7Y-03

色と深度の動画圧縮による分散並列描画サーバの性能改善

奥村直仁　齋藤豪
東京工業大学 情報理工学院

1　はじめに

ソーシャル VR のように多数のモデルの同時描画が求められるシステムでは、複数台のサーバによる並列レンダリングの採用は意味がある。

Molnar ら [1] は、並列レンダリングを描画のパイプラインにおけるソートの方法によって Sort-first 型、Sort-middle 型、Sort-last 型に分類した。Sort-first 型かつ Sort-last 型に分類される描画領域の分割による並列化 [2] や、シーン空間の分割による並列化 [3] がこれまでに提案されてきた。しかし、これらはソーシャル VR のようなモデルや視点の移動を伴うシーンでは担当するノードが切り替わってしまうために、モデルのデータ転送のコストが発生する。一方で、同じく Sort-first 型かつ Sort-last 型に分類される、モデルごとに担当するノードを割り当てた石井らの手法 [4] は、モデルのデータ転送を発生させることなくシーンを描画することが可能であり、ソーシャル VR への利用に適していると考えられる。そのため本研究では石井らの手法にモデル描画を担当するサーバの各要求に対する処理時間の短縮を目的としたハードウェア動画圧縮を導入し、このサーバがより多くの描画要求を処理することが可能かを評価した。

2　提案手法

石井らによるアーキテクチャは、単一のモデルについての描画を担当するコンテンツサーバと、深度を考慮した合成を行うマージサーバからなる。

このアーキテクチャでは特定のモデルに対する描画をそのモデルを保持するサーバに要求するため、あるモデルが同時に複数のクライアントの視界に入る場合に、一つのコンテンツサーバへ要求が集中する。コンテンツサーバは描画要求を格納するキュー構造を持ち、受信した順に処理を実行するため、一つの要求元からの描画要求は他の要求元へ描画結果を返すまでの遅延に影響する。そのため描画要求の増加に伴い、描画結果を返すまでの遅延が増加するという問題が生じる。

コンテンツサーバにおける描画処理はシーンの更新・描画・VRAM からの描画結果の読み出しという三つの処理に大きく分けられるが、これらの中で描画結果の読み出しが最も時間を要している [5]。このことから、コンテンツサーバにおける描画結果の読み出し時間を短縮するために GPU 上での動画圧縮を利用することが有効であると考えられる。

本手法では、NVIDIA 製の GPU にハードウェア実装された NVENC による H.264 動画圧縮を導入する。図 1 に示す通り、コンテンツサーバの起動時に NVENC のエンコーダを二つ初期化し、一つは RGB の色情報のエンコード用、もう一つは透明度及び深度情報のエンコード用とする。一つ目のエンコーダには 1 バイトずつ 3 チャンネルの RGB 値を入力として与え、二つ目のエンコーダには色画像の透明度 2 ビットと固定小数点数で表された深度値 22bit を 24bit の YUV 値として入力する。これらのエンコーダ入力を用意するため、描画後に圧縮前処理を新たに導入する。また、エンコーダにはロスレスの指定を行い、量子化による不可逆な変換が含まれないようにする。さらに、異なる描画要求間でエンコーダを共有するために、エンコーダにはイントラフレームのみの系列として圧縮を行うよう指定し、時系列的な相関を用いた圧縮が含まれないようにする。

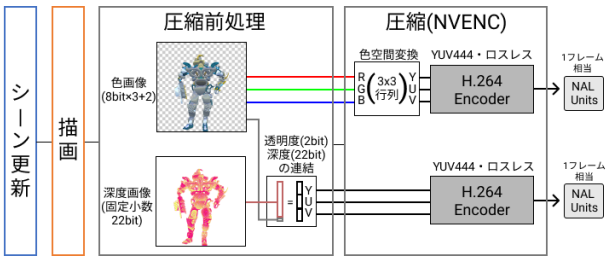


図 1: コンテンツサーバ内の 1 描画送出まで処理

3　評価と結果

3.1　評価方法

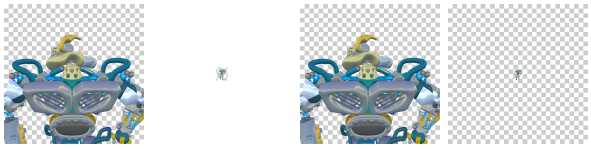
一つのコンテンツサーバに対して同時に複数の異なる視点からの描画要求を送り、描画要求に対する処理時間の構成と時間的変化を調べた。比較のため、コンテンツサーバにおける動画圧縮を用いる場合と用いない場合において実験を行った。

コンテンツサーバが保持するモデルは人型の高いポリゴン数を持つモデルとした (表 1)。モデルは初期位置から移動しないが、モーションデータが付与されており時間経過に応じてポーズが変化する。よってシー

Improving performance of distributed parallel rendering server with video compression of color and depth
Naohito OKUMURA
Suguru SAITO
School of Computing, Tokyo Institute of Technology

ン更新における処理を発生させる。また、コンテンツサーバとして表2に示す仕様の1台の計算機を用いた。

描画するシーンは要求元ごとに異なり、図2(a),2(c)に示すモデルに最も近接した視点と図2(b),2(d)に示すモデルから離れた視点の間の距離から一様乱数で決定した。圧縮を行わない場合は石井ら[4]の提案した描画領域の削減が行えるため、遠い視点の場合(図2(b))の画素の読み出し量は減少する。



(a) 圧縮なし, 近い視点　(b) 圧縮なし, 遠い視点　(c) 圧縮あり, 近い視点　(d) 圧縮あり, 遠い視点

図 2: コンテンツサーバに要求する視点

表 1: 実験に用いたモデルと描画形式

モデル (1 体あたり)	
ポリゴン数	61666 ポリゴン
関節数	59 関節
解像度	1024x1024

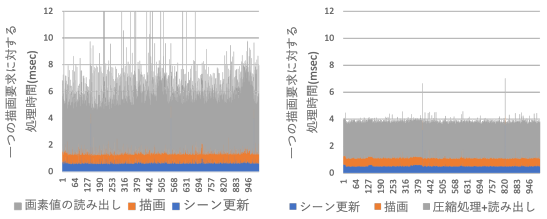
表 2: 実験に用いた環境

CPU	intel Core i5-10400F
RAM	DDR4-3200MHz 16GB × 2
GPU	GeForce RTX 2060
OS	Ubuntu 18.04.5
Linux Kernel	Linux 5.4.0
NVIDIA Driver	450.80.02

3.2　結果と考察

図3は128の要求元から1つのコンテンツサーバに対して要求を行った実験結果である。圧縮を行わずに画素値をそのままVRAMから読み出した場合(図3(a))は描画結果を返すまでに約7msec弱の時間がかかっている。一方で、ハードウェア圧縮を用いてH.264ストリームとしてVRAMから読み出した場合(図3(b))、描画結果を返すまでに約4msec程度の時間がかかっている。処理時間の内訳を見るとシーンの更新及び描画に要する時間は大きく変化しておらず、圧縮処理と圧縮結果の読み出しに要する時間の合計が圧縮なしでの描画結果の読み出しよりも短いということがわかる。動画圧縮を用いることで読み出し前に圧縮のための処理時間が発生するが、圧縮後のバイトサイズは大幅に縮小するため、圧縮なしでの画素値を読み出す場合よりも高速に描画結果を返すことができたと考えられる。

図4は要求元を32,64,128の3通りで処理時間を比較した結果であるが、圧縮を行う場合と行わない場合のいずれも同程度である。つまり、同時にコンテンツサーバへ要求をするマージサーバが高々128程度の条件下では、接続数の増加に伴う処理時間の変化は小さ



(a) 圧縮なし　(b) H.264 圧縮あり

図 3: 一つの描画要求に対する処理時間の遷移 (msec)

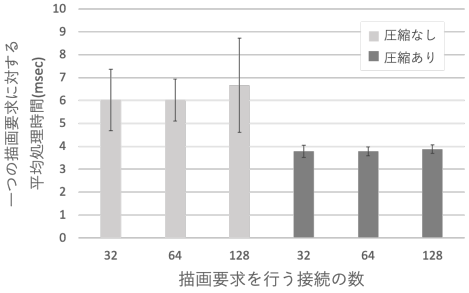


図 4: 要求元の接続数が 32,64,128 の条件での一つの描画要求に対する処理時間比較 (バーは標準偏差を示す)

いと考えられる。また時間的な変化について、圧縮なしでの描画結果を読み出す方法では処理時間の変動が大きいことが確認された。これはVRAMからの読み出し量が大きいことがCPUとGPUの間のバスの排他制御などに影響を及ぼしていることなどが考えられる。

4　結論と今後の課題

コンテンツサーバにおけるVRAMからの描画結果の読み出しにハードウェア動画圧縮を用いることで、より高速な描画要求の処理と処理時間の安定化が確認された。巨大な人数で同じシーン共有するシステムの実現に向け、サーバ構成のスケーラビリティを検証していくことが今後の課題である。

参考文献

[1] S. Molnar et al. A sorting classification of parallel rendering. Vol. 14, No. 4, pp. 23–32, 1994.

[2] R. Samanta et al. Hybrid sort-first and sort-last parallel rendering with a cluster of pcs. HWWS '00, pp. 97–108, 2000.

[3] 渡部雅人, 齋藤豪, 中嶋正之. 空間領域分割による分散レンダリングにおける画像合成並列化手法. 信学会 総合大会 D-11-104, 2008.

[4] 石井翔, 齋藤豪. 多種多様なコンテンツへのスケーリングに特化したパラレルレンダリングを用いたサーバーサイドレンダリングアーキテクチャによる合成 CG 作成法. 情処研究報告 2019-CG-173,5, pp. 1–7, 2019.

[5] 奥村直仁, 齋藤豪. 多種モデル描画のための階層的分散並列レンダリングサーバにおける深度情報の構成に対する検証と多数のクライアント下における負荷評価. 情処研究報告 2020-CG-179,2, 2020.