「コピー&トランスファー」操作に基づく音声・動画像データの分散共有機構

坂本弘達[†] 岡田義広[†]
下川俊彦[†] 牛島和夫[††]

著者らは，3次元仮想空間を分散協調作業に用いる環境について研究を行っている．特に本研究の目的は，音声・動画像通信機能がある分散協調作業空間を容易に構築可能とするための機構を明らかにすることである．本論文では,「コピー&トランスファー」と呼ぶ概念に基づいた音声・動画像データを分散共有する機構について述べる．この概念は，計算機画面上で直接操作可能な可視化されたソフトウェア部品を複製し，それを別の計算機へ転送することにより，データの分散共有を実現するものである．音声・動画像を扱う機能をこのように部品化することで，この部品の「コピー&トランスファー」操作によって，プログラミング知識のない作業者でも，音声・動画像通信機能のある分散協調作業空間の構築が可能となる．本論文では,「コピー&トランスファー」操作に基づく音声・動画像データの分散共有の実現機構を説明し，応用例をあげることによりその有用性を述べる．

Distributed Audio-Video Sharing by Copy-and-Transfer Operation

Hirotatsu Sakamoto,[†] Yoshihiro Okada,[†]
Toshihiko Shimokawa[†] and Kazuo Ushijima[††]

This paper treats a distributed audio-video sharing mechanism for collaborative virtual environments. Especially the authors propose a *copy-and-transfer* concept. This concept is that making a copy of a visible, manually operable software component and transfering it to another computer enable users to share it. If a facility that manages audio or video data is realized as such a component, even end-users can easily and rapidly build networked audio-video communication environments through the *copy-and-transfer* operation. This paper explains realization mechanisms of a *copy-and-transfer* concept and describes its availability by showing some application examples.

1. はじめに

近年，計算機ハードウェア性能の向上により，3次元空間の表示や音声・動画像通信を実時間で扱うことが可能になっている．また，ネットワーク技術の進歩とインターネットの爆発的な普及にともない，広帯域ネットワークの利用が可能となっている．そして現在，音声・動画像通信の機能を持つ多くの応用システムが開発および利用されている．また，ネットワークを介した分散環境下において，複数の利用者が協調して作業を行う分散協調作業環境についても研究が行われている．著者らは，3次元仮想空間を分散協調作業に用いる環境について研究・開発を行っている．

本研究の目的は，音声・動画像通信機能がある分散協調作業空間を容易に構築するための機構を明らかにすることである．そのため，音声・動画像データの分散共有を，計算機画面上で直接操作可能な可視化されたソフトウェア部品の対話操作によって実現することを目指している．

本論文では,「コピー&トランスファー」と呼ぶ概念を提案している[1),2)]．この「コピー&トランスファー」操作は，従来のウィンドウ操作で用いられる「コピー&ペースト」操作を拡張したものと考えることができる．これは，計算機画面上で直接操作可能な可視化されたソフトウェア部品を複製し，別の計算機へ転送することにより，データの分散共有を実現するものである．音声・動画像を扱う機能を可視化されたソフトウェア部品とすることにより，その部品の「コピー&トラ

† 九州大学大学院システム情報科学研究院
Graduate School of Information Science and Electrical Engineering, Kyushu University
†† 財団法人九州システム情報技術研究所
Institute of Systems & Information Technologies /KYUSHU

ンスファー」操作によって，音声・動画像通信環境を容易に構築可能となる．本論文では,「コピー&トランスファー」の実現機構を述べ，応用例をあげることによりその有用性を明らかにする．

本論文は以下の構成になっている．2 章では,「コピー&トランスファー」の概念について詳しく説明するとともに，その新規性を述べる．3 章では，研究基盤システムとして用いた *IntelligentBox* [3] の基本機能を説明する．4 章では，音声・動画像データの分散共有の実現機構を述べる．5 章では応用例や開発手順，実際の性能について考察する．最後に 6 章で，まとめと今後の課題を述べる．

2. コピー&トランスファー

現在のソフトウェア開発は，オブジェクト指向プログラミングが主流である．従来のオブジェクト指向プログラミングでのソフトウェア部品化は，ソースコードレベルで行われている．このソースコードの編集・作成によるソフトウェア開発は，プログラミング知識のない利用者には行えないのが現状である．著者らは，ソースコードレベルの部品化では十分ではなく，計算機の画面上に表示され，マウスなどのデバイスにより直接操作可能な形式で部品化されている必要があると考えている．その際に，部品は現実世界に存在しているものと同じ機能，見え方で定義されていることが望ましい．そのように部品化されていると，部品の機能の連携，部品の複製，別の計算機への転送など，開発の際の作業が視覚的・直感的に行えるようになる．

図 1 は，ネットワークにつながれた 2 台の計算機における「コピー&トランスファー」操作を示している．左側の計算機を使用している作業者は，ソフトウェア部品の複製を生成し（「コピー」），それを右側の計算機に転送する（「トランスファー」）．これは，マウスデバイスを用いた画面上での直接操作のみにより行われる．さらに，これら分散された 2 つのソフトウェア部品が持つデータの整合性を保つ機構を実現することにより，2 台の計算機それぞれの利用者間で，そのデータが共有されることとなる．以上のように，音声・動画像データを扱うソフトウェア部品の「コピー&トランスファー」操作によって，音声・動画像データの分散共有が可能となる．

画面上での直接操作が可能な可視オブジェクトとしてソフトウェアを部品化するという概念を持つ 3 次元ソフトウェア開発システムとして，すでに *IntelligentBox* がある．本論文で提案する「コピー&トランスファー」操作の実現機構とその有用性を明らかにするため，*IntelligentBox* を研究基盤システムとして用い実装を行った．

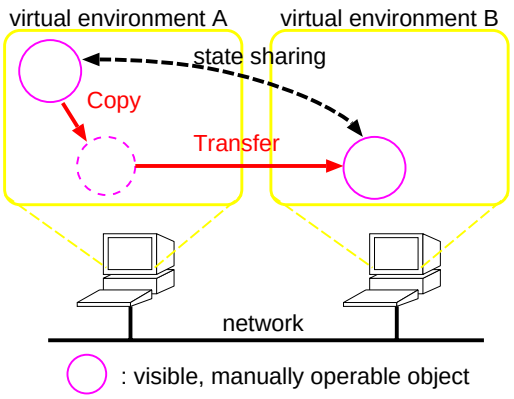

図 1 コピー&トランスファーの概念
Fig. 1 Concept of *copy-and-transfer*.

［関連研究］

仮想空間構築ツールとして，DIVE [4),5)]，MASSIVE [6)]，dVS [7)]，MERL [8),9)]，MR Toolkit [10)] などがある．DIVE は，ソースコードの編集・作成によって開発する仮想現実感構築ツールである．ネットワーク機能が拡張され，音声と文字データによる通信機能がある．すでに，DIVE を元にした協調作業システム [5)] が開発されている．MASSIVE は，仮想空間を共有することによる遠隔会議システムである．文字データを音声に変えることにより会話する機能があるが，動画像の通信機能はない．dVS は，商用の仮想現実感構築ツールである．ネットワークを介した通信機能があり，多人数参加型の応用システムも開発可能であるが，音声・動画像の通信機能はない．MERL は，多人数参加型仮想空間構築システムである．文字列の吹き出しによる会話や，音声通信による会話が可能である．MR Toolkit は，仮想現実感システム開発ツールであり，ライブラリベースのツールキットである．多人数参加型の仮想現実感応用システムの開発も可能である．本論文で提案するような，音声・動画像を扱う機能を可視化されたソフトウェア部品として表現し，さらにその「コピー&トランスファー」という直感的な操作により，音声・動画像通信を実現する機能はこれらのシステムにはない．

3. IntelligentBox

本研究では研究基盤システムとして *IntelligentBox* を採用している．*IntelligentBox* は様々なソフトウェア部品を，*Box* と呼ぶ画面上で直接操作可能な可視部品として提供する．また，画面上の直接操作により，

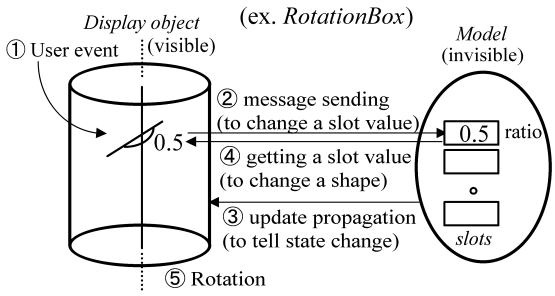

図 2 MD 構造と内部メッセージ
Fig. 2 An MD (Model-Display object) structure of a *Box* and its internal messages.

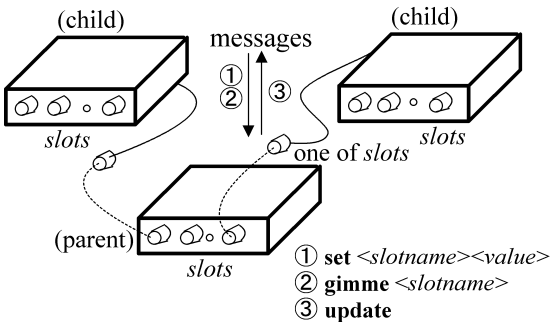

図 3 *Box* 間標準メッセージ
Fig. 3 Standard messages between *Boxes*.

Box を組み合わせ機能合成する機構を持つ．

IntelligentBox における機能合成の機構は，*IntelligentPad* [11] の機構を採用している．*IntelligentPad* は，北海道大学で研究開発を行っているシンセティックメディアシステムである．*IntelligentPad* における紙のイメージを持つ機能オブジェクトである *Pad* を，3 次元形状を持つ機能オブジェクトへと拡張したシステムが *IntelligentBox* である．

3.1 Box の基本構造

図 2 に示すように，各 *Box* はモデルとディスプレイオブジェクトから構成される．この構造を MD（Model-Display object）構造と呼ぶ．モデルは *Box* の状態情報を保持している．状態情報はスロットと呼ばれる変数に格納される．ディスプレイオブジェクトには，画面上に表示される *Box* の形状と，利用者の操作イベントに対する振舞いが定義される．

図 2 は，*RotationBox* と呼ぶ回転部品の例である．*RotationBox* は，状態情報として回転角度を保持している．その情報は，*ratio* と呼ぶスロットを介してアクセスされる．利用者の操作イベントによりモデルのスロット値が変化し（①②），それに対応してディスプレイオブジェクトが表示を変える（③④）．結果的に，利用者の操作に対して *Box* が反応することになる（⑤）．

3.2 スロット結合による機能合成

IntelligentBox では *Box* 間に親子関係を定義できる．親子関係が定義されている 2 つのボックス間でそれぞれのスロットを指定することにより，データの授受が行われ機能の合成が行われる．このデータ結合機構をスロット結合と呼ぶ．**図 3** に示すように，3 つのメッセージ①②③により，データの授受が行われ，スロット結合が実現される．①*set* メッセージにより，子 *Box* の指定されたスロットの値が親 *Box* の指定されたスロットに格納される．②*gimme* メッセージにより，親 *Box* の指定されたスロットの値が読み出され，子 *Box* の指定されたスロットへ格納される．③*update* メッセージにより，親 *Box* の状態変化がその子 *Box* に伝わる．このように，これら 3 つのメッセージによって，親 *Box* と子 *Box* のスロットが結合され，それぞれのボックスが持つ機能が合成される．

3.3 共有コピーと分散環境でのモデル共有

MD 構造では，複数の *Box* が同じ共通のモデルを共有してもよい．この機構をモデル共有と呼ぶ．モデル共有をしたディスプレイオブジェクトを複製することを共有コピーと呼ぶ．共有コピーにより生成された *Box* はすべてのスロットを共有する．

また，モデル共有されている *Box* の 1 つが別の計算機に転送されると，その計算機で対応する *Box* が自動的に生成される．そしてネットワークを介してメッセージが授受され，スロット値の一貫性が保持される．これにより，分散環境でのモデル共有が可能となる．

本論文で提案する「コピー＆トランスファー」は，*Box* の共有コピーを生成しそれを別の計算機に転送する操作であり，これにより分散環境でのモデル共有を実現する．

3.4 RoomBox による分散協調作業環境の構築

IntelligentBox には，分散協調作業環境を構築するために *RoomBox* [12] と呼ぶ *Box* がすでに提供されている．**図 4** は，*RoomBox* による分散協調作業環境の実現機構を示している．*RoomBox* は *event* というスロットを持つ．*RoomBox* の子孫 *Box* に対する利用者の操作イベントは，すべてこのスロットにいったん格納される．*RoomBox* の共有コピーを生成し，別の計算機に転送することにより，図 4 に示す構造が作られ，利用者の操作イベントが共有される．

4. 音声・動画像データ共有機構の設計と実装

前述のとおり，*IntelligentBox* では *RoomBox* を用

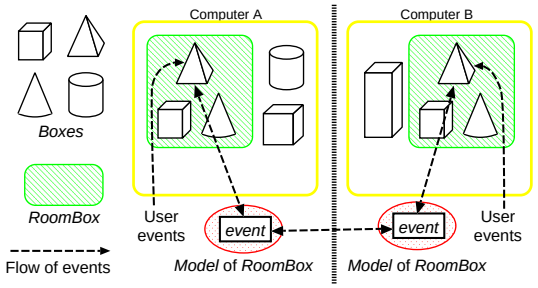

図 4 *RoomBox* の共有コピーによる分散協調作業環境
Fig. 4 Message flow between two *RoomBoxes* for network collaboration.

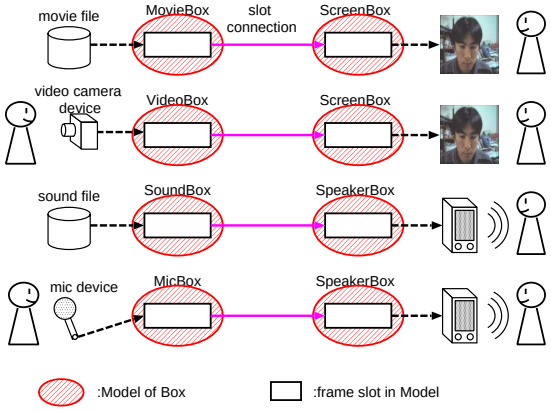

図 5 音声・動画像データを扱う *Box* とその組合せ
Fig. 5 Audio-video managing *Boxes* and their combinations.

いることで容易かつ即座に分散協調作業環境を構築可能である．*IntelligentBox* に音声・動画像データを扱う *Box* を追加することにより，分散協調作業環境における音声・動画像データの分散共有が可能となる．著者らは音声・動画像データを扱う以下の 6 つの *Box* を設計し，大半の実装を終了している．

- *MovieBox*：動画像ファイルから動画像データを入力する機能を持つ．
- *VideoBox*：ビデオカメラから動画像データを入力する機能を持つ．
- *ScreenBox*：計算機画面上に動画像データを出力する機能を持つ．
- *SoundBox*：音声ファイルから音声データを入力する機能を持つ．
- *MicBox*：マイクデバイスから音声データを入力する機能を持つ．
- *SpeakerBox*：スピーカデバイスを通して音声データを出力する機能を持つ．

IntelligentBox にはすでに，動画像を扱う *Box* として *MovieBox* と *VideoBox* があるが，これらには，動画像データの分散共有機構はない．「コピー&トランスファー」操作により動画像データの分散共有環境構築を可能とするため，*MovieBox* と *VideoBox* に対して，以下で説明する拡張を行った．

4.1 音声・動画像データを扱う Box の設計

前述のとおり，動画像を扱う部品として *MovieBox*, *VideoBox* と *ScreenBox*, 音声を扱う部品として *SoundBox*, *MicBox* と *SpeakerBox* の開発を行っている．これらを使用する際の組合せ例を**図 5** に示す．

このように，データを入力する機能を持つ部品とデータを出力する機能を持つ部品を分けて部品化することにより，それぞれの部品の再利用性が向上する．また，マイクやビデオカメラ，スピーカやスクリーンといった現実世界に存在するものと同じ機能単位で部品化することは，利用者にとって，分かりやすいといえる．動画像を扱う部品である *MovieBox*, *VideoBox* と *ScreenBox* の実装はすでに終えている．音声を扱う部品については，その設計を終え，実装の途中である．

4.2 動画像を扱う Box の構造

MovieBox と *VideoBox*, *ScreenBox* は動画像を扱う．*MovieBox* と *VideoBox* の違いは，入力源の違いである．*MovieBox* の場合は動画像ファイルである．*VideoBox* の場合はビデオカメラである．

これらの *Box* では，3 次元仮想空間上に 2 次元画像を表示するために，テクスチャマッピングを使用している．ファイルやカメラからの動画像データを，1 フレームごとにテクスチャイメージとして仮想空間内の物体に描画することで，動画像の表示を実現している．これらの *Box* には共通の設計として，このテクスチャイメージを格納する *frame* スロットが存在する．

前述のように，*IntelligentBox* には *MovieBox* と *VideoBox* が存在していたが，それらの設計では，テクスチャイメージは *Box* の *frame* スロットには格納されていなかった．*frame* スロットにはテクスチャイメージへのポインタが格納されていた．そのために *Box* の共有コピーを生成し別の計算機に転送しても，動画像通信環境を構築することできなかった．そこで，*frame* スロットにテクスチャイメージ自体を格納できるように *MovieBox* と *VideoBox* を改良した．

図 6 に，改良後の *MovieBox* と *VideoBox* の構造を示す．これらの *Box* はそのモデル内にテクスチャ画像を格納するための *frame* スロットを持つ．動画像ファイルやビデオカメラから入力された 1 フレーム分の画像データは，このスロットに格納される．さらに，ディスプレイオブジェクトへ送られることで *Box*

の表面に表示される．*MovieBox*，*VideoBox* は単独で使用することも可能であるが，*ScreenBox* と組み合わせて用いることにより，その応用範囲が広がる．

また，前述したとおり，共有コピーにより生成された *RoomBox* は，自分の子孫 *Box* で発生した利用者の操作イベントを共有することで，分散協調作業環境を提供する．ところが，*frame* スロットの更新イベントは操作イベントではない．そこで，*MovieBox* や *VideoBox* の *frame* スロットの更新イベントを操作イベントと同様に処理できるように改良した．

4.2.1 MovieBox

MovieBox はモデル内に次のスロットを持ち，これらの機能により動画像表示を実現している．

moviefile	再生する動画像ファイルの名前を保持する．
frame	画面に描画するフレーム画像を保持する．
TRIGGER	フレーム再描画の契機を促す．
currentNo	現在表示されているフレームのフレーム番号を格納する．
increment	次に描画すべきフレームが何フレーム後かを表す．

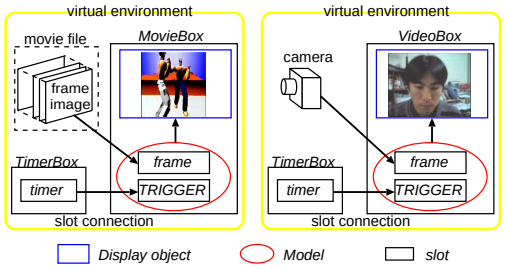

図 6 *MovieBox* と *VideoBox* の構造
Fig. 6 The mechanism of *MovieBox* and *VideoBox*.

図 6 左に，*MovieBox* に関するデータの流れを示す．

MovieBox が再生可能な動画像フォーマットには，QuickTime と AVI がある．動画像ファイルは，ファイルごとに画像の描画速度（frame per second）が異なる．そこで，*increment* を利用することで，動画の描画速度変化を実現する．*TRIGGER* がアクセスされるたびに，*currentNo* の現在の値と *increment* の値の合計が *currentNo* の次の値となり，フレームが更新される．*TRIGGER* への定期的なアクセスは，時間を扱う *TimerBox* により実現している．*TimerBox* は一定時間ごとに増加するタイマを持ち，その値は *timer* スロットにより参照可能である．*timer* と *TRIGGER* をスロット結合させることで，*timer* の値の変化が *TRIGGER* に伝わる．これにより一定の描画速度を得ることが可能である．

4.2.2 VideoBox

VideoBox はモデル内に次のスロットを持っており，それらの機能により動画像表示を実現している．

DriverNo	ビデオカメラのドライバのインデックスを保持する．
frame	画面に描画するフレーム画像を保持する．
TRIGGER	フレーム再描画の契機を促す．

図 6 右に，*VideoBox* に関するデータの流れを示す．

1 台の計算機にビデオカメラのドライバは複数存在可能であるため，*DriverNo* によりその 1 つを特定する．*TRIGGER* がアクセスされると，ビデオカメラから新しいフレーム画像を取得する．そしてそれを *frame* に格納する．*MovieBox* と同様に，*TRIGGER* の定期的なアクセスは *TimerBox* により実現している．

図 7 に，計算機 A のビデオカメラからの動画像と計

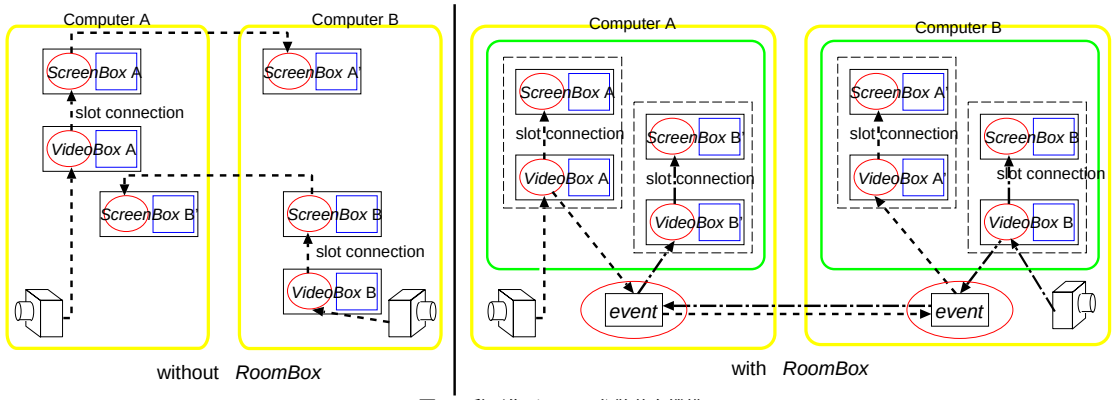

図 7 動画像データの分散共有機構
Fig. 7 Distributed video data sharing mechanisms.

算機 B のビデオカメラからの動画像をそれぞれ分散共有する例を示す．図 7 左は，*RoomBox* を用いない場合である．*ScreenBox* の「コピー&トランスファー」よって実現している．図 7 右は，*RoomBox* を用いる場合である．*VideoBox*（あるいは *MovieBox*）を *ScreenBox* と合成し，この合成 *Box* を，事前に「コピー&トランスファー」している *RoomBox* の子孫 *Box* とすることで実現している．

4.2.3 ScreenBox

ScreenBox はモデル内に *frame* スロットを持つ．*frame* スロットはテクスチャイメージを格納するスロットである．このスロット内のイメージが更新されると，そのイメージはディスプレイオブジェクトに受け渡され，テクスチャマッピングにより画面に表示される．

4.3 音声データを扱う Box の構造

SoundBox は音声ファイルから入力される音声データを自身に保持する．*MicBox* はマイクから入力される音声データを自身に保持する．*SpeakerBox* は自身が保持する音声データをスピーカから出力する．これらの *Box* は単独では利用せず，*SoundBox* と *SpeakerBox*，*MicBox* と *SpeakerBox* のように組み合わせて利用する．

これらの *Box* には音声データを扱うために共通な設計を行っている．それは，音声バッファと *frame* スロットである．**図 8** は音声バッファと *frame* スロットの関係を示している．音声バッファはデバイスから読み込んだデータを書き込んだり，あらかじめ書き込まれているデータをデバイスに書き込むために使用される．音声バッファとデバイスは 1 対の関係にある．*frame* スロットは，この音声バッファの一部をマッピングしたものである．*frame* スロットの大きさは固定長で，これはシステムから提供される．*frame* スロットへのマッピングの契機は *always* スロットや *TRIGGER* スロット（後述）により制御される．サンプリング周波数，量子化ビット数，チャネル（モノラル/ステレオ）に関しては，現在の設計では固定の値となっている．

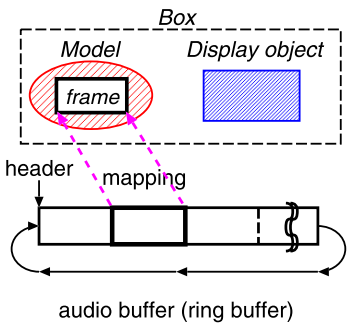

図 8 音声バッファと *frame* スロットの関係
Fig. 8 A relationship between an audio buffer and a *frame* slot.

4.3.1 SoundBox

SoundBox はモデル内に次のスロットを持つ．

soundfile	音声ファイルの名前を保持する．
frame	ファイルからの音声データを格納する．
always	音声データ格納の契機を非同期的に促す．
TRIGGER	音声データ格納の契機を同期的に促す．

再生可能なフォーマットには wave がある．*always* が真になると，音声ファイルからデータを音声バッファに書き込み続ける．*always* が偽になると，音声バッファへの書き込みを停止する．書き込みの最小単位は *frame* の大きさである．*always* が偽の場合には，*TRIGGER* のアクセスのたびにデータを音声バッファに書き込むこともできる．*always* と *TRIGGER* の違いは，音声バッファに書き込むタイミングである．*always* が真の場合は，音声バッファに連続して書き込みが行われるが，*TRIGGER* の場合は，そのアクセスに同期して音声バッファに書き込みが 1 度だけ行われる．*always* の状態の変更は，スイッチ機能を持つ *ToggleSwitchBox* により行う．*ToggleSwitchBox* は，真・偽の状態を持つ *State* スロットを持つため，この *State* と *always* とをスロット結合させればよい．*TRIGGER* へのアクセスは，*MovieBox* や *VideoBox* の場合と同様 *TimerBox* を利用する．

4.3.2 MicBox

MicBox はモデル内に次のスロットを持つ．

DriverNo	マイクのドライバのインデックスを保持する．
frame	マイクからの音声データを格納する．
always	音声データ格納の契機を非同期的に促す．
TRIGGER	音声データ格納の契機を同期的に促す．

1 台の計算機にマイクのドライバは複数存在可能であるため，*DriverNo* によりその 1 つを特定する．*always* または *TRIGGER* がアクセスさせると，マイクから入力された音声データを *frame* に格納する．音声データ格納の契機に関しては *SoundBox* と同様である．

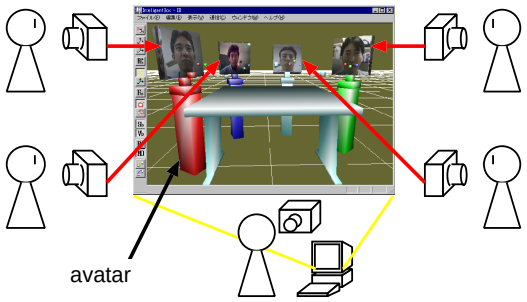

図 9 応用例：TV 会議システム
Fig. 9 A TV conference system example.

4.3.3 SpeakerBox

SpeakerBox はモデル内に次のスロットを持つ．

frame	スピーカデバイスへの音声データを格納する．
always	音声データ再生の契機を非同期的に促す．
TRIGGER	音声データ再生の契機を同期的に促す．

always または *TRIGGER* がアクセスされると，まず音声バッファに音声データを格納する．すると *frame* へのマッピングが起こり，その *frame* 内のデータをスピーカで再生する．音声データ格納の契機に関しては *SoundBox* や *MicBox* と同様である．

5. 考　　察

5.1 応　用　例

5.1.1 TV 会議システム

図 9 は，TV 会議システムの画面を示している．会議の参加者は，アバタとして表現されている．この例で示す構造は，各々の参加者が，自分のアバタの共有コピーを作成し他の計算機に転送するという処理を，他の参加者人数回繰り返すことで構築される．ここで用いているアバタの頭部は，*CameraBox* と *VideoBox* の合成 *Box* である．*CameraBox* によって視点を変えることができるため，各参加者は，自分のアバタの頭の位置を視点とした視野で，他の参加者を見ることができる．アバタの胴体は，マウスデバイスの操作により 3 次元空間を移動する *MoverBox* であり，各参加者は，自由に場所を変えて会議に参加することができる．

5.1.2 教育支援システム

図 10 は，教育支援システムの画面を示している．実行画面に表示されているオブジェクトはプリンタをモデリングした合成 *Box* である．*VideoBox* の機能により先生の顔が画面上に表示されている．これらの *Box* は，*RoomBox* の子孫 *Box* として定義されている．先生はこのプリンタの構造や操作方法を，実時間デモンストレーションにより生徒に教えることができる．

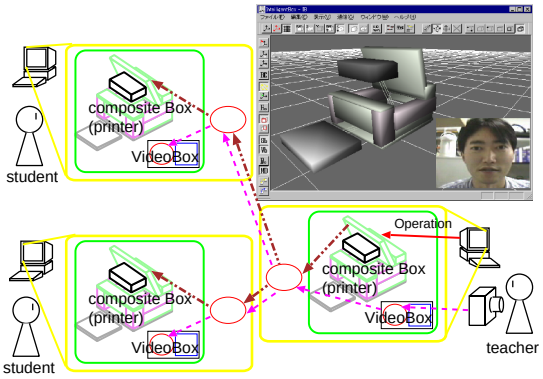

図 10 応用例：教育支援システム
Fig. 10 A tutoring system example.

5.2 開 発 手 順

VideoBox，*RoomBox*，そして *TimerBox* を用いて，最も単純な動画像通信環境の応用例を作成する手順は次のとおりである．

(1) *TimerBox* と *VideoBox* から合成 *Box* を作成する．
(2) *TimerBox* の *timer* スロットと *VideoBox* の *TRIGGER* スロットをスロット結合する．
(3) この合成 *Box* を *RoomBox* の子孫 *Box* にする．
(4) *RoomBox* の共有コピーを生成し（「コピー」），別の計算機に転送する（「トランスファー」）．

上の手順の (1)，(2) に関しては，あらかじめ合成 *Box* を作成してファイルに保存しておけば，その *Box* のファイルを読み込むだけで済む．音声通信環境の応用例に関しても，ほぼ同様の手順で作成できる．

このように，「コピー＆トランスファー」操作に基づく音声・動画像通信環境の構築過程は，利用者にとって非常に単純かつ容易である．前節であげた応用例は，マウスデバイスを使った画面上での直接操作のみにより開発でき，ソースコードの編集・作成によるプログラミングを行う必要はない．

5.3 性　　能

動画像通信の性能を評価するために，2 台の計算機上で *VideoBox* を単に「コピー＆トランスファー」した環境を作成し，実験を行った．実験環境は以下のとおりである．

- 計算機 A
  - CPU: Pentium III 800 MHz
  - Memory: 256 MB

  - Network: 100 Base-TX
  - OS: Windows 98
  - Graphics Card: Geforce
- 計算機 B
  - CPU: Pentium III 450 MHz
  - Memory: 256 MB
  - Network: 100 Base-TX
  - OS: Windows NT
  - Graphics Card: Cobalt

パラメータとして，次のものがある．

- 単方向通信か双方向通信か
- フレームレート（frame/sec）
- 解像度（縦 pixel 数 × 横 pixel 数）
- 色数（byte/pixel）

以上の実験環境でパラメータを変化させながら *VideoBox* のフレームレートを測定する．**表 1** に結果を示す．たとえば，左下の数字 2.5 とその下の 0.2 は，次の設定におけるフレームレートを意味している．双方向通信（bi-direction）で，解像度は 128×128，色数は 4 byte/pixel（RGB α 各 1 byte，α はアルファチャネル）の場合である．前述のとおり，共有コピーで生成された *Box* では，スロット値の更新は同期して行われる．そのためにフレームレートは共有コピーと同じ値となる．2.5 は，計算機 A から B への通信におけるフレームレートであり，0.2 は，逆に，計算機 B から A への通信におけるフレームレートである．この差が大きいのは，計算機の性能が B より A の方が非常に高いためである．CPU 資源を多く必要とする処理を複数の計算機で協調して行う場合，性能の高い計算機から出される要求が処理される割合が一方的に高くなると考えられるからである．いずれにしても，この表の数値は良い性能を示してはいるとは限らない．しかし，少なくとも，これらのフレームレートで双方向通信が可能であることを実証している．すなわち，著者らの提案する「コピー&トランスファー」操作により，動画像データの分散共有環境を構築可能であることを実証している．データの圧縮技術や特別な通信プロトコルを採用することにより，通信性能の向上を図ることは技術的に可能であり，それは今後の実装上の課題である．

表 1 *VideoBox* のフレームレート
Table 1 Framerate of *VideoBox*.

resolution (pixel) depth (byte/pixel)		128 x 128 4	128 x 128 1	64 x 64 4	64 x 64 1
stand-alone	A	18.0	18.0	18.0	18.0
	B	6.6	6.6	6.1	6.1
uni-direct	A -> B	2.6	18.1	10.3	18.1
	B -> A	0.3	0.9	0.9	3.4
bi-direct	A -> B	2.5	17.0	10.3	18.0
	B -> A	0.2	0.8	0.9	3.1

frame rate (frame/sec)

6. おわりに

本論文では，分散協調作業空間構築のための音声・動画像データの分散共有機構について述べた．特に，「コピー&トランスファー」と呼ぶ概念を提案し，「コピー&トランスファー」操作により音声・動画像データの分散共有環境が容易に構築可能であることを述べた．そしてその実現機構を示し，有用性を明らかにした．

もしソフトウェア部品が画面上で直接操作可能な可視部品として表現されているならば，その複製を作成したり，それを別の計算機へ転送したりという作業が，画面上の操作のみでできるようになる．すなわち，「コピー&トランスファー」操作により，部品の複製を作成しそれを別の計算機へ転送可能となる．この際，別の計算機に転送された部品の複製が元の部品と通信し，つねに同一の状態を保つように設計されていれば，この部品は，分散環境で共有されていると見なすことができる．したがって，音声や動画像データを扱う可視部品を導入することにより,「コピー&トランスファー」操作のみにより，音声・動画像通信の分散共有が可能となる．3 次元ソフトウェア開発システムである *IntelligentBox* を研究基盤システムとして用い，これに，音声・動画像データを扱う可視部品を導入した．そして，本論文で提案する「コピー&トランスファー」操作により，音声・動画像通信の分散共有環境が構築可能であることを実証した．また，応用例を示し，開発手順と性能について述べた．

今後の課題として，まず *SoundBox*, *MicBox* と *SpeakerBox* の実装を完了することがあげられる．そしてこれらを用いた音声・動画像通信機能のある応用システムを実際に構築し実証実験を行う．たとえば，応用システムとしては，ネットワーク対戦型ゲームやネットワークを介した教育用システムなどがあげられる．

謝辞 日頃の研究活動において様々な助言をいただきました研究室の皆様，また，論文を書くうえで多くのご助言をいただきました九州大学情報基盤センターの伊東栄典助教授に，つつしんで感謝の意を表します．

参考文献

1) Sakamoto, H., Okada, Y., Shimokawa, T. and Ushijima, K.: Component Based Video Communication Tool for Collaborative Virtual En-

vironment, *Proc. 15th International Conference on Information Networking*, pp.375–380 (2001).
2) 坂本弘達，岡田義広，下川俊彦，牛島和夫：3次元分散協調作業空間構築のためのビデオデータの分散共有フレームワーク，情報処理学会第62回全国大会講演論文特別トラック1， pp.37–40 (2001).
3) Okada, Y. and Tanaka, Y.: IntelligentBox: A Constructive Visual Software Development System for Interactive 3D Graphic Applications, *Proc. Computer Animation '95*, pp.114–125, IEEE Computer Society Press (1995).
4) Hagsand, O.: Interactive Multiuser VEs in the DIVE System, *IEEE Multimedia*, Vol.3, No.1, pp.30–39 (1996).
5) Frecon, E. and Nou, A.A.: Building Distribute Virtual Environments to Support Collaborative Work, *ACM VRST'98*, pp.105–113 (1998).
6) Greenhalgh, C. and Benford, S.: MASSIVE: A Collaborative Virtual Environment for Teleconferencing, *ACM Trans. Computer-Human Interaction*, Vol.2, No.3, pp.239–261 (1995).
7) Ghee, S.: dVS — A Distributed VR System Infrastructure, *SIGGRAPH '95 CourseNotes* (1995).
8) Anderson, D.B., Barrus, J.W., et al.: Building Multiuser Interactive Multimedia Environments at MERL, *IEEE Multimedia*, Vol.2, No.4, pp.77–82 (1995).
9) Barrus, J.W., Waters, R.C. and Anderson, D.B.: Locals and Beacons: Efficient and Precise Support For Large Multi-User Virtual Environments, *Proc. IEEE Virtual Reality Annual Int. Symp.* (*VRAIS-96*) (1996).
10) Shaw, D., Green, M., Liang, J. and Sun, Y.: Decoupled Simulation in Virtual Reality with the MR Toolkit, *ACM Trans. Information Systems*, Vol.11, No.3, pp.287–317 (1993).
11) Tanaka, Y.: Meme Media and a World Wide Meme Pool, *Proc. ACM Multimedia'96*, pp.175–186 (1996).
12) Okada, Y. and Tanaka, Y.: Collaborative Environments in IntelligentBox for Distributed 3D Graphics Applications, *The Visual Computer* (*CGS special issue*), Vol.14, No.4, pp.140–152 (1998).

(平成13年 6 月 7 日受付)
(平成13年12月18日採録)

坂本 弘達

2000年九州大学工学部電気情報工学科卒業．同年同大学大学院システム情報科学府情報工学専攻修士課程入学．音声・動画像通信，仮想空間等の研究に従事．

岡田 義広（正会員）

1993年北海道大学大学院工学研究科電気工学専攻博士課程修了．博士（工学）．1993年4月同大学工学部電気工学科助手．1997年4月同大学大学院工学研究科電子情報工学専攻助手．1999年1月九州大学大型計算機センター助教授．2000年4月同大学大学院システム情報科学研究院助教授．3次元グラフィックス，ソフトウェアアーキテクチャ，分散システム，仮想現実感等の研究に従事．電子情報通信学会，日本ソフトウェア科学会，ヒューマンインタフェース学会，ACM，IEEE各会員．

下川 俊彦（正会員）

1990年九州大学工学部情報工学科卒業．1992年同大学大学院工学研究科情報工学専攻修士課程修了．同年（株）東芝入社．1997年九州大学大学院システム情報科学研究科情報工学専攻助手．2000年同大学大学院システム情報科学研究院情報工学部門助手．博士（情報科学）．広域分散協調処理，インターネット等の研究に従事．

牛島 和夫（正会員）

1937年生．1961年東京大学工学部応用物理学科（数理工学コース）卒業．1963年九州大学工学部講師．1977年同教授（情報工学科計算機ソフトウェア講座）．1996年九州大学大学院システム情報科学研究科長併任．2001年九州大学停年退職．同年（財）九州システム情報技術研究所長．同年4月から本会アクレディテーション委員会委員長．電子情報通信学会，日本ソフトウェア科学会，ACM，IEEE各会員．