6P-09

キャラクタの精細度を維持し データ通信量を抑えたゲーム配信手法

駒牧 潤也† 中島 克人‡
東京電機大学未来科学部情報メディア学科†‡

1 はじめに

近年，コンピュータゲームのストリーミング配信が人気である．例えば，ゲームのライブ配信が中心のプラットフォーム「Twitch」の同時視聴者数は 2021 年 4 月時点で 310 万人[1]である．しかし，動画ストリーミングサービスの配信はサーバや通信回線の従量制コストの負担が大きく，視聴者側でもモバイル回線での動画視聴では通信量および通信料金の負担は大きい．解像度を落として通信量を削減する方法では動画の品質低下によりゲームの重要な情報が欠落する可能性がある．

本研究では，関心領域であるキャラクタ等の精細度を保ちつつ，他の領域の情報削減によって，動画配信の通信量を削減する手法を提案する．

2 関連研究

動画品質を保持しつつ送信データ量を減らす手法として，元動画の解像度を落として送信し，受信側のスマートフォンにおいて CNN を用いた超解像技術(SRCNN)を適用し，元動画を復元する手法[2]が提案されている．この手法はマルチビットレートでの配信を想定しており，低い解像度(854×480 px など)で受信した場合に超解像を行うことで必要なビットレートは高解像度の動画を受信する場合と比べて約 10 分の 1 に抑えられ，結果的には配信に必要な帯域幅は元動画の約半分に抑えられることが示されている．課題点は SRCNN を行う受信側である程度高性能な GPU の使用が前提となることや，シーンによってはゲームキャラクタ等のユーザの関心領域において超解像処理が効果を十分発揮しないことである．

3 提案手法

我々はキャラクタ等の関心領域の解像度を保ちつつ，それ以外の領域の解像度低下によって通信量削減を図る手法を提案する．図 1 に処理の流れを示す．

まず，物体検出器によって関心領域であるキャラクタ等を検出し，その外接矩形領域のみ切り取った動画とそれ以外の領域（以下，背景領域と呼ぶ）のみの動画を生成する．背景領域動画は解像度を下げた上で，これら 2 つの動画を送信し，視聴者側の端末で合成する．

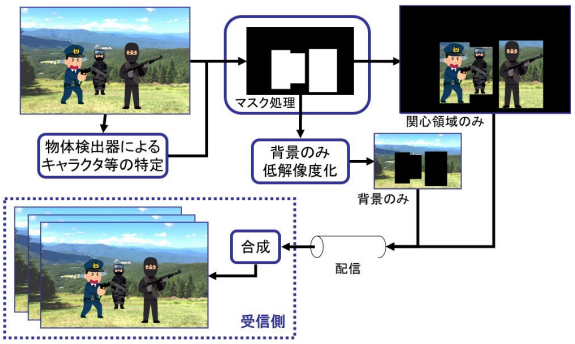


図 1 処理の流れ(画像はイメージ)

リアルタイム配信を目指すために，物体検出器には検出精度と速度の両立した YOLOv5x [3]を使用し，検出するキャラクタ等は対象ゲーム内の画像にアノテーションを付けて学習を行う．

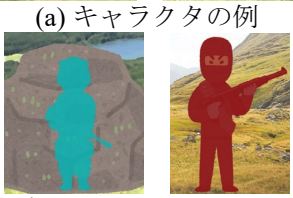
4 実験と評価

4.1 対象ゲームと学習データ

今回はゲーム配信にて一定の人気を誇る「Apex Legends」を対象とした．学習データセットは，2021 年 8 月時点で公開されている 3 種類の対戦マップにおいて，プレイヤー視点で様々なスキンのキャラクタが映し出される複数のプレイ動画からキャプチャしたものであり，解像度は 1920×1080 px である．訓練画像 911 枚，検証画像 270 枚，テスト画像 130 枚からなる．

関心領域として検出するのは図 2 に示すような「キャラクタ」とキャラクタが障害物に遮られた際に単色で表示される「シルエット」とした．

(a) キャラクタの例



(b) 単色で示されたキャラクタのシルエット例

図 2 関心領域のイメージ

A Video Game Streaming Method to Reduce Traffic while Keeping Character Resolution
† Junya Komamaki・Tokyo Denki University
‡ Katsuto Nakajima・Tokyo Denki University

4.2 関心領域検出の評価

テスト用画像 130 枚に対して，IoU≧0.4 を条件に関心領域である「キャラクタ」もしくは「シルエット」の信頼度 0.2 以上での検出数の混同行列を表 1 に示す．適合率は 89.1%，再現率は 89.5%，両者の調和平均である F 値は 89.3%となった．

表 1 関心領域の混同行列

		推論結果(検出数)	
		関心領域	背景
真値	関心領域	197	23
	背景	24	ー

4.3 動画像の加工と通信量

検出した関心領域の外接矩形内以外の画素値を 0 としたフレーム画像と，（関心領域の画素値を 0 にした）元画像を縮小したフレーム画像，および，音声からなる送信用の動画像データを作成し，受信側でそれらを組み立てて元の解像度の動画に戻す．このプログラムを画像処理ライブラリの OpenCV，および，動画像・音声の変換用の FFmpeg [4]を用いて作成した．

30~40 秒程度のプレイ動画のサンプルに対してこのプログラムで送信用動画像データを生成した場合のデータ量を，元の動画，および，単純に縮小した場合の動画のそれぞれのデータ量と比較した．元動画の解像度は 1920×1080 px，提案手法の背景を含め，縮小時の解像度は 854×480 px である．データ量の比較結果を表 2 に示す．

表 2 プレイ動画のデータ量 (MB)

	サンプル 1	サンプル 2	サンプル 3
元動画(1080p)	92.1	118.0	72.3
元動画(480p)	13.0	13.3	8.4
提案手法	16.4	17.1	12.4

提案手法により，動画のデータ量は元動画と比べて 0.14～0.18 倍に圧縮されたことが分かる．

4.4 動画品質の定量評価

サンプル動画 1,2,3 の中から代表的シーンの画像を 1 枚切り出し，SSIM による品質評価を行った結果を表 3 に示す．比較対象は 1920×1080 px の元動画からのシーン画像である．

表 3 SSIM による画像品質

	サンプル 1	サンプル 2	サンプル 3
SSIM(480p)	93.36%	95.52%	95.80%
SSIM (提案手法)	94.65%	95.64%	95.92%
関心領域の比率	18.12%	2.03%	1.69%

当然ではあるが，シーン内の関心領域の割合が多い程，低解像度画像に比べて画像品質の低下が抑制されている．

4.5 動画品質の定性評価

提案手法による動画の主観評価を得るためにゲーム動画の視聴経験がある学生 27 名に対して先ほどのサンプル動画 1,2,3 を視聴して貰い，アンケート調査を行った．結果を表 4 に示す．なお，表 4 のサンプル 3 は印象比較のために背景領域の解像度をさらに低く設定している．

表 4 アンケート調査の結果

	サンプル 1	サンプル 2	サンプル 3
背景領域＼解像度	854×480 px	854×480 px	426×240 px
元の解像度の動画と比較して品質に変化が無い	25.9%	29.6%	0%
低解像度の動画よりキャラクタの動きが分かりやすい	88.9%	81.5%	70.4%
低解像度の動画より動画内の状況を把握しやすい	59.3%	51.9%	29.6%

サンプル 1，2 のように背景領域をある程度低解像度化しても，元の解像度のキャラクタを埋め込み表示することで，単に低解像度化した動画よりもゲーム動画視聴の満足度が大幅に向上することがわかった．

5 まとめ

本研究ではゲームシーンに登場するキャラクタ等の関心領域の精細度を維持しつつ，その他の領域を低解像度に変換することによってゲーム配信に要するデータ量の削減を行う手法を提案した．

実験・評価の結果，独自学習を行った YOLOv5x による関心領域の検出では F 値が 89%となり，キャラクタ等の位置特定にはそれ程問題とならないことを示した．また，動画送信用の加工プログラムで生成した動画は，何も加工しない元動画と比較してデータ量を 0.14~0.18 倍に圧縮することができた．画質の客観指標である SSIM では本提案手法の効果は大きくないが，ゲーム視聴の立場での主観評価では単純に解像度を落とした動画に比べてかなり高評価を得ることができた．

提案手法の python/OpenCV/FFmpeg による現実装では GPU 搭載 PC(@i7-9700K+RTX 2070 SUPER)での送信が約 7 fps，受信が約 8 fps のため，速度向上のための実装見直しも今後の予定である．

参考文献

[1] TwitchTracker: https://twitchtracker.com/statistics, 2021/10/29 参照.
[2] “LevelUp: A thin-cloud approach to game livestreaming” Landon P. Cox et al.2020 IEEE/ ACM Symposium on Edge Computing.
[3] G.Jocher,“Yolov5” https://github.com/ultralytics/yolov5, 2021/10/26 参照.
[4] FFmpeg: http://www.ffmpeg.org/, 2021/11/16 参照.