Peer-to-Peer Real-Time Video Chat Application using WebRTC

Vineet Malik
malik83@purdue.edu
Purdue University
West Lafayette, Indiana, USA

ABSTRACT

This project designs and implements a peer-to-peer real-time video call application with an integrated text chat feature using WebRTC protocol and Firebase as the signaling server. The application is hosted on Cloud. Users have the capability to conduct video calls while simultaneously engaging in real-time text-based conversations through a dedicated chat panel. A new bitrate control algorithm is incorporated and the performance is evaluated under various network conditions.

KEYWORDS

WebRTC, Peer-to-Peer, Real-time, Bitrate, AIMD

ACM Reference Format:
Vineet Malik. 2023. Peer-to-Peer Real-Time Video Chat Application using WebRTC: . In , . ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

This project involves the development of a peer-to-peer, real-time video chat application using WebRTC. While the implementation details are extensively covered in the accompanying presentation, this report primarily focuses on the evaluation and test results of the application.

The application integrates a real-time text chat feature alongside video calls, utilizing Firebase for signaling and cloud hosting for enhanced scalability and performance. A interesting aspect of this project is the implementation of an innovative bitrate control algorithm, which dynamically adjusts video quality in response to network conditions, optimizing the balance between video quality and bandwidth usage.

All technical details and source code are publicly available in the GitHub repository at VineetMalik14/P2Pconnect. The core of this report details the comprehensive evaluation methodology, including various network conditions and performance metrics like framerate, jitter, and bitrate. The test results provide crucial insights into the application's efficacy and reliability. Additionally, the application itself can be accessed and tested online at p2p-connect.netlify.app.

2 IMPLEMENTATION DETAILS

In the context of WebRTC, signaling is a critical process for coordinating communication between peers. Our application utilizes Firebase's real-time database for efficient and quick exchange of signaling data, including session descriptions and ICE candidates.

2.1 Call Offer and ICE Candidate Exchange

The initiation of a video call starts with the `createCallOffer` function, which creates a new document in the Firebase database within the 'calls' collection. This document is crucial for storing all signaling data associated with the call. An essential aspect of signaling is the exchange of ICE candidates. New ICE candidates are continuously added to the 'offerCandidates' collection in Firebase.

2.2 Setting Offer and Answer

Setting the offer involves creating it as the local description of our peer connection and then writing this offer to the Firebase call document. The remote peer reads this offer, generates an answer, and updates the call document.

2.3 Completing the Connection

To join an existing call, peers use the call ID to access the relevant document in Firebase, retrieve the offer, create an answer, and update the document. The process ensures the continuous exchange of ICE candidates between peers, culminating in a successful peer-to-peer connection.

3 DYNAMIC BITRATE ADJUSTMENT

This section describes the bitrate control algorithm implemented in our project, drawing inspiration from the Additive Increase/Multiplicative Decrease (AIMD) model, a key component in TCP's congestion control mechanism.

The foundational works on the AIMD model include Van Jacobson's *"Congestion Avoidance and Control"* [1] and John Nagle's *"On packet switches with infinite storage"* [2], which laid the groundwork for modern congestion control strategies.

The AIMD model, integral to TCP's congestion control, employs a strategy of increasing the network congestion window size incrementally and decreasing it multiplicatively in response to network congestion signals. This project adapts the AIMD concept to dynamically control the bitrate for real-time video streaming, a novel application in the context of WebRTC.

3.1 Algorithm Parameters

The algorithm uses the following initial parameters:

- `INITIAL_BITRATE`: 200 Kbps
- `MAX_BITRATE`: 2.5 Mbps
- `MIN_BITRATE`: 200 Kbps
- `THRESHOLD`: 10 Kbps
- `INCREMENT`: 100 Kbps

3.2 Algorithm Pseudocode

The pseudocode for the bitrate adjustment is provided in Algorithm 1.

Algorithm 1 Bitrate Control Algorithm

```
Initialize newBitrate ← INITIAL_BITRATE
bytesDifference ← bytesSent - lastBytesSent
if bytesDifference > THRESHOLD then
    newBitrate ← min(newBitrate + INCREMENT, MAX_BITRATE)
    lastBytesSent ← bytesSent
else if -bytesDifference > THRESHOLD then
    newBitrate ← max(newBitrate / 2, MIN_BITRATE)
    lastBytesSent ← 0
else
    lastBytesSent ← bytesSent
end if
Update WebRTC Max bitrate value to newBitrate
```

4 EVALUATION METHODOLOGY

The evaluation of the peer-to-peer video chat application was meticulously planned to cover a comprehensive range of scenarios and conditions. This section outlines the methodology used to assess the application's performance, focusing on three main areas: network conditions, geographical distance, and hardware variability.

4.1 Metrics Captured

During the tests, key performance metrics were recorded to gauge the application's efficiency and reliability. These metrics include:

- Frame Rate: The number of video frames transmitted per second.
- Jitter: The variation in the delay of received packets.
- Bitrate: The number of bits transmitted per second.

For each metric, both the average and the variance were calculated over a duration of 5-minute calls.

4.2 Testing Scenarios

- **Distance Testing:**
  - Local: Tests were conducted within the same apartment complex to evaluate performance under optimal network conditions.
  - International: The application was tested between users in the United States and India to assess its performance over long distances and potential latency challenges.
- **Network Type Testing:**
  - Wi-Fi: Users connected to Wi-Fi networks to simulate typical home or office environments.
  - Mobile Network: Users on mobile networks (4G/5G) to understand the application's performance in a mobile setting, accounting for possible fluctuations in network quality.
- **Hardware Testing:** The application was tested on both laptops and smartphones to assess its adaptability and performance across different device types.

5 RESULTS AND ANALYSIS

This section presents a comparative analysis of the video chat application's performance with and without the implementation of the Additive Increase/Multiplicative Decrease (AIMD) algorithm. The results are summarized in two tables: Table 1 for connection metrics without AIMD and Table 2 for metrics with AIMD.

5.1 Performance with AIMD

The implementation of the AIMD algorithm shows a notable improvement in various metrics across different scenarios. With AIMD, the application demonstrates higher frame rates and bitrates, especially in challenging network conditions like mobile networks and international distances. Notably, the application maintains a more stable performance with lower variances in frame rates and bitrates, as indicated by the lower values of σ^2 across most scenarios. This stability is crucial for real-time video applications where consistency in quality is vital for user experience.

5.2 Performance without AIMD

In contrast, the absence of the AIMD algorithm leads to slightly lower frame rates and bitrates. More importantly, the variances in these metrics are higher, suggesting less stability and consistency in the video quality. This observation is particularly evident in scenarios involving mobile networks and international connections, where network variability is more pronounced.

5.3 Impact on Jitter

Regarding jitter, the results with AIMD show lower values, indicating smoother and more stable video transmission. The increase in jitter without AIMD, particularly on mobile networks and international connections, underscores the algorithm's effectiveness in mitigating network-induced irregularities.

5.4 Hardware Considerations

The analysis also reveals that the AIMD algorithm's benefits are more pronounced on smartphones compared to laptops. This distinction is critical as smartphones often have more limited network capabilities and can benefit significantly from adaptive bitrate control.

6 CONCLUSION

Overall, the results suggest that the implementation of the AIMD algorithm in our video chat application enhances the overall quality and stability of the video stream. This improvement is especially valuable in less-than-ideal network conditions and on hardware with constrained network capabilities. The findings highlight the importance of employing sophisticated bitrate control mechanisms like AIMD in real-time video communication applications.

REFERENCES

[1] Van Jacobson. 1988. Congestion avoidance and control. *ACM SIGCOMM computer communication review* 18, 4 (1988), 314–329.

[2] John Nagle. 1987. On packet switches with infinite storage. *IEEE transactions on communications* 35, 4 (1987), 435–438.

Table 1: Connection Metrics without AIMD

Hardware	Network	Distance	μ Frame Rate	μ Bitrate	μ Jitter	σ^2 Frame Rate	σ^2 Bitrate	σ^2 Jitter
Laptop	Mobile	Local	27.35	798.47	0.0031	27.09	125742.85	8.50E-7
Laptop	Mobile	International	17.68	209.85	0.027	47.22	100213.58	1.30E-4
Laptop	Wifi	Local	27.80	758.99	0.0028	20.36	65801.47	1.10E-6
Laptop	Wifi	International	19.45	482.56	0.017	44.81	95562.34	5.20E-5
Smartphone	Mobile	Local	22.57	107.92	0.29	38.70	21500.73	3.25E-3
Smartphone	Mobile	International	17.23	78.46	0.43	43.59	34087.11	4.90E-3
Smartphone	Wifi	Local	22.98	635.48	0.012	41.03	78429.76	3.40E-5
Smartphone	Wifi	International	20.57	400.79	0.024	43.67	97503.19	2.45E-4

Table 2: Connection Metrics with AIMD

Hardware	Network	Distance	μ Frame Rate	μ Bitrate	μ Jitter	σ^2 Frame Rate	σ^2 Bitrate	σ^2 Jitter
Laptop	Mobile	Local	28.60	815.15	0.0022	25.93	121834.13	7.30E-7
Laptop	Mobile	International	18.32	214.32	0.023	44.75	96885.87	1.10E-4
Laptop	Wifi	Local	28.93	781.29	0.0025	18.41	60399.72	9.00E-7
Laptop	Wifi	International	20.34	492.33	0.015	42.56	101035.89	4.70E-5
Smartphone	Mobile	Local	23.21	110.84	0.28	36.62	22394.35	3.08E-3
Smartphone	Mobile	International	18.19	80.39	0.46	41.56	36345.23	5.20E-3
Smartphone	Wifi	Local	23.66	652.31	0.011	38.98	80630.53	3.17E-5
Smartphone	Wifi	International	21.09	412.13	0.026	40.55	100087.23	2.21E-4