

Joint Optimization of Handoff and Video Rate in LEO Satellite Networks

Kyoungjun Park*, Zhiyuan He†, Cheng Luo†, Yi Xu‡, Lili Qiu*†, Changhan Ge*, Muhammad Muaz*,
Yuqing Yang†

*UT Austin †Microsoft Research Asia ‡USTC

Abstract

Low Earth Orbit (LEO) satellite communication presents a promising solution for delivering Internet access to users in remote regions. Given that video content is expected to dominate network traffic in LEO satellite systems—as it does across the broader Internet—this study presents a new video-aware mobility management framework specifically designed for such networks. By combining simulation models with real-world datasets, we highlight the critical role of handoff strategies and throughput prediction algorithms in both single-user and multi-user video streaming scenarios. Building on these insights, we introduce a suite of innovative algorithms that jointly determine satellite selection and video bitrate to enhance users' quality of experience (QoE). Initially, we design model predictive control (MPC) and reinforcement learning (RL) based methods for individual users, then extend the approach to manage multiple users sharing a satellite. Notably, we incorporate *centralized training with distributed inference* in our RL design to develop distributed policies informed by a global view. The effectiveness of our approach is validated through trace-driven simulations and testbed experiments.

1 Introduction

Motivation. Recently we have witnessed a rapid increase in Low Earth orbit (LEO) satellite network. Several companies, such as SpaceX, Amazon, and OneWeb, have launched or planned to launch thousands of satellites into space to build their commercial satellite networks. LEO satellite networks are attractive due to their ability to provide global network coverage at an affordable price and their lower latency and higher throughput than geostationary satellites [3, 10]. On the other hand, a LEO satellite moves at a rapid speed—it moves at around 7.5 km/s relative to ground stations, circulating the Earth every 90 to 110 minutes. Due to this rapid movement, user terminals maintain connections with satellites or durations ranging from a few tens of seconds up to 3 minutes, necessitating frequent handoffs. Zhao *et al.* [32] report that Starlink switches the primary link to another satellite approximately once every 15 seconds. Therefore, such frequent handoffs call for an effective handoff strategy to ensure smooth and high performance.

Note that although the handoff frequency in LEO satellite communication is high, the satellites' movement follows a deterministic pattern. This characteristic is very different from traditional Wi-Fi and cellular networks where mobility is less predictable. Such a predictable mobility pattern presents a valuable opportunity to optimize handoff strategies, yielding significant potential benefits.

Meanwhile, we also observe a rapid growth in video streaming traffic on the Internet. Video streaming has become the majority of Internet traffic. LEO satellite providers offer Internet access to remote regions (e.g., Africa, remote villages), where satellite links

are their only way to access the Internet. We expect remote users to have similar traffic patterns as traditional Internet users to enjoy video streaming, which accounts for most Internet traffic.

Studies have shown that video quality is crucial to retaining viewers [5, 22]. Adaptive video streaming addresses fluctuating network throughput, where the sender divides a video into several chunks and encodes each video chunk at several different data rates. Clients dynamically select the appropriate bitrate based on current network conditions. Many interesting algorithms are proposed to select the video bitrate to optimize the video quality of experience (QoE), which is determined by three major factors: video quality, rebuffering time, and smoothness of quality. Some use optimization while others use reinforcement learning (RL) to adapt the video bitrate.

Our approach. While both video streaming and LEO satellite networks are important research topics, there is little work on video streaming in LEO satellite networks. This motivates our exploration of this topic. We first conduct simulations and real-world measurements to identify key challenges and opportunities in the video streaming process in LEO environments. Our results indicate that (i) throughput pattern is important in the video QoE and we should explicitly incorporate the LEO satellites' movement along with the environmental factors for prediction, (ii) widely used handoff algorithms, such as maximizing received signal strength (RSS) or maximizing the satellite serving time, do not yield good video QoE, and (iii) multiple users' sharing the satellite's bandwidth further complicates the design of video rate adaptation.

To address these challenges, we observe a strong interaction between handoff and adaptive bitrate (ABR) algorithms and propose novel methods that jointly optimize satellite selection and video bitrate. We show that by explicitly incorporating the impact of satellite selection into video QoE, we can effectively balance the tradeoff between handoff numbers and RSS. Moreover, to support multiple users competing for limited satellite throughput, we design novel algorithms to jointly select satellites and video bitrates for all users.

Our contributions can be summarized as follows:

- We first use measurements to identify challenges of video streaming in LEO satellite networks. We show that it is important to design a handoff strategy considering video performance. We also show the importance of throughput prediction in both single-user and multi-user scenarios.
- We develop the first algorithms that jointly manage mobility and adapt video bitrate in LEO satellite networks. Our suggested algorithms are based on (i) MPC, and (ii) RL. In particular, we develop a *centralized training and distributed inference-based RL algorithm* to explicitly consider the interaction between multiple satellites and ground stations while supporting practical use.

- We conduct both trace-driven simulation and testbed experiments. Our results show that joint video rate and satellite selection lead to 8.8% to 62.2% QoE improvement across different settings.

We plan to release our dataset and source code to the community.

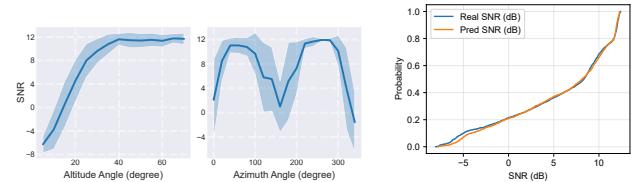
2 Related Works

This section examines recent trends in LEO satellite communication, bitrate adaptation in video streaming, and mobility management.

LEO Satellite Communication. LEO networks utilize satellites to establish communication links across the Earth. Compared to geostationary and middle Earth orbit satellites, LEO satellites, which orbit at 250–1000 km altitude, have lower propagation delay in satellite-to-ground communication due to their proximity to the Earth. Note that electromagnetic waves in inter-satellite communication travel close to the speed of light, while light travels about 31% slower through fiber optic cables [4]. As a result, LEO satellite communication systems offer low end-to-end latency of around 20 ms, making them suitable for latency-sensitive applications like video chat, cloud gaming, and remote control. Furthermore, for distances over 3000 km, LEO satellite communication systems can provide superior performance compared to existing terrestrial fiber optic networks [6]. In addition, their proximity to the Earth also makes it possible to use high-frequency wireless links, e.g., K_u -band, K_a -band, or even THz, which can offer high data rates.

Several studies have improved LEO satellite data access by designing space and ground station networking models. Handley *et al.* [6] study Starlink's dynamic topology and report its end-to-end latency based on SpaceX's public FCC filing. Starfront [13] and Ho *et al.* [8] propose content distribution networks over LEO constellations and minimize their latency and operational costs. Orbitcast [14] proposes a geo-location-driven scheme that uses LEO constellations to efficiently deliver Earth observation data from remote sensing satellites to end users. Vasishth *et al.* [28] develop L2D2, a system that maximizes downlink capacity and minimizes switching penalties by mapping ground stations to satellites using the Hungarian algorithm, predicting channel conditions based on satellite trajectory and weather. SkyTube [16] explores the joint optimization of satellite selection and super-resolution by leveraging each user's local state.

Bitrate Adaptation. To address variations in network performance, there have been works that improve the QoE in video streaming using the ABR approach. The previous studies can be mainly divided into three categories: rate-based, buffer-based, and considering both together. The rate-based approach tries to predict the future network throughput and then choose the highest bitrate possible [11, 27]. FESTIVE [11], which is one of the rate-based studies, predicts the future throughput by applying harmonic mean to the past five video chunks. On the other hand, buffer-based research considers solely the client's playback buffer status when deciding bitrates [9, 26]. BOLA [26], which is one of the recent buffer-based studies, uses Lyapunov optimization to maximize the QoE. BOLA also proposes a heuristic that abandons the previous download decision and recalculates the decision when rebuffering is impending even in the middle of downloading. Besides, some other studies consider both bitrates and buffer size [9, 18, 30]. Yin *et al.* [30] proposes RobustMPC



(a) Relationship between SNR and the altitude (b) Model-based SNR prediction vs. empirical observations

Figure 1: The relationship between the SNR, altitude, and azimuth angles measured from NOAA satellites in (a). (b) depicts a cumulative distribution function (CDF) plot of the SNR prediction based on the ML model.

that utilizes both rate-based throughput estimates and buffer-based occupancy information to maximize the QoE. Pensieve [18] uses a RL based algorithm [20] to decide the bitrates of future video chunks. Pensieve applies a multi-agent tactic to speed up the training time and encourage exploration. It is robust to network variation.

Satellite handoff. LEO satellites operate at high speeds of about 7.5 km/s [17]. As a result, when a satellite moves out of a client's coverage area, a handoff to a new satellite is required. This handoff process, as reported by [29], results in around 150 – 300 ms disconnection at the physical layer. The impact of this disconnection is further amplified at higher layers, as noted in [15]. The handoff strategy determines when to switch and which satellite to switch to. Akyildiz *et al.* [2] provide a nice survey of handoff management in LEO satellite networks with a focus on voice traffic and Nguyen *et al.* [21] present an architecture for handoff in LEO based on IPv6. Juan *et al.* [12] leverage the LEO satellite trajectories to maximize the amount of time of staying connected with a satellite. It further uses simulation to evaluate the impact of user mobility, antenna radiation error, and satellite steering error.

3 Background and Motivation

In this section, we present several unique challenges and opportunities that arise in supporting video streaming in LEO satellite networks, which motivates us to develop novel algorithms tailored to LEO satellite networks.

3.1 Predictability of Satellite Signal

LEO satellites follow fixed and predictable trajectories, enabling the forecasting of RSS from these satellites. Starlink is the largest LEO satellite deployment, but unfortunately, its ground station cannot report RSS. Instead, we measure the signal-to-noise ratio (SNR) from National Oceanic and Atmospheric Administration (NOAA) satellites, which are LEO satellites for weather monitoring. While there are fewer NOAA satellites, they share similar characteristics as Starlink (*e.g.*, predictable trajectories and comparable speeds).

Figure 1a shows a highly predictable relationship between the received SNR and the altitude and azimuth angles. We develop a machine learning (ML)-based model similar to that of [28] to predict the SNR, and the results on the test set show that our model has a mean absolute error (MAE) of only 0.84 dB, as shown in Figure 1b.

ABR algorithms (*e.g.*, [11, 27, 30]) can exploit the predictability of satellite signals for optimization. Specifically, we can map the predicted SNR to the MAC-layer data rate, which can be further

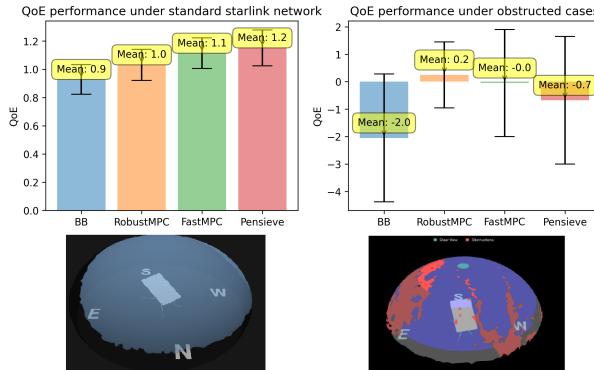


Figure 2: The video QoE in the Starlink network at two locations: one is free from obstructions and the other has several obstructions around it. The figures show the visibility map generated by the Starlink APP. The red portion is the detected obstruction.

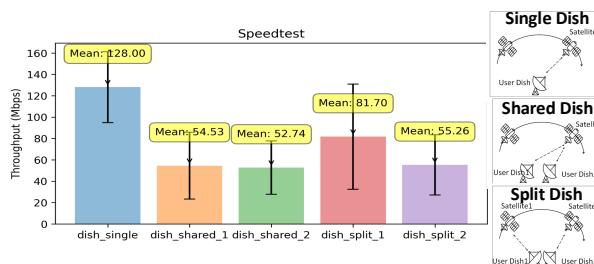


Figure 3: Throughput measurement for Starlink network at different settings: (i) A single dish, (ii) Two dishes facing the same direction, and (iii) Two dishes facing different directions.

used to estimate video bitrates based on the number of users and protocol overhead.

3.2 Video Performance in LEO Network

To study the real video-watching experience of end users in the LEO satellite network, we measure the QoE of video streaming in Starlink and find that obstructions have a big impact on it. Video QoE is usually quantified through the following formula [30]:

$$QoE = \sum_{k=1}^N [\mu_1 Q(R_k) - \mu_2 T(R_k)] - \mu_3 \sum_{k=1}^{N-1} |Q(R_{k+1}) - Q(R_k)| \quad (1)$$

where R_k denotes the bitrate of the video chunk k , $Q()$ represents the video quality calculated from bitrate, $T()$ is the rebuffering time, N is the optimization horizon (i.e., the number of future video chunks considered in optimization), the last term quantifies the smoothness of video quality, and μ_i is a relative weight for each term. To decide the bitrate R_k , we adopt several popular ABR algorithms, including RobustMPC [30] and Pensieve [18].

We evaluate the video QoE at two locations: one free from obstructions, and the other has several obstructions, including buildings and trees mainly in the southeast direction. Figure 2 shows that current ABR algorithms offer stable QoE when there are no obstructions. However, when obstructions occur, the data rate drops to below 0.1 Mbps , which increases the rebuffering time and causes the QoE to degrade significantly.

To mitigate the impact of obstructions, the ABR algorithm can exploit the predictability of satellite signals, as mentioned in Section 3.1. By predicting when the signal is obstructed, the algorithm can select a low bitrate in advance to accumulate buffer time and avoid rebuffering. Furthermore, we can perform *joint selection* by proactively switching to a new satellite to maintain a high bitrate when the current satellite is about to disconnect. In this paper, we demonstrate that joint selection is essential for enhancing video QoE in satellite networks.

3.3 Satellite Handoff Selection

LEO satellites travel at approximately 7.5 km/s [17]. When a satellite moves out of a client's horizon, a handoff to another satellite is necessary. The frequency of handoffs depends on different handoff strategies, occurring every few minutes or seconds. However, these handoffs can impact the quality of video streaming. They may interrupt communication, delay data transfer, and decrease the video's quality and buffer length. According to [29], each handoff costs about $150 - 300 \text{ ms}$ disconnection at the physical layer, which is further amplified at a higher layer [15]. In Starlink, RTT fluctuation due to satellite handoff occasionally caused a 5-second buffer drop, and this is an issue that is not reported in the Starlink mobile application [32]. Such disconnection can cause unexpected rebuffering time and sub-optimal bitrate choices. Therefore, unlike other networks, such as WiFi and LTE, we should consider the unique characteristics of LEO networks in video streaming, e.g., irregular fluctuation of network bandwidth due to handoff, satellite movement, and obstacles. Currently, satellite hardware vendors, like Starlink, do not provide a handoff selection option. We advocate opening this selection, since it can help to significantly enhance the application performance in LEO networks.

In the LEO satellite network, various handoff strategies have been developed to determine appropriate handoff timing and which satellite to switch to. The maximum visible time strategy is to stay connected with a satellite until its signal becomes weak, then switch to a new satellite with the longest visible time. This reduces the number of handoffs but may lead to worse signal strength. On the other hand, selecting a satellite with maximum RSS may result in better signal strength but more handoffs and disconnections, leading to unexpected rebuffering and poor bitrate choices.

Ultimately, which metric to use for satellite selection depends on the application metric for end users. Since video accounts for most traffic in the network, choosing the metric that can optimize video performance is desirable. Video QoE is affected by both the number of handoffs and RSS. How to balance the two highly depends on the video QoE. Therefore, we incorporate various factors involved in handoff into the video QoE model and directly optimize it.

3.4 Throughput Sharing in LEO Network

Throughput is a crucial factor in video streaming, as higher and more stable throughput typically results in better QoE. To evaluate throughput in the LEO network, we use the Starlink APP with speed tests. With a single user terminal, it achieves an average throughput of 120 Mbps from Starlink satellites. However, when two user terminals are placed at the same location, each terminal receives only 50 Mbps due to their competition for the limited link capacity as

shown in Figure 3. These results highlight the significant impact of multi-user competition.

4 Our Approach

We formulate the joint optimization of video bitrate adaptation and satellite selection in Section 4.1. Then, we describe our joint optimization for a single user in Section 4.2 and joint optimization for multiple users in Section 4.3.

4.1 Joint Optimization Problem Formulation

To support video streaming in LEO networks, one option is to adopt the same strategy as on the Internet that decouples the video bitrate selection and satellite selection. While simple, this strategy leads to sub-optimal performance since satellite selection significantly impacts video streaming QoE. Therefore, we propose having clients perform joint satellite and video bitrate selection.

Moving the handoff decision from the satellite to the end host is consistent with the trend in cellular networks, which have also moved from network-based handoff to client-based handoff to not only reduce latency but also offload base stations. A LEO client can determine when to handoff and which satellite to handoff to because the satellites' trajectories are known in advance according to [1]. Moreover, the handoff can be performed efficiently on a per-packet basis by applying the corresponding beamforming weight to the received signals across its antenna array and passing the combined signal to the decoder for further processing. As the client knows its performance objective, having it select the satellite and video rate improves performance.

We define the QoE metric as follows to capture the impact of the satellite selection:

$$QoE_{SAT} = \sum_{k=1}^N [\mu_1 Q(R_k) - \mu_2 T(R_k, Sat_k)] - \mu_3 \sum_{k=1}^{N-1} |Q(R_{k+1}) - Q(R_k)| \quad (2)$$

Our joint ABR algorithm tries to optimize the QoE in Equation 2. The only difference from existing separate ABR algorithms is that Sat_k is considered an optimization variable.

4.2 Single-user QoE Optimization Algorithms

In this section, we present two methods, MPC and RL, in LEO networks and analyze their respective characteristics.

4.2.1 Joint MPC Algorithm. MPC is effective for video bitrate adaptation on the Internet [7, 30]. It uses predicted throughput to optimize the QoE for the next few video chunks. It finds the best QoE by calculating all possible combinations of video quality for future several video chunks. Therefore, its performance highly depends on the quality of throughput prediction. To accommodate throughput prediction errors, there are several variants of MPC. RobustMPC improves the robustness of prediction against error by estimating prediction errors and using $C = \frac{Predict}{1+\max\{error\}}$ as input. We adopt RobustMPC in our problem due to its resilience against errors.

We propose two types of satellite-specific MPC: joint exhaustive MPC and joint pruned MPC. Joint exhaustive MPC only considers all combinations of satellites and bitrates for the future chunks download, which is the same logic as traditional MPC but additionally includes the satellites in the combination. However, it requires significant computational resources and time for calculation. Its time

Algorithm 1 Joint pruned MPC

```

Input:  $Sat$ : satellite,  $t$ : timestamp,  $k$ : video chunk index,  $C, \hat{C}$ : throughput,  $Q$ : QoE reward,  $R$ : bitrate,  $B$ : buffer size,  $Sat_c$ : the current satellite,  $H$ : handoff point
1:  $\hat{C}_{Sat_c, t_k} = ThroughputPred(C_{Sat_c, [t_{k-N}, t_k]})$ 
2:  $Q_{best}, R_{best} = f_{mpc}(R_{k-1}, B_{k-1}, \hat{C}_{Sat_c, t_k})$ 
3: for  $Sat_n$  in  $VisibleSatellites(t_k)$  do
4:    $\hat{C}_{Sat_n, t_k} = ThroughputPred(C_{Sat_n, [t_{k-N}, t_k]})$ 
5:   for  $H = 0$  to  $F - 1$  do            $\triangleright F$ : future prediction length
6:      $Q, R = f_{mpc}^{sat}(R_{k-1}, B_{k-1}, H, \hat{C}_{Sat_c, t_k}, \hat{C}_{Sat_n, t_k})$ 
7:     if  $Q_{best} < Q$  then
8:        $Q_{best} = Q$ 
9:        $R_{best} = R$ 
10:       $Sat_{best} = Sat_n$ 
11:       $H_{best} = H$ 
12: return  $Q_{best}, R_{best}, Sat_{best}, H_{best}$ 

```

complexity for a single calculation would be $O(|R|^F \times |Sat|^F)$ where $|R|$ is the number of data rates, $|Sat|$ is the number of satellites, and F is the number of future chunks considered. This cost is too high to run in real time. Therefore, we propose a joint pruned MPC that assumes only one handoff during the next F video chunks in the optimization horizon. This is a reasonable assumption since F is generally small (*e.g.*, $F = 5$ in existing work) and multiple handoffs in such a short time window are expensive. Joint pruned MPC can significantly reduce computational overhead over joint exhaustive MPC. Its time complexity is $O(|R|^F \times |Sat| \times F)$ where F represents the possible handoff point. We use joint pruned MPC in evaluation.

Algorithm 1 shows joint pruned MPC that considers both bitrates and satellites. The video player starts downloading a video chunk k at t_k . We then predict the future throughput of the connected satellite S_c using $ThroughputPred()$. We employ two prediction methods and compare their performances: (i) Harmonic mean over historical throughput; (ii) A model similar to [28] which takes elevation angle, azimuth angle, and weather information as input, and leverages machine learning models to predict future throughput. With the predicted throughput, $f_{mpc}()$ applies the MPC algorithm. This algorithm tries to find the best QoE by iterating over all satellite candidates $VisibleSatellites()$ and handoff points H when downloading future chunks. At each handoff point H , we use $f_{mpc}^{sat}()$ to compute the utility by taking into account the handoff delay and throughput change when switching from Sat_c to Sat_n at t_{k+H} .

4.2.2 RL-based Algorithm. Our optimization problem naturally fits in the RL framework, where the user aims to select the satellite and video bitrate to optimize QoE. We propose RL algorithms based on proximal policy optimization (PPO) [25], a state-of-the-art actor/critic method. It trains a critic network to estimate the value function and an actor network to optimize the policy based on the value function. To prevent the policy from changing too much in each step and improve training stability, it enforces the objective to be within a small range. Figure 4 shows our networks, where S denotes the state space after downloading a chunk t , v is the value function, a is the action space, and π represents the state transition probability.

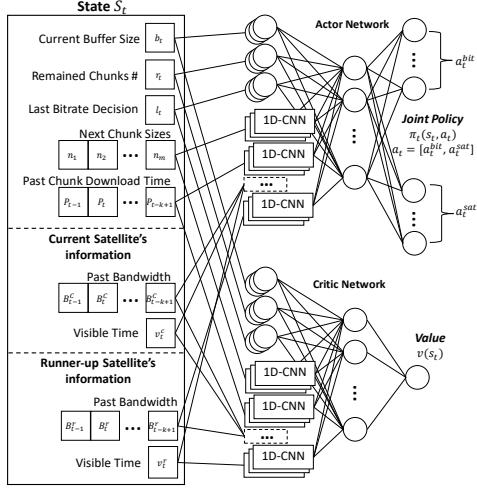


Figure 4: The PPO-based RL algorithm generates joint optimization policies.

State: For each video chunk, the state inputs to the actor network and the critic network can be represented as $S_t = (b_t, \gamma_t, l_t, n_m, P_t, B_t, v_t)$. b_t is the current buffer level; γ_t is the number of chunks remaining in the video; l_t is the bitrate at which the last chunk was downloaded; n_m is a vector of m available sizes for the next video chunk; P_t is the download time of the past k video chunks, which represents the time interval of the throughput measurements; B_t is the network throughput measurements of two satellites for the past k video chunks; and v_t indicates the visible time elapsed.

Most state inputs are similar to [18] but we add the following new states: B_t^r and v_t for both the current satellite and runner-up satellite to support the LEO network. These additional inputs are crucial to our problem since we jointly decide on bitrate and handoff together. Note that we only pass the previous states for RL to implicitly predict based on the historical data and determine the appropriate action. We no longer need to design a separate predictor.

Action: Our action selects the satellite a_t^{sat} and video bitrate a_t^{bit} for the next video chunk at time t , denoted as $a_t = [a_t^{bit}, a_t^{sat}]$.

Reward: We use video QoE as shown in Equation 2 as the reward since it is the ultimate metric we want to optimize.

Policy: The policy specifies the probabilities of taking different actions at each state, namely $\pi_t(s_t, a_t)$, which is the probability of selecting the satellite a_t^{sat} and video rate a_t^{bit} at state s_t . a_t^{sat} is selected from a set of candidate satellites visible at the time t .

To enhance efficiency, in our evaluation, we reduce the joint RL algorithm complexity by setting the number of candidate satellites to 2 (i.e., $N_{sat} = 2$), which considers only one runner-up satellite with the best-predicted throughput.

Policy Update: As pointed out in Equation 2, the joint optimization algorithm aims to maximize the QoE. For our RL algorithm to learn an optimal policy that maximizes the objective, the environment provides a reward r for every chunk download. The critic network guides the update of the actor network by estimating the gradient, which can be computed as the policy gradient training of [25].

The training process of RL-based algorithms can be described as follows: 1) Collect user data: The algorithm needs to gather data

about each user's state and satellite throughput, 2) Calculate individual QoE: Based on the collected data, the algorithm calculates each user's individual QoE. This can be done using different micro-benchmarks, 3) Adjust the hyperparameters: The hyperparameters of the system, such as learning rates and model weights, are adjusted to improve the performance of the system, 4) Re-calculate QoE: After the hyperparameters are optimized, the algorithm re-calculates the QoE for all users. This step is important to ensure that the optimization does not degrade the other users' QoE, 5) Make a decision: Based on the updated QoE scores, the algorithm makes a decision on the overall QoE, and 6) Repeat the process: The algorithm should continuously monitor the users' states and re-optimize the communication if necessary. This process is repeated until the communication session ends or the user's status changes significantly.

4.3 Multi-user QoE Optimization Algorithms

In Section 4.2, we finally introduce several optimization algorithms for a single-user scenario. In practice, multiple user terminals can share the same satellite and compete for limited network resources. As shown in Section 3.4, the throughput of each user significantly decreases when sharing with others.

There are several ways to address this issue: (i) Let users independently make their own decisions using the single-user algorithm described in Section 4.2. A user's throughput is affected by the cross traffic, which in turn influences the rate and satellite selection; however, this approach is sub-optimal and can lead to fluctuation as it optimizes for each user's own utility rather than the overall utility. (ii) Use a centralized algorithm, which considers information from all users and makes decisions that can achieve high utility for all users; however, it faces deployment issues in practice as it is challenging to control all users' selections of satellites and video rates. Motivated by the limitation of the above options, (iii) we design the RL that uses other users' states as input to optimize the global objective but only takes an action for a specific user at a time. This design is easier to implement and deploy because it does not have to control the actions of all users. Last but not least, this RL design involves collecting other user information in real time, which may still be difficult in some deployments. Therefore, (iv) we introduce *centralized training and distributed inference* RL design. This approach uses all users' states as the input during training but only uses the current user's state during the inference, which not only simplifies the deployment but allows us to explicitly consider the interaction between multiple satellites and ground stations.

Our multi-user algorithms primarily aim to optimize the QoE for users engaged in video streaming. However, in real-world scenarios, LEO networks support a diverse range of applications beyond video streaming, including online gaming and web browsing. To better reflect these practical environments, we incorporate multiple users with dynamic resource allocation mechanisms, simulating the diverse network demands of different applications. This approach ensures a more realistic and adaptable optimization framework.

4.3.1 Multi-user RL (Joint RL (L)). We apply the single-user PPO-based RL to the multi-user scenario. Our multi-user process can be represented by the following four tuples: (S_t^j, A^j, π^j, R^j) , where j is the user id, S_t^j represents the user j 's state, A is the distributed action space, π is the state transition probability, and R is the local

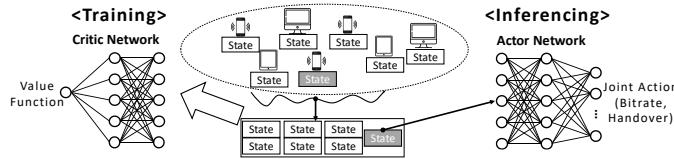


Figure 5: The PPO-based RL algorithm generates joint optimization policies with centralized training and distributed inference.

reward value. The decision-making process of a single-user RL can be directly applied to multi-user optimization. This RL uses only the current user's state, so we call it the local state. It uses the throughput history of the current user's satellite to implicitly learn the network condition when sharing its satellite with other users.

For multi-user training and testing, we use a critic network for distributed training. The input state of the critic network is a distributed environment (i.e., S_t^j for each user). The critic network estimates the value of action decisions generated by the actor network. The state transition probability π is then used to update the subsequent actor steps, evaluate each actor, and guide their parameter updates. The parameters and gradients of the critic network and actor network are shared by all agents.

4.3.2 Multi-user RL with Global Status (Joint RL (G)). This approach leverages all users' states to maximize all users' overall QoE, hence called Joint RL with global information (Joint RL (G)). This RL algorithm works by substituting the input states in the multi-user RL design to the following: (S_t, A^j, π^j, R^j) for each user j . The only difference is that S_t represents all users' states, not just one user. We feed all users' information to the critic network to train a single global policy. During training, the centralized critic network is used to update the actor networks of all users. The gradient of the critic network is computed based on the actions taken by all users, and updates in the actor network are based on gradients in the critic network. Since it requires global information not only for training but also for the inference phase, it requires users to frequently exchange their states with other users that may share the same satellites.

4.3.3 Multi-user RL with Centralized Training and Distributed Inference (Joint RL (L+)). Inspired by the asymmetric actor-critic structure from the RL community [23, 24, 31], we use a centralized critic architecture and a decentralized actor architecture, as shown in Figure 5. During training, the centralized critic network takes the global state as input and outputs a single value for each state, which is used to update the actor networks of all users. This speeds up convergence and improves performance. During inference, the decentralized actor network takes the local state of each user as input and outputs a local action for that user. In other words, *centralized training and distributed inference* have a hybrid concept of considering the global objective during training and making decisions at the user-level state during inference.

4.3.4 Centralized MPC. So far, we describe several RL algorithms that can run in practice. We also consider the centralized MPC algorithm as another baseline. This algorithm assumes we have full control of all users' actions. While this assumption is strong and may not hold in practice, this is an interesting baseline. Without prediction error and with a large enough optimization horizon, the

oracle centralized MPC establishes the upper bound for multi-user video QoE. We extend the single-user MPC to a centralized MPC by (i) treating all users' selected satellites and video bitrates as the optimization variables and (ii) taking the sum of all users' utilities as the objective. Since MPC calculates all the possible combinations, the search space increases exponentially with the number of users. To reduce running time, we prune the satellites with low throughput.

5 System Implementation

When applying our methods to real-world scenarios, we encounter a challenge: the lack of an interface in Starlink for clients to actively change satellites. To overcome this issue, we develop our own clients and servers. In our system, the client not only can select the bitrate of video chunks but can also choose between different satellites.

We use TCP sockets to emulate different network links between the users and the satellites. Upon connecting to a satellite, we establish a dedicated TCP socket on the server side, which is managed by a satellite controller. The satellite controller regulates throughput by limiting data rates in the socket, based on throughput traces collected from the actual Starlink network or other simulated traces for experimental purposes. Note that the Starlink throughput traces are acquired from the connection between the video server and end users, which includes both uplink and downlink; therefore, it represents end-to-end throughput in a real-world scenario when they are replayed in our system. When a client decides to change the satellite to associate with, we terminate the current socket and establish a new one, controlled by a new satellite controller. Multiple clients can connect to the same satellite, in which case throughput is shared, and it is also managed by the satellite controller. In addition, the states of other users, such as the buffer size and download speed, are required in Joint RL (G). To implement it, every client reports its states to the server and gets back other users' states in a separate API call, which happens immediately after downloading a video chunk.

RL Implementation. We feed the input states using densely connected neural network (NN) layers with 128 neurons and 1D convolutional NN layers with 128 filters of size 1 with stride 1. We set the length of the past chunk download time and bandwidth as 8. These settings of neurons and filters and the past measurement length are set based on the microbenchmark evaluation. The actor network uses the softmax function at the output layer to generate the policy, and the critic network uses a linear function to update the policy parameters. The discount factor, which determines how much current choices affect future choices, is set to 0.99. The learning rates for both networks are set to 10^{-4} . Finally, we apply the adaptive entropy weights to reduce the intensity of exploration according to the learning rate.

Multi-user Scenarios. We implement a simulator with 20 users engaged in either video streaming or other applications. To better reflect real-world environments, we apply dynamic resource allocation to users not involved in video streaming, simulating diverse network conditions. Furthermore, to facilitate multi-user decision-making within the RL framework, we apply an encoder-decoder architecture in our neural network layers, enabling efficient representation learning and decision output generation. While our current RL model is designed for 20 users, it can be scaled to accommodate a larger number of users by increasing the model size.

6 Evaluation

We conduct extensive experiments to evaluate the proposed MPC and RL algorithms in both single-user and multi-user scenarios. The algorithms are tested in simulation and testbed as described in Section 5. We use a video with 1080p, and it is split into 49 chunks. We assume the video chunks are stored in the origin video server and are transmitted through satellite networks. Every chunk lasts for 2 seconds and is encoded into three different bitrates: 300, 1200, and 2850 $kbps$. Video servers typically support six or more bitrates. For a fair comparison, we use three bitrate levels because the complexity of centralized MPC is too high to operate with more than three levels. We further experiment with **Joint RL (L+)** with six bitrate levels, i.e., 300, 750, 1200, 1850, 2850, and 4300 $kbps$, to show that our proposed RL models are scalable to support all bitrates. In the QoE equation, we set 1, 4.3, and 1 for μ_1 , μ_2 , and μ_3 , respectively. Link RTT is set to 80 ms and handoff delay is set to 200 ms.

6.1 Satellite Network Dataset

We create three datasets: simulated, NOAA, and Starlink. Each dataset is divided into one training set and one test set. The simulated dataset is constructed based on the orbital trajectories of Starlink satellites, utilizing a free space model to simulate the throughput dynamics. The NOAA dataset is acquired through measured SNR data of NOAA satellites, whereas the Starlink dataset consists of end-to-end measured throughput data obtained from a real Starlink link between a client and a video server.

To evaluate the MPC-style algorithms, we directly apply them to the test set. To evaluate the RL methods, we first train it on the training set of the simulated dataset. We then fine-tune the trained model on the specific training set (such as NOAA or Starlink) when we test the model.

To generate the simulated dataset, we compute the trajectories of Starlink phase I satellites for the following cities: London, Boston, Shanghai, Hong Kong, Los Angeles, Paris, New York, Tokyo, Chicago, Singapore, San Francisco, Sydney, Toronto, Mexico City, Taipei, Washington, Beijing, Rome, Berlin, Dublin, Sao Paulo, Moscow, Osaka, and Seoul. The throughput trace is generated by: $b_t = \alpha \times b_{max} \times \frac{d_{min}^2}{d_t^2} + \epsilon$, where b_{max} is the maximum throughput of a single Starlink satellite in $Mbps$, d_{min} is the upright height of the satellite to Earth, d_t is the distance between the satellite and the client, α is a scale factor, and ϵ is a noise following Normal distribution $N(-2, 1)$. The formula employs the free space path loss model, commonly used in satellite communication. This model indicates that as the distance from a transmitter grows, the signal's strength reduces at a rate of the inverse of the distance square. In our throughput equation, this is seen as throughput dropping when the distance d_t between the satellite and the client increases. To test the algorithm under limited resources, we use a scaling factor α to scale the throughput to a suitable range. We generate 24 traces for each city, and every trace has throughput traces lasting for 10 minutes. We set 7 cities as the test set and the remaining as the training set.

We collect SNR and throughput from clients to satellites to obtain more realistic network traces. However, we find it difficult to fetch SNR in Starlink devices, so we collect SNR from 3 NOAA satellites (NOAA-15, NOAA-18, and NOAA-19) instead. NOAA satellites are also classified as LEO satellites with orbit heights of 808 km, 854 km,

and 850 km respectively. These heights are relatively close to the 550 km of the Starlink phase I satellites. Considering their similar orbital heights, the velocity of Starlink phase I satellites is 7.6 km/s, while the velocity of NOAA satellites is approximately 7.4 km/s. Such similarity in movement suggests NOAA satellite traces are beneficial for our evaluation of handoff algorithms. We collect traces from major cities in the US and China and create a mapping from the elevation angle to the SNR. We randomly choose one NOAA trace for each Starlink satellite pass, and assign the SNR value to the pass by applying the elevation-angle-to-SNR mapping. Throughput is then generated from the SNR value. We create one training set with 24 traces and one testing set with 12 traces, confirming that the NOAA traces in these 2 sets do not overlap.

We also measure end-to-end throughput in the Starlink network to obtain the dataset. An iperf download is conducted from a Starlink RV terminal to a video cloud server node in the US. Note that the throughput traces include uplink, downlink, and terrestrial links between the Starlink dish and the Internet video server. Thus, they provide an accurate representation of the actual video streaming conditions within the Starlink network. The iperf logs with timestamps are collected at a granularity of 1 second and a length of 13 hours. To simulate the multiple satellites, time division multiplexing is utilized. Specifically, we transform a 16-minute iperf log into a 4-minute log with four satellites. Therefore, 48 multiplexed logs are created from a 13-hour log. We use 40 logs as the training set and 8 as the test set.

6.2 Baseline Methods

Our methods can select satellites and bitrates simultaneously. We compare them with popular satellite selection strategies and ABR algorithms. The following satellite handoff strategies are tested:

- Maximum visible time (MVT): Switch to the satellite with the longest visible time when the currently connected satellite is about to leave the client's horizon.
- Maximum received signal strength (MRSS): Switch to the satellite with the highest signal strength.
- Maximum bandwidth (MB): Switch to the satellite with the maximum available bandwidth when the currently connected satellite is about to leave the client's horizon.

6.3 Simulation Results

RobustMPC [30] and Pensieve [18] are used in ABR algorithms. For simplicity, we refer to RobustMPC as "MPC". The satellite handoff strategy and the ABR algorithm are combined and tested. For example, MPC (MRSS) uses MRSS to select satellites and RobustMPC to decide bitrates. We pair Pensieve with the MB due to its superior performance. We highlight **Joint RL (L+)** in bold to indicate that it is our final proposed model. We experiment with three scenarios involving 20 users: (i) one user streaming video while 19 users engage in other applications, (ii) five users streaming video while 15 users engage in other applications, and (iii) all 20 users streaming video.

6.3.1 Overall Comparison. The simulation results on three different datasets with different user numbers are presented in Figure 6. The proposed joint methods consistently outperform separate methods, particularly when multiple users are present. Compared to the best result achieved by the separate selection, joint selection methods

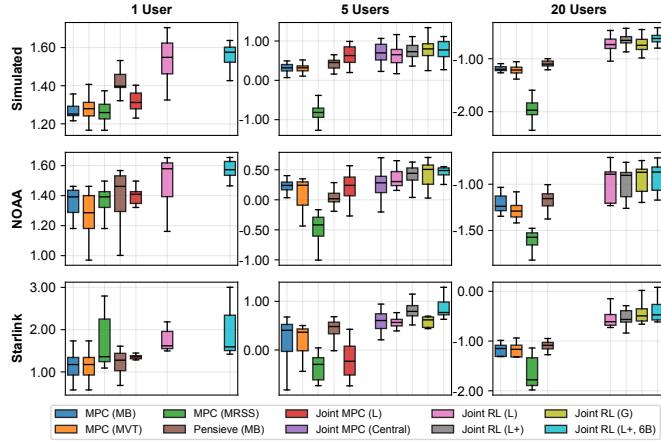


Figure 6: QoE results of the nine models (i.e., MPC, RL). The results are from simulated, NOAA, and Starlink datasets. All models use three bitrate levels for fair comparison except Joint RL (L+, 6B). The number of users refers to how many users are streaming video out of 20 users.

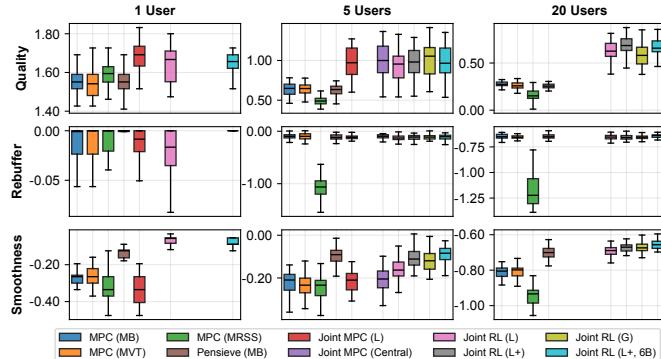


Figure 7: QoE breakdown with three terms: 1) quality reward, 2) rebuffer penalty, and 3) smoothness penalty. The results are from the simulated dataset. All models use 3 bitrate levels for fair comparison except Joint RL (L+, 6B). The number of users refers to how many users are streaming video out of 20 users.

can improve QoE by 18%, 68%, and 57% for 1 user, 5 users, and 20 users, respectively, in the Starlink dataset. The main reason for the rapid change in QoE difference depending on the number of users is that, in our network resource settings, one user is provided with sufficient resources, but when there are more than three users, a lot of competition occurs. These results indicate that our joint optimization performs better than separate models, especially in resource-limited environments. In the simulated dataset, we train in 17 cities and test in 7 other cities. Our RL methods generalize well to unseen cities. Overall, the joint RL (G) has the best performance. Joint RL (L) and **Joint RL (L+)** are also competitive but slightly worse than Joint RL (G). Note that **Joint RL (L+)** always yields a better QoE than Joint RL (L), ranging from 0.8% to 25%. It shows that our *centralized training and distributed inference approach* effectively considers both sides of the satellite network: periodicity of satellite movement and irregular network throughput.

We present the breakdown of QoE into three terms (quality, rebuffer, and smoothness) in Figure 7. Based on Figure 7, separate selection methods generally produce lower-quality rewards when compared to joint selection methods. This can be attributed to a

mismatch between the satellite selection strategy and the video QoE. Joint selection methods can actively change to a new satellite, which potentially brings QoE improvement. MB is the best handoff strategy to maximize the throughput among separate selection schemes, but it is still worse than the joint selection in quality score.

It is observed that MPC-based methods are often less smooth than RL-based ones, possibly due to the limited optimization window of MPC. This limitation also exists in our proposed Joint MPC method. Thus, in the 1-user scenario, although Joint MPC (L) produces higher quality rewards than Pensieve (MB), its final QoE score falls behind that of Pensieve (MB) due to worse smoothness. Due to the limited calculation window in MPC, we can see that the performance of Joint MPC (Central), which knows the network status within the window, is similar to or even underperforms our Joint RL.

Pensieve (MB) rarely picks the highest bitrate, even though it has a sufficient buffer. Hence it has lower quality scores than our proposed Joint RL, as shown in Figure 7. We interpret Pensieve (MB) as making a conservative decision, as it does not jointly consider satellites. When Pensieve (MB) increases the video quality due to enough buffers, it encounters many cases where a rebuffing penalty occurs due to sudden drops in satellite throughput.

We find that Joint RL actively decides the handoff to achieve a high QoE, i.e., 2 to 3 times more handoffs than the MB strategy. Joint RL does not hesitate to choose the highest quality when it has enough buffer compared to Pensieve (MB). Interestingly, Joint RL does the handoff only when the buffer is enough, which indicates a low possibility of rebuffing. Conversely, when the buffer is sufficient, Joint RL actively performs actions to achieve high throughput, such as performing multiple handoffs within a few seconds.

By incorporating global status, Joint RL (G) achieves better results than Joint RL (L). Joint RL (G) can learn interesting policies maximizing the overall QoE of all users, even if it slightly degrades one user. Joint RL (G) learns to tactically establish each user's role, such as determining handoff frequency or video quality. For example, one user does a lot of handoffs when encountering resource sharing and only maintains modest quality downloads. This allows other users to download the highest quality without resource contention. It is quite attractive that our RL models learn these policies to optimize the overall QoE without any special a priori knowledge.

Compared to the simulated trace and Starlink trace for 5 users, the MPC models have long tails in the boxplot, indicating huge variances in QoE. This is because the real trace contains unstable latencies at specific timestamps, i.e., handoff. To cope with this unexpected case, users changed the quality abruptly as shown in the smoothness in Figure 7. On the other hand, RL variants, including our proposal **Joint RL (L+)** have quite stable QoE results, indicating that RL models handle harsh network conditions more wisely. Note that a sudden one-time quality change incurs a severe degradation of QoE.

6.3.2 Obstructions. In Section 3.2, we highlight that obstructions decrease video QoE significantly in a satellite network. We find our proposed methods can effectively handle the obstructed scenario. In the Starlink dataset, sometimes bandwidth drops abruptly, degrading QoE as shown in Figure 2. If the bandwidth falls below the 25th percentile for over 10 seconds, it is considered an "obstruction period". In all, 158 obstruction periods were identified in the dataset, accounting for 5.8% of the total duration. Our Joint RL effectively

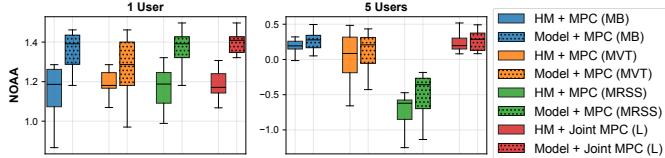


Figure 8: QoE results of MPC methods on the NOAA dataset with and without the model prediction. The Y-axis represents QoE. "HM" indicates the harmonic mean, while "Model" uses ML for prediction.

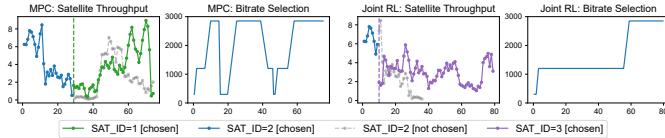


Figure 9: An obstruction scenario, with the X-axis denoting time (sec). Initially, both Joint MPC (L) and **Joint RL (L+)** connect with satellite 2. However, satellite 2 experiences an obstruction period from 15th to 40th. Joint MPC (L) switches to a different satellite around 30th with significant bitrate fluctuation. Conversely, **Joint RL (L+)** transits to an alternative satellite when disturbed, effectively maintaining stable bitrates.

handles these scenarios as shown in Figure 6, and we provide a case study in Figure 9. Unlike other methods that have bitrate fluctuations when faced with obstructions, RL methods smoothly switch to a different satellite and maintain consistently high bitrates.

6.3.3 Effects of throughput prediction. We employ an ML-based model similar to [28] to predict the throughput of the satellite using the NOAA dataset. The input features for the model are the azimuth angle, elevation angle, weather status, and the satellite's movement. The model uses a stacked ensemble of gradient-boosting decision trees and NN models to predict the future throughput, achieving a mean absolute error of 0.1304 Mbps on the test set.

Our prediction model is useful for throughput-based ABR algorithms. We compare the QoE of MPC-style methods with and without our model's predictions to evaluate its efficiency. Specifically, we compare the performance of MPC methods that use our model's predictions with those that use the harmonic mean of historical throughput to select bitrates, considering several handoff strategies.

As shown in Figure 8, model prediction leads to improved QoE for both traditional MPC methods and our proposed joint MPC. The throughput prediction enables better bitrate selection for traditional MPC methods, while for our proposed methods, it enables not only better bitrate selection but also a better satellite selection.

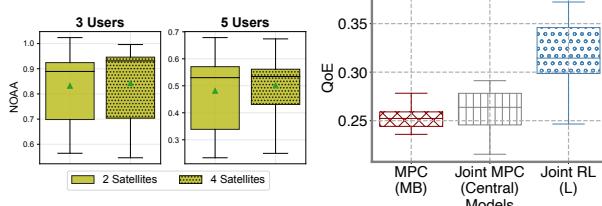


Figure 10: QoE results of **Joint RL (L+)** on the NOAA dataset varying satellite selection coverage.

Figure 11: QoE results of three models for the multi-session problem. Three sessions are used.

6.3.4 Consideration of multiple satellites. Our models consider 2 satellites in decisions. Figure 10 shows how much the QoE can be improved by considering more than 2 satellites. Since there are up to 4 visible satellites in NOAA, we evaluate the **Joint RL (L+)** by extending the input/output states to account for 4 satellites. The QoE of the model considering all satellites improved by 1.3% and 4.1% on average for 3 users and 5 users, respectively. Our RL design is scalable to support multiple satellites without major changes.

Table 1: QoE results for Joint MPC (Central) with two resource sharing strategies.

Sharing Strategy	3 Users	5 Users
Fair Resource	0.999	0.641
Prioritized Ratio	1.032	0.761

6.3.5 Prioritized ratio strategy. When total bandwidth is limited, the different resource-sharing strategies may matter. We experiment with two strategies (i) fair resource sharing and (ii) prioritized ratio sharing on the simulated dataset using Joint MPC (Central). We use sequential least squares programming every second to find the optimized resource ratio of users. We filter only those tracks where more than two users are connected to the same satellite. We also apply a heuristic to calculate the rebuffering time by reducing the user's current buffer size by 70%. This is to accommodate the unexpected events because even a small rebuffering time highly degrades QoE. When we tested using the original buffer sizes in the objective function, the algorithm fully utilized the buffered data; however, this is risky under prediction errors, which can generate rebuffering.

Table 1 shows the prioritized ratio strategy outperforms the fair strategy by about 3.3% for 3 users and up to 18.7% for 5 users. It indicates that a prioritized ratio strategy is beneficial when there is a lot of resource competition. Note that this prioritized strategy can only be applied in a centralized model because it should gather the states of all users, such as buffer sizes and previous bitrates, and make decisions for all users.

6.3.6 Multi-session. We evaluate the three models in multiple sessions, sharing network resources at a single user terminal. We assume that only one session can make a handoff decision to avoid unexpected issues (e.g., each session performing the handoff at different times). Since all sessions share the same user terminal, handoffs must be performed simultaneously. Figure 11 shows that **Joint RL (L)** outperforms MPC (MB) and Joint MPC (Central), which is in line with the other evaluations. This result indicates that our proposed models can work well in multi-session problems.

Table 2: QoE comparison among cities generated by Joint RL (G).

City	Hong Kong	Seoul	Beijing	Chicago	Toronto	Paris	London
Latitude	22.3	37.5	39.9	41.9	43.7	48.9	51.5
QoE - 3 Users	0.87	1.04	0.96	1.10	1.08	1.16	1.31
QoE - 5 Users	0.50	0.71	0.75	0.90	0.94	1.08	1.16

6.3.7 Effects of geographical locations. According to the data presented in Table 2, cities at higher latitudes tend to have higher video QoEs. This is likely because LEO satellites are more concentrated in high latitudes, whereas there are fewer satellites in low latitudes [19]. These results come from a simulation. In the real world, other factors such as ground station distribution, obstruction, and user density can also affect the QoE.

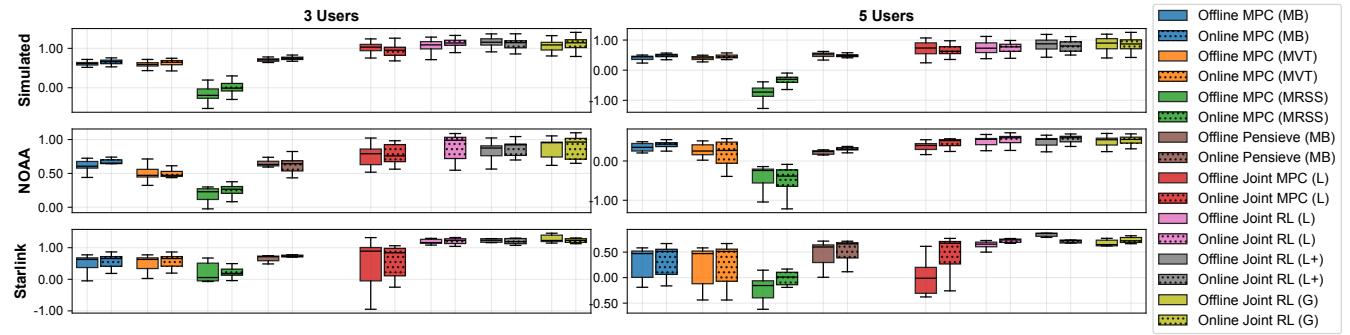


Figure 12: Comparison between testbed results and simulation results. Testbed results are marked with dots.

6.4 Testbed Results

To validate the effectiveness of our algorithms in practical scenarios, we implement them in a testbed system as described in Section 5. As shown in Figure 12, the results of the online (from the testbed) and offline (from the simulation) are consistent in most cases. It indicates that joint selection techniques can enhance the video QoE, and Joint RL can outperform other methods.

The major difference we observed in our testbed system is that HTTP requests may get stuck due to network fluctuation and packet loss in an online environment. To address this issue, we have set a timeout threshold of 10 seconds to download each chunk and immediately stop it on the client side if a request exceeds this limit, switch to another satellite, and attempt downloading again. This setting can improve performance, especially on Starlink datasets where obstacles can cause connection timeouts, as shown in Figure 12. Besides, in the testbed, the latency becomes more unstable than in simulation, and the runtime of each algorithm can also affect performance. We found that our methods are efficient as the bitrate decision of each chunk takes less than 10 ms. Thus the runtime does not adversely impact system performance.

6.5 Summary of Evaluations

We have conducted extensive experiments using a simulation and testbed. The results suggest jointly selecting bitrates and satellites is crucial for optimal video streaming in LEO satellite networks. We improve QoE up to 68% compared to separate selection strategies. We investigate how resource-sharing strategies, geographical locations, different RL designs, and multi-session can affect video QoE. Last but not least, our *centralized training and distributed inference* in RL helps the model understand the global perspective. Overall, our findings contribute to a deeper understanding of the factors that impact the video streaming of LEO satellite networks.

7 Conclusion

We propose a joint satellite selection and adaptive bitrate algorithm based on RL and MPC for LEO satellite networks. According to the experiment results, joint optimization can optimize satellite selection and bitrate decisions dynamically according to the network conditions. Our results show that joint selection yields significant benefits in a single-user scenario and its benefit further increases

over a multi-user scenario. Future directions involve leveraging neural network-based video enhancement techniques at receivers to further optimize video streaming in LEO satellite networks (*e.g.*, video super-resolution and interpolation upon packet losses).

References

- [1] 2023. Starlink Simulator. <https://starlink.sx>.
- [2] Ian F. Akyildiz, Huseyin Uzunalioglu, and Michael D. Bender. 1999. Handover Management in Low Earth Orbit (LEO) Satellite Networks. In *Mobile Networks and Applications*.
- [3] Prakash Chitre and Ferit Yegenoglu. 1999. Next-generation satellite networks: architectures and implementations. In *IEEE Communications Magazine*.
- [4] Joseph Coffey. 2023. Latency in Optical Fiber Systems. <https://www.commscope.com/globalassets/digizuite/2799-latency-in-optical-fiber-systems-wp-111432-en.pdf>.
- [5] Florin Dobrian, Vyas Sekar, Asad Awan, Ion Stoica, Dilip Joseph, Aditya Ganjam, Jibin Zhan, and Hui Zhang. 2011. Understanding the impact of video quality on user engagement. In *ACM SIGCOMM*.
- [6] Mark Handley. 2018. Delay is not an option: Low latency routing in space. In *ACM HotNets*.
- [7] Jian He, Mubashir Adnan Qureshi, Lili Qiu, Jin Li, Feng Li, and Lei Han. 2018. Favor: Fine-grained video rate adaptation. In *ACM MMSys*.
- [8] Cuong Manh Ho, Anh Tien Tran, Chunghyun Lee, Duc Thien Hua, and Sungrae Cho. 2022. Handover in mobility-aware caching strategy for LEO satellite-based overlay system with content delivery network. In *ACM MobiHoc*.
- [9] Te-Yuan Huang, Ramesh Johari, Nick McKeown, Matthew Trunnell, and Mark Watson. 2014. A buffer-based approach to rate adaptation: Evidence from a large video streaming service. In *ACM SIGCOMM*.
- [10] Abbas Jamalipour and Tracy Tung. 2001. The role of satellites in global IT: trends and implications. In *IEEE Personal Communications*.
- [11] Junchen Jiang, Vyas Sekar, and Hui Zhang. 2012. Improving fairness, efficiency, and stability in http-based adaptive video streaming with festive. In *ACM CoNEXT*.
- [12] Enric Juan, Mads Lauridsen, Jeroen Wigard, and Preben Mogensen. 2022. Handover solutions for 5G low-earth orbit satellite networks. In *IEEE Access*.
- [13] Zeqi Lai, Hewu Li, Qi Zhang, Qian Wu, and Jianping Wu. 2021. Cooperatively constructing cost-effective content distribution networks upon emerging low earth orbit satellites and clouds. In *IEEE ICNP*.
- [14] Zeqi Lai, Qian Wu, Hewu Li, Mingyang Lv, and Jianping Wu. 2021. Orbitcast: Exploiting mega-constellations for low-latency earth observation. In *IEEE ICNP*.
- [15] Xu Li, Feilong Tang, Long Chen, and Jie Li. 2017. A state-aware and load-balanced routing model for LEO satellite networks. In *IEEE GLOBECOM*.
- [16] Po-Hsun Lin and Wanjiun Liao. 2023. Space-Centric Adaptive Video Streaming with Quality of Experience Optimization in Low Earth Orbit Satellite Networks. In *IEEE ICC*.
- [17] Vikalp Mandawaria, Neha Sharma, Diwakar Sharma, Chitradeep Majumdar, Anshuman Nigam, Seungil Park, and Jungsoo Jung. 2022. Uplink zone-based scheduling for LEO satellite based Non-Terrestrial Networks. In *IEEE WCNC*.
- [18] Hongzi Mao, Ravi Netravali, and Mohammad Alizadeh. 2017. Neural adaptive video streaming with pensieve. In *ACM SIGCOMM*.
- [19] Jonathan C McDowell. 2020. The low earth orbit satellite population and impacts of the SpaceX Starlink constellation. In *The Astrophysical Journal Letters*.
- [20] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. 2016. Asynchronous methods for deep reinforcement learning. In *PMLR ICML*.

- [21] Hoang Nam Nguyen, Salem Lepaja, Jon Schuringa, and Harmen R van As. 2001. Handover management in low earth orbit satellite IP networks. In *IEEE GLOBECOM*.
- [22] Kyoungjun Park, Myungchul Kim, and Laihyuk Park. 2022. NeuSaver: Neural Adaptive Power Consumption Optimization for Mobile Video Streaming. In *IEEE Transactions on Mobile Computing*.
- [23] Lerrel Pinto, Marcin Andrychowicz, Peter Welinder, Wojciech Zaremba, and Pieter Abbeel. 2017. Asymmetric actor critic for image-based robot learning. *arXiv:1710.06542* (2017).
- [24] Stefan Schneider, Holger Karl, Ramin Khalili, and Artur Hecker. 2022. Deep-CoMP: Coordinated Multipoint Using Multi-Agent Deep Reinforcement Learning.
- [25] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv:1707.06347* (2017).
- [26] Kevin Spiteri, Rahul Urgaonkar, and Ramesh K Sitaraman. 2020. BOLA: Near-optimal bitrate adaptation for online videos. In *IEEE/ACM Transactions on Networking*.
- [27] Yi Sun, Xiaoqi Yin, Junchen Jiang, Vyas Sekar, Fuyuan Lin, Nanshu Wang, Tao Liu, and Bruno Sinopoli. 2016. CS2P: Improving video bitrate selection and adaptation with data-driven throughput prediction. In *ACM SIGCOMM*.
- [28] Deepak Vasisht, Jayanth Shenoy, and Ranveer Chandra. 2021. L2D2: Low latency distributed downlink for LEO satellites. In *ACM SIGCOMM*.
- [29] Bowei Yang, Yue Wu, Xiaoli Chu, and Guanghua Song. 2016. Seamless handover in software-defined satellite networking. In *IEEE Communications Letters*.
- [30] Xiaoqi Yin, Abhishek Jindal, Vyas Sekar, and Bruno Sinopoli. 2015. A control-theoretic approach for dynamic adaptive video streaming over HTTP. In *ACM SIGCOMM*.
- [31] Chao Yu, Akash Velu, Eugene Vinitsky, Jiaxuan Gao, Yu Wang, Alexandre Bayen, and Yi Wu. 2022. The surprising effectiveness of ppo in cooperative multi-agent games. In *Advances in Neural Information Processing Systems*.
- [32] Haoyuan Zhao, Hao Fang, Feng Wang, and Jiangchuan Liu. 2023. Realtime Multimedia Services over Starlink: A Reality Check. In *NOSSDAV*.