— 総説 —

大規模シミュレーションデータ可視化における研究の動向

松岡　大祐[1*]，荒木　文明[1]

シミュレーションデータや観測データを画像に変換する科学的可視化は，データに含まれる科学的な特徴や意味を直感的に理解するための効率的な手法の一つである．本論文では，可視化パイプラインにおけるプロセスの効率化，時系列データ可視化，遠隔可視化，および最先端の環境を用いた大規模データの可視化研究について調査を行い，報告する．大規模シミュレーションによって生成される大量のデータの可視化処理には膨大な時間がかかり，一台の計算機で行うのは容易ではない．そのようなデータを効率的に可視化するための典型的な手法として，可視化パイプライン中のプロセスの並列化およびデータ構造の最適化が用いられている．また，時系列データの可視化においては，データI/O 処理がステップ毎に生じる．そのため, 時系列データ可視化のための効率的なデータI/O 手法として，プリフェッチングや並列I/O，およびパイプライン並列処理が開発されている．特に，パイプライン並列処理は，I/O のボトルネックを除去し，高フレームレイトを実現するための手法として広く用いられている．ネットワーク環境を用いた遠隔可視化は，遠隔環境にある使用可能な計算資源を利用するための手法である．とりわけ，地理的に分散した複数のデータを取り扱うために，遠隔可視化の一形態である分散可視化が提案されている．同じく遠隔可視化の一手法である協調的可視化は，様々なレベルの可視化プロセスにおいて複数の解析者が参加できるものである．最後に，最先端の環境を用いた可視化について述べる．より大規模なデータをより高速に可視化するためには，スーパーコンピュータおよびマルチGPU システムの利用は有効な手法である．また，大規模シミュレーションの結果をより直感的に理解するための，高度なインタラクティブ性を持つバーチャルリアリティシステムを用いた可視化についても報告する．

キーワード：科学的可視化，大規模シミュレーション，並列分散処理，データ転送

2011年4月11日受領；2011年6月2日受理

1　独立行政法人海洋研究開発機構　地球シミュレータセンター

*代表執筆者：
松岡　大祐
独立行政法人海洋研究開発機構　地球シミュレータセンター
〒236-0001　神奈川県横浜市金沢区昭和町3173-25
045-778-5862
daisuke@jamstec.go.jp

— Review —

Survey on Scientific Data Visualization for Large-scale Simulations

Daisuke Matsuoka[1*] and Fumiaki Araki[1]

Scientific data visualization to convert simulation or observational data into images is one effective technique to intuitively understand the scientific features and meanings included in such data. This paper reviews the current status of research into largescale data visualization, including time-varying visualization, remote visualization, visualization using leading edge environments and optimization of the processes in a visualization pipeline. Visualization of a massive dataset produced by a large-scale numerical simulation takes a great deal of time and is not easy to perform on a single computer. Typical approaches to efficiently visualize such datasets are the parallelization of each process in the visualization pipeline such as filtering, mapping and rendering, as well as the optimization of the data structure. In the visualization of a time-varying dataset, data I/O processing occurs at every timestep of the dataset. Several methods of pre-fetching, parallel I/O and parallel pipeline processing have been developed as efficient data I/O techniques for time-varying visualization. In particular, parallel pipeline processing is widely used as an effective method that can reduce I/O bottlenecks and realize lower inter-frame delay. Remote visualization over a network enables users to utilize available computation resources and obtain visualization results on a desktop computer from the data that is retrieved from servers on the local network or over the Internet. In particular, distributed visualization, which is one of the configurations of remote visualization, is proposed in order to handle multiple datasets at remote sites. Collaborative visualization, also a configuration of remote visualization, allows multiple collaborative users to take part in several levels of visualization process. Finally, various visualization methods using leading edge environments are described. The utilization of massively parallel supercomputers and multiple GPU systems with tightly-coupled interconnecting backbones and a massive number of cores is effective to facilitate the faster visualization of larger datasets. In addition, virtual reality systems that enable users to interactively analyze such large-scale visualization results are also presented.

Keywords: Scientific visualization, Large-scale simulation, Distributed parallel processing, Data transfer

Received 11 April 2011 ; Accepted 2 June 2011

1 Earth Simulator Center (ESC), Japan Agency for Marine-Earth Science and Technology (JAMSTEC)

*Corresponding author:
Daisuke Matsuoka
Earth Simulator Center (ESC), Japan Agency for Marine-Earth Science and Technology (JAMSTEC)
3173-25 Showa-machi, Kanazawa-ku, Yokohama 236-0001, Japan
Tel. +81-45-778-5862
daisuke@jamstec.go.jp

1. はじめに

シミュレーションデータや観測データ，実験データ等の自然科学におけるあらゆる数値データは，そのままでは理解することが困難な数字の羅列である．そのため，その中に含まれる現象や構造を人間が直感的に認識するための手段として可視化は重要な役割を担っている．特にシミュレーションデータの場合は，計算機技術が向上するにつれて数値シミュレーションの計算規模も大規模化し，その結果として出力される膨大な量の数値データを効率的に可視化するための手法が常に必要とされている．例えば，2002年に登場した地球シミュレータでは1回のシミュレーションで数TBから数10TBの数値データも出力されており（Kageyama et al., 2008; Sasaki et al., 2008），現在開発が進められている京コンピュータにおいてもPBオーダーの数値データが出力される見込みである．大規模シミュレーションの結果を用いて円滑に研究活動を進めるには，少なくとも5 fps程度のインタラクティブな可視化パラメータ変更（小野，2008）や半日以内でのアニメーション生成（Uehara et al.，2006），またはシミュレーション実行中に途中経過を見ることができるリアルタイム処理によって作業効率を高めることが求められている．今後さらに加速していくと思われる数値シミュレーション研究をより発展させるためには，シミュレーション手法だけでなく革新的な大規模データ可視化の手法を開発することが必要不可欠である．本論文は，大規模数値シミュレーションによって出力される膨大なサイズの数値データを効率的に可視化するための国内外の研究事例について調査し紹介するものである．

本論文の構成は以下の通りである．第2章では，可視化処理の基本的な事項について述べる．第3章では，可視化パイプラインを構成するフィルタリング，マッピング，レンダリングの各プロセスにおいて可視化処理を効率化する手法を紹介する．第4章では，データI/O処理が連続的に行われる時系列データの可視化における効率化手法について紹介する．第5章では，遠隔地間においてデータ転送が発生する場合の可視化手法について紹介する．第6章では，最先端のハードウェアを用いた研究事例として，超並列のスーパーコンピュータおよびマルチGPUシステムを用いた大規模かつ高速な可視化や，バーチャルリアリティシステムによる可視化について紹介する．最後に第7章でまとめと今後の展望について述べる．

2. 可視化の基本事項

本章では，現在の可視化処理の基本となるデータフローモデルについて説明する．また，スカラ場やベクトル場等，様々な数値データの可視化手法について紹介する．

2.1. データフローモデル

3次元データの読み込みから2次元画像の生成までの可視化処理は，可視化パイプラインと呼ばれる多段階の過程全体で表される．図1に示すデータフローモデルは，可視化パイプラインにおける最も基本的なパラダイムを，各段階におけるデータと変換処理の流れで表したものである（Haber and McNabb, 1990）．

まず，シミュレーションによって生成された生データに対して可視化対象とするデータの抽出または加工等のフィルタリングを行う．これは，可視化対象とするステップや空間領域，物理成分の選択等である．また，必要に応じてデータの間引きや補間，座標変換，周波数変換等のデータ変換等も行われる．フィルタリングを施すことで，生データは可視化対象データに変換される．

次に，マッピングと呼ばれる，可視化対象データに対する様々な可視化操作を施す．可視化操作の例については2.2で述べるが，可視化対象データは断面や流線，等値面等の頂点データ（ジオメトリデータ）に変換される．このとき，補間値を色に対応させる伝達関数を通して，ジオメトリデータに対して色が割り当てられる．

最後に，レンダリングと呼ばれる，3次元空間上のジオメトリデータから画像データへの変換処理が行われる．ここで3次元空間から2次元平面への射影変換が行われるが，

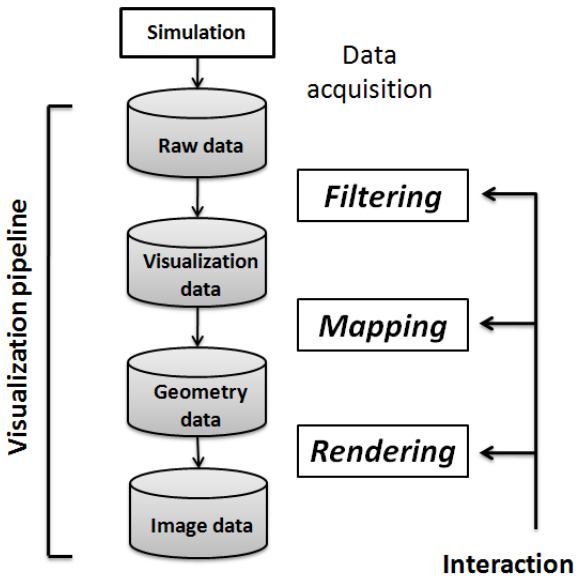

Fig. 1. Data flow model of abstract visualization pipeline (Haber and McNabb, 1990).

図1. 可視化パイプラインにおけるデータフローモデル（Haber and McNabb, 1990）.

3次元コンピュータグラフィックスにおいては平行投影および透視投影と呼ばれる二つの手法が代表的である．射影変換を行った後，設定した視点および光源情報等から陰面消去および陰影付け（シェーディング）を行い，各ピクセルの色の決定（ラスタライズ）が行われる．

上述のプロセスを実行することで，可視化結果の画像が生成される．実際の可視化処理や解析処理では，目的とする画像が得られなければ，適宜前のプロセスに戻って可視化パラメータを調整し直す．このように，可視化パラメータの設定，および可視化結果に対する解釈を対話的に繰り返しながら処理を進めるのが一般的である．

2.2. 可視化手法の例

本節では，ベクトル場およびスカラ場のデータを中心に，構造データおよび粒子データ等の基本的な可視化手法についていくつか例をかいつまんで紹介する．各説に関する詳細はHansen and Johnson (2005) を参照されたし．

2.2.1. スカラ場の可視化手法

2次元空間（平面）上に定義されたスカラ場の可視化手法には，等値線（線コンター）表示やカラーコンター（面コンター）表示がある．等値線表示は，図2 (a) に示すような，あるスカラ量が等しい点を結んだ線を表している．地形の高度分布を表す地図の等高線や，気圧分布を表す天気図の等圧線等において用いられている．図2 (b) に示すカラーコンター表示は，データの存在しない領域の補間値を求めることで，平面上の全領域の分布を表示する手法である．カラーコンター表示は，3次元データの可視化においても，任意の断面の分布を可視化する断面コンター表示として用いられる．

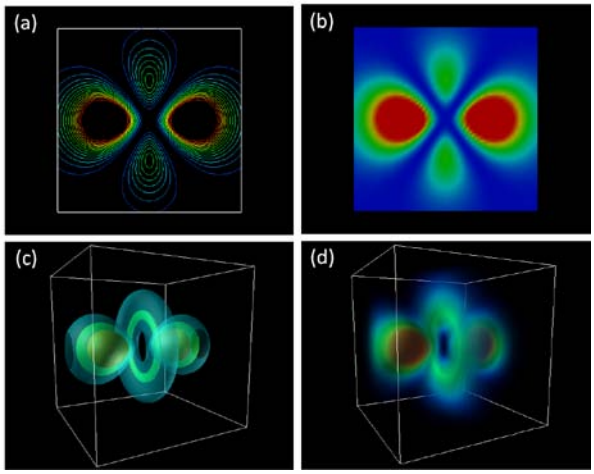

Fig. 2. Scalar field visualization. (a) Isoline (counter line), (b) counter surface, (c) isosurface and (d) volume rendering.

図2. スカラ場の可視化手法．(a) 等値線（線コンター），(b) 面コンター，(c) 等値面，(d) ボリュームレンダリング．

3次元空間上に定義されたスカラ場を可視化する手法としては，任意の値を持つ表面を抽出して表示するサーフェスレンダリングと，ボリューム全体を可視化するボリュームレンダリングがある．図2 (c) に示す等値面表示は，同じ値を表す曲面を多角形で近似することによって再構成する手法であり，アルゴリズムとしてはマーチングキューブ法（Lorensen and Cline, 1987）がよく知られている．マーチングキューブ法は，2次元データにおける等値線表示を3次元空間に拡張した手法であり，3次元データ中の特徴的な値を持つ構造を抽出することができる．マーチングキューブ法以外の等値面表示の手法としては，等値面に交差する隣接格子を再帰的に探索しながら自己増殖的に等値面を生成する手法（Isosurface propagation）（Wyvill et al., 1986）や，後述するボリュームレンダリングを応用した手法等が知られている．

ボリュームレンダリングは，スカラ場の値に対して輝度だけでなく不透明度を定義することにより，図2 (d) のように物体の内部まで透過して表現することが可能な可視化手法である．サーフェスレンダリングが特徴的な値で作られる表面構造の表現に適しているのに対し，ボリュームレンダリングは空間全体の内部構造や，明確な表面をもたない雲や煙等の構造を表現するのに適している．ボリュームレンダリングは，ボリュームデータに対して光線追跡を行うジオメトリ処理と，得られた輝度値と不透明度（アルファ値）をサンプリングしながら重ね合わせていく（アルファブレンディング）ことで画像のピクセルに色を割り当てるフラグメント処理から構成される（Levoy, 1988）．光線の追跡方法としては，視線方向の追跡（Front to back）と視線とは逆方向の追跡（Back to front）があるが，ここではアルゴリズムの容易さから広く用いられているback to frontについて説明する．逆方向の追跡におけるアルファブレンディングは，図3 (a) に示すように視点に向かう方向にボリュームデータをサンプリングし，スカラ場の値と不透明度を順に重ね合わせていく．あるサンプリング点における輝度値の入力をI_{in}，スカラ場の値をc，不透明度をαとすると，輝度値の出力I_{out}は以下のように表される．

$$I_{out} = (1 - \alpha) I_{in} + c\alpha \qquad (1)$$

アルファブレンディングの結果得られた各ピクセルの輝度値に対して色付けを行うことで，最終的な画像が生成される．

ボリュームレンダリングの手法は，レイキャスティング法等のような画像オーダーの手法と，テクスチャベース法

等のオブジェクトオーダーの手法に大別される．レイキャスティングは，図3 (b)に示すように，視点から画像の各ピクセルを通るように視線を伸ばし，その視線上にあるボリュームデータを等間隔にサンプリングしてブレンディングを行う（Roth, 1982）．視点に対して等間隔でサンプリングを行うため，ノイズの少ない画像を生成することができる．図3 (c)に示す2Dテクスチャ法は，ボリュームデータを軸方向にスライスして2Dテクスチャを生成し，奥から手前に向かってブレンディングすることでボリュームレンダリングを行う（Cabral et al., 1994; Rezk-Salama et al., 2000）．テクスチャの生成が容易なため処理が高速であるが，視点の位置や視線の方向によっては生成される画像の画質が低下するという欠点がある. 図3 (d)に示す3Dテクスチャ法は，生成する画像と平行な面でボリュームデータをスライスして3Dテクスチャを生成し，ブレンディングを行う手法である（Cullip and Neumann, 1993; Guan and Lipes, 1994）．2Dテクスチャ法や3Dテクスチャ法は, GPUのテクスチャマップ機能を用いることで高速なボリュームレンダリングが行われる．レイキャスティングやテクスチャベース法以外のボリュームレンダリング手法については，スプラッティング（Westover, 1990）やセル投影法（Max et al., 1990），粒子ベース法（Sakamoto et al., 2007）等が知られている．

2.2.2. ベクトル場の可視化手法

速度場や渦度等のような大きさと方向を持った値であるベクトル場についても，様々な表現手法が用いられている．最もポピュラーな手法は，データの定義された点を起点とした矢印（arrow glyph）で表現する手法（図4 (a)）であろう．矢印による表示は，ベクトルの大きさと方向の認識が容易であり，データの間引きによる可視化の誤差がないことから，広く用いられている手法である．矢印表示は，任意の空間におけるベクトル場の方向と大きさしか表現することができないが，ベクトル場の空間的なつながりを理解するのに適した手法として，図4 (b)に示す流線表示（streamline）がある．流線表示は，ベクトル場によって浮遊する仮想粒子の軌跡を表した可視化手法であり，ベクトル場全体の様相を容易に捉えることが可能なため様々な分野で用いられている．また，時間発展するベクトル場における追跡を表す流跡線や，同一点から次々と流れる複数の粒を結んでできる流脈線も，ベクトル場の表現手法として用いられている．矢印表示や流線表示は，起点の位置や疎密によってはベクトル場の持つ特徴が抜け落ちてしまう可能性がある．図4 (c)に示す線積分畳み込み（LIC: Line Integral Convolution）法は，ベクトル場全体を余すことなく表現することが可能な可視化手法である（Cabral and Leedom, 1993）．LIC 法では，各ピクセルを起点とした流線に沿ってホワイトノイズを線積分して畳み込むことで，流れ場の大局的な性質から局所的な性質までの特徴を詳細に表現できる．また，図4 (d)に示すように，ベクトル場の発散や収束から決定される幾何学的なトポロジー構造として

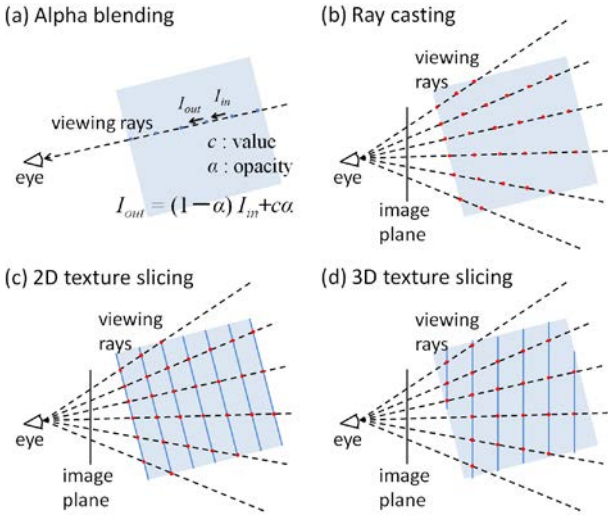

Fig. 3. Volume rendering methods. (a) Alpha blending (back to front), (b) Ray casting, (c) 2D texture slicing and (d) 3D texture slicing.

図3. ボリュームレンダリング手法．(a) アルファブレンディング（back to front），(b) レイキャスティング，(c) 2D テクスチャ法，(d) 3D テクスチャ法.

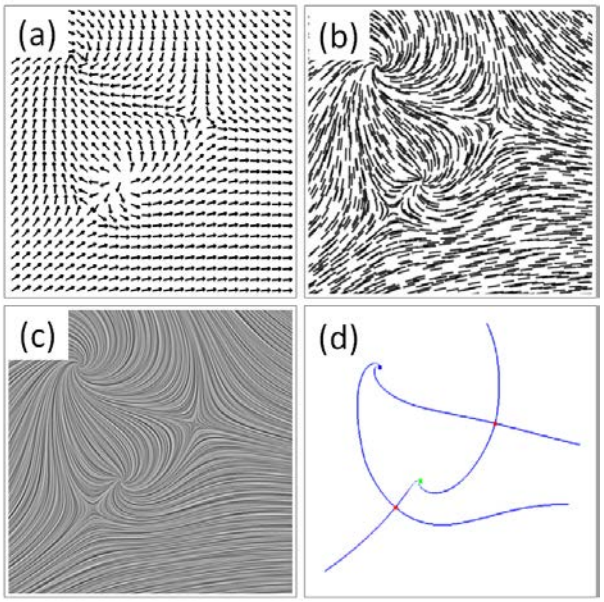

Fig. 4. Vector field visualization: (a) arrow glygh, (b) streamlines, (c) LIC and (d) topology (Weiskopf and Erlebacher, 2005).

図4. ベクトル場の可視化手法．(a) 矢印表示，(b) 流線，(c) LIC，(d) トポロジー（Weiskopf and Erlebacher, 2005）.

表現する手法も提案されている（Scheuermann and Tricoche, 2005）．これらは2次元データの可視化例を示したが，矢印表示，流線表示およびトポロジー表示については3次元データにおいても同様に適用することが可能である．また，LIC法についても，3次元に拡張した体積線積分畳み込み（VLIC: Volume Line Integral Convolution）法が考案されている（Interrante and Grosch, 1997）．

2.2.3. その他の可視化

シミュレーションデータの可視化では，場の量だけでなく粒子データや構造の可視化も行われている．粒子シミュレーションによって生成される粒子データは，点または球体で粒子の位置を表示しつつ，色でその粒子の持つ値を表現することができる（図5）．また，航空機，自動車または建物の周りの流れのシミュレーションでは，機体や車体，建造物または地形等の境界面の可視化も重要である（図6）．テンソル場の可視化についてはZhukov and Barr (2005), Zhang et al. (2005) およびScheuermann and Tricoche (2005) を参照されたし．

2.3. 代表的な可視化アプリケーション

可視化や解析を行うためのツールとしてはGnuplot やMATLAB，GrADS 等が広く用いられており，上述のような3 次元データの科学可視化に特化したアプリケーションソフトウェアも数多く開発されている．これらは，対象とするデータの可視化に特化した専用可視化アプリケーションと，様々なデータを可視化するための汎用可視化アプリケーションに分けることができる．専用可視化アプリケーションとしては，気象データ可視化のVAPORや分子構造可視化のViewMol3D，航空機の流体解析用のFIELDVIEW等が知られている．

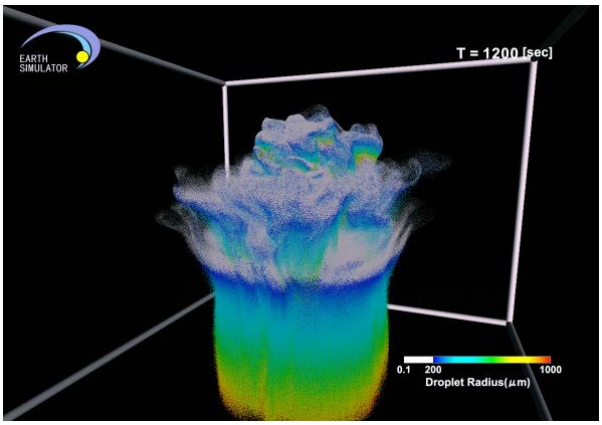

Fig. 5.　Visualization result of the particle simulation (Shima et al., 2009).

図5.　粒子シミュレーション結果の可視化例（Shima et al., 2009）．

また，汎用可視化アプリケーションとしても商用ソフトウェアのAVS/Express（Favre and Valle, 2005）やSCIRun（Weinstein et al., 2005），Iris Explorer（Walton, 2005），amira（Stalling et al., 2005），EnSightを始めとして様々なものが用いられている．例えばAVS/ExpressやSCIRunでは，図7 (a)および(b)のようにモジュール化されたデータI/O，フィルタリング，マッピング，レンダリング等の各機能をGUI（Graphical User Interface）上で接続することで，直感的な操作で容易に可視化処理を行うことができる．またEnSightやParaViewでは，図7 (c)および(d)のように，データの流れを意識せずとも可視化操作をメニュー選択するだけで可視化を行うことができる仕様になっている．これら多くの汎用可視化アプリケーションでは，前述のデータフローモデルに則ったフレームワークになっている．

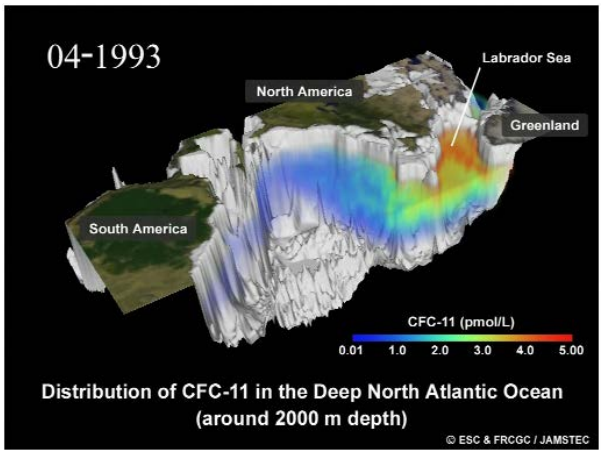

Fig. 6.　Visualization result of the submarine topography (Sasai et al., 2005).

図6.　海底地形の可視化例（Sasai et al., 2005）．

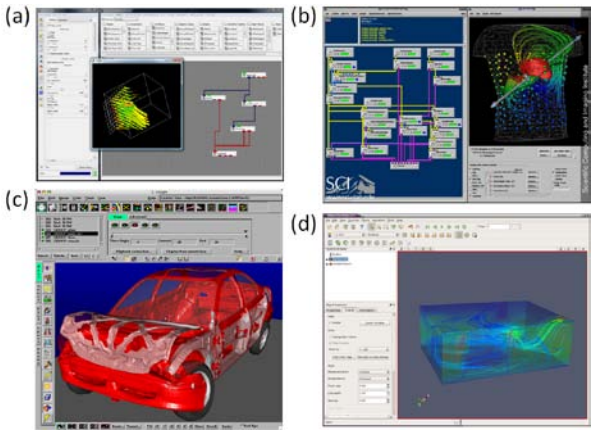

Fig. 7.　Examples of visualization software. (a) AVS/Express, (b) SCIRun, (c) EnSight and (d) ParaView.

図7.　汎用可視化アプリケーションの例．(a) AVS/Express, (b) SCIRun, (c) EnSight, (d) ParaView.

また，汎用可視化のためのクラスライブラリとして無償公開されているVTK（Visualization Tool Kit）（Schroeder et al., 2003）やユーザ独自の拡張が可能なマークアップ言語であるXML（eXtensible Markup Language）を用いることで，プラットフォームを選ばず，完全に無償の可視化アプリケーションが構築できる．その一つであるParaView（Ahrens et al., 2001; Ahrens et al., 2005）はオープンソース化されており，ユーザの用途に併せて多くの派生版が数多く開発されている（Henderson, 2004）．

詳細はThe Visualization Handbook（Hansen and Johnson, 2005）のPART IX Visualization Software and Frameworksを参照されたし．本節で名前を挙げたアプリケーションで論文引用のないものについては，参考URLを参考文献の各項目にそれぞれ記載している．

3. 可視化処理の効率化

本章では，データフローモデルにおけるプロセスの効率化手法について述べる．特に効率的な可視化を行うためのフィルタリング手法や，計算コストの高いボリュームレンダリングについて，大規模データを効率的に可視化するための取り組みを紹介する．

3.1. フィルタリングによる効率化

効率的な可視化処理を行うためのフィルタリング手法には，データサイズの削減と，最適なデータ構造への変換の2種類がある．データサイズの削減は，マッピングおよびレンダリングの高速化につながる．その手法としてはデータの間引きが考えられるが，これは，時空間の単純な間引きだけでなく，特徴（周波数）空間での間引きも含まれる．Westermann (1995)は，各タイムステップのデータに対してウェーブレット変換を用いて3次元分布の特徴的なウェーブレットスペクトラムのみを抽出することで，データ圧縮を実現した．ウェーブレット変換の応用としては，特徴的な時系列ジオメトリデータの圧縮（Sohn et al., 2002）や，異なる解像度のデータ圧縮（Guthe and Straßer, 2002）も提案されてきた．その他にも，伝達関数に合わせて表示対象外のデータを間引くことで，計算量と使用するメモリサイズを削減する手法が提案されている（Ljung et al., 2004）．

また，データの間引きや探索に適したデータ構造へ変換することで，マッピングおよびレンダリングをより効率的に行うことが可能となる．Levoy (1990)は3次元空間上の点の集合を表現するデータ構造として八分木（Octree）を考案した．八分木は，図8に示すような木構造の一種であり，ノード（または子ノード）を再帰的に八分割していくことで，異なる分解能の3次元データを効果的に表すことができる．また，Wilhelms and Gelder (1992)は，時系列データを4次元データとして扱うために，八分木と階層型ボリューム索引構造として知られる（BON: Branch-on-need）分割手法を用いたBON八分木（BONO: BON Octree）を考案した．これらのデータ構造はデータの局所的性質を利用したもので，高速なメモリアクセスが可能となる．しかし，各ノードの時空間の解像度が大きく異なる場合，データの局所性が得られずその効果を発揮しない．Linsen et al. (2002)は，異なる時空間分解能のデータを取り扱うための2の四乗根（4th-root-of-2）細分化スキームを開発した．また，Shen et al. (1999)は時系列データを効率よく処理するための時空間分割（TSP: Time-Space Partitioning）木と呼ばれる階層型データ構造を提案した．TSP木は，図9に示すように3次元空間データを八分木で，各ノードの時系列データを2分木で表現することにより，時系列のボリュームデー

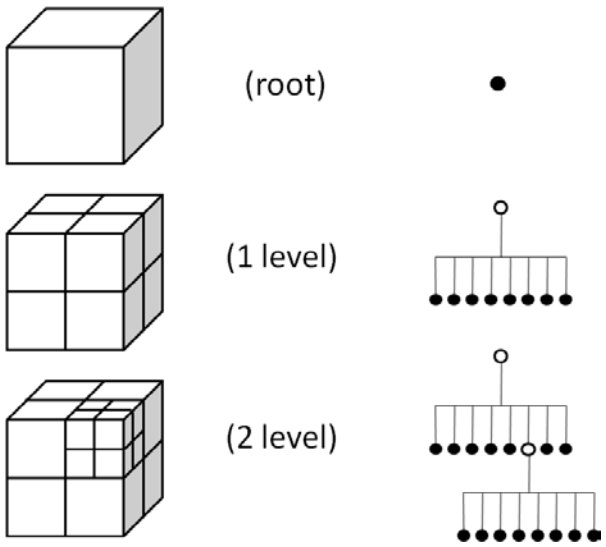

Fig. 8. Octree (Levoy, 1990).

図8. 八分木（Levoy, 1990）．

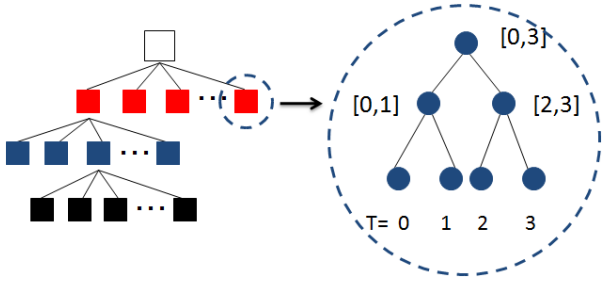

Fig. 9. TSP tree (Shen et al., 1999).

図9. TSP木（Shen et al., 1999）．

タに対するメモリアクセスおよび並列レンダリングのためのデータ分割の高速化を実現している．さらに，ウェーブレット変換を用いたウェーブレット型TSP木（WTSP tree: Wavelet-based TSP tree）（Wang and Shen, 2004）やTSP木を最適化した改良型TSP木（ETSP tree: Enhanced TSP tree）（Gao et al., 2003; Gao et al., 2004）も考案されている．

3.2. ボリュームレンダリングの効率化

ボリュームレンダリングは，ボリュームデータ全体に対する処理を行うため2.2.で挙げた可視化手法の中でも特に計算コストが高く，必要なメモリ領域も多い．そのため，高フレームレイトでのレンダリングが求められるインタラクティブ可視化やリアルタイム可視化においては，レンダリング処理の高速化が重要な課題の一つである．本節では，ボリュームレンダリングの効率化手法について述べる．

3.2.1. 並列ボリュームレンダリング

並列ボリュームレンダリングは，並列計算機が普及している現在においては汎用的に用いることのできる有効な手法である．ボリュームレンダリングの並列化手法の特徴として，Crockett (1997)はデータ並列性（data parallelism），時間並列性（temporal parallelism），タスク並列性（task parallelism）の3つに基づく手法に分類している．以下，Crockett (1997)の分類に従って，それぞれの並列性の特徴について説明する．また，タスク並列性とデータ並列性の特徴を併せ持ったハイブリッド並列性についても説明する．　一つ目の並列化手法として，データ並列性を用いた手法がある．これは，図10 (a)に示すように，各ステップにおいて処理するデータを複数に分割し，それぞれプロセッサに割り当てて同時にレンダリングする手法である．各プロセッサがレンダリングした結果を一枚の画像に合成する処理（画像重畳）においてデータ転送が発生するため，プロセッサ数の増加に伴って並列性能が低下するという問題点が挙げられる．しかし，この手法は大規模なデータの処理にも対応することができ，近年の大規模並列計算機でも最も多く用いられている．そのため，大容量データの並列ボリュームレンダリングにおいては，データ並列性に基づいた効率的なアルゴリズムが数多く研究されてきた．詳細については3.2.2.以降で述べる．

二つ目の並列化手法として，時間並列性を用いた手法がある．時間並列性に基づく手法は，図10 (b)に示すように時系列データをステップ単位で分割して各プロセッサに割り当てて並列実行する手法である．前述のデータ並列性で発生する画像重畳よりも，分割された画像の合成の方がコストは小さいため，並列化による台数効果が得られやすい．しかし，1台の計算機で処理できるデータサイズに制限があることから，大容量データの処理には向いていない．また，各ステップにおいてデータが分割されていないためフレームレイトが大きくなり，データ並列性と比べてインタラクティブな可視化処理やリアルタイム可視化には不利である．

三つ目の並列化手法として，タスク並列性を用いた手法がある．タスク並列性に基づく手法では，ボリュームレンダリングの処理要素を直列に連結し，ある要素が次の要素の入力となるように配置して処理を行う．図10 (c)に示すように各処理要素をプロセッサにそれぞれ割り当て，タイムスライス化して実行することによって，各プロセッサは常に稼働状態となり，効率的な並列処理が実現できる．これは，パイプライン処理として知られている並列化手法である．問題点として，パイプラインの中の計算コストの高い処理がボトルネックとなりうることや，プロセッサ数によってはパイプライン中のステージ数に制限があることが挙げられる．1980年代には，タスク並列性を用いた専用ハードウェアを搭載したグラフィックワークステーション

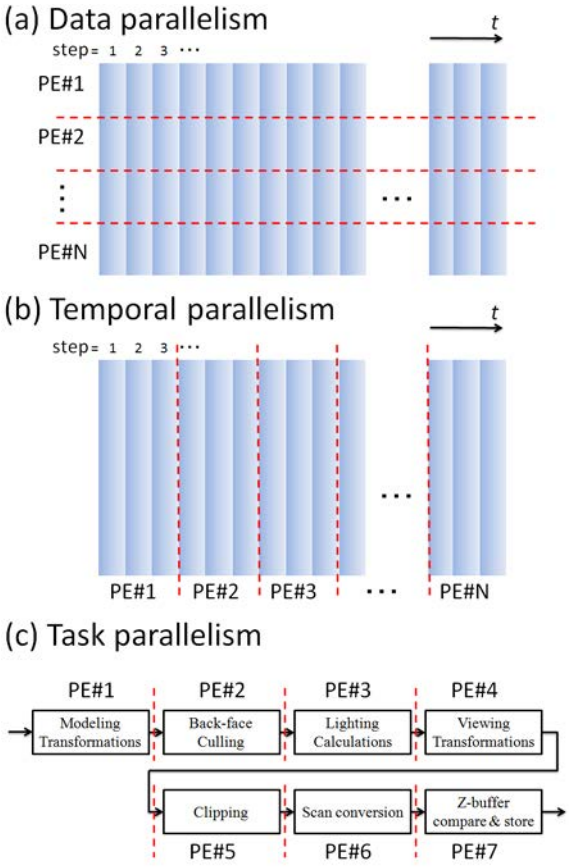

Fig. 10. Parallelization methods based on (a) data parallelism, (b) temporal parallelism and (c) task parallelism.

図10. (a) データ並列性, (b) 時間並列性, (c) タスク並列性に基づく並列化．

が流行した（Clark, 1982a; Clark, 1982b）.

また，機能並列性とデータ並列性を組み合わせたパイプライン並列性（ハイブリッド並列性）による並列化手法がある．これは，データ分割によって並列化された処理を，さらに機能ごとに分割して各プロセッサに割り当ててパイプライン処理を行う手法である．詳細については，4章の時系列データ可視化における処理の効率化で述べる．

3.2.2. ソーティングによる並列ボリュームレンダリングの分類

Molnar et al. (1994)は，データ並列性に基づくボリュームレンダリングの並列化アルゴリズムを，ソーティングのタイミングに注目してソートラスト, ソートファースト, ソートミドルの3種類に分類した．レンダリングの並列化手法は，ボリュームデータをオブジェクト空間で分割する手法と, 画像空間で分割する手法に大別される．前者の手法は，図11 (a)に示すように，分割されたブロック要素を各プロセッサに割り当て，ジオメトリ処理およびフラグメント処理を行い，中間画像を生成する．各プロセッサで生成された複数の画像は重なりあう部分が存在するため，奥行き方向のソーティング処理を行いながら画像を合成（画像重畳）することで最終画像が生成される. ソートファーストでは，図11 (c)に示す通り，スクリーン空間でデータが分割されるように，ジオメトリ処理の前にデータに対するソーティングを行って各プロセッサに割り当てる．スクリーン空間でデータを分割しているため，各プロセッサで生成した画像は重なりあう部分がない最終画像である．図11 (b)に示すソートミドルは，両者の特徴を併せ持った手法であり，データをオブジェクト空間とスクリーン空間の両方で分割し，ジオメトリ処理を行う．分割された各スクリーンの中でジオメトリデータをソーティングした後に，フラグメント処理を行って最終画像を生成する．ソートラストはジオメトリデータの存在する3次元空間でデータを分割して処理を行うオブジェクト並列性，ソートファーストは生成する画像空間でデータを分割する画像並列性，ソートミドルは両者の特徴を併せ持つ並列化手法である.

Molnar et al. (1994)の提案した3つのソーティング手法において，特にソートラストはソートファースト，ソートミドルと比べて並列計算時の負荷分散に優れ，メモリの制限等も少なく汎用性が高いため広く用いられてきた（例えばWylie et al., 2001; Nonaka et al., 2004; Eilemann and Pajarola, 2007）．ソートラストでは，画像重畳（image compositing）時に中間画像のデータ転送によるオーバヘッドが発生する．そのため，ソートラストにおける効率的な画像重畳の通信手法の研究も以下に述べるように盛んに行なわれてきた．ダイレクトセンド（direct send）型画像重畳は, 図12 (a)に示すように各ノードが自身の生成した中間画像を，全ノードに対してデータ転送を行いながら交換を行う単純なアルゴリズムである（Hsu, 1993; Neumann, 1993; Neumann,

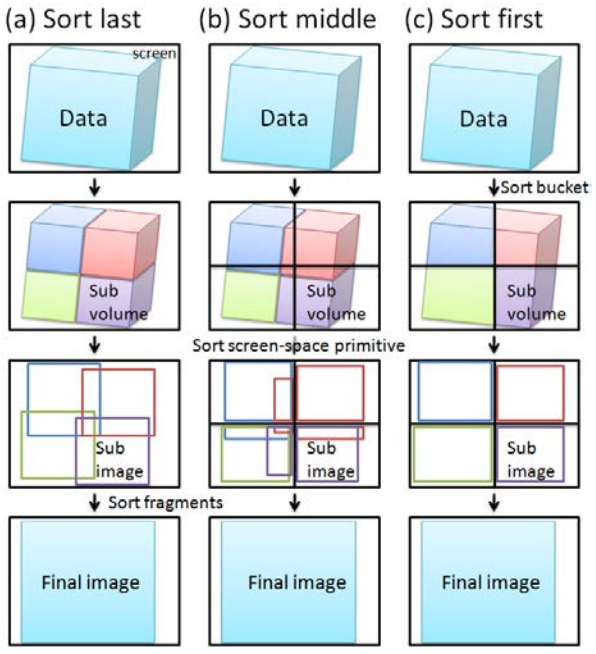

Fig. 11. Parallel rendering methods. (a) sort-last, (b) sort-middle and (c) sort-first.

図11. 並列レンダリング手法．(a) ソートラスト，(b) ソートミドル，(c)ソートファースト.

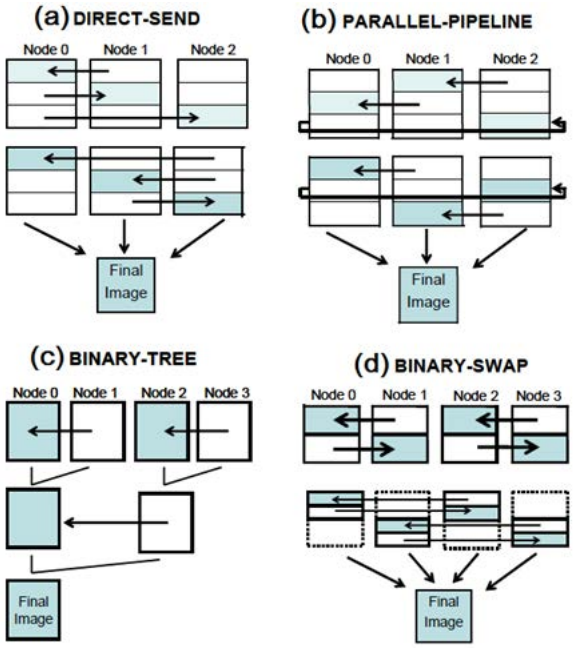

Fig. 12. Image compositing method.

図12. 画像重畳手法.

1994）．実装が容易であるため広く用いられてきたが，all-to-allのデータ転送であるため，Nプロセッサ使用時の通信コストは$O(N^2)$となる．Neumann (1993)やLee et al. (1996)は，ダイレクトセンド型を基として，並列パイプライン型の画像重畳を提案した．図12 (b)に示すように，常に隣り合ったノード間でのデータ交換を行うことで，通信の競合が回避され，データ転送の順番が最適化される手法である．通信コストは，ダイレクトセンド型と同じく$O(N^2)$である．Ma et al. (1993)およびMa et al. (1994)は，通信コストの削減を目的としてバイナリツリー（binary tree）型の画像重畳を提案した．バイナリツリー型の手法は，図12 (c) に示すように重畳ノードでペアを作り，一方のノードにデータを転送しながら最終画像を生成していくものである．2 分木法でデータ転送が行われるため，通信コストは$O(N \log_2 N)$である．ダイレクトセンド型と比べて大規模データの可視化に適し，プロセッサ間通信の負荷バランスに優れている．一方，ペアの片方しか重畳処理が行われないため，ステージが進むにつれてビジー状態となるノードが増えることが問題である．Ahrens and Painter (1998)やYang et al. (1999)は，この問題を解決するための手法としてバイナリスワップ（binary swap）型の画像重畳を提案した．バイナリスワップ型は，図12 (d) に示すように重畳ノードのペア間で半分ずつデータを交換しながら重畳処理を進めていく手法である．ステージ数はバイナリツリー型と同じく$\log_2 N$回であるが，転送されるデータサイズはステージが進むにつれて小さくなるため，効率的な重畳処理が可能になる．

3.2.3. ソートラスト型並列ボリュームレンダリングの最適化手法

前節で述べたソートラスト型並列ボリュームレンダリングは，大容量データに適した手法として様々な研究が行われてきた．これらは，一般的な並列計算における効率化問題と同様に，(1)負荷バランスの最適化，(2)データ転送の高速化，(3)パイプライン処理の最適化に大別できる．

一つ目の各プロセッサにおける負荷バランスの最適化は，台数効果を上げるための有効な手法である．並列レンダリングでは処理を高速化するために，光線が背景ボクセルに達した時点で追跡を打ち切る手法（ERT: Early Ray Termination）（Levoy, 1990）が多く用いられているが，これは各プロセッサにおいて光線の追跡距離にばらつきが出るため，負荷バランスが低下する．Lee et al. (1996)は，背景ピクセルの処理をスキップするための多重バウンディングボックスを用いることで負荷バランスの最適化を図った．また，Lee et al. (1996)およびYang et al. (1999)が提案した微粒子画像空間分割法（Finer-grain image space partition）は，用いるプロセッサの台数以上に細かく分割したデータを，ラウンドロビン方式で割り当てることによって，各プロセッサへの負荷分散を行う手法である．同様の手法をバイナリスワップに実装した例も報告されている（Takeuchi et al., 2003）．

二つ目は，データ転送の高速化である．データ転送の高速化は，負荷バランスの最適化と並んで並列処理の並列化効率を向上させるために有効な手法である．そこで，画像重畳時に発生するデータ転送において，転送される画像データを圧縮するさまざまな手法が提案されている．その一つとして，連長圧縮（ランレングス圧縮）を用いて画像重畳時の中間画像を圧縮する手法が実用化されている．連長圧縮は，ある連続したデータを，そのデータ一つと連続した長さで表現することでデータ圧縮を行う手法である．また，Stompel et al. (2003)はダイレクトセンドにおけるデータ転送の高速化手法として，線形計画画像合成（SLIC: Scheduled Linear Image Compositing）法を提案した．SLICは，ボリュームデータの3次元分割時において通信回数が分割数の最も多い辺に依存することに着目した，画像重畳時のスケジューリングの最適化手法である．Strengert et al. (2004)は，フレームバッファからのデータ読み込み，ネットワーク通信，合成のコストを最小化するためにデータ階層構造におけるブロック要素のメモリ計算を考慮する合成スキームを提案した．その他，3.1で述べたボリュームデータの圧縮によるデータ転送の高速化手法や，データ構造の最適化によるメモリアクセスの高速化もこの分類に入る．

三つ目は，データの読み込み，転送，画像重畳までを通じた全パイプラインにおける最適化である．例えば，Cavin et al. (2005)は，マルチスレッドを用いて上述の処理を重ね合わせる手法を，バイナリスワップのパイプライン・ソートラストとして実装した．これらの性能向上は全て，ダイレクトセンドのためのN^2オーダーのメッセージパッシングの項が残り，さらに2 のべき乗台のプロセッサ使用時に限られるという制約条件があった．従来のバイナリスワップでは2 のべき乗台のプロセッサ使用時でしか効率的な並列処理ができなかったという問題点を解決するため，任意の台数のプロセッサ使用時においても効率的な並列ボリュームレンダリングを行うための手法が提案されている（Yu et al., 2008）．2-3スワップ重畳と呼ばれるこの手法は，ダイレクトセンドとバイナリスワップの両方の長所を併せ持つアルゴリズムである．任意の台数のプロセッサでも効率が良いというのは，ダイレクトセンドの長所であり，数千プロセッサ使用時においても高い並列化効率が得られている．

3.2.4. ハードウェアを用いたボリュームレンダリングの効率化

上述した以外のボリュームレンダリングの効率化手法として，専用のハードウェアを用いる手法がある．初期のボリュームレンダリング用の商用ハードウェアとしては，VOGUE（Knittel and Straßer, 1994）やCubeシリーズ（Kaufman and Bakalash, 1988; Bakalash et al., 1992），VolumeProシリーズ（Pfister et al., 1999; Wu et al., 2003），VIZARDシリーズ（Meißner et al., 1998）等がある．これらは，高速なメモリアクセスやデータの補間，サンプリング機能を実装し，高速なレイキャスティングを実現するシステムであった．また，画像重畳機能のみを搭載したハードウェアとしては，Lighting-2（Stoll et al., 2001）とSepia-2（Lombeyda et al., 2001）が開発されている．これは，データ転送による通信コストの高い画像重畳を，高速な専用ハードウェアを用いて処理するシステムである．Pugmire et al. (2007)は分散レンダリングシステムにおけるハードウェア型画像重畳のためにNPU（Network Processing Unit）を利用する手法を提案した．専用ハードウェアを用いた並列レンダリングは，ソフトウェアによる並列レンダリングと比較して高速な処理が可能である一方，一般的に処理の柔軟性に欠け，高価であるという問題点がある．専用ハードウェアを用いた可視化処理については，Kaufman and Mueller (2005)に詳しい．

また，PCに搭載されたグラフィックス用の汎用ハードウェア（GPU: Graphics Processing Unit）においても，可視化への応用が盛んに行われている．1990年代に入ってGPUのグラフィックスライブラリが標準化されると，ボリュームレンダリングを高速に行うためのハードウェアアクセラレータとしてのGPUの利用が進んできた．最初の例としては，SGI Reality Engineワークステーションのテクスチャマッピング機能を用いた，テクスチャベースのボリュームレンダリングが報告されている（Cabral et al., 1994）．その他にも，メモリ領域に奥行き情報をもたせて陰影処理を高速化するZバッファや，アルファブレンディング機能を利用した例も報告されている（Van Gelder and Kim, 1996; Meißner et al., 1998）．

2000年代に入ると，GPUは固定機能シェーダからユーザによる実装が可能なプログラマブルシェーダへと移行し，画像処理以外の目的に応用が可能なGPGPU（General Purpose Computation on GPU）としての利用が進んできた．2Dテクスチャ法（Rezk-Salama et al., 2000）や3Dテクスチャ法（Schroeder et al., 2002），レイキャスティング（Kruger and Westermann2003; Röttger et al., 2003; Bitter et al., 2004; Engel, 2004），スプラッティング（Chen et al., 2004）等についてGPUを用いた高速化の研究例が報告されている．また，データの存在しない空間を読み飛ばす手法（empty-space skipping）や，前述のEarly ray terminationもGPU上に実装され，処理の高速化が図られている（Kruger and Westermann, 2003）．GPGPUを用いた高速な可視化については，GPU-Based Interactive Visualization Techniques（Weiskopf，2007）に詳しい．

GPUは高速なレンダリングが可能なためインタラクティブな可視化処理を行う場合に適しているが，GPU用のビデオメモリ（VRAM）の容量制限や，GPUからCPUへのデータ転送（リードバック）速度が遅いという問題点から，大規模なデータの可視化処理には適していないとされてきた．このような問題の改善策として，高速な探索が可能なデータフォーマットであるk-d木や，データ圧縮のためのウェーブレット圧縮，並列負荷分散のためのデータ分割手法であるMulti-brickingの利用が行われている（Weiskopf et al., 2004）．また，近年では複数のGPUを搭載したマルチGPUや，GPU搭載PCによるGPUクラスタの高性能化が進み，数値計算だけでなく可視化処理においても利用が進められつつある．マルチGPUやGPUクラスタを用いた大規模かつ高速な可視化処理への取り組みについては6.2. で述べる．

4. 時系列データ可視化における処理の効率化

時系列データを可視化する場合，複数ステップのデータ読み込み（データI/O）が連続的に発生するため，ステップを更新するたびに発生するデータI/Oが大きなボトルネックとなりうる．本章では，データI/Oの効率化手法や，データI/Oも含めた可視化パイプライン全体の効率化手法を用いた時系列可視化（Time-varying visualization）について述べる．

4.1. パイプライン並列化

時系列データの可視化においては，特徴追跡（Feature tracking）や時間発展のある速度場に沿った粒子追跡のような特別な場合を除いて，各ステップの処理は独立である．そのため，現在のステップのデータを処理している間に，次のステップのデータに対しても並列に処理を行うことが可能である．この処理の独立性を用いた，連続的に繰り返される可視化パイプライン全体の並列化手法として，3.2.1.で述べたパイプライン並列性を用いた手法がある．パイプライン並列化は，図13に示すように時間ステップに対応した複数の可視化パイプラインを，時間をずらして並列実行することにより，連続的に転送されてくるデータに対する処理を高効率に行う手法である．各プロセッサのビジー状態を減らし，負荷バランスを向上させることも

できる．図13において，PE#1とPE#2間ではフレーム間の遅延（Interflame delay）はデータが送られてくるタイミングと同等であるが，PE#3とPE#0間では遅延が大きくなる．パイプライン数を最適化することで（図の例では一つ増やすことで），このような問題を解決し，最終画像の生成や表示を連続的に行うことが可能となる．

パイプライン並列では，一ステップのデータを可視化する一つのパイプライン中に，複数台のプロセッサをグループとして割り当てることができる．パイプライン中に複数台のプロセッサを割り当てた場合，パイプライン中ではデータ並列性に基づく並列処理になる．可視化に用いる環境やデータ転送の間隔に合わせて，最適なパイプライン数または一パイプライン中に割り当てられるプロセッサ数を決定することで，効率的な時系列データの可視化を実現できる．図14に，並列パイプライン処理におけるパイプライン数と各パイプライン中のプロセッサ数の関係を示す．図14は，一ステップ分のデータ可視化に用いるプロセッサを四角形で表し，一つのパイプラインを構成するプロセッサ群を枠で囲ってグループ化して表している．図14(a)はデータ並列性に基づいており，可視化パイプラインを並列化せず一つのパイプラインで一ステップの全データを並列化する手法である．(c)は時間並列性に基づいており，

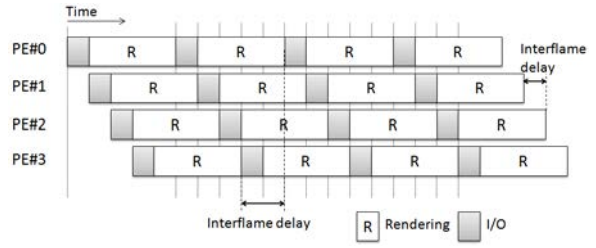

Fig. 13. Parallel pipeline for time-varying data visualization (Ma and Lum, 2005).

図13. 時系列データ可視化のためのパイプライン並列化（Ma and Lum, 2005）.

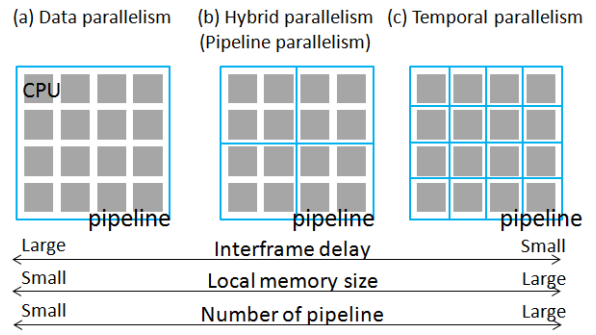

Fig. 14. Data assignment for pipeline parallelism.

図14. パイプライン並列性のデータ割り当て.

データを分割せずに可視化パイプラインのみを並列化し，一つの可視化パイプラインで一ステップのデータを処理する手法である．(b)のハイブリッド並列性はパイプライン並列処理であり，ステップに対応したパイプラインを複数本用意し，それぞれのパイプライン中ではデータを分割して並列処理を行う．

図14において，(a)の手法のようにパイプライン数を減らして一ステップのデータ可視化に用いるプロセッサ数を増やせば，フレーム間の遅延は大きくなるが，各プロセッサに必要なメモリサイズは小さくなる．一方，(c)の手法のようにパイプライン数を増やせば，各プロセッサが必要とするメモリサイズは大きくなるがフレーム間の遅延は小さくなるというメリットがある．これは，後述する実時間可視化において有効な手法である．環境や目的に合わせて，パイプライン数と各パイプライン中のプロセッサ数に有限個のプロセッサをどのように割り当て，最適化するかが重要である．

4.2. パイプライン並列におけるデータI/O処理の隠ぺい

データサイズの大規模化によって，ストレージからのデータ転送やメモリ読込み等のデータI/O処理が，全体の処理時間において占める割合が大きくなる．データI/Oのコストを軽減するための手法として，図15に示すように負荷の大きいデータI/O処理を，レンダリングを行うプロセッサとは別のプロセッサで行う手法が広く用いられている（Thakur et al., 1998; Thakur et al., 1999）．入力用のプロセッサは，数値データが保存されているストレージからレンダリング用のプロセッサまでのデータ転送を行い，出力用のプロセッサは，レンダリング用プロセッサで生成された画

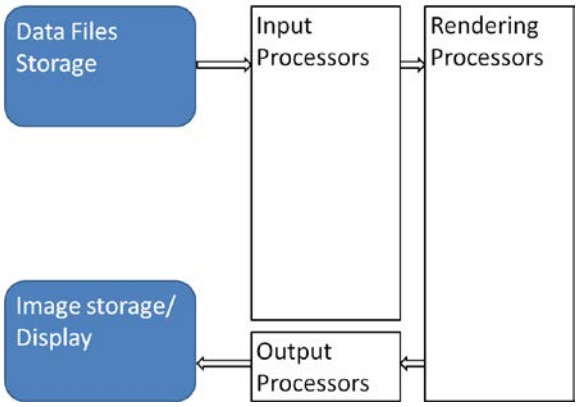

Fig. 15. System configuration for time-varying data visualization (Thakur et al., 1999).

図15. 時系列データ可視化のシステム構成（Thakur et al., 1999）.

像データの出力や，ストレージへの保存に関する処理を行う．

このようなシステム構成では，データI/O処理とレンダリングがそれぞれ別のプロセッサで実行されるため，現在のステップのデータをレンダリングしている間に次のステップのデータをストレージから読み込むことが可能になる．このデータ読み込みの隠ぺい手法はプリフェッチング（Prefetching）と呼ばれる（Arunachalam and Choudhary, 1995, Thakur et al., 1994）．図16は，プリフェッチングによるデータI/Oの隠ぺいの効果を模式的に表している．図16 (a)では，レンダリングに要する時間とI/O処理に要する時間が一致しており，I/Oに要する時間は完全に隠ぺいされている．このような場合，プリフェッチングによる効果が最大である．一方，図16 (b) に示すように，レンダリング時間よりもI/Oにかかる時間の方が長い場合は，レンダリング用のプロセッサに待ち時間が生じることになる．この問題を解決するために，複数台のプロセッサをI/O用プロセッサとレンダリング用プロセッサに効果的に割り当てる，1DIP（1-Dimensional Input Processors）と2DIP（2-Dimensional Input Processors）が提案された（Yu and Ma，2005）．これは，I/O処理の時間とレンダリング時間の比率からそれぞれの処理を行うプロセッサ数の構成を最適化し，I/O処理を最大限隠ぺいする手法である．

データI/O処理にかかる時間は完全に隠ぺいできたとしても，フレーム間の遅延はレンダリング時間によって決定される．そのため，インタラクティブな可視化処理を行うためには，高いフレームレイトを得るためにレンダリング処理を短縮する必要がある．それと同時に，データ処理の時間も短縮しなければならない．また，近年ではメモリのサイズと比べて処理するべきデータのサイズがますます増大化している．そのため，スワップ領域を含めたメモリ領域ではデータ全てを処理できない場合が多い．このような場合，データをいったんメモリに格納せず，仮想的なメモリスワッピングによって直接処理するストリーミング技術（Data streaming）も用いられてきている（Law et al., 1999; Ahrens et al., 2001）．その他の時系列データ可視化におけるデータ処理に関連した研究として，高速な専用ハードウェアを用いる手法，専用の転送プロトコルの開発，スパコンを用いたその場（in-situ）可視化によるデータ転送およびデータアクセスの軽減等が行われている．

4.3. データI/O処理の効率化

データI/Oのボトルネックは，データのアクセスパターン（サイズやファイル数，場所等）に大きく依存する．小さいサイズのデータに対して不連続なディスクアクセスが多い場合，ネットワークの帯域幅性能が低くなり，レイテンシのコストも増加するため，データI/Oのボトルネックは大きくなる．この問題の解決手法として，小さくて不連続なディスクアクセスを，大きくて連続なディスクアクセスにマージするデータシービング（Data sieving）が提案された（Thakur et al., 1994; Thakur and Choudhary, 1996; Thakur et al., 1996a; Thakur et al., 1996b）．データシービングの概念図を図17に示す．データシービングでは，ファイル上に不連続に存在する細切れのデータを，連続的なデータの塊としてメモリに読み込む．次に，ユーザからのリクエストのあった部分領域のデータをユーザのバッファにコピーする．連続かつ一定の大きさのデータのブロックを転送するため，不連続かつ細切れのデータ転送と比べて高効率な処理が可能となる．

一方，並列レンダリングではデータの読み込み後にレンダリング処理のためのデータ分割と再分配を行う必要がある．そのため，不連続で小さなサイズのデータアクセスが大量に発生し，結局はボトルネックとなってしまう．そこで，並列ディスク上に分散したデータを並列レンダリングのために再配置するところまでを考慮し，ディスクアクセス，メモリフェッチを2段階のフェーズに分けた集

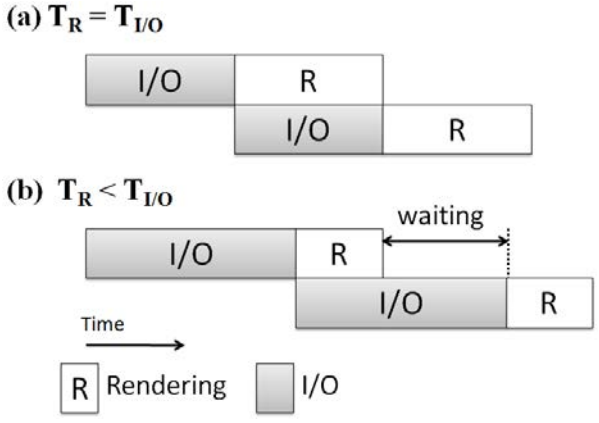

Fig. 16. Prefetching and hidden I/O.

図16. プリフェッチングとI/Oの隠ぺい．

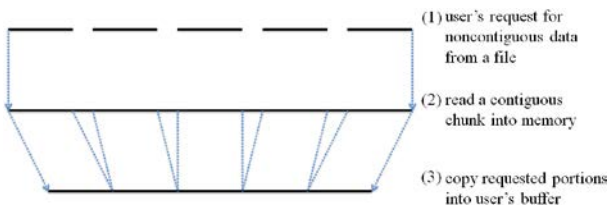

Fig. 17. Data sieving (Thakur et al., 1994).

図17. データシービング（Thakur et al., 1994）．

合的I/O（Collective I/O）が提案された（Rosario et al., 1993; Seamons et al., 1995; Kotz, 1997）．集合的I/Oの処理の流れを図18に示す．まず，最初のフェーズでは，ファイル上の細切れのデータを，データシービングによってバッファにマージして読み込む．次のフェーズでは，マージされたデータを，並列レンダリングのためのデータ分割を考慮して再配置した上でまとめてメモリに読み込み直す．これによって，ディスクアクセスの効率化を実現している．

5. ネットワーク環境における可視化処理の効率化

本章では，ネットワークを介して行われる遠隔可視化（Remote visualization）について紹介する．遠隔可視化によって，膨大な量のシミュレーションデータそのものをユーザの手元に転送する必要がなくなり，遠隔地にある高性能な計算資源を用いることも可能になる．また，分散環境を用いた分散可視化や，複数の解析者間で協調して可視化を行う協調的可視化についても触れる．

5.1. 遠隔可視化

本節では，遠隔可視化をデータ転送の手法およびシステム構成によって分類し，データ転送の高速化手法や実用例について紹介する．

5.1.1. クライアント・サーバモデルによる分類

遠隔可視化は，可視化パイプラインにおいてデータ転送が発生する（プロセス間の）場所によって分類することができる．遠隔地の計算機を利用する場合，クライアント・サーバ型のアプリケーションを用いることが多い．サーバ側とクライアント側のどちらにどのプロセスを割り当てるかによって転送されるデータの種類が異なり，図19に示す4つの構成に分けられる．

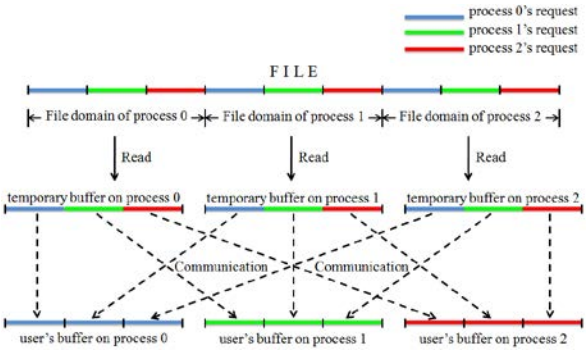

Fig. 18. Collective I/O (Rosario et al., 1993).

図18. 集合的I/O（Rosario et al., 1993）．

図19 (1)は，データが保存されているストレージのみがサーバ側にある場合で，クライアント側に数値データを転送してフィルタリングからレンダリングまでの処理を行うものである．(2)は，サーバ側においてデータのフィルタリングを行い，可視化に用いるデータのみをクライアント側に転送してジオメトリ変換やレンダリング処理等を行う．(3)は，サーバ側でデータのフィルタリングとジオメトリ変換を行い，ジオメトリデータを転送してクライアント側でレンダリング処理を行う．(4)は，サーバ側でレンダリングまでの全ての処理を行い，生成された画像データだけをクライアント側に転送して表示する．特に，シミュレーションを行ったスーパーコンピュータ等の環境をサーバとして用いて可視化を行った場合は，その場可視化（in-situ visualization）と呼ばれる．その場可視化については6.1.1.で述べる．

ここで注意しなければならないのは，転送されるデータの種類によってデータサイズが異なることである（Heermann and Pavlakos, 2005）．一般に，可視化パイプラインでは，処理が下流に進むほど処理するデータのサイズは小さくなる．最もサイズが大きいのは数値データであり，そこから必要なデータのみをフィルタリングし，ジオメトリデータに変換し，最終的には2次元の画像データになる．データセットのサイズが小さい場合は，数値データをクライアント側に転送し，クライアント側でそのまま可視化することができる．逆に，データセットのサイズが大きい場合，そのまま転送するのはコストが大きくなるため，サイズの小さいジオメトリデータや画像データとしてクライアント側に送る手法が有効である．

また，一般に可視化パイプラインの各プロセスは，下流のプロセスほど頻繁に繰り返されることが多い．つまり，データ転送の発生する場所が上流になるほど，下流のプロセス間ではデータ転送が発生しないため，可視化処理のインタラクティブ性は向上する．一方，インタラクティブ性

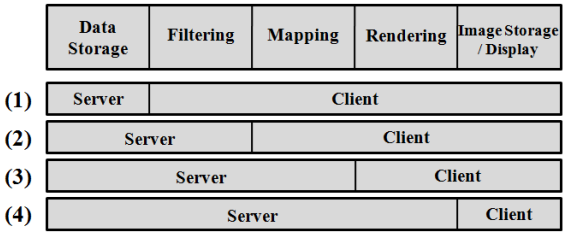

Fig. 19. Client server model of remote visualization pipeline.

図19. 遠隔可視化パイプラインにおけるクライアントサーバモデル．

を要求しない場合は，データ転送をなるべく下流で行う方が転送されるデータサイズが一般には小さくなるため有利である．データ転送を含めた効率的な可視化処理を行うためには，データセットのサイズや目的とする処理，サーバ側とクライアント側の環境に合わせて適切な手法を選択することが重要である．

5.1.2. データ転送の高速化

遠隔地間において，特にインタラクティブな可視化で要求されるような遅延の少ない処理を行うためには，データ転送の高速化が必要不可欠である．データ転送を高速化する手法としては，データ圧縮により転送データサイズを小さくする手法と，データ転送のスループットを高める手法が考えられる．データ圧縮としては，まず，差分符号化（difference encoding）が考えられる．差分符号化は，連続する時間の配列に対する差分情報のみに変換する，最も単純な可逆変換のデータ圧縮手法である．Shen and Johnson (1994)は，シミュレーションによって出力された時系列データに対して差分符号化によるデータ圧縮を施し，非圧縮時と比べて90％以上のデータサイズの軽減とレンダリング時間の高速化を実現した．非可逆変換としては，量子化八分木と差分符号化を非構造格子データに適用し，複雑な時系列データを圧縮した研究例が報告されている（Ma and Shen, 2000）．また，高速なハードウェアレンダリングを目的として，グラフィックスカードを用いた時系列データの複合化に関する研究例も報告されている（Lum et al., 2001）．

さらにデータ転送のスループットを高める手法としては，大規模データを高速に転送することが可能なプロトコルの使用も考えられる．複数のパケットが同時に流れる可能性のあるネットワークにおいては，信頼性が高く実装の容易なTCP（Transmission Control Protocol）が主に用いられてきた．しかし，TCPは通信トラフィックの混雑を避けるための輻輳制御等が実装されており，大容量のデータの長距離伝送には向いていない．そこで，転送の確認応答を必要としないUDPを用いた遠隔可視化システム（Chen et al., 2001）が開発された．TCP使用時と比べて大幅なスループットの向上が見られたものの，通信のトラフィックが多い一般回線では，パケットが欠落し実用性にかける等の問題が残った．Tanaka et al. (2005)は，UDPに信頼性を付加させたプロトコルであるReliable Blast UDP（RBUDP）を遠隔可視化システムに応用した．帯域幅を考慮した送信レートでRBUDPを使用することにより，伝送遅延の削減が確認されている．また，通信トラフィックの発生しにくい専用の高速ネットワークを用いた可視化システムとしては，UDPをベースとしたUDTの利用例がある（Murata et al., 2007）．UDTでは，定期的に受信側が利用できる帯域幅を測り，それに基づいたウィンドウサイズを送信側に伝えるため，常に一定のスループットで伝送することが可能である．また，TCPにおけるウィンドウサイズの動的な調整や，コネクションの並列化によって高速な転送を図るGridFTP等の例も報告されている（Allcock et al., 2002）．その他の手法としては，データ転送に適した形式に変換することで転送スループットを効率化する手法や，データ転送を含むデータI/O処理をレンダリング中に隠ぺいする手法が考えられる．これらの詳細については5.2で述べる．

5.2. 分散可視化

大規模なデータセットの可視化処理に適した環境は，他の処理の実行終了を待たなければならないことも多い．そこで，グリッド（Grid）等のネットワーク上に分散されたプロセッサやストレージ等の計算資源を用いた分散可視化（Distributed visualization）が注目されている．分散可視化によって，目的に応じた規模の可視化処理を，目的に応じた規模の環境で即時的に実行することが可能となる．以下では，分散可視化のシステム構成や実装例について紹介し，それらの特徴や問題点についても述べる．

5.2.1. 分散可視化のシステム構成

分散可視化はストレージやプロセッサ，解析者のいずれかがネットワーク上に分散された状態における可視化処理を指す．そのシステム構成は，図20のような3つに分類できる（Rodrigues et al., 2010）. (a)はネットワーク上に分散して保存されているデータを，一つの計算環境に集めて可視化する場合である．分散計算環境を用いたシミュレーションによって生成されるデータを可視化するのに適している．(b)はプロセッサが分散している場合であり，分散した計算環境にそれぞれデータを分配して可視化を行う．ビジー状態にある分散計算環境を用いることで，即時的な可

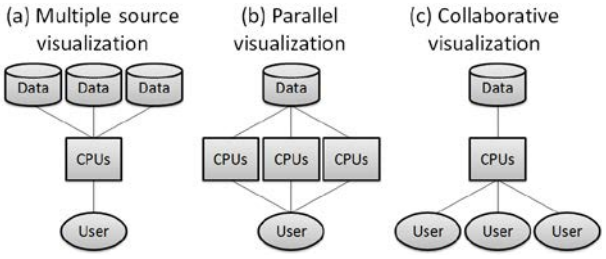

Fig. 20. Quantitative based classification of distributed visualization (Rodrigues et al., 2010).

図20. 分散可視化の分類（Rodrigues et al., 2010）.

視化処理が可能になる．(c)は複数の解析者が分散して存在する場合であり，協調的可視化とも呼ばれる．解析者間で可視化手法や可視化結果を共有することが可能である．その他の例として，分散ストレージに接続されたプロセッサを用いた分散処理（(a)と(b)の組み合わせ）や，複数の解析者の計算環境間における協調分散可視化（(a)と(c)または(b)と(c)の組み合わせ）も考えられる．協調的可視化および協調分散可視化については，5.3.で詳しく述べる．

5.2.2. 分散環境におけるデータ転送の効率化

分散可視化では，分散ストレージまたは分散プロセッサ間におけるデータ分配や画像重畳のためのデータ転送が発生する．地理的に分散した環境でのデータ転送は，同一システム内でのデータ転送と比べてコストが高いため高速化が必要である．分散ストレージ上にあるデータを可視化するためのアプリケーションとして，TCPを用いたデータ転送機能を組み込んだVisapultが開発された（Bethel, 2000; Bethel et al., 2000）．Visapultでは，図21に示すように分散ストレージにそれぞれ接続された分散並列ストレージシステム（DPSS: Distributed Parallel Storage System）サーバを用いることで，高速なデータ転送が実現された．しかし，インターネット上で行った実験では，サーバとクライアント間のデータ転送コストは小さいものの，サーバとストレージ間でのデータ転送がボトルネックとなった．Bethel et al. (2002)では，データ転送性能の制御機能を追加したUDPを用いることでVisapultを改良した．信頼性のあるTCPで高速なUDPを挟み込み，低帯域のネットワーク上で高速な分散可視化を実現した例も報告されている（Thibault et al., 2002）．専用の高速ネットワークで接続されたグリッド環境を用いた例として，UDPをベースとした高速性と信頼性を有するプロトコルであるRBUDP（He et al., 2002）を用いたSuper SINET上での分散可視化システムが報告されている（Tanaka et al., 2005）．

5.2.3. 異機種間における分散可視化

ネットワーク上に存在する分散計算環境は，同一機種のシステムではなく，異機種（ヘテロ）で構成されている場合が多い．処理能力やデータ転送性能が異なるヘテロな分散環境では，データや処理をどの計算機に割り当てるかを考えることが必要不可欠である．そこで，可視化に係わらず，性能の異なる計算機に適切にデータや処理を割り当てるための，手法が数多く提案されてきた（例えばCasanova et al., 1999, Berman et al., 2003, Desprez and Vernois, 2005 等）．また，分散計算環境は複数のユーザによって同時に利用されることも多く，現在の利用状況を確認しながら動的に処理を割り当てる共スケジューリング（Co-scheduling）手法も必要である．Liu et al. (2006)は，Casanova et al. (1999)が提案したスケジューリング手法を分散可視化システムに実装し，可視化処理とデータ転送の動的な負荷分散を実現した．Liu et al. (2006)のシステムでは，図22に示すように複数のユーザ(ワークステーション)から分散ノード(サーバ)に対して可視化処理が実行される．このとき，データの複製を異なるサーバで共有して保持する（レプリケーション）．各サーバの処理能力やサーバから各ワークステーション間のデータ転送性能を確認する．それによって，各ユーザにとって最適となる分散処理が可能となるサーバ，データの割り当てを行う．

異機種間での効果的なデータ管理を目的とした例としては，ETSP木と論理ネットワークを用いた分散可視化システムも提案されている（Gao et al., 2005）．また，分散計算環境における各計算資源，データストレージ，ネットワークの性能を数値モデル化し，可視化パイプラインの最適化

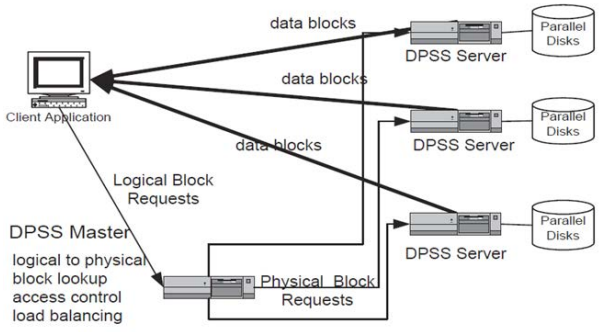

Fig. 21. Distributed Parallel Storage System architecture (Bethel et al., 2000).

図21. 並列分散ストレージシステムの構成（Bethel et al., 2000）.

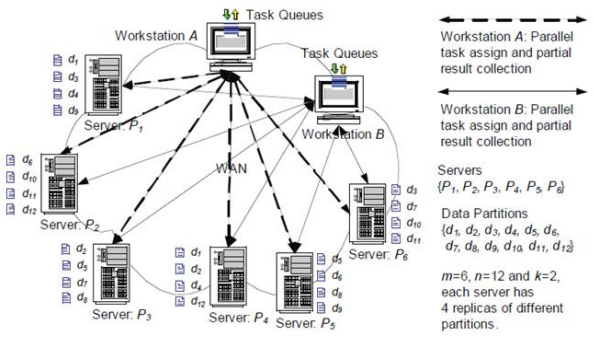

Fig. 22. A typical structure of task parallel supplications on replicated datasets (Liu et al., 2006).

図22. データセットの複製における並列タスクの要求の例（Liu et al., 2006）.

を行う研究も報告されている（Wu et al., 2008）．ヘテロな分散計算環境では，計算機の処理能力だけでなくOSやライブラリ等の違いから，一つのアプリケーションが全ての計算ノードで実行できない可能性もある．そのため，異なるOSやライブラリ，アーキテクチャの計算ノードでも動作する汎用性の高い可視化アプリケーションが求められている．Brodlie et al. (2004)は，汎用可視化アプリケーションであるIRIS Explorerと，統一記述言語であるXMLをもちいることで，異なるOSで構成される分散環境においても利用可能な可視化アプリケーションを開発した．このようなシステムでは，データの流れや使用する計算ノードの場所，複数の計算ノードへのログイン等が複雑になるが，ユーザはそれらを意識せずに処理を行うことができる．また，グリッドの問題点として常に指摘されているセキュアかつ透過性のある認証も実現されている（Huang et al., 2007）．

5.3. 協調的可視化

協調的可視化（Collaborative visualization）は，主に医療分野の遠隔医療において，ビデオチャットの動画像と音声に加えてボリュームデータを遠隔地間で共有しながら情報交換を行うことを目的として始まった．これは，ボリュームコミュニケーションと呼ばれる．初期の協調的可視化システムであるTeleInViVo^{TM}は，CTスキャンによって得られたデータに対して，遠隔地の複数のユーザ間でデータや可視化パラメータ，可視化結果の共有を実現している（Coleman et al., 1996）．また，ユーザを撮影した動画や音声も共有することができ，ビデオ会議を行いながらの遠隔医療が可能である．医療分野における協調的可視化については，Manssour and Freitas (1998)やManssour and Freitas (2000)に詳しい．

一方，シミュレーションデータにおいても，複数の解析者間で協調しての可視化処理が行われている．近年では，SciDACやINCITE等，スーパーコンピュータの高度利用とシミュレーションの高度化を目的とした大規模な研究コミュニティも存在する．また，Access Gridや，ITBL，VizGrid等のプロジェクトも，独自のネットワークで分散環境を構築し，シミュレーション研究を進めている．（これらのコミュニティおよびプロジェクトについては，それぞれURLを参考文献の各項目に記載）これらの大規模コミュニティでは，シミュレーションコードの開発や解析処理は，複数人で構成されるチームで行われていることが多い．また，地理的に分散した研究グループから構成されていることが多いため，複数の研究者間で協調してデータの可視化や解析を行うことが特に重要である．

一般的な協調的可視化のモデルを図23に示す（Wood et al., 1997）．協調的可視化は，可視化パイプラインのデータフローモデルに基づいており，図23に示すようにユーザ間において，各プロセスのパラメータ制御とデータの共有が行われる．必ずしも全ユーザが自身の環境で全ての可視化処理を行う必要はなく，他ユーザが可視化した結果を受け取るだけでも良い．例えば数値データをユーザAのみが保持している場合，ユーザAの環境でフィルタリング，マッピングを行った後にジオメトリデータをユーザBに転送し，それぞれの環境でレンダリングを行う手法も考えられる．このとき，レンダリングは各ユーザの環境で行うため，視点位置や視点方向等のパラメータを共有することで，全ユーザが同じ可視化結果を共有することができる．また，複数のユーザ間でパラメータやデータを共有するために，パラメータ変更やデータ変更のための制御権が用いられることが多い．制御権を持ったユーザによるパラメータ変更のみ，他ユーザの可視化処理や可視化結果に反映される．協調的可視化は，「可視化環境，データ，パラメータそれぞれについて何を共有し，何を転送し合うか」によって転送されるデータサイズが異なる．可視化パイプラインにおけるどの処理が頻繁に発生するか等を考慮して最適な構成にすることが必要である．全ユーザがそれぞれ可視化環境を有している場合，それぞれの環境で処理を並列実行することで効率的な可視化が可能になる．これは，分散協調可視化（Distributed and collaborative visualization）と呼ばれる（例えばHutanu et al., 2006; Okuda et al., 2007）．

協調的可視化の実装例をいくつか紹介する．Bajaj and Cutchin (1999)は，Web上でシミュレーションの実行から可視化，解析処理までの一連のプロセスを協調して行うためのアプリケーションを実装した．参加する各ユーザは，シミュレーションの実行，データの選択，可視化処理のパラメータ決定，解析結果に基づくシミュレーションのパラメータ決定等の処理を，インタラクティブに制御することが可能である．VizGridの可視化システムでは，制御権をもったユーザが必要なパラメータを遠隔地にある可視化用クラスタに送信し，可視化した結果である画像を受信す

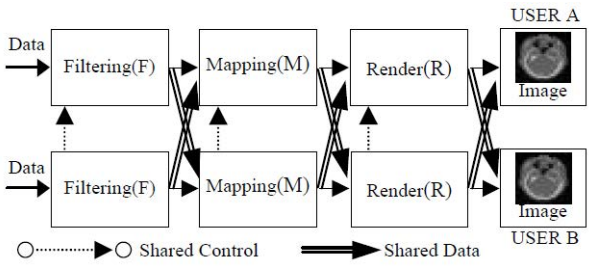

Fig. 23. A general model of collaborative visualization (Wood et al., 1997).
図23. 協調的可視化の一般的なモデル（Wood et al., 1997）．

る（Karube and Matsukura, 2004）．受信した結果は，制御権を持ったユーザが各ユーザにそれぞれ転送するというものである．このシステムでは，各ユーザの元で可視化処理を行う必要がないため，高性能な可視化用計算機が一台あるだけでよいというメリットがある．さらに，バーチャルリアリティとテレビ会議を融合したテレイマージョン（臨場感通信）もVizGridでは実現されている（Matsukura et al., 2005）．ITBLでは，リアルタイム可視化アプリケーションであるPATRAS（Mulder et al., 1995）を用いた協調的可視化を行っている．Webブラウザ上でシミュレーション結果をリアルタイムに可視化しながら，可視化結果の共有やステアリングを行うことができる．Iwamoto et al. (2006)のシステムでは，可視化に用いるデータおよび計算機は各メンバーがそれぞれ保持している場合が想定されている．視点情報や時刻などのパラメータのみを通信することで，インタラクティブ性の高いボリュームコミュニケーションも実現されている．各メンバーは可視化に必要なパラメータ（データの種類や伝達関数，視点位置・視線方向等の各種パラメータ）のみを転送して共有しあうため，インタラクティブ性に優れている．その他にも，没入型バーチャルリアリティシステムを用いた協調的可視化環境が数多く構築されている（Leigh et al., 1999; Matsuzawa, 2003; Manssour and Freitas, 2000）．

6. 高性能な計算機環境の利用

3章および4章では，大規模データを高速に可視化するためのデータ構造やアルゴリズムに焦点を絞って紹介した．それらの手法を実施する環境として，5章ではネットワーク環境を用いた例について述べたが，本章では高性能な計算機環境を用いた例について紹介する．

6.1. スーパーコンピュータを用いた可視化処理の大規模化

6.1.1. その場可視化

前章で述べたネットワーク環境を用いた可視化処理は，データ転送コストの削減と大規模な計算環境の利用が大きな目的であった．5.1.1.で示した遠隔可視化のさまざまな手法のうち最もデータ転送コストが削減できる手法は，シミュレーションを実行したスーパーコンピュータで全ての可視化処理を行うその場可視化（in-situ visualization）である（図19の(4)）．生成された画像データだけを解析者のもとに転送することにより，大幅なデータ転送コスト削減が考えられる．また，シミュレーション実行と同一の計算環境であるため，一般に可視化処理に対して十分な計算資源を有している場合が多い．

スーパーコンピュータの可視化処理への利用は，1990年代後半から行われてきた．Ma and Crockett (1997)やMa and Crockett (1999)では，当時の最高レベルの分散メモリ型スーパーコンピュータを用いた大規模可視化例が報告されている．Ma and Crockett (1997)では，IBM SP2の128プロセッサを用いて10万要素の非構造格子データの並列ボリュームレンダリングを行った．レイキャスティングと比べて非構造格子データに対する負荷バランスの最適化が容易であるセル投影（Cell-Projection）法を用い，128プロセッサ使用時において約70％の並列化効率と約5万要素毎秒（約2 fps）という結果が得られている．さらに，Ma and Crockett (1999)ではCray T3Eの511プロセッサを用いて，従来よりも大きなサイズの非構造格子データの可視化にも挑戦している．プロセッサ数の増加に伴う効率の低下が通信コストよりも負荷バランスの低下にあることを突き止め，静的負荷分散のためのラウンドロビン法と微粒子インタリーブ分割法，k-d木による効率的なデータ割り当てを適用して問題の解決を試みた．その結果，511プロセッサ使用時において1800万要素のデータを約74％の並列化効率，約900万要素毎秒（約2.0 fps）で可視化したという結果が得られている． プロセッサ数の増加に伴うオーバヘッドがどの処理において発生するかを突き止め，システムに合わせてアルゴリズムやパラメータを最適化することで，スーパーコンピュータを用いた可視化処理は高い効率が得られることを示した．

Yu et al. (2007)は，同じくCray T3Eを用いて，1億要素の非構造格子データに対する並列ボリュームレンダリングを行った．3.で説明した集合的I/Oとデータシービング，並列I/Oの最適化のための2DIPを実装し，64プロセッサ使用時に約1億要素毎秒（1 fps）と高い効率を得た．近年では，ParaView等のオープンソースの可視化ソフトウェアにおいても，スーパーコンピュータ向けの最適化が施されている（Moreland et al., 2007）．

現在報告されている最も大きなデータサイズの可視化例は，Peterka et al. (2008a), Peterka et al. (2009a)によるIBM BlueGene/Pを用いた並列ボリュームレンダリングである．特にPeterka et al. (2009a)では，約90億グリッドのデータを，BlueGene/Pの32,000コアまでを用いて実験を行っている．並列I/Oのための集合的I/Oとデータシービング，データ転送のためのストリーミング技術に加え，時系列データ可視化のための並列パイプライン処理の最適化とI/Oの隠ぺい，改良版ダイレクトセンドによる高効率な画像重畳を実装している．図24は，可視化パイプラインの各プロセス別の処理時間を表している．図から分かるように，I/Oとレンダリングはプロセッサ数の増加に比例して減少するが，プロセッサ間でのデータ転送が必要となる画像重畳は

処理時間が指数関数的に増加する．そのため，画像重畳のアルゴリズムを改良し，また，両者のトレードオフを考慮して最適なプロセッサ数を決定した結果，16,000プロセッサ使用時に最も高速に処理できることが分かる．さらに，図25からも分かるように，並列化数を増加させると，I/O処理にかかる時間の比率は増加し，32,000プロセッサ使用時にはI/O処理の比率は全体の90％を超える．プロセッサ数の増加に伴う並列性能の低下をなるべく抑えるために，全体の処理時間の大部分を占めるI/O処理をより高速に行うことが必要不可欠である．ペタスケールの大規模データ可視化におけるI/O 処理の高速化手法に関する研究も進められている（Ross et al., 2008）.

6.1.2. シミュレーション時可視化

可視化処理をシミュレーション実行後のポスト処理として行う場合，可視化プロセスが実行されるまでに待ち時間が発生する．このような場合，シミュレーションを実行するスーパーコンピュータを用いて数値計算と同時に可視化処理を行うことで効率化を図ることができる．これは，シミュレーション時可視化（Simulation-time visualization）と呼ばれる．データをストレージから可視化環境に転送する必要がないため，I/Oコスト，転送コストを除去できるというその場可視化の特徴に加え，データをストレージ出力することなくメモリから直接可視化できるというメリットがある．また，シミュレーション実行時における結果のモニタリングやパラメータ修正等のステアリングも可能となる．

これらの機能は古くから実現されており，Vista（Tuchman et al., 1991）やVASE（Jablonowski et al., 1993），SCIRun（Parker and Johnson, 1995; Johnson et al., 2000），CUMULVS（Geist et al., 1997），RVSLIB（van der Ven et al., 1998），PATRAS（Mulder et al., 1995）等のアプリケーションやライブラリが開発されている．その他にも，大規模シミュレーションにおけるシミュレーション時可視化の研究例として，I/O，通信，計算のバランスがとれたパイプライン処理のスケジューリング手法（Fujishiro and Chen, 2001）や，それらを始めとするさまざまな効率化手法を用いたTBオーダーの大規模データ可視化への取り組み（Ma et al., 2002; Tu et al., 2006; Yu et al., 2006）等が報告されている．

6.2. マルチGPU システムを用いた大規模かつ高速な可視化

6.1.では，より大規模なサイズのデータを可視化するために，スーパーコンピュータのようにプロセッサを大量に接続した計算機環境を用いた実験例について紹介した．3.2.4.では，より高速に可視化するために，GPUを汎用目的で用いたGPGPUによる可視化処理について紹介した．後者は前者より密結合な並列性を持つ．本節では，より大規模なサイズのデータをより高速に可視化するために，密結合の並列システムと疎結合の並列システムを融合した環境であるマルチGPUやGPUクラスタを用いた実験例を紹介する．

マルチGPUによるシステムは，その構成によっていく

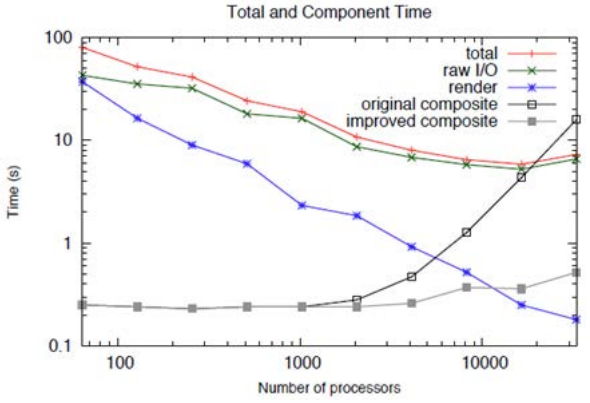

Fig. 24. Total frame time as well as individual components of visualization pipeline (Peterka et al., 2009a).

図24. I/O, レンダリング, 画像重畳の処理時間（Peterka et al., 2009a）.

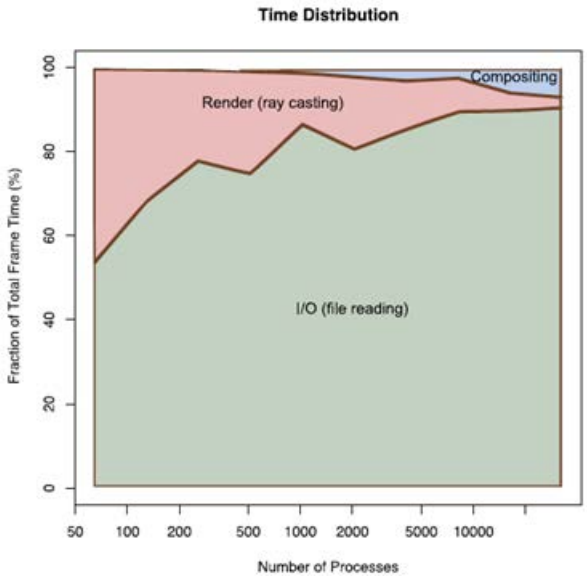

Fig. 25. The relative percentage of time spent in I/O, rendering, and compositing as a function of system size (Peterka et al., 2009a).

図25. I/O, レンダリング, 画像重畳の処理時間の割合（Peterka et al., 2009a）.

つかの種類に分類できる．NVIDIA社のTeslaシリーズのように複数のGPUを搭載したビデオカードを複数枚搭載したコンピュータもその一つである．また，通常のPCクラスタの各ノードにシングルGPUのビデオカードが1枚搭載された構成や，各ノードにマルチGPUを複数枚搭載した構成がある．Marchesin et al. (2008)は，4台のPCにそれぞれシングルGPUを搭載したGPUクラスタと，シングルCPU と4コアのマルチGPUからなるGPU クラスタ（以下，マルチGPU）を用いて並列ボリュームレンダリング時のベンチマークテストを行った．本実験では，VRAMからメインメモリへのリードバックの高速化手法や，並列化の際のbrickサイズの最適化手法も提案されている．PCクラスタでは，GPUでレンダリングを行った結果をいったんメインメモリにリードバックし，分散メモリ間での画像重畳を行う（図26（左））．マルチGPUでは，GPUでレンダリングを行った結果をVRAMにリードバックした後にダブルバッファを用いてVRAM上で画像重畳し，最後にCPUを用いてメインメモリ上（CPU画像重畳）もしくはVRAM上（GPU画像重畳）で画像重畳を行う（図26（右））．

図27は，PCクラスタおよびマルチGPUにおいて，それぞれCPU画像重畳とGPU画像重畳を行った実験の結果である．それぞれの実験において，1GBおよび128MB のデータを用いて，異なる解像度の画像のレンダリングに要した時間を示している．GPU画像重畳では画像解像度が高いとき，CPU画像重畳では画像解像度が低いときに高いフレームレイトが得られていることが分かる．これは，画像解像度が高い時は，VRAMからメインメモリへのリードバックがボトルネックとなるため，GPU画像重畳の方が有利となるためである．また，画像解像度にかかわらず，データサイズが小さい場合（128MB）はマルチGPUを用いてGPU画像重畳をするのが速く，データサイズが大きい場合はGPUクラスタでCPU画像重畳をするのが速いという結果が出ている．

次に，図28は，マルチGPUとGPUクラスタそれぞれにおけるGPU コア数とフレームレイトの関係を示している．1024 × 768 の解像度の画像を，CPU画像重畳を用いてレンダリングした場合である．マルチGPUは，コア数を増やしても性能が低下しにくく，GPUクラスタは画像の解像度を高くしても性能が低下しにいということが分かる．GPUのコア数を増やす場合，ノード間で分散するのではなくノード内のGPU数を増やす方がコア数増加にともなう並列性能の低下を抑えられる．

6.3. バーチャルリアリティ技術を応用した可視化

コンピュータが作りだしたデータを人間に提示する一手法に，バーチャルリアリティ（VR: Virtual Reality）技術がある．VRは日本語では仮想現実と訳され，VR技術を用いることで人間はバーチャル空間に仮想的に入り込むことができる．Sutherland (1968)によって開発されたヘッドマウ

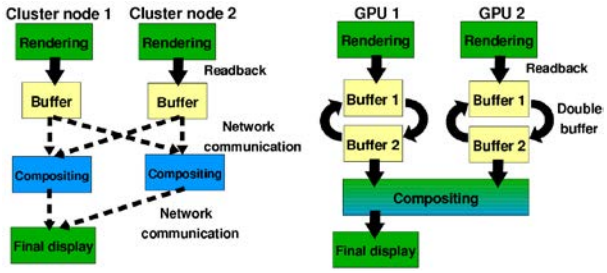

Fig. 26. The sort-last rendering pipeline on a GPU-cluster (left) and on a multi-GPU (right).(Marchesin et al., 2008).

図26. GPU クラスタとマルチGPU におけるソートラスト型レンダリングパイプライン（Marchesin et al., 2008）．

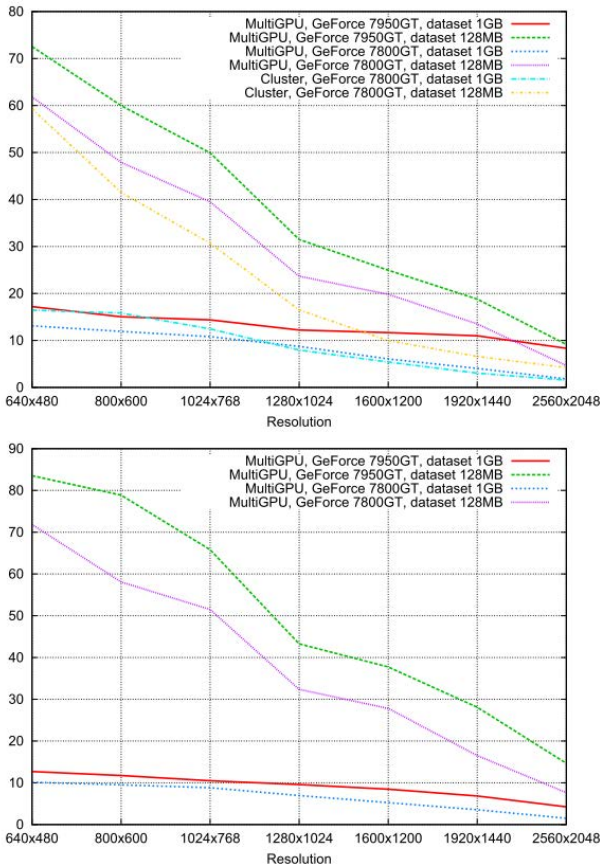

Fig. 27. Scalability with the screen resolution using CPU composition (top) and GPU composition (bottom) (Marchesin et al., 2008).

図27. CPU 画像重畳（上）とGPU 画像重畳（下）の画像解像度の増加に伴うスケーラビリティ（Marchesin et al., 2008）．

ントディスプレイ（HMD: Head Mounted Display）が，最初のバーチャルリアリティシステムとされている．

近年では，3次元シミュレーションデータの可視化結果を表示する手法および装置としても，バーチャルリアリティの応用が進められている．初期の応用例としては，BOOMと呼ばれるNASAが開発したゴーグル型ディスプレイがある（Bryson and Levit, 1992）．BOOMでは，ゴーグル型のディスプレイを覗くことで，シミュレーション空間に入り込んで航空機の周りの定常流を見ることができる．3次元可視化結果を立体視し，人間がVR 空間に入り込んで解析を行うことで，立体構造の認識や3次元的な現象の解釈がより容易になる．

その後，より没入感の高いVRシステムとしてCAVE（Cruz-Neira et al., 1993）等に代表される没入型VR システムが開発された．CAVEは，図29に示すように複数のスクリーンから構成され，各面に立体映像が投影される．通常のディスプレイと異なり視野のほとんどがVR空間に覆われているため，ユーザは高い没入感を得ることができる．また，ヘッドトラッキングシステムや専用コントローラを用いることで，没入感だけでなく高い対話性を得ることが可能である．廣瀬らは，前，上下，左右の5面構成の没入型VRシステムCABINを開発し，シミュレーションデータの可視化に応用した（廣瀬ほか，1997）．その他にも，陰山らは核融合科学研究所に設置されたCAVEシステムCompleXcopeを用い，核融合プラズマシミュレーションのデータを可視化した（Kageyama et al., 1999）．また，CAVE型バーチャルリアリティシステム用の汎用可視化ソフトウェアVFIVE（Kageyama et al., 2000）も開発された．特にVFIVEは，複雑なベクトル場の可視化を目的とした様々な可視化機能が実装されている．さらに，海洋研究開発機構のCAVEシステムBRAVE（Araki et al., 2006）を用いて大規模データをインタラクティブに可視化するための高速ボリュームレンダリング（Ohno and Kageyama, 2007）およびGPGPUを利用した可視化手法（川原，2010）の開発が進められている．

このように3次元可視化結果の解釈を容易にするCAVEであるが，これには解析者からのリクエストに対する即時的なフィードバックが要求される．一般にはデータサイズの大規模化と処理の高速化は相反するため，VRシステム上で大規模データを可視化するのは不向きであるとされていた．この問題の対処法としては，データに対して関心領域（ROI: Region of Interest）の設定や詳細度制御（LOD: Level of Detail）機能を用いることで高フレームレイトを得ようとする手法が提案されている（Ohno and Kageyama, 2010）．

一方，大規模データを間引きすることなく可視化するための取り組みも行われている．Peterka et al. (2008b)やPeterka et al. (2009b)は，前述のBlueGene/Pでの大規模可視化実験（Peterka et al., 2008a; Peterka et al., 2009a）で用いた高度なアプリケーションを改良し，大規模データの高フレームレイトでの立体視に挑戦している．Peterka et al. (2009b)では，左右両目用の可視化パイプラインに4096プロセッサを用い，6億4000万格子のデータを2 fps で可視化することに成功している．

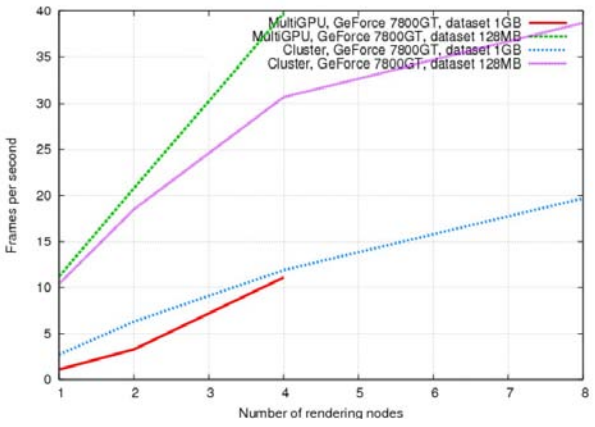

Fig. 28. Multi-GPU vs GPU-Cluster scalability with the number of GPUs (Marchesin et al., 2008).

図28. マルチGPU とGPU クラスタのGPU 数の増加に伴うスケーラビリティ（Marchesin et al., 2008）．

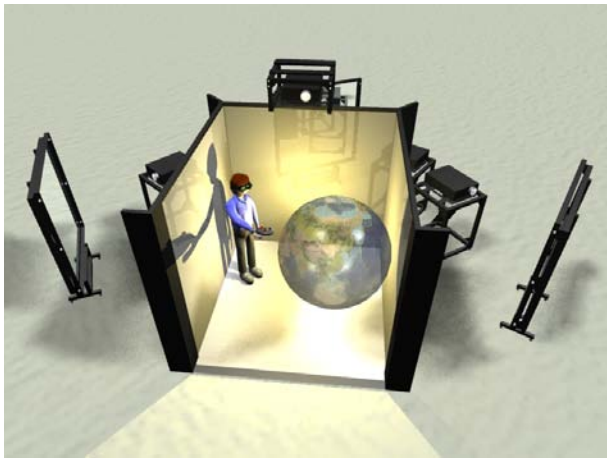

Fig. 29. CAVE-type virtual reality system.

図29. CAVE 型バーチャルリアリティシステム．

近年ではスクリーン投影法ではなく高画質な液晶ディスプレイを複数接続した方式の裸眼立体視が盛んに研究されている（図30）．大規模データから生成された高解像度の画像を表示するために，Sandin et al. (2005) のDynallaxでは35台，Peterka et al. (2007) のDynallaxでは65台もの液晶ディスプレイを用いて並列処理による立体視を行っている．詳細は，Advances in Computer Displays（Leigh et al.，2009）やThe Future of CAVE（DeFanti et al., 2010）を参照されたし．

7. おわりに

本論文では，スーパーコンピューティング技術の進歩にともなう大規模可視化研究の創成期から現状までの調査結果を体系的にまとめた．特に，可視化データフローモデルにおけるプロセスの効率化，可視化パイプライン全体の効率化，ネットワーク環境における可視化処理の効率化，最先端のハードウェアを用いた可視化処理の効率化研究について紹介した．ペタスケール時代の大規模シミュレーションのためのデータ可視化においては，より大規模なデータをより高速に可視化することが求められる．そのためには，シミュレーションを行うスーパーコンピュータ，ストレージからデータ転送を行う情報通信ネットワーク，可視化を行う計算機までの全てを一つのシステムと捉え，目的や環境，データに応じてボトルネックの少ない最適な可視化手法を検討することが必要である．また，可視化処理時のデータ読み込みから転送，レンダリングまでのパイプライン全体におけるボトルネックを最小化するため，シミュレーション実行時においてもデータ出力のタイミングやフォーマット，出力場所を検討することが必要不可欠である．

本論文は，主に計算機が行う処理の側面から大規模データ可視化の研究事例の調査を行ったものであるが，実際に可視化および解析作業を行うユーザの側面から見た効率性も重要視されている（小野，2006; 小野，2008）．例えば，興味のある現象や構造を可視化するためには，最適な可視化パラメータの決定に至るまでの人手を伴うトライアンドエラーの繰り返しが必要であり，これらの作業も大きなボトルネックの一つとなり得る．それゆえ，このような作業を自動化または半自動化できれば，効率よく可視化作業を遂行する上で大いに役立つであろう．特にペタスケール時代の可視化では人力による可視化作業の破たんが予想されるため，可視化パラメータの決定から結果の解釈まで，人間の作業をも含めた可視化プロセスの効率化手法の確立が急務の要件になると考えられる．本論文では触れなかったが，ビジュアルデータマイニングまたは知的可視化と呼ばれる，可視化プロセスにおける必要な情報の選別や取得を機械的に行うことで，大量の数値データから必要な情報を得たり，または新しい知見を発見したりしようとする手法（Soukup and Davidson, 2002; 白山, 2006）が，この問題を解決する重要な鍵となるかもしれない．

計算機技術が飛躍的に進歩している現在，数値シミュレーション研究は今後ますます高度化していくと思われる．先進的なシミュレーションから得られる大規模データから本質的な現象を効率的に理解し新たな発見を得るためには，常にその時代に合わせた先進的な可視化処理手法の研究が必要不可欠となる．計算機技術，シミュレーション技術とともに大規模データ可視化研究を推し進めることによってこそ，数値シミュレーション研究はますます発展していくであろう．

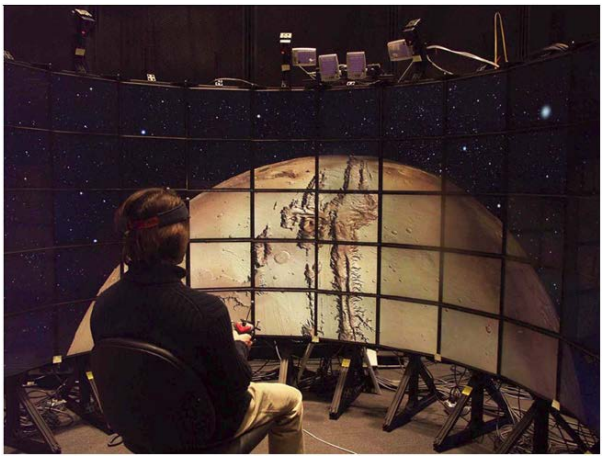

Fig. 30. 60-panel autostereo display (DeFanti et al., 2010).

図30. 60パネル裸眼立体ディスプレイ（DeFanti et al., 2010）．

謝辞

地球シミュレータセンターシミュレーション高度化研究開発プログラムの川原慎太郎氏には，本論文の3.2.4. および6.2. に関する調査において助言をいただいた．ここに感謝の意を表する．

参考文献

Access Grid <http://www.accessgrid.org/>.

Ahrens, J. P. and J. S. Painter, Efficient Sort-Last Rendering Using Compression-Based Image Compositing, Proceedings of EurographicsWorkshop on Parallel Graphics and Visualization 1998, 145-151.

Ahrens, J., K. M. Martin, B. Geveci, and C. Law (2001), Large-scale data visualization using parallel data streaming, *IEEE Computer Graphics and Applications, 21 (4)*, 34-41.

Ahrens, J., B. Geveci, and C. Law (2005), ParaView: An EndUser Tool for Large-Data Visualization, Hansen, C. D., and C. R. Johnson (eds.), The Visualization Handbook, 690-717.

Allcock, B., J. Bester, J. Bresnahan, A. L. Chervenak, I. Foster, C. Kesselman, S. Meder, V. Nefedova, D. Quesnel, and S. Tuecke (2002), Data management and transfer in high-performance computational grid environments, *Parallel Computing, 28 (5)*, 749-771.

Araki, F., H. Uehara, N. Ohno, S. Kawahara, M. Furuichi, and A. Kageyama (2006), Visualizations of Largescale Data Generated by the Earth Simulator, *Journal of the Earth Simulator, 6*, 25-34.

Arunachalam, M. and A. Choudhary (1995), A prefetching prototype for the parallel file systems on the Paragon, Proceedings of the 1995 ACM SIGMETRICS joint international conference on Measurement and modeling of computer systems, 321-322.

Bajaj, C. and S. Cutchin (1999), Web based Collaborative Visualization of Distributed and Parallel Simulation, Proceedings of the 1999 IEEE Symposium on Parallel Visualization and Graphics, 47-54.

Bakalash, R., A. Kaufman, R. Pacheco, and H. Pfister (1992), An extended volume visualization system for arbitrary parallel projection, Proceedings of Eurographics Workshop on Graphics Hardware '92.

Berman, F. D., R. Wolski, H. Casanova, W. Cirne, H. Dail, M. Faerman, S. Figueira, J. Hayes, G. Obertelli, J. Schopf, G. Shao, S. Smallen, N. Spring, A. Su, and D. Zagorodnov (2003), Adaptive computing on the grid using apples, *IEEE Transactions on Parallel and Distributed Systems, 14(4)*, 369-382.

Bethel, W. (2000), Visapult: A prototype remote and distributed visualization application and framework, Proceedings of the Conference Abstracts and Applications, Sketches and Applications, ACM SIGGRAPH 2000.

Bethel,W., B. Tierney, J. Lee, D. Gunter, and S. Lau (2000), Using high-speed WANs and network data caches to enable remote and distributed visualization, Proceedings of the IEEE/ACM 2000 Conference on Supercomputing (CD-ROM).

Bethel, W., R. Frank, and J. D. Brederson (2002), Combining a multithreaded scene graph system with a tiled display environment, Proceedings of the 2002 IS&T/SPIE Conference on Electronic Imaging and Technology.

Bitter, I., N. Neophytou, K. Mueller, and A. E. Kaufman (2004), Squeeze: Numerical-precision-optimized volume rendering, 2004 Eurographics/SIGGRAPHWorkshop on Graphics Hardware, 25-34.

Brodlie, K., D. Duce, J. Gallop, M. Sagar, J. Walton, and J. Wood (2004), Visualization in Grid Computing Environments, Proceedings of the conference on Visualization '04.

Bryson, S. and C. Levit (1992), The virtual wind tunnel, *IEEE Computer Graphics & Applications, 12 (4)*, 25- 34.

Cabral, B. and C. Leedom (1993), Imaging Vector Field Using Line Integral Convolution, SIGGRAPH93, Conference Proceeding, 263-270.

Cabral, B., N. Cam, and J. Foran (1994), Accelerated volume rendering and tomographic reconstruction using texture mapping hardware, 1994Workshop on Volume Visualilzation, 91-98.

Cavin, X., C. Mion, and A. Filbois (2005), COTS Cluster-Based Sort-Last Rendering: Performance Evaluation and Pipelined Implementation, Proceedings of IEEE Visualization 2005 Conference, 111-118.

Casanova, H., M. Kim, J. S. Plank, and J. Dongarra (1999), Adaptive scheduling for task farming with grid middleware, *International Journal of High Performance Computing, 12 (3)*, 231-240.

Chen, P., R. Ge, Y. Ma, J. Mayo, Y. Zhang, and C. Zhao (2001), A TCP/UDP Protocol Visualization Tool: Visual TCP/UDP Animator (VTA), Technical report, 01-10, Computer Science Department, Michigan Technological University.

Chen, W., L. Ren, M. Zwicker, and H. Pfister (2004), Hardware-accelerated adaptive EWA volume splatting, IEEE Visualization '04, 67-74.

Clark, J. (1982a), A VLSI Geometry Processor for Graphics, *Computer 13 (7)*, 59-68.

Clark, J. (1982b), The Geometry Engine, A VLSI Geometry System for Graphics, *Computer Graphics, 16 (3)*, 127-133.

Coleman, J., A. Goettsch, A. Savchenko, H. Kollmann, K. Wang, E. Klement, and P. Bono (1996), TeleInViVo ™ : towards collaborative volume visualization environments, *Computers and Graphics, 20 (6)*, 801-811.

Crockett, T. W. (1997), An introduction to parallel rendering, *Parallel Computing, 23 (7)*, 819-843.

Cruz-Neira, C., D. J. Sandin, and T. DeFanti (1993), Surround-screen projection-based virtual reality: The design and implementation of the CAVE, Proceedings of SIGGRAPH

93, 135-142.

Cullip, T. J. and U. Neumann (1993), Accelerating volume reconstruction with 3D texture mapping hardware, Technical Report TR93-027, University of North Carolina at Chapel Hill.

DeFanti, T. A., D. Acevedo, R. A. Ainsworth, M. D. Brown, S. Cutchin, G. Dawe, K. Doerr, A. E. Johnson, C. Knox, R. Kooima, F. Kuester, J. Leigh, L. Long, P. Otto, V. Petrovic, K. Ponto, A. Prudhomme, R. Rao, L. Renambot, D. J. Sandin, J. P. Schulze, L. Smarr, M. Srinivasan, P.Weber, and G.Wickham (2010), The Future of the CAVE, (in press).

Desprez, F. and A. Vernois (2005), Simultaneous scheduling of replication and computation for data-intensive applications on the grid, Technical Report RR2005-01.

Eilemann, S. and R. Pajarola (2007), Direct Send Compositing for Parallel Sort-Last Rendering, Proceedings of Eurographics Symposium on Parallel Graphics and Visualization 2007, 29-36.

Engel, K. (2004), High-quality volume rendering, SIGGRAPH 2004 Course #28 Notes: Real-Time Volume Graphics, 114-144.

ENSIGHT <http://www.ensight.com/>.

Favre, J. M. and M. Valle (2005), AVS and AVS/Express, Hansen, C. D., and C. R. Johnson (eds.), The Visualization Handbook, 493-510.

FIELDVIEW <http://www.vinas.com/jp/ seihin/fieldview/>.

Fujishiro, I. and L. Chen (2001), Parallel visualization of gigabyte datasets in GeoFEM, *Journal of Concurrency and Computation: Practice and Experience, 14 (6-7)*, 521-530.

Gao, J., J. Huang, H.-W. Shen, and J. Kohl (2003), Visibility Culling Using Plenoptic Opacity Function for Large Scale Volume Visualization, Proceedings of IEEE Visualization 2003, 341-348.

Gao, J., H. -W. Shen, J. Huang, J. A. Kohl (2004), Visibility Culling for Time-Varying Volume Rendering Using Temporal Occlusion Coherence, Proceedings of the conference on Visualization '04, 147-154.

Gao, J., J. Huang, C. R. Johnson, S. Atchely, and J. Kohl (2005), Distributed Data management for Large Volume Visualization, Proceedings of IEEE Visualization 2005, 183-189.

Geist, G. A., J. A. Kohl, and P. M. Papadopoulos (1997), CUMULVS: Providing Fault-Tolerance, Visualization and Steering of Parallel Applications, *International Journal of High Performance Computing Applications, 11(3)*, 224-236.

GrADS <http://grads.iges.org/grads/>.

Gnuplot <http://www.gnuplot.info/>.

Guan, S. and R. Lipes, Innovative volume rendering using 3D texture mapping, Image Capture, Formatting and Display, *Medical Imaging, 2164*, 382-392.

Guthe, S. andW. Straßer (2002), Real-Time Decompression and Visualization of Animated Volume Data, Proceedings of IEEE Visualization '02, 349-356.

Haber, R. B. and D. A. McNabb (1990), Visualization idioms: A conceptual model for scientific visualization systems, Visualization in Scientific Computing, IEEE Computer Society Press, 74-93.

Hansen, C. D., and C. R. Johnson (2005), The Visualization Handbook, Elsevier.

He, E., J. Leigh, O. Yu, and T. A. DeFanti (2002), Reliable blast udp: Predictable high performance bulk data transfer, CLUSTER ’02: Proceedings of IEEE International Conference on Cluster Computing, 317.

Heermann, P. D. and C. Pavlakos (2005), Desktop Delivery: Access to Large Datasets, Hansen, C. D., and C. R. Johnson (eds.), The Visualization Handbook, 493-510.

Henderson, A. (2004), The ParaView Guide, Kitware.

廣瀬通孝, 小木哲朗, 山田俊郎, 石綿昌平(1997), 没入型多面ディスプレイ(CABIN) を利用した数値シミュレーションの可視化, *日本バーチャルリアリティ学会大会論文集, 2*, 141-144.

Hsu, W. M. (1993), Segmented Ray Casting for Data Parallel Volume Rendering, Proceedings of IEEE Symposium on Parallel Rendering 1993, 7-14.

Huang, J., J. Gao, and T. Moore (2007), Dynamic Sharing of Large-Scale Visualization, *IEEE Computer Graphics and Applications, 27 (1)*, 1-25.

Hutanu, A., G. Allen, S. D. Beck, P. Holub, H. Kaiser, A. Kulshrestha, M. Liska, J. MacLaren, L. Matyska, R. Paruchuri, S. Prohaska, E. Seidel, B. Ullmer, and S. Venkataraman (2006), Distributed and collaborative visualization of large data sets using high-speed networks, *Future Generation Computer Systems, 22 (8)*, 1004-1010.

INCITE <http://www.er.doe.gov/ascr/incite/index.html>.

Interrante, V. and C. Grosch (1997), Strategies for Effectively Visualizing 3D Flow with Volume LIC, Proceedings of IEEE Visualization '97, 421-424.

ITBL Information Technology Based Laboratory <http://www.itbl.jp/>.

Iwamoto, K., K. T. Murata, E. Kimura, M. Umehara, and H.

Miyachi (2006), 3-D volume communication system and its application to e-Learning, 2006 AGU Fall Meeting, American Geophysical Union, (CD-ROM).

Jablonowski, D. J., B. Bliss, J. D. Bruner, and R. B. Haber (1993), VASE: The Visualization and Application Steering Environment, Proceedings of Supercomputing '93, 560-569.

Johnson, C. R., S. G. Parker, and D. Weinstein (2000), Large-Scale Computational Science Applications Using the SCIRun Problem Solving Environment, Proceedings of the Supercomputing 2000, (CD-ROM).

Kageyama, A., Y. Tamura, and T. Sato (1999), Scientific Visualizatoin in Physics Research by CompleXcope CAVE System, *Transactions of the Virtual Reality Society of Japan, 4 (4)*, 717-722.

Kageyama, A., Y. Tamura, and T. Sato (2000), Visualization of Vector Field by Virtual Reality, *Progress of Theoretical Physics Supplment, 138*, 665-673.

Kageyama, A., T. Miyagoshi, and T. Sato (2008), Formation of current coils in geodynamo simulations, *Nature, 454*, 1106-1109.

Karube, Y. and R. Matsukura (2004), Virtualized Collaboration Environment with 3D Images: VizGrid, *FUJITSU. 55 (2)*, 116-120.

Kaufman, A. and R. Bakalash (1998), Memory and processing architecture based on a 3-D voxel-based imagery, *IEEE Computer Graphics & Applications, 8 (6)*, 10-23.

Kaufman, A. and K. Mueller (2005), Overview of Volume Rendering, Hansen, C. D., and C. R. Johnson (eds.), The Visualization Handbook, 127-174.

川原慎太郎（2010）, GPGPU を用いたVR 装置用可視化ソフトウェアの高速化手法, 可視化情報学会全国講演会講演論文集, 217-218.

Knittel, G. and W. Straßer (1994), A compact volume rendering accelerator, Volume Visualization Symposium Proceedings, 67-74.

Kotz, D. (1997), Disk-directed I/O for MIMD multiprocessors, *ACM Transactions on Computer Systems (TOCS), 15 (1)*, 41-74.

Kruger, J. and R. Westermann (2003), Acceleration Techniques for GPU-based Volume Rendering, Proceedings of IEEE Visualization 2003, 287-292.

Law, C. C., K. M. Martin,W. J. Schroeder, and J. E. Temkin (1999), A multithreaded streaming pipeline architecture for large structured data sets, Proceeding of IEEE Visualization 1999, 225-232.

Lee, T. -Y., C. S. Raghavendra, and J. B. Nicholas (1996), Image Composition Schemes for Sort-Last Polygon Rendering on 2D Mesh Multicomputers, *IEEE Transactions on Visualization and Computer Graphics, 2 (3)*, 202-217.

Leigh, J., A. E. Johnson, and L. Renambot (2009), Advances in Computer Displays, *Advances in Computers, 77*, 58-79.

Leigh, J., A. Johnson, M. Brown, D. Sandin, and T. DeFanti (1999), Visualization in teleimmersive environments, *IEEE Computer, 32 (12)*, 66-73.

Levoy, M., Volume Rendering: Display of Surfaces from Volume Data, *IEEE Computer Graphics and Applications, 8 (5)*, 29-37.

Levoy, M. (1990), Efficient ray tracing of volume data, *ACM Transactions on Graphics, 9 (3)*, 245-261.

Linsen, L., V. Pascucci, M. Duchaineu, B. Hamann, and K. Joy (2002), Hierarchical representation of timevarying volume data with '4th-root-of-2' subdivision and quadrilinear B-spline wavelets, Proceedings of the 10th Pacific Conference on Computer Graphics and Applications - Pacific Graphics 2002, IEEE Computer Society Press, 346-355.

Liu, H., M. Beck, and J. Huang (2006), Dynamic Co- Scheduling of Distributed Computation and Replication, 6th IEEE International Symposium on Cluster Computing and the Grid (CCGRID '06), 592-600.

Ljung, P., C. Lundstrom, A. Ynnerman, and K. Museth, Transfer function based adaptive decompression for volume rendering of large medical data sets, 2004 IEEE Symposium on Volume Visualization and Graphics, 25-32.

Lombeyda, S., L. Moll, M. Shand, D. Breen, and A. Heirich (2001), Scalable Interactive Volume Rendering Using Off-the-Shelf Components, Proceedings of IEEE Symposium on Parallel and Large-Data Visualization and Graphics 2001, 115-121.

Lorensen, W. W. and H. E. Cline (1987), Marching cubes: A high resolution 3D surface construction algorithm, *Computer Graphics, 21 (4)*, 163-169.

Lum, E. B., K.-L. Ma, and J. Clyne (2001), Texture Hardware Assisted Rendering of Time-Varying Volume Data, Proceedings of the IEEE Visualization Conference 2001, 263-270.

Ma, K. -L., J. S. Painter, C. D. Hansen, and M. F. Krogh (1993), A Data Distributed, Parallel Algorithm for Ray-Traced Volume Rendering, Proceedings of Parallel Rendering Symposium 1993, 15-22.

Ma, K. -L., J. S. Painter, C. D. Hansen, and M. F. Krogh (1994), Parallel Volume Rendering using Binary Swap Image Composition, *IEEE Computer Graphics and Applications, 14(4)*, 59-68.

Ma, K. -L. and T. W. Crockett (1997), A scalable parallel cell-projection volume rendering algorithm for three-dimensional unstructured data, Proceedings of the IEEE symposium on Parallel rendering, 95-104.

Ma, K. -L. and T.W. Crockett (1999), Parallel visualization of large-scale aerodynamics calculations: a case study on the Cray T3E, Proceedings of the 1999 IEEE symposium on Parallel visualization and graphics, 15-20.

Ma, K. -L. and H.-W. Shen (2000), Compression and accelerated rendering of time-varying volume data, Proceedings of the 2000 International Computer Symposium - Workshop on Computer Graphics and Virtual Reality, 82-89.

Ma, K. -L., G. Schussman , B. Wilson , K. Ko , J. Qiang, and R. Ryne (2002), Advanced visualization technology for terascale particle accelerator simulations, Proceedings of the 2002 ACM/IEEE conference on Supercomputing, 1-11.

Ma, K. -L. and E. B. Lum (2005), Techniques for Visualizing Time-Varying Volume Data, Hansen, C. D., and C. R. Johnson (eds.), The Visualization Handbook, 511-531.

Manssour, I. H. and C. M. D. S. Freitas (1998), An Introduction to Collaborative Visualization, *Semana Academica do CPGCC, 3*, 145-152.

Manssour, I. H., and C. M. D. S. Freitas (2000), Collaborative visualization in medicine, Proceedings of WSCG 2000 - The 8th International Conference in Central Europe on Computer Graphics, Visualization and Interactive Digital Media 2000.

Marchesin, S., C. Mongenet, and J. -Mi. Discheler (2008), Multi-GPU Sort-Last Volume Visualization, Eurographics Symposium on Parallel Graphics and Visualization, 1-8.

MATLAB <http://www.mathworks.co.jp/products/matlab/>

Matsukura, R., K. Koyamada, Y. Tan (2005), Teleimmersive Collaboration Environment With Three Dimensional Images: VizGrid, *Special Issue on Virtual Reality and Immersive Technology, 3 (1)*, 21-30.

Matsuzawa, T. (2003), VizGrid:Achievement of real experimental environment on super computer network, *The Institute of Systems, control and Information Engineers, 47 (2)*, 65-70.

Max, N., P. Hanrahan, and R. Crawfis (1990), Area and volume coherence for efficient visualization of 3D scalar functions, *Computer Graphics, 24(5)*, 27-33.

Meißner, M., U. Kanus, and W. Straßer (1998), VIZARD II: A PCI-card for real-time volume rendering, Proceeding of SIGGRAPH/EUROGRAPHICS Workshop on Graphics Hardware '98, 61-67.

Molnar, S., M. Cox, D. Ellsworth, and H. Fuchs (1994), A Sorting Classification of Parallel Rendering, *IEEE Computer Graphics and Applications, 14(4)*, 23-32.

Moreland, K., L. Avila, and L. A. Fisk (2007), Parallel Unstructured Volume Rendering in ParaView, Visualization and Data Analysis 2007, Proceedings of SPIEIS& T Electronic Imaging, 64950F-1-12.

Mulder, J. D. and J. J. van Wi (1995), 3D Computational Steering with Parameterized Geometric Objects, Proceeding of Visualization '95, IEEE Computer Society Press, 304-311.

Murata, K. T., E. Kimura, Y. Kitamura, H. Shimazu, K. Fukazawa, T. Obara, H. Kumagai, J. Tanaka, T. Ikeda, H. Miyachi, Y. Matsumoto, M. Yoshikawa, D. Matsuoka and K. Yamamoto (2007), A Virtual Earth TV set via Real-time Data Transfer from Super Computer, SC2007 Bandwidth Challenge, (CD-ROM).

Neumann (1993), U., Parallel Volume-Rendering Algorithm Performance on Mesh-Connected Multicomputers, Proceedings of IEEE Symposium on Parallel Rendering 1993, 97-104.

Neumann (1994), U., Communication Costs for Parallel Volume-Rendering Algorithms, *IEEE Computer Graphics and Applications, 14(4)*, 49-58.

Nonaka, J., N. Kukimoto, N. Sakamoto, H. Hazama, Y. Watashiba, X. Liu, M. Ogata, M. Kanazawa, and K. Koyamada (2004), Hybrid Hardware-Accelerated Image Composition for Sort-Last Parallel Rendering on Graphics Clusters with Commodity Image Compositor, Proceedings of IEEE Symposium on Volume Visualization and Graphics 2004, 17-24.

Ohno, N., A. Kageyama, and K. Kusano (2006), Virtual Reality Visualization by CAVE with VFIVE and VTK, *Journal of Plasma Physics, 72 (6)*, 1069-1072.

Ohno, N. and A. Kageyama (2007), Scientific Visualization of Geophysical Simulation Data by the CAVE VR System with Volume Rendering, *Physics of Earth and Planetary Interiors, 163*, 305-311.

Ohno, N. and A. Kageyama (2010), Region-of-Interest Visualization by CAVE VR System with Automatic Control of Level-of-Detail, *Computer Physics and Communications, 181*, 720-725.

Okuda, M., Y. Karube, R. Matsukura, K. Hirota, M. Watan-

abe, and T. Matsuzawa (2007), Development of a three-dimensional object visualization system by a distributed processing method, *Journal of Visualization Society of Japan, 27 (7)*, 61-68.

小野謙二（2006）, 可視化研究の動向と次世代可視化システム, 第12回ビジュアリゼーションカンファレンス, <http://kgt.cybernet.co.jp/avsconso/event/vc12/summary/data/2-3.pdf>.

小野謙二（2008）, 次世代HPCにおける大規模データの可視化環境, SS研HPCフォーラム2008ペタスケール・アプリを支える基盤技術, <http://www.ssken.gr.jp/MAINSITE/download/newsletter/2008/20080827-sci-1/lecture-4/ppt.pdf>.

Parker, S. G. and C. R. Johnson (1995), SCIRun: A Scientific Programming Environment for Computational Steering, Proceedings of 1995 ACM/IEEE Supercomputing Conference, Vol.2, 1419-1439.

Peterka, T., R. Kooima, J. Girado, J. Ge, D. Sandin, and T, DeFanti (2007), Evolution of the varrier autostereoscopic vr display, Proc. IS&T / SPIE Electronic Imaging 2007.

Peterka, T., H. Yu, R. Ross, and K. -L. Ma (2008), Parallel Volume Rendering on the IBM Blue Gene/P, Proceedings of Eurographics Symposium on Parallel Graphics and Visualization 2008 (EGPGV '08).

Peterka, T., R. B. Ross, H. -W. Shen, K. -L. Ma,W. Kendall, and H. Yu (2009a), Parallel Visualization on Leadership Computing Resources, Journal of Physics: Conference Series SciDAC 2009.

Peterka, T., R. L. Kooima, D. Sandin, A. Johnson, J. Leigh, and T. DeFanti (2008b), Advances in the dynallax solid-state dynamic parallax barrier autostereoscopic visualization display system, *IEEE Transactions on Visualization and Computer Graphics, 14 (3)*, 487-499.

Peterka, T., R. Ross, H. Yu, and K. -L. Ma (2009b), Autostereoscopic display of large-scale scientific visualization, Proceedings of IS&T/SPIE Electronic Imaging 2007.

Pfister, H., J. Hardenbergh, J. Knittel, H. Lauer and L. Seiler (1999), The VolumePro real-time ray-casting system, Proceedings of SIGGRAPH '99, 251-260.

Pugmire, D., L. Monroe, C. C. Davenport, A. DuBois, D. DuBois, and S. Poole (2007), NPU-Based Image Compositing in a Distributed Visualization System, *IEEE Transactions on Visualization and Computer Graphics, 13(4)*, 798-809.

Rezk-Salama, C., K. Engel, M. Bauer, G. Greiner, and T. Ertl (2000), Interactive colume rendering on standard PC graphics hardware using multi-textures and multistage rasterization, Eurographics/SIGGRAPH Workshop on Graphics Hardware 2000, 109-118.

Rodrigues Jr., J. F., A. R. Balan, L. Zaima, and A. J. M. Traina (2010), A Survey on Distributed Visualization Techniques over Clusters of Personal Computers, (in press).

Rosario, J. M. del , R. Bordawekar, and A. Choudhary (1993), Improved parallel I/O via a two-phase run-time access strategy, *ACM SIGARCH Computer Architecture News, 21 (5)*, 31-38.

Ross, R. B., T. Peterka, H. -W. Shen, Y. Hong, K. -L. Ma, H. Yu, and K. Moreland (2008), Parallel I/O and Visualization at Extreme Scale, *Journal of Physics: Conference Series SciDAC 2008*.

Roth, S. D. (1982), Ray Casting for Modeling Solids, *Computer Graphics and Image Processing, 18 (2)*, 109-144.

Röttger, S., S. Guthe, D. Weiskopf, T. Ertl, and W. Straßer (2003), Smart hardware accelerated volume rendering, Eurographics/IEEE TCVG Symposium on Visualization '03, 231-238.

Sakamoto, N., J. Nonaka, K. Koyamada, and S. Tanaka (2007), Particle-based Volume Rendering, 2007 6th International Asia-Pacific Symposium on Visualization (APVIS 2007), 129-132.

Sandin, D., T. GE J. Margolis, J. Giradl, T. Peterka, and T. Defanti (2005), The Varrier™ Autostereoscopic Virtual Reality Display, *ACM Transactions on Graphics, Proceedings of ACM, 24 (3)*, 894-903.

Sasai, Y., A. Ishida, H. Sasaki, S. Kawahara, H. Uehara, and Y. Yamanaka (2005), Spreading of Antarctic Bottom Water examined using the CFC-11 distribution simulated by an eddy-resolving OGCM, *Polar Meteorology and Glaciology, 19*, 15-27.

Sasaki, S., M. Nonaka, Y. Masumoto, Y. Sasai, H. Uehara, and H. Sakuma (2008), An eddy-resolving hindcast simulation of the quasiglobal ocean from 1950 to 2003 on the Earth Simulator, Hamilton, K., and W. Ohfuchi (eds.), High Resolution Numerical Modeling of the Atmosphere and Ocean, 157-185.

Scheuermann, G. and X. Tricoche (2005), Topological Methods for Flow Visualization, Hansen, C. D., and C. R. Johnson (eds.), The Visualization Handbook, 341-356.

Schroeder, W. J. , K. M. Martin, and B. Lorensen (2002), The Visualization Toolkit, 3rd ed. Kitware Inc., 496.

Schroeder,W. J., K. M. Martin, andW. E. Lorensen (2003), The Visualization Toolkit: An Object-Oriented Approach to

Computer Graphics, 3rd ed., Kitware.

SciDAC DOE's Scientific Discovery through Advanced Computing <http://www.scidac.gov/>.

Seamons, K. E., Y. Chen, P. Jones, J. Jozwiak, and M. Winslett (1995), Server-directed collective I/O in Panda, Proceedings of the 1995 ACM/IEEE conference on Supercomputing, (CD-ROM), 57.

Shen, H. -W. and C. R. Johnson (1994), Differential Volume Rendering: A Fast Volume Visualization Technique for Flow Animation, Proceedings of IEEE Visualization Conference, 180-187.

Shen, H. -W., L. -J. Chiang, and K.-L. Ma (1999), A Fast Volume Rendering Algorithm for Time-Varying Fields Using a Time-Space Partitioning (TSP) Tree, Proceedings of IEEE Visualization Conference 1999, 371-377.

白山晋(2006), 知的可視化, 計算力学レクチャーシリーズ, 丸善.

Shima, S., K. Kusano, A. Kawano, T. Sugiyama, and S. Kawahara (2009), The super-droplet method for the numerical simulation of clouds and precipitation: a particle-based and probabilistic microphysics model coupled with a non-hydrostatic model, *The Quarterly Journal of the Royal Meteorological Society, 135 (642)*, 1307-1320.

Sohn, B. S., C. L. Bajaj, and V. Siddavanahalli (2002), Feature Based Volumetric Video Compression for Interactive Playback, Proceedings of IEEE Symposium on Volume Visualization 2002, 89-96.

Soukup, T. and I. Davidson (2002), Visual Data Mining, John Wiley & Sons, Inc.

Stoll, G., M. Eldridge, D. Patterson, A. Webb, S. Berman, R. Levy, C. Caywood, M. Taveira, S. Hunt, and P. Hanrahan (2001), Lightning-2: A High-Performance Display Subsystem for PC Clusters, Proceedings of ACM SIGGRAPH 2001 Conference, 141-148.

Stalling, D., M. Westerhoff, and H. -C. Hege (2005), amira: A Highly Interactive System for Visual Data Analysis, Hansen, C. D., and C. R. Johnson (eds.), The Visualization Handbook, 749-769.

Stompel, A., K. -L. Ma, E. B. Lum, J. P. Ahrens, and J. Patchett (2003), SLIC: Scheduled Linear Image Compositing for Parallel Volume Rendering, Proceedings of IEEE Symposium on Parallel and Large-Data Visualization and Graphics 2003, 33-40.

Strengert, M., M. Magall, D. Weiskopf, and T. Ertl (2004), Hierarchical Visualization and Compression of Large Volume Datasets Using GPU Clusters, Proceedings of Eurographics Symposium on Parallel Graphics and Visualization 2004, 41-48.

Sutherland, I.E. (1968), The Ultimate Display, Proceedings of IFIP Congress 65, Vol.2, 506-508, 582-583.

Takeuchi, A., F. Ino, and K. Hagihara (2003), An Improved Binary-Swap Compositing for Sort-Last Parallel Rendering on Distributed Memory Multiprocessors, *Parallel Computing, 29 (11-12)*, 1745-1762.

Tanaka, T., Y. Ebara, H. Sone, and K. Koyamada (2005), Study on Speed-up for Remote Visualization in Grid Computing Environment, *IPSJ SIG Notes, 2005 (116)*, 109-114.

Thakur, R., R. Boardawekar, A. Choudhary, R. Ponnusamy, and T. Singh (1994), PASSION runtime library for parallel I/O, Proceedings of the 1994 Scalable Parallel Libraries Conference, 119-128.

Thakur, R. and A. Choudhary (1996), An Extended Two- Phase Method for Accessing Sections of Out-of-Core Arrays, *Scientific Programming, 5(4)*, 301-317.

Thakur, R., A. Choudhary, R. Bordawekar, S. More, and S. Kuditipudi (1996a), Passion: Optimized I/O for Parallel Applications, *Computer, 29(6)*, 70-78.

Thakur, R., W. Gropp and E. Lusk (1996b), An Abstract-Device Interface for Implementing Portable Parallel- I/O Interfaces, Proceedings of the 6th Symposium on the Frontiers of Massively Parallel Computation, 180-187.

Thakur, R.,W. Gropp, and E. Lusk (1998), A case for using MPI's derived datatypes to improve I/O performance, Proceedings of the 1998 ACM/IEEE conference on Supercomputing, (CD-ROM).

Thakur, R., W. Gropp, and Ewing Lusk (1999), Data Sieving and Collective I/O in ROMIO, Proceedings of the The 7th Symposium on the Frontiers of Massively Parallel Computation, 182.

Thibault, S., X. Cavin, O. Festor, and E. Fleury (2002), Unreliable transport protocol for commoditybased opengl distributed visualization, Workshop on Commodity-Based Visualization Clusters.

Tu, T., H. Yu, L. Ramirez-Guzmanz, J. Bielak, O. Ghattas, K.-L. Ma, and D. R. O'Hallaron (2006), From Mesh Generation to Scientific Visualization: An End-to-End Approach to Parallel Supercomputing, Proceedings of ACM/IEEE Supercomputing 2006 Conference.

Tuchman, A., D. Jablonowski, and G. Cybenko (1991), Vista: A system for remote data visualization, Technical Report 1067, Center for Supercomputing Research and Development, University of Illinois at Urbana- Champaign.

Uehara, H., S. Kawahara, N. Ohno, M. Furuichi, F. Araki, and A. Kageyama (2006), MovieMaker: a parallel movie-making software for large scale simulations, *Journal of Plasma Physics, 72 (6)*, 841-844.

Van Gelder, A., and K. Kim (1996), Direct volume rendering with shading via three-dimensional textures, 1996 Symposium on Volume Visualization, 23-30.

VAPOR <http://www.vapor.ucar.edu/>

van der Ven, H., B. C. Schultheiss, S. Doi, H. Matsumoto, K. Sugihara, and T. Takei (1998), Real-time visualization of large data sets on NLR's NEC SX-4, *Lecture notes in Computer Science, 1401*, 397-402.

View Mol3D <http://redandr.ca/vm3/>.

VisIt <https://wci.llnl.gov/codes/visit/>.

VizGrid <http://www.vizgrid.org/>.

Walton, J. (2005), NAG' s Iris Explorer, Hansen, C. D., and C. R. Johnson (eds.), The Visualization Handbook, 633-654.

Wang, C. and H. -W. Shen (2004), A framework for rendering large time- varying data using waveletbased time- space partitioning (WTSP) tree, Technical Report No. OSU-CISRC-1/04-TR05, Department of Computer Science and Engineering, The Ohio State University.

Weinstein, D. M., S. Parker, J. Simpson, Kurt Zimmerman, and G. M. Jones (2005), Visualization in the SCIRun Problem-Solving Environment, Hansen, C. D., and C. R. Johnson (eds.), The Visualization Handbook, 593-614.

Weiskopf, D., M. Weiler, and T. Ertl (2004), Maintaining constant frame rates in 3D texture-based volume rendering, Proceedings of IEEE Computer Graphics International (CGI), 604-607.

Weiskopf, D. and G. Erlebacher (2005), Overview of Flow Visualization, Hansen, C. D., and C. R. Johnson (eds.), The Visualization Handbook, 261-278.

Weiskopf, D.(2007), GPU-Based Interactive Visualization Techniques, Mathmatics + Visualization, Springer.

Westermann, R. (1995), Compression Domain Rendering of Time-Resolved Volume Data, Proceeding of IEEE Visualization '95, 168-175.

Westover, L. (1990), Footprint evaluation for volume rendering, Computer Graphics, Proceedings of SIGGRAPH '90, 367-376.

Wilhelms, J. and A. Van Gelder (1992), A Octrees for faster isosurface generation, *ACM Transactions on Graphics, 11 (3)*, 201-227.

Wood, J., H. Wright and K. Brodlie (1997), CSCV - Computer Support for Collaborative Visualization, Visualization & Modeling, 13-25.

Wu, Y., V. Bhatia, H. Lauer and L. Seiler (2003), Shearimage order ray-casting volume rendering, ACM SIGGRAPH Symposium on Interactive 3D Graphics '03, 152-162.

Wu, Q., J. Gao, M. Zhu , S. Nageswara, V. Rao , J. Huang, and S. Iyengar (2008), Self-Adaptive Configuration of Visualization Pipeline Over Wide-Area Networks, *IEEE Transactions on Computers, 57 (1)*, 55-68.

Wylie, B., C. Pavlakos, V. Lewis, and K. Moreland (2001), Scalable Rendering on PC Clusters, *IEEE Computer Graphics and Applications, 21(4)*, 62-70.

Wyvill, G. C., McPheeters, and B. Wyvill (1986), Data structure for soft objects, *The Visual Computer, 2*, 227-234.

Yang, D. -L., J. -C. Yu, and Y. -C. Chung (1999), Efficient Compositing Methods for the Sort-Last-Sparse Parallel Volume Rendering System on Distributed Memory Multicomputers, Proceedings of International Conference on Parallel Processing 1999, 200-207.

Yu, H., K. -L. Ma, and J.Welling (2007), A Parallel Visualization Pipeline for Terascale Earthquake Simulations, Proceedings of the 2004 ACM/IEEE conference on Supercomputing, 49.

Yu, H. and K. -L. Ma (2005), A Study of I/O methods for Parallel Visualization of Large-scale Data, *Parallel graphics and visualization, 31 (2)*, 167-183.

Yu, H., T. Tu, J. Bielak, O. Ghattas, J. C. L'opez, K.-L. Ma, D. R. O' Hallaron, L. Ramirez-Guzmanz, N. Stone, R. Taborda-Rios, and J. Urbanic (2006), Remote Runtime Steering of Integrated Terascale Simulation and Visualization, HPC Analytics Challenge, ACM/IEEE Supercomputing 2006 Conference, 2006.

Yu, H., C. Wang, and K. -W. Ma (2008), Massively parallel volume rendering using 2-3 swap image compositing, Proceedings of IEEE/ACM Supercomputing 2008 Conference, 1-11.

Zhang, S., D. H. Laidlaw, and G. Kindlmann (2005), Diffusion Tensor MRI Visualization, Hansen, C. D., and C. R. Johnson (eds.), The Visualization Handbook, 690-717.

Zhukov, L. and A. H. Barr (2005), Oriented Tensor Reconstruction, Hansen, C. D., and C. R. Johnson (eds.), The Visualization Handbook, 690-717.