7ZF-07

多人数空間共有に適した分散レンダリングにおける GPU サーバ構成に関する研究

奥村直仁　Fang Hao　齋藤豪
東京工業大学 情報理工学院

1　はじめに

3D レンダリングを利用した空間共有では、ユーザ同士が固有のアバタ用モデルを介してコミュニケーションを行うことが想定される。このようなアプリケーションでは多種類かつ多数のモデルを低遅延でレンダリングすることが求められるため、より多人数の場面へ対応しづらい。

本稿では、分散レンダリングの手法を用いて複数の GPU サーバによるレンダリングを行い、さらにこれらの GPU サーバを複数のユーザ間で共有するシステムの構築について報告する。

2　システムの設計

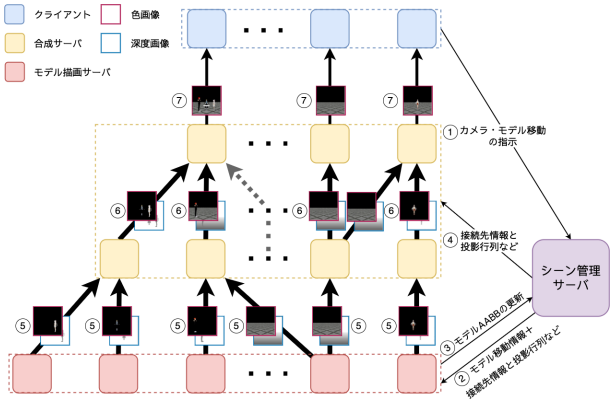

図 1: 分散レンダリングシステムの概要

本システムは石井らの分散レンダリング手法 [1] を基本設計とした報告 [2] のシステムを改良したものである。この手法は、レンダリングを「各モデルごとの部分生成画像の生成」と「部分生成画像の合成」の二種類の過程によって達成している。それぞれ「モデル描画サーバ」と「合成サーバ」と呼ばれるサーバプロセスがこの過程を処理する。部分生成画像の合成は、3D 空間上に配置された四角形ポリゴンの上にテクスチャマッピングを行うことで達成される。この際、より正確な深度比較を行うために RGBA に加えて深度値を転送し、深度スプライト法 [3] によって合成する。

図 1 は、今回検証を行う分散レンダリングシステムの概要である。本研究では、従来研究で実装実験が行われていなかった、共有空間を多数のクライアントへ個別の視点からのレンダリング結果を送信する場面を想定する。さらに、クライアント数の増加に対応するために 2 点の変更を加える。

一点目の変更は、GPU 上でのデータ圧縮である。部分生成画像の転送には、モデル描画サーバ上でのエンコードと合成サーバ上でのデコード・エンコードが必要となる。複数のクライアント間でサーバを共有するためには、部分生成画像のスループットをより高くする必要性があるため、CPU 圧縮を使用するとバス経由の GPU-GPU 間の転送が多くの処理時間を費やしてしまうためである。実装には可逆圧縮用のライブラリである nvCOMP[4] を使用して効率化を行う。圧縮方式には、最も高速に処理できることから任意回のランレングス圧縮と差分符号化の組み合わせから構成される Cascaded 方式を選択する。報告 [2] とは異なるライブラリを使用しているのは、可逆圧縮を GPU 上で行う目的でより適しているためである。

二点目の変更は、シーン管理を行うサーバの追加である。図 1 に示すように、新たに追加された「シーン管理サーバ」は他のサーバプロセス及びクライアントと通信を行い、モデル移動やカメラ移動などの情報を同期し続ける。さらに、シーン管理サーバでは各モデルの軸平行バウンディングボックスを利用した視錐台カリングを実行し、視界の範囲内に含まれるモデルのみをレンダリングするようにモデル描画サーバへ命令し、同時に合成サーバへ合成対象を修正するように命令する。

3　評価と結果

システムの評価として、TSUBAME3.0[5] 上のノードを用いてレンダリングを実行した。3D 空間上に表 1 に示すそれぞれ異なるアバタ用モデルを 100 体と静止したテーブル及び床を図 2 のように配置した。各アバターの頭部に固定したカメラ姿勢に対するレンダリングを行い、それぞれのクライアントへ完成したフレームを転送する。解像度は全てのクライアントで幅 640px・高さ 480px を使用した。サーバプロセスとして表 2 に示す通り、モデルと同数のモデル描画サーバとクライアントと同数の合成サーバを起動した。クラ

GPU Server structure of distributed rendering of shared space for multiple clients
Naohito Okumura
Fang Hao
Suguru Saito
School of Computing, Tokyo Institute of Technology

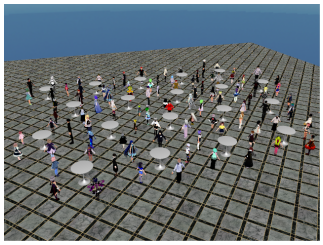
図 2: 俯瞰視点


図 3: クライアント 1


図 4: クライアント 2


図 5: クライアント 3

イアントとモデル描画サーバの間には一段の合成サーバを配置した。

図 3 から図 5 は、クライアントが受信したレンダリング結果の一例である。これは、多種類かつレンダリングコストの高いモデルのレンダリングを多人数に対して転送できていることを示している。単一の GPU でレンダリングした場合において多数のモデルをレンダリングする際に生じる GPU メモリの消費やロード時間の発生がなく、性能が低いデバイスでも高品質なレンダリング結果を表示できている。

表 1: 使用したモデル

アバタ用モデル (100 種類)	
形式	VRM(GLTF)
平均ポリゴン数	29602
平均テクスチャ枚数	14.95
平均ファイルサイズ	16.05MB

表 2: 実験条件

モデル数	121
クライアント数	100
モデル描画サーバ数	121
合成サーバ数	100
シーンサーバ数	1
使用した TSUBAME ノード数	81

表 3 はサーバにおけるフレーム生成に要する処理時間をサーバプロセスの種類ごとにまとめたものである。モデル描画サーバの平均フレーム生成時間は、1 つの視点についての部分生成画像を転送するまでの時間を表している。したがって、同じモデルを視界に入れているクライアントの数に平均フレーム生成時間を乗じた時間が、クライアントから見てそのモデルが更新されるまでの遅延となる。計測の結果から、少ない人数から見られている場合には高い更新頻度でフレームを送信できていると言える。3D 空間上でのコミュニケーションを行う上では、近接した数モデルについての更新頻度が重要であり、図 2 に示したようなテーブルを囲む人同士のやりとりであれば遅延は問題になりにくいと考えられる。一方で、特定のモデルを多くのクライアントが注視しているような状況ではモデルの更新頻度が不足する可能性があるが、この場合はモデル 1 体あたりに複数のモデル描画サーバを割り当てる方法で対処できる。

合成サーバの平均フレーム生成時間は、クライアントが受信するフレームの間隔を表している。合成サーバは最新のカメラ視点を考慮して深度スプライト法で合成を行うため、クライアントの能動的な視点移動に対して低遅延でフレームを更新することができていることがわかる。平均デコード時間は新しく受信した全てのフレームのデコード処理を含んでいるため、合成対象の枚数とモデル描画サーバからのフレームの受信頻度に影響を受ける。より多くの枚数のデコードを行う場面では、合成サーバの多段接続が有効だと考えられる。

表 3: 各サーバの計測結果

モデル描画サーバ	
平均レンダリング時間	2.83ms
平均エンコード時間	1.99ms
平均フレーム生成時間	5.14ms
合成サーバ	
平均デコード時間	11.24ms
平均レンダリング時間	0.81ms
平均エンコード時間	2.01ms
平均フレーム生成時間	14.38ms

4 結論と今後の課題

多人数の空間共有を想定した分散レンダリングシステムの設計と評価を行った。必要とするサーバ資源とクライアントに対する遅延の間で生じるトレードオフ関係についての詳細な検証は今後の課題である。

参考文献

[1] 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.

[2] 奥村直仁, 齋藤豪. 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. 情報処理学会第 83 回全国大会, 7Y-03, 2pages, 2021.

[3] Gernot Schaufler. Nailboards: A rendering primitive for image caching in dynamic scenes. In Julie Dorsey and Philipp Slusallek, editors, *Rendering Techniques '97*, pp. 151–162, Vienna, 1997. Springer Vienna.

[4] nvCOMP. `https://github.com/NVIDIA/nvcomp` (最終閲覧日:2022 年 01 月 05 日).

[5] 松岡聡, 遠藤敏夫, 額田彰, 三浦信一, 野村哲弘, 佐藤仁, 實本英之, Drozd Aleksandr. Overview of tsubame3.0, green cloud supercomputer for convergence of hpc, ai and big-data. *Tsubame ESJ. : e-science journal*, Vol. 16, pp. 2–8, nov 2017.