thus has higher convergence time due to the increased amount of satellites.



(a) Delivery latency under different number of users.



(b) Delivery latency under different number of EO sources.

Fig. 8: Latency results of different approaches as the number of users/sources increases.

We next examine the network performance of each approach as the number of sources or users increases. Figure 8 shows the latency results of each approach when we change the number of source sensing satellites from 1 to 25 and tune the user amount. Note that the number of simultaneously supported spacecrafts for a GEO satellite relay in TDRSS is limited, we do not plot the results of TDRSS for many EO sources. Figure 8a shows that as the number of users increases, the delivery latency of DGS keeps stable because they first download EO data to ground operation and data storage centers, and then distribute EO data to users. The increase of user amount does not impose addition network traffic on satellite-ground downlinks of DGS and TDRSS. While SR-SP calculates and uses the shortest path in LEO constellations to route data, the increasing number of users imposes congestion on ISLs and the available throughput for each user on average is reduced. In ORBITCAST, inter-satellite traffic is saved as data flow splits in the block close to end users, and thus it achieves better network efficiency by avoiding redundant traffic over ISLs. Hence ORBITCAST obtains the lowest delivery latency as user amount increases.

Similarly, Figure 8b plots the latency when multiple sources simultaneously collect and deliver EO data to all users. The average delivery latency increases when we add more sensing satellites as the data source, since many delivery flows rendezvous in ISLs and compete for bandwidth. ORBITCAST stays in the lowest delivery latency since it aggregates delivery traffic for different users and reduces strain on network links.

D. Traffic Overhead and Power Consumption

Finally we evaluate the traffic overhead imposed by delivering EO data through LEO satellite networks. We calculate the average transferred data volume carried by each ISL when delivering data to a number of terrestrial users. We measure

Authorized licensed use limited to: University of Regina IEL. Downloaded on November 20,2025 at 03:38:45 UTC from IEEE Xplore. Restrictions apply.

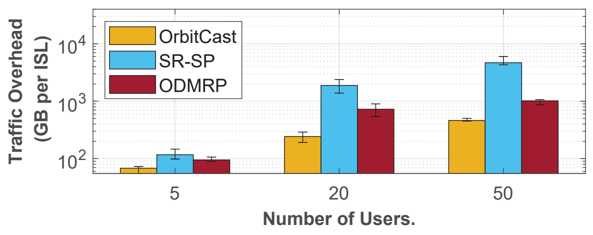
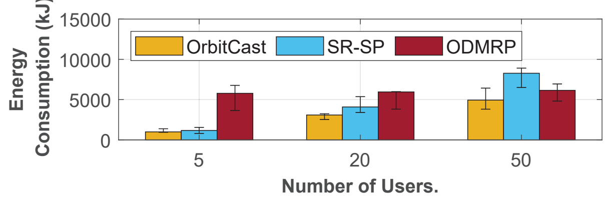
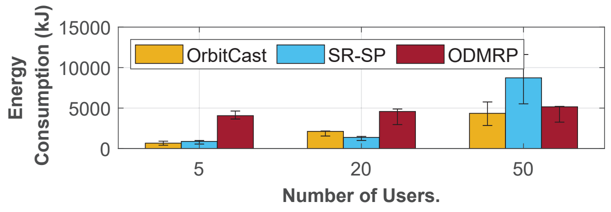


Fig. 9: Average traffic overhead incurred in each ISL when delivering EO data to a collection of users.



(a) Energy consumption in SpaceX's Starlink.



(b) Energy consumption in Amazon's Kuiper.

Fig. 10: Estimation of power consumption.

the traffic overhead of these three approaches that rely on LEO constellations. Figure 9 plots the average data volume carried by each ISL when completely delivering EO data collected in one day to all users. Because SR-SP source-routes data to all users via independent flows, it achieves the highest per-ISL traffic overhead. ORBITCAST attains the lowest traffic overhead in each constellation as it supports multicast and avoids redundant inter-satellite traffic.

Note that the usage of ISL can significantly affect the power consumption in each satellite. We use the power model in [47], [53] with optical power of 2W to set the power of ISLs during the full-on period, and estimate the total power consumption involved by EO data delivery. As plotted in Figure 10. ORBITCAST achieves the lowest energy consumption as it reduces the transmission time and traffic overhead involved by inter-satellite communication.

VI. RELATED WORKS

Delivering Earth observation data. As we have introduced in Section II, most existing approaches for delivering EO data can be summarized as GS-based [66] and geostationary satellite based [71]. Authors in [37] propose to offload partial computation to the satellite to reduce the downlink traffic overhead, and thus decrease the delivery latency. However, in certain scenarios, with insufficient knowledge of end applications, such a pre-processing approach might omit critical information that is important to users [66]. Different with prior efforts, ORBITCAST exploits a hybrid model upon LEO constellations and GSaaS to deliver entire EO data to users.

Exploring emerging satellite networks. There is an increasing number of studies focusing on building a satellite network upon emerging LEO constellations [27], [35], [36], [42], [42]–[44], [67]. Mark *et al.* studied how to use the laser links or ground stations to provide a network and studied the problem of routing in this network [43], [44], based on public details from the FCC filings. Giacomo *et al.* studied a path-aware networking architecture in which end-hosts obtain information and control over network paths [29]. Yannick *et al.* explored the network behaviors resulting from whether ISLs are enabled [45]. Prior efforts [54], [57] have studied geocast routing protocols mainly for mobile ad-hoc networks. Those previous explorations complement our work. Our location-driven routing scheme proposed in this paper exploits the predictable characteristics of mega-constellations (*e.g.,* satellite connectivity and trajectory), and focuses on building a geocast network for efficiently delivering EO data from the sensing satellite to a number of terrestrial users.

Satellite simulator. To support the assessment of system and protocol design in satellite networks, several simulation tools were proposed in previous studies. SNS3 [19] is a high-fidelity ns3-based simulator for satellite communications. However SNS3 is built on a static system configuration, with only one geostationary satellite and does not support LEO constellations in its current version. Hypatia [49] is a framework for simulating and visualizing the packet-level network behavior of these constellations by incorporating their unique characteristics, and StarPerf [52] is a simulation platform that enables the estimation of the achievable performance under a variety of constellation options. Different from previous simulator, the testbed proposed in this work can load real EO traffic and run network protocol suite, and thus support the emulation of protocol designs above the network layer.

VII. CONCLUSION

This paper presents ORBITCAST, which exploits ground station services and emerging mega-constellations that consist of thousands of LEO satellites to construct a low-latency and scalable delivery network in space. Specifically, to handle the high-dynamicity in the satellite network and enable long-duration, high-throughput communication from the remote sensing satellite to terrestrial users, we propose a geo-location driven delivery algorithm to route and forward the packet, and support multicast if the number of users increases. Extensive evaluations show that ORBITCAST enables low-latency EO data distribution for representative EO data traces, and scales well as the number of data sources and users increases.

VIII. ACKNOWLEDGEMENTS

We gratefully appreciate the feedback by the anonymous ICNP reviewers, and we thank Tim Griffin for the caring and shepherding of the paper. This work is supported by National Key Research and Development Program of China (No. 2018YFB1800303), NSFC (No. 61832013) and China Postdoctoral Science Foundation (No. 2021M691786).

Authorized licensed use limited to: University of Regina IEL. Downloaded on November 20,2025 at 03:38:45 UTC from IEEE Xplore. Restrictions apply.

References

[1] Amazon kuiper. https://www.geekwire.com/2019/amazon-project-kuiper-broadband-satellite/.

[2] Atmosphere infrared sounder (airs). https://earthdata.nasa.gov/earth-observation-data/near-real-time/download-nrt-data/airs-nrt.

[3] Azure orbital: Satellite ground station and scheduling service connected to azure for fast downlinking of data. https://azure.microsoft.com/en-us/services/orbital/.

[4] Copernicus emergency management service. https://emergency.copernicus.eu/.

[5] Copernicus sentinel-6: Monitoring the global ocean. https://www.eumetsat.int/.

[6] European data relay system (edrs). https://www.aerospace-technology.com/projects/european-data-relay-system-edrs/.

[7] European forest fire information system (effis). https://effis.jrc.ec.europa.eu/.

[8] H3 hexagonal hierarchical spatial index. https://eng.uber.com/h3/.

[9] Itu-rpy. https://github.com/iportillo/ITU-Rpy.

[10] Lemur 2 zo satellite information. https://www.n2yo.com/satellites/?c=49.

[11] Mininet. http://mininet.org/.

[12] Moderate resolution imaging spectro radiometer. https://earthdata.nasa.gov/earth-observation-data/near-real-time/download-nrt-data/modis-nrt.

[13] Multi-angle imaging spectro radiometer (misr). https://earthdata.nasa.gov/earth-observation-data/near-real-time/download-nrt-data/misr-nrt.

[14] Nasa earth observation data. https://earthdata.nasa.gov/earth-observation-data/near-real-time/download-nrt-data.

[15] National oceanic and atmospheric administration. https://www.noaa.gov.

[16] On-demand multicast routing protocol (odmrp) for ad hoc networks. https://tools.ietf.org/html/draft-gerla-manet-odmrp-05.

[17] Satellite toolkit agi. https://www.agi.com/products.

[18] Satellitebased earth observation market by type, application, and geography forecast and analysis 2020-2024. https://www.technavio.com/report/satellite-based-earth-observation-market-industry-analysis.

[19] Sns3. https://www.sns3.org/content/home.php.

[20] Spacex starlink satellites could cost $250,000 each and falcon 9 costs less than $30 million. https://www.nextbigfuture.com/2019/12/spacex-starlink-satellites-cost-well-below-500000-each-and-falcon-9-launches-less-than-30-million.html.

[21] Spacex's starlink user terminal. https://arstechnica.com/information-technology/2020/12/teardown-of-dishy-mcflatface-the-spacex-starlink-user-terminal/.

[22] Starlink. https://www.starlink.com/.

[23] Starlink beta users report positive results even in a forest. https://www.slashgear.com/starlink-beta-users-report-positive-results-even-in-a-forest-02645539/.

[24] Ucs satellite database. https://www.ucsusa.org/resources/satellite-database.

[25] Evelyn Arevalo. Spacex's starlink 'ufo on a stick' user terminal prototypes revealed in photos. https://www.tesmanian.com/blogs/tesmanian-blog/ufo-starlink-terminal.

[26] AWS. Amazon ground station launches new antenna location. https://aws.amazon.com/about-aws/whats-new/2020/11/aws-ground-station-launches-new-antenna-location-in-hawaii-usa/.

[27] Debopam Bhattacherjee, Waqar Aqeel, Ilker Nadi Bozkurt, Anthony Aguirre, Balakrishnan Chandrasekaran, P. Brighten Godfrey, Gregory Laughlin, Bruce Maggs, and Ankit Singla. Gearing up for the 21st century space race. In *Proceedings of the 17th ACM Workshop on Hot Topics in Networks*, HotNets'18, pages 113–119, New York, NY, USA, 2018. Association for Computing Machinery.

[28] Debopam Bhattacherjee, Simon Kassing, Melissa Licciardello, and Ankit Singla. In-orbit computing: An outlandish thought experiment? In *Proceedings of the 19th ACM Workshop on Hot Topics in Networks*, HotNets '20, page 197–204, 2020.

[29] Debopam Bhattacherjee and Ankit Singla. Network topology design at 27,000 km/hour. In *Proceedings of the 15th International Conference on Emerging Networking Experiments And Technologies*, pages 341–354, 2019.

[30] Neal Cardwell, Yuchung Cheng, C Stephen Gunn, Soheil Hassas Yeganeh, and Van Jacobson. Bbr: Congestion-based congestion control: Measuring bottleneck bandwidth and round-trip propagation time. *Queue*, 14(5):20–53, 2016.

[31] Kristen C Castonguay. Additive manufacture of propulsion systems in low earth orbit. Technical report, Air Command and Staff College, Air University Maxwell AFB United States, 2018.

[32] Celestrak. Norad two-line element set format description. https://www.celestrak.com/NORAD/documentation/tle-fmt.php.

[33] CelesTrak. Norad two-line element sets current data. https://www.celestrak.com/NORAD/elements/.

[34] Diane Davies, Karen Michael, Sherry Harrison, Daniel Ziskin, Phil B Durbin, Colin Seftor, Jessica Braun, Min Minnie Wong, Matthew Cechini, Ryan Boller, et al. Nasa's land, atmosphere near real-time capability for eos (lance): Delivering data and imagery to meet the needs of near real-time applications. 2018.

[35] I. del Portillo, B. Cameron, and E. Crawley. Ground segment architectures for large leo constellations with feeder links in ehf-bands. In *2018 IEEE Aerospace Conference*, pages 1–14, 2018.

[36] Inigo del Portillo, Bruce G Cameron, and Edward F Crawley. A technical comparison of three low earth orbit satellite constellation systems to provide global broadband. *Acta Astronautica*, 159:123–135, 2019.

[37] 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, page 939–954, 2020.

[38] B. Deng, C. Jiang, L. Kuang, S. Guo, J. Lu, and S. Zhao. Two-phase task scheduling in data relay satellite systems. *IEEE Transactions on Vehicular Technology*, 67(2):1782–1793, 2018.

[39] L. Dreyer. Latest developments on spacex's falcon 1 and falcon 9 launch vehicles and dragon spacecraft. In *2009 IEEE Aerospace conference*, pages 1–15, 2009.

[40] H. Fenech, S. Amos, A. Tomatis, and V. Soumpholphakdy. High throughput satellite systems: An analytical approach. *IEEE Transactions on Aerospace and Electronic Systems*, 51(1):192–202, 2015.

[41] Warren Frick and Carlos Niederstrasser. Small launch vehicles-a 2018 state of the industry survey. 2018.

[42] Giacomo Giuliari, Tobias Klenze, Markus Legner, David Basin, Adrian Perrig, and Ankit Singla. Internet backbones in space. *SIGCOMM Comput. Commun. Rev.*, 50(1):25–37, March 2020.

[43] Mark Handley. Delay is not an option: Low latency routing in space. In *Proceedings of the 17th ACM Workshop on Hot Topics in Networks*, HotNets '18, page 85–91, New York, NY, USA, 2018. Association for Computing Machinery.

[44] Mark Handley. Using ground relays for low-latency wide-area routing in megaconstellations. In *Proceedings of the 18th ACM Workshop on Hot Topics in Networks*, HotNets '19, page 125–132, New York, NY, USA, 2019. Association for Computing Machinery.

[45] Yannick Hauri, Debopam Bhattacherjee, Manuel Grossmann, and Ankit Singla. "internet from space" without inter-satellite links. In *Proceedings of the 19th ACM Workshop on Hot Topics in Networks*, HotNets '20, page 205–211, 2020.

[46] L. He, J. Li, M. Sheng, R. Liu, K. Guo, and D. Zhou. Dynamic scheduling of hybrid tasks with time windows in data relay satellite networks. *IEEE Transactions on Vehicular Technology*, 68(5):4989–5004, 2019.

[47] Li Zhengda Zhang Junjun Deng Rong Huang Feijiang, Lu Xiaochun and LiGuangcan. Design and implementation of the inter-satellite link budget software. *http://docsdrive.com/pdfs/academicjournals/jse/2015/230-241.pdf*, 2015.

[48] Anthony Iemole. Spacex launches first starlink mission of 2021. https://www.nasaspaceflight.com/2021/01/spacex-launch-first-starlink-mission-2021/.

[49] Simon Kassing, Debopam Bhattacherjee, André Baptista Águas, Jens Eirik Saethre, and Ankit Singla. Exploring the "internet from space" with hypatia. In *Proceedings of the ACM Internet Measurement Conference*, IMC '20, page 214–229, 2020.

[50] Eric Blossom Joseph Breu Bryan Klofas Kyle Colton Ryan Kingsbury Kiruthika Devaraj, Matt Ligon. Planet high speed radio: Crossing gbps from a 3u cubesat.

[51] Mark Krebs. Satellite constellation. us20170005719a1/en. *https://patents.google.com/patent/*, 2016.

Authorized licensed use limited to: University of Regina IEL. Downloaded on November 20,2025 at 03:38:45 UTC from IEEE Xplore. Restrictions apply.

[52] Z. Lai, H. Li, and J. Li. Starperf: Characterizing network performance for emerging mega-constellations. In *2020 IEEE 28th International Conference on Network Protocols (ICNP)*, 2020.

[53] LI Li, ZHANG Xuejiao, ZHANG Jianhua, XU Changzhi, and JIN Yi. Advanced space laser communication technology on cubesats. *ZTE Communications*, 18(4):45–54, 2021.

[54] Christian Maihofer. A survey of geocast routing protocols. *IEEE Communications Surveys Tutorials*, 6(2):32–42, 2004.

[55] Morgan McFall-Johnsen. About 1 in 40 of spacex's starlink satellites may have failed.. https://www.businessinsider.com/spacex-starlink-internet-satellites-percent-failure-rate-space-debris-risk-2020-10.

[56] Deimos Space. Murray Kerr. Novel satellite architectures for very low latency eo data products that meet societal needs. http://www.unoosa.org/documents/pdf/psa/activities/2019/UNAustria2019/KerrSatelliteArchitecture.pdf.

[57] Julio C. Navas and Tomasz Imielinski. Geocast—geographic addressing and routing. In *Proceedings of the 3rd Annual ACM/IEEE International Conference on Mobile Computing and Networking*, 1997.

[58] N. Pachler, J. J. G. Luis, M. Guerster, E. Crawley, and B. Cameron. Allocating power and bandwidth in multibeam satellite systems using particle swarm optimization. In *2020 IEEE Aerospace Conference*, pages 1–11, 2020.

[59] Gian Maria Pinna and Francesco Ferrante. The esa earth observation payload data long term storage activities.

[60] HK Ramapriyan. The role and evolution of nasa's earth science data systems. 2015.

[61] Injong Rhee, Nallathambi Balaguru, and George N Rouskas. Mtcp: Scalable tcp-like congestion control for reliable multicast. In *INFOCOM.*, volume 3, pages 1265–1273. IEEE, 1999.

[62] Luigi Rizzo. pgmcc: a tcp-friendly single-rate multicast congestion control scheme. *ACM SIGCOMM Computer Communication Review*, 30(4):17–28, 2000.

[63] SatNOGS. Open source global network of satellite ground-stations. https://satnogs.org/.

[64] Ted Sobchak, Donald W Shinners, and Harry Shaw. Nasa space network project operations management: Past, present and future for the tracking and data relay satellite constellation. In *2018 SpaceOps Conference*, page 2358, 2018.

[65] 2020. The European Space Agency. Types of orbits. https://www.esa.int/Enabling_Support/Space_Transportation/Types_of_orbits.

[66] Deepak Vasisht and Ranveer Chandra. A distributed and hybrid ground station network for low earth orbit satellites. In *Proceedings of the 19th ACM Workshop on Hot Topics in Networks*, HotNets '20, page 190–196, 2020.

[67] Florian Vidal, Hervé Legay, George Goussetis, Maria Garcia Vigueras, Ségolène Tubau, and Jean-Didier Gayrard. A methodology to benchmark flexible payload architectures in a megaconstellation use case. *International Journal of Satellite Communications and Networking*, 2020.

[68] O. Vidal, G. Verelst, J. Lacan, E. Alberty, J. Radzik, and M. Bousquet. Next generation high throughput satellite system. In *2012 IEEE First AESS European Conference on Satellite Telecommunications (ESTEL)*, pages 1–7, 2012.

[69] Y. Wang, M. Sheng, W. Zhuang, S. Zhang, N. Zhang, R. Liu, and J. Li. Multi-resource coordinate scheduling for earth observation in space information networks. *IEEE Journal on Selected Areas in Communications*, 36(2):268–279, 2018.

[70] Wikipedia. Earth remote sensing satellite introduction. https://en.wikipedia.org/wiki/Earth_observation_satellite.

[71] Wikipedia. Tracking and data relay satellite system. https://en.wikipedia.org/wiki/Tracking_and_Data_Relay_Satellite_System.

[72] ESA Witting, Harald Hauschildt, Andrew Murrell, Jean-Pascal Lejault, Josep Perdigues, Jean Lautier, Cedric Salenc, Khalil Kably, Heli Greus, Francois Garat, Hermann Moeller, Silvia Mezzasoma, DLR Meyer, Bjoern Guetlich, Sabine Philipp-May, Anke Pagels-Kerp, Frank Heine, Stefan Seel, Konrad Panzlaff, and Herbert Schuff. Status of the european data relay satellite system. 10 2012.

[73] Y. Yang, M. Xu, D. Wang, and Y. Wang. Towards energy-efficient routing in satellite networks. *IEEE Journal on Selected Areas in Communications*, 34(12):3869–3886, 2016.

[74] Z. Yang, H. Li, Q. Wu, and J. Wu. Analyzing and optimizing bgp stability in future space-based internet. In *2017 IEEE 36th International Performance Computing and Communications Conference (IPCCC)*, pages 1–8, 2017.

[75] G. Zheng, N. Wang, R. Tafazolli, and X. Wei. Geosynchronous network grid addressing for integrated space-terrestrial networks. In *2020 IEEE 28th International Conference on Network Protocols (ICNP)*, pages 1–6, 2020.

Authorized licensed use limited to: University of Regina IEL. Downloaded on November 20,2025 at 03:38:45 UTC from IEEE Xplore. Restrictions apply.