社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2008－DPS－135 (15)
2008／6／20

マルチディスプレイを用いた 高解像度ディスプレイ環境の提案

千葉豪 石田智之 柴田義孝

岩手県立大学大学院 ソフトウェア情報学研究科

近年, ブロードバンドネットワークサービスの普及によりIP ネットワーク上で DV や HDV のような高精細映像の配信や, 高性能ビデオカードを搭載した複数の PC と複数ディスプレイやプロジェクタを組み合わせたレンダリングクラスタの構築がソフトウェアレベルで可能となり, 高精細で大画面かつ高臨場感を持つディスプレイ環境が実現できるようになった. そこで本研究では, 大規模高解像度によるバーチャルリアリティや 2，3 次元映像による協調作業を実現するため, 複数 PC から成るクラスタと液晶ディスプレイを組み合わせ, これらを高速ネットワークで接続することにより, 実質的にクラスタサイズに比例した高解像度を達成できるプレゼンテーションシステムを開発する. 本システムはアプリケーションと独立してミドルウェアとして実装し, ピクセルデータのストリーミング機能やディスプレイ間同期表示機能や多地点間映像表示機能を実現し, 高速ネットワーク上で柔軟に制御を可能とする. 本稿では, 本システムのアーキテクチャや実現方法およびプロトタイプシステムについて述べる.

High Resolution Display Environment using Multi-Displays

Go Chiba Tomoyuki Ishida Yoshitaka Shibata

Graduate School of Software and Information Science
Iwate Prefectual University

Recently, delivery of high quality picture such as DV and HDV on a IP networks by spread of broadband networks and the construction of the rendering cluster which is a combination of multiple PCs and displays or projectors with cost effective vedeo card have been realized at software level. These have made it possible to a construct of high-definition, big screen and high sence of reality display environment.
In our research, we develop a presentation environment system which can achieve high definition in proportion to cluster size substantially by combination of multiple PC cluster and displays interconnecting on high speed network. This system is application independent and implemented as middleware and realizes pixel streaming and synchronous display functions and enable a control on high speed network flexibly. In this paper, we describe an architecture and implementation method of our system and prototype.

1 はじめに

近年, ブロードバンドネットワークサービスの普及により IP ネットワーク上における DV や HDV のような 高精細映像の配信や, 三次元仮想空間におけるオブジェクト情報, テクスチャ情報等を利用した大容量データ通信, 高解像度の地図データの利用したといったことが実現されてきている. また, 高性能ビデオカードを搭載した複数の計算機と複数ディスプレイやプロジェクタを組み合わ

せたレンダリングクラスタの構築がソフトウェアレベルで可能となり, 過去の可視化, VR 分野において利用されてきた専用グラフィックスワークステーションやマルチスクリーンを利用せずとも, 一般的に利用されている計算機, ディスプレイ環境を応用することで容易に高精細で大画面かつ高臨場感を持つディスプレイ環境が実現できるようになった [1].

そこで本研究では, 対象アプリケーションとしてバーチャルリアリティシステムや, 双方向ビデオ通信システム, 高解像度地図画像データを用いたシステム等を想定し, これらのアプリケーションを大規模高解像度環境において実現するため, 複数 PC クラスタと液晶ディスプレイを組み合わせ, これらを高速ネットワーク上で相互接続することにより, 容易にアプリケーションの出力結果をストリームとして配信し, 実質的にクラスタサイズに比例した高解像度ディスプレイ環境を達成できるプレゼンテーションシステムを開発する.

本システムはアプリケーションと独立してミドルウェアとして実装し, 複数のディスプレイからなる高解像度出力装置への描画処理を行うクラスタ, それらクラスタの管理やストリーム配信制御を行うための管理ノード, 実際にアプリケーションが動作しユーザへのインターフェースを提供するクライアントノードの 3 つの要素から構成される. 本システムの提供する機能としてピクセルデータのストリーミング機能やディスプレイ間同期表示機能や多地点間映像表示機能を実現し, 高速ネットワーク上で柔軟に各ディスプレイへの描画サイズ, ストリーム送信先の指定といった制御を可能とする. 本稿では, 本システムのアーキテクチャや実現方法およびプロトタイプシステムについて述べる.

2 関連研究

本研究における関連研究として DMX[2], Chromium[3], SAGE[4][5] がある.

DMX はネットワークを介し複数の X Window サーバを相互接続することで仮想的に単一の高解像度デスクトップ環境を実現するもので Linux をベースとしたシステムである. しかし, あくまでもクライアントの単一のデスクトップ画面を表示するもので複数アプリケーションを並列で動作, 表示するといった機能は提供されていない.

また, Chromium では OpenGL のコマンドをダンプし, SPU(Stream Processing Unit) を用いたストリームベースの分散レンダリングライブラリを提供している. このシステムでは OpenGL ライブラリを用いた Virtual Reality, 三次元可視化といったシステムを想定しており, 並列アプリケーションの分散レンダリングを許容しておらず また, 複雑なシーン描画においてネットワークの利用帯域が増大してしまうといった問題がある.

近年では, イリノイ大学が開発した SAGE(Scalable Adaptive Graphics Environment) といったものがシステムが存在する. このシステムではピクセルデータをキャプチャ仮想フレームバッファを利用し, 擬似的に高解像度なディスプレイ環境を実現しており並列アプリケーション等をサポートしているが, グリッド環境における複数クラスタ環境を想定しているため構成が複雑であり, IP ユニキャストのみを利用しており, ネットワーク帯域への負荷が大きい, またライブラリとして提供されており, アプリケーションへの適用が困難であるといった問題点を含んでいる.

3 タイルドディスプレイシステム

近年, ブロードバンドネットワークサービスの普及により IP ネットワーク上で DV や HDV のような高精細映像の配信や, 高性能ビデオカードを搭載した複数の PC と複数ディスプレイやプロジェクタを組み合わせたレンダリングクラスタの構築がソフトウェアレベルで可能となり, 高精細で大画面かつ高臨場感を持つディスプレイ環境が実現できるようになった [1] [4].

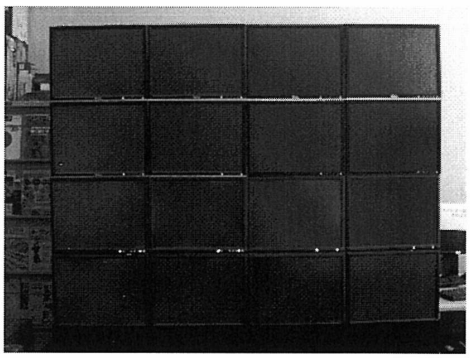

Figure 1: タイルドディスプレイ

タイルドディスプレイとは図 1 に示すような複数の液晶ディスプレイを組み合わせたディスプレイ環境を指し, 専用のグラフィックスハードウェアを利用せずにソフトウェアを利用した仮想的な高解像度ディスプレイ環境を構築できる, 一般のディスプレイを利用することで用意にシステム構築が可能である, プロジェクターを用いた分散レンダリングシステムに比べキャリブレーション

処理が不要であるといったメリットがある反面, 個々のディスプレイには明度, 表示色数といった性能の差が存在する, ディスプレイ枠の存在により完全にシームレスな表示ができない, 台数に比例して電力が必要になるといったデメリットも存在する.

本研究では, 大規模高解像度によるバーチャルリアリティや2, 3次元映像による協調作業を実現するため, 複数PCクラスタと液晶ディスプレイを組み合わせ, 高速ネットワークを介し相互接続することにより, 実質的にクラスタサイズに比例した高解像度を達成できるプレゼンテーションシステムを開発する. 本システムはアプリケーションと独立してミドルウェアとして実装し, ピクセルデータのストリーミング機能やディスプレイ間同期表示機能や多地点間映像表示機能を実現し, 高速ネットワーク上で柔軟にタイルドディスプレイへの描画処理の制御を可能とする.

3.1 ネットワーク構成

本システムのネットワーク構成は図2のようになっており各拠点には出力装置としてタイルドディスプレイ, タイルドディスプレイへの描画処理を行う Rendering Cluster, 各タイルの状態を管理する Tile Manager が存在しており, ユーザは Application Host 上でビデオ通信, CVE, 画像表示ツールといったアプリケーションを動作させ, そのピクセルデータを Rendering Cluster へ送信することで各アプリケーションの出力結果をタイルドディスプレイ上に描画することを可能とする.

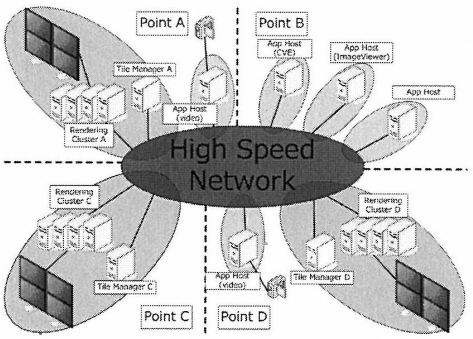

Figure 2: システム構成

3.2 システムアーキテクチャ

本システムのアーキテクチャを図3に示す. 出力装置として前述したタイルドディスプレイを用いることを想定しており, Application Host, Tile Manager, Rendering Cluster といった3つの構成要素から成る.

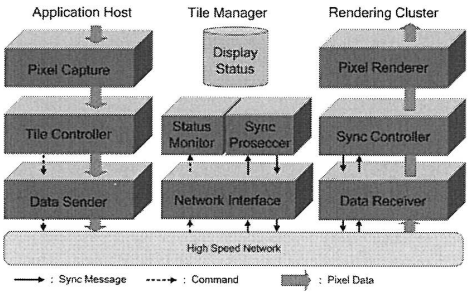

Figure 3: システムアーキテクチャ

3.3 Application Host

Application Host ではアプリケーションを提供すると同時に, Pixel Capture モジュールがピクセルデータをシステムのフレームバッファから取得, バッファへと格納される. その後, Data Sender がバッファ内のデータを Rendering Cluster へピクセルストリームとして送信を行う. また, Message Controller では同期処理等を実現するメッセージを生成し, Data Sender モジュールを介して Tile Manager へと送信を行う.

3.4 Tile Manager

Tile Manager では 内部データとしてタイルドディスプレイにおける各タイルのサイズ, セッション情報, クライアントにおける画像サイズといった情報を保持し, Application Host から受け取ったメッセージをもとにこれらの情報へアクセスしタイルドディスプレイへの描画処理の制御を行う. また, Rendering Cluster との同期メッセージのやりとりにより各タイル間におけるフレーム同期処理を実現する.

3.5 Rendering Cluster

Rendering Cluster では Application Host から送信されたピクセルストリームを受信後それらのデータをバッファへ格納, その後 Window Manager モジュールが同期処理用メッセージをもとにバッファからピクセルデータを取得, タイルに適合するようリサイズなどの処理を加え, Pixel Renderer が担当するタイルへの描画処理を行う.

4 システムの機能

4.1 ピクセルキャプチャ

Application Host 上でのピクセルデータの取得には図 4 に示すとおり OpenGL の機能である glReadPixels 関数によって行い, 毎フレームをフロントバッファから RGB にて各ピクセル値を取得する. まず, glReadBuer では読み込む先のフレームバッファの種類を指定する. その後, glPixelStorei にてバッファの読み取り方を 1 byte ずつ読み込むように指定し, glReadPixels によって読み取る領域の左下の座標 (x,y), 読み取る領域の幅と高さ (width, height), 取得する色情報の形式 (GL RGB), 読み取ったデータを保存する配列の型 (GL UNSIGNED BYTE) を指定してフレームバッファからピクセルデータを取得する.

```
glReadBuffer(GL_FRONT)
glPixelStorei(GL_UNPACK_ALIGNMENT, 1)
pixeldata = glReadPixels(x,
                y,
                width,
                height,
                GL_RGB,
                GL_UNSIGNED_BYTE)
                        .
                        .
                        .
```

Figure 4: キャプチャメソッド

4.2 分割処理

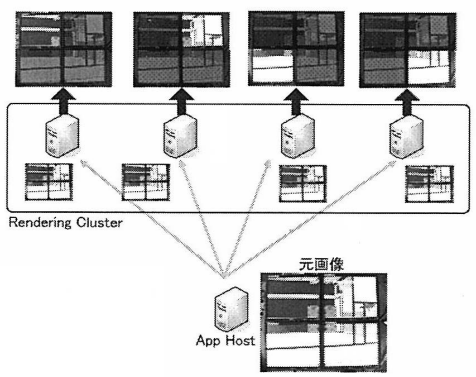

Figure 5: 分割処理

本システムにおける分割処理は図 5 のように行われる. Application Host でキャプチャされたピクセルデータを転送元でのサイズを保持したまま, 各 Rendering Cluster へと転送される. その後, ピクセルデータを受け取った各クラスタでは担当するタイルのサイズに合わせるため分割, リサイズといった処理を行う. これは, クライアント側での分割処理による CPU 負荷の増大を防ぐためであり, これによりクライアント側ではアプリケーションの処理, ピクセルデータのキャプチャ処理といったタスクにリソースを集中できる.

4.3 ディスプレイ間同期

ディスプレイ間の同期はメッセージのやりとりによって行われ, そのタイミングは図 7 の示す通りとなる.

まず, Application Host はキャプチャしたピクセルデータをストリームとして各クラスタへ送信を開始する. 各クラスタはそれぞれピクセルデータの受信が完了後描画の準備ができたことを Tile Manager へと通知する. Tile Manager では全てのクラスタからの通知を受け取ったら, 各クラスタに更新メッセージを送信し, 描画処理を開始するよう命じる.

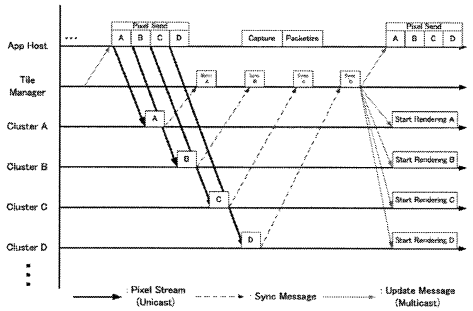

Figure 6: ユニキャスト同期フロー

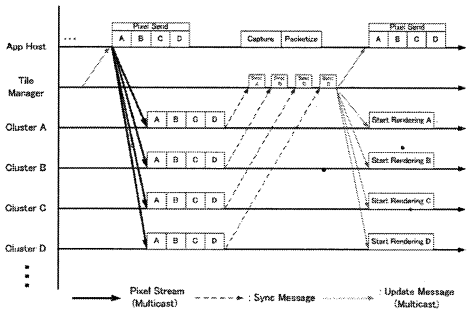

Figure 7: マルチキャスト同期フロー

4.4 IP マルチキャスト

本システムではピクセルデータのストリーミングにおいて IP マルチキャストの利用を可能とし, 表示アプリケーションを一つのセッションとして管理する. また, ここで先のアーキテクチャで述べた Application Host からのメッセージ機能を用い動的にセッションへの参加, 離脱をすることで図 8 が示すように各セッション単位での柔軟な表示ディスプレイの割り当てを実現する. これにより, システムにおけるセッション管理が容易になる, タイルの枚数の増加させた際のピクセルストリームにおけるネットワークの利用帯域の軽減を実現するといったことが期待される.

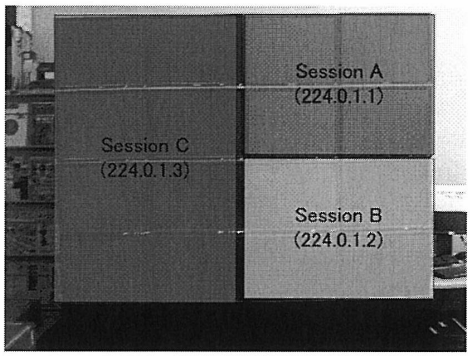

Figure 8: マルチキャストによるセッション管理

5 プロトタイプシステム

現在プロトタイプシステムとして Application Host, Rendering Cluster の 2 つのコンポーネントを構築しギガビットネットワークでの相互接続を行い実際に 16 面のタイルドディスプレイへの描画の確認を行った. 各タイルを担当するディスプレイは 1600x1200 の解像度に対応しており, 描画処理を担当する Rendering Cluster はそれぞれデュアルディスプレイでの出力設定を行い計 8 台の PC を利用しており, Application Host, RenderingCluster におけるマシンの性能は以下に示すとおりとなる.

- Application Host

  CPU Pentium D 3.20GHz
  Memory 2GB RAM

- Rendering Cluster

  CPU Pentium 4 3.73GHz
  Memory 1GB RAM
  Video nVidia Quadro FX 1400

また適用事例として図 9 では仮想空間を用いた非同期型協調作業支援システム [6] へ適用した場合の様子を, 図 10 では全方位映像を用いた監視システム [7] へ適用した場合の様子を示している.

これらは Application Host 上で図 11 に示すような作業ウィンドウを提示, ウィンドウ領域内のピクセルデータをシステムのフレームバッファからキャプチャし IP マルチキャストを用いて 8 台の各クラスタへとピクセル情報をストリーミングしている.

各 Rendering Cluster 上では Application Host から受け取ったピクセルデータを元に担当するタイルのサイズへリサイズ, 分割処理を行う. その後各クラスタが OpenGL 命令を用いてタイルへの描画処理を行っている.

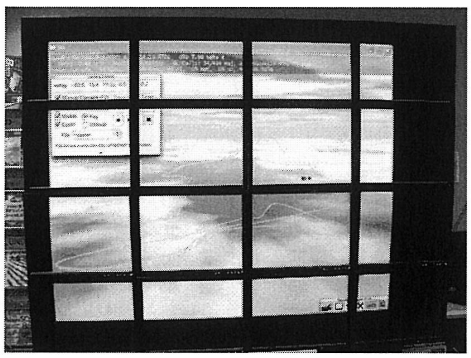

Figure 9: VR アプリケーションへの適用

Figure 10: ビデオアプリケーションへの適用

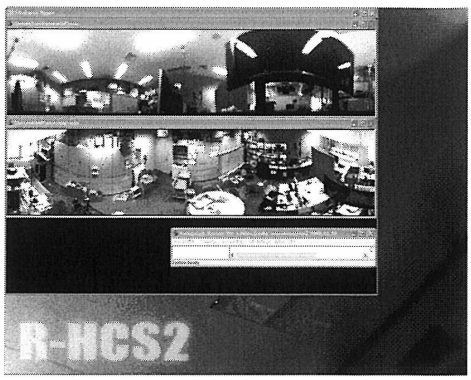

Figure 11: クライアント画面

6 評価

本システムのプロトタイプ完成後は"フレームレート", "利用帯域"に関して性能評価を行う. フレームレートはタイルドディスプレイにおいて描画を行うタイル数を変化させた際のフレームレートの変化を, 利用帯域に関しては Application Host, Tile Manager, Rendering Cluster の 3 地点間での利用帯域の違いに関して評価していく.

また, 関連研究である SAGE[4] [5] との比較評価として VLC を用いた双方向でのビデオ通信, CVE(Collaborative Virtual Environment) を利用した複数人での協調作業といったアプリケーションへの適用を行った際の性能を比較していく予定である.

7 まとめ

本研究では複数ディスプレイを用いた高解像度プレゼンテーション環境を提案し, ビデオ通信, バーチャルリアリティといった分野のアプリケーションへと適用した. これは専用のグラフィックスハードウェアを用いず, 安価なディスプレイを複数台利用することで用意に高解像度環境を実現し, これにより高精細な三次元仮想空間やビデオイメージの表現が可能になるといったことが期待される.

今後はさらにプロトタイプシステムの構築を進め, 描画を行うタイル数の変化をさせた場合のフレームレートの変化, CAVE システムを利用した場合とタイルドディスプレイを利用した場合の没入感, 臨場感の違いを評価していく.

また, 現時点でのプロトタイプではキャプチャするピクセルデータのサイズがクライアントのディスプレイサイズによって制限されてしまう, クライアントでのキャプチャ負荷が大きいといった問題が存在している. そこでクライアントでのディスプレイサイズに依存しないキャプチャ手法や負荷の軽減手法に関して今後検討していきたいと考えている.

References

[1] Bruno Raffin and Luciano Soares. Pc clusters for virtual reality. *IEEE Virtual Reality Conference*, pages 215–222, March 2006.

[2] Distributed multihead x project. `http://dmx.sourceforge.net`.

[3] Greg Humphreys, Mike Houston, Ren Ng, Randall Frank, Sean Ahern, Peter D.Kirchner and James T.Kloswski. Chromium: A stream-processing framework for interactive rendering on clusters. *Proc. ACM International Conference on Computer Graphics and Interactive Techniques Conference(SIGGRAPH 2002)*, pages 693–702, July 2002.

[4] Byungil Jeong, Luc Renambot, Rajvikram Singh, Andrew Johnson, and Jason Leigh. High-performance scalable graphics architecture for high-resolution displays. *Technical Paper*, 2005.

[5] Byungil Jeong, Luc Renambot, Ratko Jagodic, Rajvikram Singh, Julieta Aguilera, Andrew Johnson and Jason Leigh. High-performance dynamic graphics streaming for scalable adaptive graphics environment. *Proc. ACM/IEEE SC 2006 Conference (SC 2006)*, November 2006.

[6] Hiroki Ogasawara and Yoshitaka Shibata. Asynchronous collaborative virtual environment support system by using revision tree presentation method. *International Workshop on Network-based Virtual Reality and Tele-existence (INVITE 2008)*, March 2008.

[7] Yousuke Sato, Koji Hashimoto and Yoshitaka Shibata. A new remote camera work system for teleconference using a combination of omni-directional and network controlled cameras. *Proc. IEEE 22nd International Conference on Advanced Information Networking and Applications(AINA 2008)*, pages 502–508, March 2008.