クロスレイヤによるマルチユーザー向け全天周映像伝送システムの実装と評価

研究代表者　　グエンヴァンドゥック　　東北工業大学情報通信工学科・講師

1 はじめに

高速なモバイルネットワークと格安 VR ヘッドセットの普及で、人々は今やいつでもどこからでも仮想現実コンテンツを資料することができる。最も有望な VR コンテンツの一つは、サッカーの試合や音楽のコンサートなどのイベントからの 360 度映像配信である。従来の 2D 映像とは異なり、360 度映像はあらゆる方向のシーンを捉える[1]。VR ヘッドセットで 360 度映像を視聴する際、ユーザーは自由に視点を制御して、好きな視点から映像を観ることができる。このような体験は「没入体験」と呼ばれ、VR の主要な特徴の一つである[2]。

モバイルネットワークの時間変動する特性および 360 度映像の高いデータレートのため、モバイルネットワークを介したライブ 360 度映像のストリーミングにおいて優れたユーザー体験品質を提供することが難しい。360 度映像は少なくとも 4K の解像度が必要であり、従来の 2D 映像よりもはるかに高いビットレートを持つ。例えば、解像度 4K の圧縮された 360 度映像のビットレートは 35-45Mbps であり、これは典型的な HD 映像のビットレートの 4〜5 倍高い [3]。さらに、モバイルネットワークのスループットは、ユーザーの移動や電波状況の悪さなど、多くの要因の影響で時間とともに大きく変動する[4]。特に、ネットワークスループットの急激な低下は、映像ストリーミングシステムに悪影響を与える。特に、再生中の一時的な停止（再バッファリング）[5]を引き起こし、ユーザー体験品質に低下する[6]。

これまで、ネットワーク上での 360 度映像ストリーミングに関する研究は、主にユーザーが知覚する映像品質を最大化することに焦点を当てていた。そのために、360 度映像のデータレートを削減するための技術が提案された [7]–[10]。特に、タイルを用いたビューポート適応型ストリーミング手法は、360 度映像のデータレートを 8 割まで削減することに成功した[11]。スループットの変動に対処するため、これまでの研究では将来のネットワークスループット予測手法を使用した[7,9,10]。さらに、個々の映像のセグメントのタイルのバージョンの決定は、ビデオタイルのダウンロードの前に一度だけ行われる。この手法はネットワークスループットの急な低下に対して対処することができない問題点がある。特に、バッファサイズが小さい設定では、1 つ以上のタイルが再生の期限後に受信する可能性が非常に高くなり、映像の再生が強制的に停止される再バッファリングイベントが発生するおそれがある。

本研究では、モバイルネットワークを介した低遅延の 360 度映像ストリーミングに対する新しい適応方法を提案する。ネットワークスループットの変動や小さなバッファサイズにおいても、途切れない再生を確保しながら高い映像品質を提供できる。

公益財団法人電気通信普及財団
研究調査助成報告書 第38号 2023年度

2 提案手法

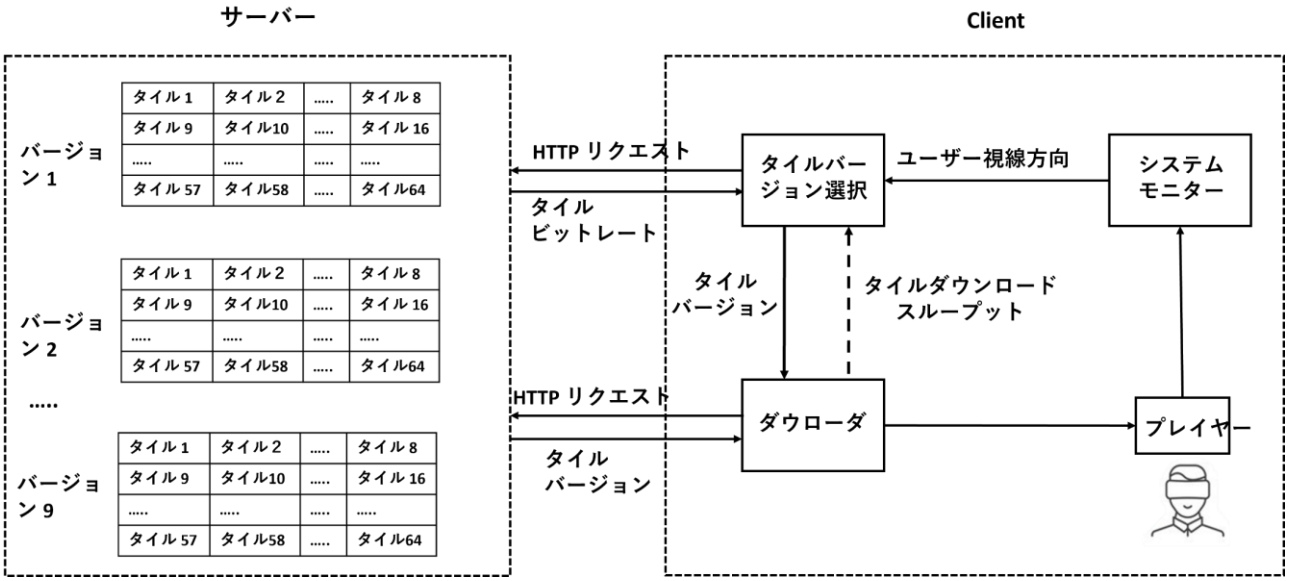

図 1 提案システムの概要図

提案システムの構成図は図 1 に示される。提案システムは、ストリーミングサーバーとユーザーデバイス（VR ヘッドセット）上で動作するクライアントから構成される。サーバーは、360 度映像のタイルを複数のバージョンにエンコードして保存する。タイルのバージョンのビットレート/品質情報やその他の説明情報はメタデータとして保存される。クライアントでは、タイルのバージョン選択、ダウンローダー、プレーヤー、およびシステムモニターの４つのコンポーネントが含まれる。タイルのバージョン選択には 2 つのタスクがある。1 つ目のタスクは、タイルの基本バージョンの選択である。2 つ目は、表示されるタイルのバージョンと再生速度の更新である。一つ目のタスクでは、タイルのビットレート情報とシステムモニターから受信したユーザーの視聴方向を使用して、各タイルのバージョンを選択する。2 つ目のタスクでは､スループットの低下が発生した場合にネットワークの状態に基づいてタイルのバージョンを更新し、再生速度を低下させる。プレーヤーは、タイルのバージョンをデコードし、360 度映像を再構築する。その後、プレーヤーはユーザーの現在の視聴方向に対応するビューポートを抽出して画面に表示する。タイルのバージョン選択からの更新を受け取ると、更新情報に従って再生速度を調整する。システムモニターは 2 つの役割を果たす。1 つ目は、ユーザーの視聴方向の監視で、2 つ目は、定期的にタイルのバージョン選択にフィードバックを送信することである。

2.1 問題定義

提案システムでは、360 度映像を T 個のタイルに空間的に分割され、各タイルは N 個のバージョンにエンコードされるとする。各タイルバージョンはさらに時間的にセグメントに分割され、再生時間がΩ秒のセグメントが生成される。セグメントの数Sは、Wを映像の長さとして、$S = W/\Omega$で与えられる。ここで、

$R(s,t,n)$と$Q(s,t,n)$は、セグメント$s(0 \le s < S)$のタイル$t(0 \le t < T)$のバージョン$n(0 \le n < N)$のビットレートと品質をそれぞれ示す。タイルのバージョンはリアルタイムで生成され、新しいセグメントはΩ秒ごとに生成される。クライアントでは、ネットワークの変動に対応するために長さB（秒）のバッファを維持する。本研究では、初期バッファサイズがセグメントの再生時間と等しい、つまり$B = \Omega$の遅延の低いストリーミングを綱領する。クライアントは、最初のセグメントのすべてのタイルが完全にダウンロードされた後に 360 映像の再生を開始する。その後、クライアントのタイルのバージョン選択は、ネットワークのスループット、タイルのビットレート、およびユーザーの視聴方向を考慮して、次のセグメントのタイルのバージョンを決定する。$V(s,t)$は、セグメントsのタイルtに選択されたバージョンとし、セグメントsのすべてのタイルをダウンロードするのにかかる時間は、以下の式で計算される。

$$\tau(s) = \frac{\sum_{t=0}^{T-1} R\big(s,t,V(s,t)\big) \times \Omega}{RC(s)}$$

ここで、$RC(s)$はセグメントsのダウンロード中の平均ネットワークスループットを示す。ダウンロード時間がダウンロード開始時のバッファサイズよりも長い場合、再バッファリングイベントが発生し、映像の再生が一時停止される。また、次のセグメントのタイルのダウンロードが、サーバーで生成された後のみ開始することが許される。セグメントsのすべてのタイルが完全にダウンロードされた時刻を$\pi^f(s)$で示すと、セグメント$(s+1)$のダウンロードまでの遅延時間$\pi^d(s)$は、以下の式で与えられる。

$$\tau^d(s) = \max\,(0, (s+1) \times \Omega - \tau^f(s))$$

セグメントsにおける再バッファリング時間$Rebuf(s)$は、以下の式で与えられる。

$$Rebuf(s) = \max\,(0, \tau(s) + \tau^d(s-1) - b(s-1))$$

式(3)の$b(s-1)$はセグメント$(s-1)$のダウンロード後のバッファサイズを示しす。セグメントsのダウンロード後、1 つのセグメントがクライアントのバッファに追加される。したがって、バッファサイズ$b(s)$は以下のように更新される。

$$b(s) = \max\,((b(s-1) - \tau(s) - \tau^d(s-1), 0) + \Omega$$

ネットワークのスループット$RC(s)$とユーザーの視聴方向$U(s)$が与えられた場合、タイルの品質値、ユーザーの視聴方向、再バッファリング時間の関数である総合品質OQを最大化するための最適な値$\{V(s,t), 0 \le s < S, 0 \le t < T\}$を求める。

$$OQ = f(Q\big(s,t,V(s,t)\big), U(s), Rebuf(s))$$

一般に、タイルの品質値が高いほど、総合品質OQも向上する。一方、再バッファリングはユーザーの体験に悪影響を及ぼすため、できるだけ小さくする必要がある。

2.2 タイルのバージョン選択

利用可能なスループット$RC(s)$とユーザーの視聴方向$U(s)$に基づいて、図２で示すセグメントのタイルの適切なバージョンを決定するアルゴリズムを提案した。視聴方向に対応する可視タイルと不可視タイルのリストはそれぞれＥとＰで示され、不可視タイルに最低品質のバージョンを割り当てる。次に、不可視タ

イルのすべてのビットレートの合計R^Pを計算し、残りのビットレートRC^2を計算する。各バージョンで、そのバージョンの可視タイルのビットレートの合計R^Eを計算する。R^EがRC^2よりも大きい場合、現時点のバージョンが保存される。すべてのタイルを伝送する必要があるため、可視タイルにバージョン$(i-1)$を割り当てる。タイルのバージョンの選択結果 $\{V(s,t), 0 \le t < T\}$は、ダウンローダーにてサーバーから順番にダウンロードする。タイル t のダウンロードが完了した後、$\tau(s,t)$はタイルtのダウンロード時間とおくと、タイルtのダウンロードスループットである$thrp(s,t)$は以下のように計算される。

$$thrp(s,t) = R\big(s,t,V(s,t)\big) \times \frac{\Omega}{\tau(s,t)}$$

```
可視タイルリスト E を作成;
   不可視タイルリスト P を作成;
不可視タイルに最低品質のバージョンを割り当てる:
for t ∈ P do
  | V(s,t) ← 0 ;
end
不可視タイルの合計ビットレート R^P の計算;
R^P ← 0;
for t in P do
  | R^P ← R^P + R(s,t,0)
end
RC_2 ← RC(s) − R^P;
可視タイルのバージョンを決定;
for i ← 0 to N − 1 do
  | R^E ← 0;
  | for j in E do
  |   | R^E ← R^E + R(s,j,i)
  | end
  | if R^E > RC_2 then
  |   | break;
  | end
end
if i=0 then
  | i++;
end
可視タイルのバージョンを i-1 にする;
for t in E do
  | V(s,t) ← i − 1;
end
```

図２タイルバージョン選択アルゴリズム

タイルのダウンロードスループットに基づいて、ネットワークスループットの低下を検出して対応する方法を提案した。利用可能なスループット$R(s)$は、前セグメントのダウンロード中のネットワーク状況から推定される。特に、現在のセグメントの最後のタイルのダウンロードスループットを次のセグメントの推定ダウンロードスループットとして使用する。

2.3. タイルのバージョンと再生速度の適応

```
1  if (t == 0 ∨ thrp(s,t) < RC(s−1))∧
     (thrp(s,t) < thrp(s,t−1)) then
2    空リスト L を作成;
3    for t′ ← t to T − 1 do
4      if V(t′) ≠ 0 then
5        t′ を L に追加する;
6      end
7    end
8    if |L| > 0 ∨ V(s,L(0)) ≠ 0 then
9      for v ← V(s,L(0)) to 0 do
10       for t′ in L do
11         V(s,L(t′)) ← v;
12       end
13       Rr^E ← 0;
14       for t′ ← t + 1 to T − 1 do
15         Rr^E ← Rr^E + R(s,t′,V(s,t′));
16       end
17       τ^E ← Rr^E ÷ thrp(s,t);
18       if (Σ_{t′=0}^{t} τ(s,t′) + τ^E) ≤ b(s−1) ∧ v = 0
          then
19         if (Σ_{t′=0}^{t} τ(s,t′) + τ^E) > b(s−1) then
20           b_cur = b(s−1) − Σ_{t′=0}^{t} τ(s,t′);
21           ς(s) = α × b_cur / τ^E;
22         end
23         break;
24       end
25     end
26   end
27 end
```

図３　タイルのバージョンと再生速度の適応

ネットワークスループットの急激な変化に対処するために、図３で示すタイルダウンロード中にタイルのバージョンを更新するためのアルゴリズムを提案した。基本的な考え方は、タイルレベルでネットワークスループットを観測し、ネットワークスループットの低下が観測された場合にタイルのバージョンを調整することである。さらに、最低品質のバージョンをサポートするためのスループットが不十分の場合、スムーズな再生を維持するために再生速度を減らすようにする。ネットワークスループットが前のタイルのダウンロード時のスループットと比較して低下した場合、まだダウンロードされていない可視タイルをリストLに追加し、これらのタイルのバージョンを更新して、現在のセグメントのすべてのタイルが再生期限前に完全にダウンロードされるようにする。Lのタイルのバージョンを順番に低下させ、残りのタイル（$t+1$から$T-1$）の総ビットレートを計算し、タイルの総ダウンロード時間τ^Eを計算する。以前にダウンロードされたタイルと残りのタイルの総ダウンロード時間がバッファーサイズよりも小さい場合、ループを抜けて更新されたタイルのバージョンでダウンロードを続ける。ただし、最低のバージョンに達した場合でも、タイルの予想ダウンロード時間がバッファーサイズよりも大きい場合、再生速度を減速することをする。具体的には、バッファーレベルをb_{cur}とし、再生速度$\varsigma(s)$は以下のように計算される。

$$\varsigma(s) = \alpha \times \frac{b_{cur}}{\tau^E}$$

ここで、αは[0, 1]の範囲の閾値で、本研究では、αの値を 0.8 に設定する。更新された再生速度により、バッファーサイズはΔBの量増加される。

$$\Delta B = b_{cur} \times (\frac{1}{\varsigma(s)} - 1)$$

この増加したバッファーサイズの量により、残りのすべてのタイルが再生期限前にダウンロードされることが可能となる。

3. 評価結果

実験では、解像度が 3840x1920 の Equirectangular 形式の 360 度映像を使用した。サーバー上では、映像を 64 タイルに分割されて、各タイルの解像度は 240x240 で、9 つのバージョンにエンコードされる。セグメントの持続時間 Ω は 1 秒である。タイルバージョンを保存するために、Ubuntu 20.04 マシンで実行されている Apache HTTP サーバーを使用した。モバイルネットワークからの 3 つのネットワークスループットトレース[17]を使用した。クライアントは Java で実装され、8GB の RAM と 2.30GHz の CPU を搭載した Laptop Lenovo ThinkPad X1 Carbon Gen 4 で実行され、初期バッファーサイズ B は 1 秒に設定される。提案手法を 2 つの既存手法と比較した。既存手法＃１では、タイルのバージョンは前のセグメントの平均ダウンロードスループットに基づいてセグメントの開始時に決定される。既存手法＃２では、ネットワークスループットは最後の 3 つのセグメントの平均ダウンロードスループットとして推定される。実験は 3 つの異なるネットワークで行う。

手法	ネットワーク＃１			ネットワーク#2			ネットワーク#3		
	再バッファリング数	再バッファリング時間(秒)	画質(dB)	再バッファリング数	再バッファリング時間(秒)	画質(dB)	再バッファリング数	再バッファリング時間(秒)	画質(dB)
既存#1	6	0.42	38.86	5	1.45	39.08	6	2.35	39.17
既存#2	6	1.21	38.94	4	2.30	39.26	7	3.37	39.25
提案手法	0	0.00	38.86	1	0.008	39.14	4	1.02	39.20

表１ 再バッファリングイベントの数、総再バッファリング時間、および平均ビューポート品質

提案手法と 2 つの既存手法の再バッファリングイベントの数、総再バッファリング時間、および平均ビューポート品質を表１で示す。異なるネットワークスループットトレースでも、提案手法は常に最も少ない再バッファリングイベントと再バッファリング時間を実現していることが分かる。特に、ネットワークスループット#2 では、提案手法はわずか 1 回の再バッファリングイベントを経験し、再バッファ

リング時間は 0.008 秒に対して、既存手法＃1 および既存手法＃2 はそれぞれ 5 回と 4 回の再バッファリングイベントがある。また、総再バッファリング時間はそれぞれ 1.45 秒と 2.3 秒である。ネットワーク#3 では、提案手法はストリーミングセッションの開始時に主に 4 つの再バッファリングイベントを経験した。提案手法の総再バッファリング時間は 1.02 秒であり、2 つのベースライン手法よりもはるかに低い。また、提案手法の再バッファリングの回数もベースライン手法よりも低くい。ビューポートの品質に関しては、提案手法は 3 つのネットワークスループットトレースにおいて、2 つのベースライン手法と同様のパフォーマンスを実現する。これらの結果から、提案手法は再バッファリングを削減する効果があり、ベースライン手法と比較して同等のビューポート品質を実現していることが示される。

4. まとめ

本研究では、モバイルネットワーク上での低遅延 360 度ビデオストリーミングのための新しい適応方法を提案した。提案手法では、ネットワークスループットの変化を迅速に検出し、ネットワークスループットの低下時に再バッファリングを回避するためにコンテンツを適切に適応できる。さらに、再生速度を適応させ、途切れのない映像再生を維持することを実現した。実験結果では、バッファサイズが 1 秒であっても、提案手法が強力なネットワーク変動下で再バッファリングを回避しつつ高品質な画質を実現するのに効果的であることを証明した。

【参考文献】

[1] “360-degree video,” https://en.wikipedia.org/wiki/360-degreevideo, accessed: 2022-01-27.

[2] “Virtual reality,” https://en.wikipedia.org/wiki/Virtual-reality, accessed: 2022-01-27.

[3] “Wiredrive 360 video specs,” https://support.wiredrive.com/hc/enus/articles/115000282194-Wiredrive-360-Video-Specs, accessed: 2022-01-27.

[4] J. van der Hooft, S. Petrangeli, T. Wauters, R. Huysegems, P. R. Alface, T. Bostoen, and F. De Turck, “HTTP/2-Based Adaptive Streaming of HEVC Video Over 4G/LTE Networks,” *IEEE Communications Letters*, vol. 20, no. 11, pp. 2177–2180, 2016.

[5] H. T. Le, D. V. Nguyen, N. P. Ngoc, A. T. Pham, and T. C. Thang, “Buffer-based bitrate adaptation for adaptive http streaming,” in *2013 International Conference on Advanced Technologies for Communications (ATC 2013)*, Ho Chi Minh city, Vietnam, Jul. 2013, pp. 33–38.

[6] H. T. T. Tran, N. P. Ngoc, A. T. Pham, and T. C. Thang, “A multifactor qoe model for adaptive streaming over mobile networks,” in *2016 IEEE Globecom Workshops (GC Wkshps)*, Washington DC, US, 2016, pp. 1–6.

[7] D. V. Nguyen, H. T. T. Tran, A. T. Pham, and T. C. Thang, “An optimal tile-based approach for viewport-adaptive 360-degree video streaming,” *EEE Journal on Emerging and Selected Topics in*

Circuits and Systems, vol. 9, no. 1, pp. 29–42, 2019.

[8] M. Graf, C. Timmerer, and C. Mueller, "Towards Bandwidth Efficient Adaptive Streaming of Omnidirectional Video over HTTP," in *Proc. Of the 8th ACM Multimed. Syst. Conf., MMSys'17*, Taipei, Taiwan, 2017, pp. 261–271.

[9] L. Xie, X. Zhang, and Z. Guo, "360probdash: Improving qoe of 360 video streaming using tile-based http adaptive streaming," in *Proc. 2017 ACM Multimedia (MM) Conference*, CA, USA, Oct. 2017, pp. 315–323.

[10] J. He, M. A. Qureshi, L. Qiu, J. Li, F. Li, and L. Han, "Rubiks: Practical 360-degree streaming for smartphones," in *Proceedings of the 16th Annual International Conference on Mobile Systems, Applications, and Services*, ser. MobiSys '18. New York, NY, USA: Association for Computing Machinery, 2018, p. 482–494.

[11] F. Qian, L. Ji, B. Han, and V. Gopalakrishnan, "Optimizing 360 video delivery over cellular networks," in *Proceedings of the 5th Workshop on All Things Cellular: Operations, Applications and Challenges*, ser. ATC '16. New York, NY, USA

[12] Y. Ban, L. Xie, Z. Xu, X. Zhang, Z. Guo, and Y. Wang, "Cub360: Exploiting cross-users behaviors for viewport prediction in 360 video adaptive streaming," in *2018 IEEE International Conference on Multimedia and Expo (ICME)*, CA, USA, 2018, pp. 1–6.

[13] C.-L. Fan, J. Lee, W.-C. Lo, C.-Y. Huang, K.-T. Chen, and C.-H. Hsu, "Fixation prediction for 360° video streaming in head-mounted virtual reality," in *Proceedings of the 27th Workshop on Network and Operating Systems Support for Digital Audio and Video*, ser. NOSSDAV'17, Taipei, Taiwan, 2017, p. 67–72.

[14] T. C. Thang, H. T. Le, A. T. Pham, and Y. M. Ro, "An evaluation of bitrate adaptation methods for http live streaming," *IEEE Journal on Selected Areas in Communications*, vol. 32, no. 4, pp. 693–705, 2014.

[15] T. Lohmar, T. Einarsson, P. Frojdh, F. Gabin, and M. Kampmann, ¨ "Dynamic adaptive http streaming of live content," in *2011 IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks*, Lucca, Italy, 2011, pp. 1–8.

[16] C. Gutterman, B. Fridman, T. Gilliland, Y. Hu, and G. Zussman, "Stallion: Video adaptation algorithm for low-latency video streaming," in *Proceedings of the 11th ACM Multimedia Systems Conference*, ser. MMSys '20, Istanbul, Turkey, 2020, p. 327–332.

[17] C. Muller, S. Lederer, and C. Timmerer, "An evaluation of dynamic adaptive streaming over http in vehicular environments," in *Proceedings of the 4th Workshop on Mobile Video*, ser. MoVid '12, Chapel Hill, North Carolina, 2012, p. 37–42.

〈発　表　資　料〉

題　　名	掲載誌・学会名等	発表年月
LL-VAS: Adaptation Method for Low-Latency 360-degree Video Streaming over Mobile Networks	2022 IEEE Symposium on Computers and Communications (ISCC)国際学会	2022 年 6 月 ２ ９ 日