3次元仮想空間における空間のコンポーネント化の実現

伊藤正彦[†,☆] 田中譲[††,†††]

さまざまなメディアを扱うことのできる3次元のコンポーネントウェアを利用して，計算機上の空間を利用する際に，ユーザの扱うオブジェクトの量は膨大なものとなる．そのさい，限られたディスプレイ空間を効率良く利用する工夫が必要とされる．また，3次元オブジェクト空間をWWWのような情報へのアクセス空間として利用することが考えられる．このとき，3次元空間に情報アクセスのための仕組みを導入する必要がある．本論文では，3次元空間自体をコンポーネントとして扱うことで，これらの要求に対応する手法を提案する．このコンポーネントをWorldMirror，WorldBottleと呼ぶ．WorldMirror，WorldBottleはそれぞれ平面形状，立体形状のオブジェクトとして3次元空間中で表される．これらは，他の空間のシーンをオブジェクト内部に写し込み，中に入ることで作業空間の切替えを可能にする．これらにより，3次元空間にマルチ・ウィンドウ・システムのような仕組みを導入できる．また，コンポーネントとして扱うことで，他のコンポーネントとの機能連携を実現する．

WorldMirror and WorldBottle: Components for Embedding Multiple Spaces in a 3D Virtual Environment

Masahiko Itoh[†,☆] and Yuzuru Tanaka[††,†††]

In this paper we propose and demonstrate mechanisms that address efficient access to large number of information objects within a restricted 3D display space. Our mechanisms allow us to embed multiple 3D spaces in a single virtual environment, and enable us to navigate through these different spaces. An embedded space can be represented either as a flat, window-like WorldMirror or as a spherical WorldBottle; in either case, the user can see the contents of the embedded space from outside, and jump into the embedded space to change his or her current working context. We describe here how these components can be implemented in the IntelligentBox architecture, a constructive and visual software-development system architecture for interactive 3D graphical applications.

1. はじめに

近年の計算機を取り巻く環境の発達は目覚ましいものがある．これは，(1) ハードウェアの発達とグラフィックス技術の進歩，(2) ネットワーク環境の発達とインターネットの普及，(3) コンポーネントウェア技術の発達などによりもたらされたものである．このような環境の発達は，ユーザが自分の計算機環境にネットワーク上に分散しているデータやコンポーネントなどの資源をも取り込み，自由に扱うことを可能にした．このため，ユーザが扱える情報の量は膨大なものとなってきている．さまざまなメディアを扱うことのできる3次元のコンポーネントウェアを利用して，計算機上の空間をインタラクティブなメディア空間として活用する際にも，ユーザの扱うオブジェクトの量は膨大なものとなりうる．

これら膨大な量のオブジェクトを3次元オブジェクト空間で扱うために，(1) 限られた作業空間を効率良く利用する手法とともに，流通オブジェクトに対する管理，編集，再流通を行うために，(2) それらのオブジェクトを組織化し，効率の良い情報探索とアクセスを行うための仕組みを3次元オブジェクト空間へ導入する必要がある．

3次元オブジェクト空間において，上記の2つの要求を満たすような空間を構築するには，空間構築コンポーネントを，それ自体もまたメディア・オブジェクトとして3次元空間に導入することが望ましい．

† 北海道大学工学研究科電子情報工学専攻
Graduate School of Engineering, Hokkaido University
†† 北海道大学工学研究科電子情報工学専攻
Graduate School of Engineering, Hokkaido University
††† 北海道大学知識メディアラボラトリー
Meme Media Laboratory, Hokkaido University
☆ 現在，株式会社ビー・ユー・ジー
Presently with B.U.G., Inc.

システムのユーティリティ機能として情報管理とアクセスを試みたものや，ユーザインタフェース部品を用いて効率の良い空間利用と情報アクセスを試みたシステムはすでに多数存在する[3)~5)]．従来システムにおけるこれらの機能はシステム開発者によりシステム内にあらかじめ組み込まれたものである．一方，インタラクティブなコンポーネントウェアでは，空間の効率の良い利用と情報アクセスを行うためのしくみを，エンドユーザが他のコンポーネントと同様に利用しうるコンポーネントの形で，導入されることが望まれる．

本論文では，空間自体を空間構築コンポーネントとして部品化し，3 次元メディア・オブジェクトとして 3 次元オブジェクト空間中に持ち込むことを提案する．

以降，本論文では空間を，空間名と座標を持ち，そこにおいて描画されるオブジェクトのリストを持つ 3 つ組として定義する．一方，任意の空間が与えられたとき，その空間に存在するすべてのオブジェクトの表示を行うコンポーネントを空間コンポーネントと定義する．

上記の空間コンポーネントを実現するためには，空間を埋め込む 3 次元形状オブジェクトの内部へその空間のシーンを描画する機能，その埋め込まれた空間内へユーザが移動する機能，ユーザがいる空間と，そこに埋め込まれている別の空間との間でオブジェクトを出し入れする機能，空間コンポーネントと他のコンポーネントとの機能連携により，空間コンポーネントやそこに埋め込まれた空間を制御する機能，空間を組み合わせる機能，新しい空間を登録する機能などが必要になる．

これらの機能を持つコンポーネントを WorldMirror[12)]，WorldBottle[13)] と呼ぶコンポーネント・オブジェクトとして 3 次元空間に導入した．これらは，2 次元ディスプレイの場合のマルチ・ウィンドウ・システムにおけるオーバラッピング・ウィンドウに相当し，異なる作業空間を，同時に 1 つのディスプレイの表示空間で扱うことを可能にする．そして，それらの中に他の作業空間を写し込むことでディスプレイの表示空間を仮想的に広く利用することが可能になる．さらに，それら個々の空間を出入りする機能を用意することで，空間を渡り歩くことが可能になる．また，空間をコンポーネント化することにより，他のコンポーネントと機能連携が可能になる．

本論文では，空間を表すコンポーネントである WorldMirror と WorldBottle について，2 章でその概要を述べ，3 章で実現機構を述べる．4 章では WorldMirror/Bottle を用いた応用例を述べる．

2. 空間の埋め込みとコンポーネント化のフレームワーク

限られたディスプレイ空間を効率良く利用し，関連情報へのアクセスを行うための空間コンポーネントとして WorldMirror，WorldBottle を実現するには，以下の機能が必要である．

(1) 空間のオブジェクト化．
(2) オブジェクト内へのシーンの映し込み．
(3) 空間の分割と管理．
(4) 他の空間への出入り口．
(5) 空間の間での情報の出し入れ．
(6) 空間の組合せ．
(7) 空間コンポーネントと他のコンポーネントとの機能連携．
(8) (3)〜(7) の直接操作による実現．

(1) により，空間をオブジェクトとして取り扱うことが可能になり，ユーザによる直接操作が可能になる．(1) の「空間のオブジェクト化」には，(2) の「オブジェクト内へのシーンの映し込み」の機能が必要となる．これにより，他の空間（ターゲット空間）のシーンを現在いる空間（ソース空間）内のオブジェクト内に映すことが可能になる．ある空間の中で別の空間をコンポーネントとして扱うために，(3) の「空間の分割と管理」を行う機能が必要となる．また，作業空間の切替えを行ったり，空間を渡り歩いたりするために (4) の「他の空間への出入り口」としての機能が必要である．また，2 次元マルチ・ウィンドウ・システムにおけるファイルの移動に相当する操作を行うために (5) の「空間の間での情報の出し入れ」を行う機能が必要である．空間の中に複数の空間を入れたり，空間コンポーネントの中にさらに空間を埋め込むのに (6) の「空間の組合せ」を行う機能が必要となる．(7) の「空間コンポーネントと他のコンポーネントとの機能連携」を行う機能により，たとえば空間を表すコンポーネントとスライドメータとを合成することで，空間を映し込むサイズをスライドメータの直接操作により変更することが可能になる．インタラクティブなメディア空間をコンポーネントとして扱うためには (8) の機能が必要である．

2 次元空間で，空間自体をオブジェクトとして扱い，他の空間をオブジェクト中に映し込んだシステムとしては，GMW ウィンドウ・システム[16)] がある．これは，「空間コンポーネントと他のコンポーネントとの機能連携」の機能を持たない．上記の 8 つの機能について，**表 1** に他の 3 次元システムと WorldMirror/Bottle

表 1 空間のコンポーネント化に必要とされる機能の有無
Table 1 Provision of facilities needed for expressing spaces as component objects.

	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)
The Virtual Tricorder	○	○	×	×	×	×	×	×
Flat Lenses	○	○	×	×	×	×	×	×
WIM	○	○	○	×	×	?	×	?
Worldlets	○	△	○	○	×	○	×	×
Volumetric Lenses	○	○	×	×	×	×	×	×
Locales and Beacons	×	×	○	○	○	○	×	×
VRML	○	×	○	○	×	×	×	×
Information Visualizer	×	×	○	○	×	×	×	×
WorldMirror	○	○	○	○	○	○	○	○
WorldBottle	○	○	○	○	○	○	○	○

○：有 ×：無 △：制限付きで有 ?: 不明
(1)：空間のオブジェクト化
(2)：オブジェクト内へのシーンの映し込み
(3)：空間の分割と管理
(4)：他の空間への出入り口
(5)：空間の間での情報の出し入れ
(6)：空間の組合せ
(7)：空間コンポーネントと他のコンポーネントとの機能連携
(8)：(3)〜(7) の直接操作による実現

との比較を示す．The Virtual Tricorder [1),11)] と 3D Magic Lenses システム[10)] における Flat Lenses，は 3 次元仮想空間中に 2 次元平面のインタフェースを持ち込んだシステムである．これらは，「空間のオブジェクト化」と「オブジェクト内へのシーンの写し込み」の機能のみを持つ．WIM（World In Miniture）[9)]，Worldlets [6)] および 3D Magic Lenses システムにおける Volumetric Lenses は 3 次元仮想空間中に 3 次元形状で他のシーンを映し込んだシステムである．Volumetric Lenses は Flat Lenses と同じく「空間のオブジェクト化」と「オブジェクト内へのシーンの写し込み」の機能のみを持つ．WIM は，これらの機能に「空間の分割と管理」を加えた機能を持ち，Worldlets では，さらに「他の空間への出入り口」と「空間の組合せ」の機能が加わるが，「空間の間での情報の出し入れ」と「空間コンポーネントと他のコンポーネントとの機能連携」の機能は持たない．Locales and Beacons [2)] は，バーチャルリアリティ空間における空間分割を扱ったシステムである．これは，「空間のオブジェクト化」と「オブジェクト内へのシーンの写し込み」と「空間コンポーネントと他のコンポーネントとの機能連携」の 3 つの機能を持たない．ほかに，VRML（The Virtual Reality Modeling Language）と Information Visualizer を比較対象としてあげる．

2.1 WorldMirror

WorldMirror は，可視的でユーザによる直接操作可能な平面コンポーネントとして空間中に埋め込まれた 3 次元仮想空間である．WorldMirror は**図 1** のように計算機内の複数の 3 次元空間を結ぶための，出入り口，および覗き窓の機能を持つ．この WorldMirror の中にはターゲット空間が映し込まれる．ただし，WorldMirror は平面オブジェクトなので WorldMirror の裏からは空間を覗くことはできない．ユーザは WorldMirror の中に入ることにより，空間の移動ができる．さらに，ターゲット空間に任意のオブジェクトを移動することができる．これにより，ターゲット空間に他の WorldMirror や WorldBottle を埋め込むことも可能である．

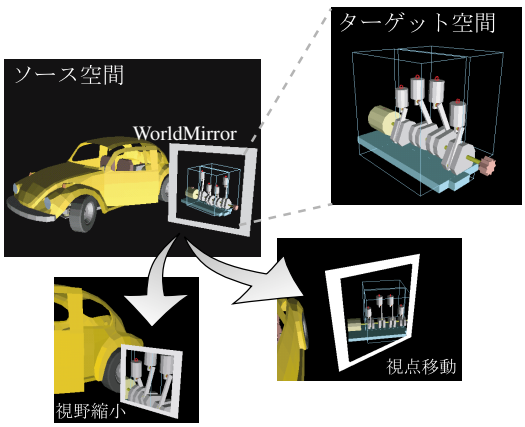


図 1 WorldMirror によるターゲット空間の写し込み
Fig. 1 Reference to another space using a WorldMirror.

2.2 WorldBottle

WorldBottle は，可視的でユーザによる直接操作可能な立体コンポーネントとして空間中に埋め込まれた 3 次元仮想空間である．WorldBottle は 3 次元のオブジェクト自体にターゲット空間を閉じ込めた表現をとる．WorldBottle もまた，**図 2** のように，計算機内の複数の 3 次元空間を結ぶための出入り口の機能と，覗き窓としての機能を提供する．WorldBottle は立体オブジェクトなのでユーザは任意の角度から空間を覗く

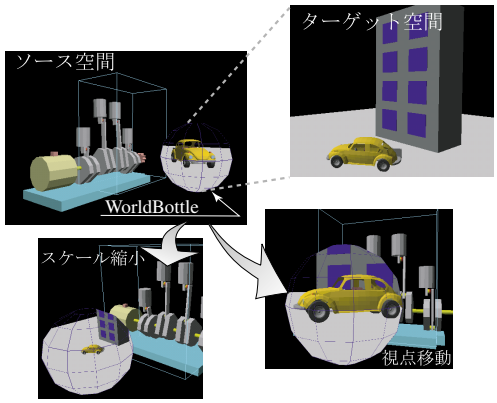


図 2 WorldBottle によるターゲット空間の写し込み
Fig. 2 Reference to another space using a WorldBottle.

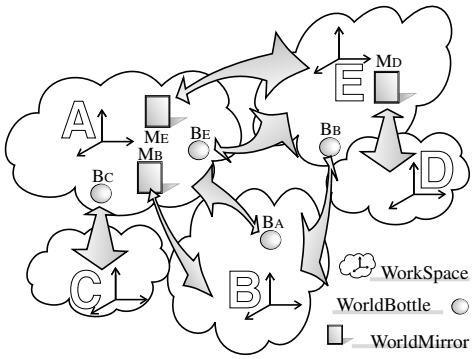


図 3 WorldMirror および WorldBottle による複合空間
Fig. 3 The nesting of spaces using WorldMirrors and WorldBottles.

ことができる．また，ターゲット空間に任意のオブジェクトを移動できる．さらに，他のコンポーネントとの機能連携を定義することにより他のコンポーネントから制御を行い，オブジェクト内に映されるターゲット空間のスケールなどの種々のパラメータを変更することもできる．

2.3 複合空間の実現

複数の空間を 1 つの 2D ないし 3D の空間に配置することを，空間の折り畳みと呼ぶ．折り畳まれた空間の中には別の空間を折り畳んで置いておくことができる．このように入れ子状にされた空間を複合空間と呼ぶ[15]．

2 つの空間をつなぐための道具である WorldMirror/Bottle を用いることで，複合空間を実現できる．このように複数の空間どうしをつなぐことにより，3 次元空間はハイパーリンク構造をとる．これにより，**図 3** に示すように 1 つの空間を他の複数の WorldMirror/Bottle の中に映し込むことも許される．この際，これらのうちの 1 つの WorldMirror/Bottle に入ってターゲット空間で行った操作は他の WorldMirror/Bottle にも反映される．

3. 実現アーキテクチャ

著者らは，基盤システムとして 3 次元のコンポーネントウェアである IntelligentBox[8),14)] を用い WorldMirrorBox と WorldBottleBox を開発することにより，IntelligentBox のユーザ環境を空間コンポーネントに導入した．(1)，(7) の機能は IntelligentBox を用いることで実現した．さらに，(2) の機能には OpenGL に用意されているステンシルバッファを利用した．(3)，(4)，(5)，(6) の機能は，WorldMirrorBox および WorldBottleBox の機能として実現した．また，(8) の機能は WorldMirrorBox，WorldBottleBox をインタラクティブな 3 次元システムである IntelligentBox のコンポーネントとして提供することで実現した．

表 2 表 1 にあげた機能の実現に必要とされるスロットリスト
Table 2 The list of required slots for WorldMirror/Bottle.

スロット名☆	種別	機能
#SpaceName	◎	ターゲット空間名の保持
#SpacePosition	◎	ターゲット空間座標の保持
#Size	◎	スケール変換に用いるパラメータ
#Enter	◎	ユーザ移動のトリガ
#ObjEnter	◎	オブジェクト移動のトリガ
#CenterP	◎	視点制御手法の切替え
#RealDistance	△	ステレオモードにおける視差
#StopP	△	ターゲット空間の視点固定
#Fovy	△	パースを変える
#Alpha	△	内部の透明度を変える
#AlphaFocus	△	透明度のフォーカスを変える
#BGColor	△	ターゲット空間の背景色を変える

◎：すべての空間コンポーネントが持つスロット
△：特殊効果を与える空間コンポーネントが持つスロット

3.1 IntelligentBox

IntelligentBox において，すべてのコンポーネントはボックスと呼ばれる 3 次元形状を持った機能オブジェクトで表される．そして，それらのボックスに親子関係を与え組み合わせ，スロット結合と呼ばれる機能合成を行うことで合成オブジェクトを生成できる．これらはすべて画面上での直接操作により行われる．各ボックスはスロットと呼ばれるデータの受け渡し口を持つ．このスロット中にそれぞれのボックスの持つ状態値を保持し，各ボックスのスロットを結びつけスロット結合することで状態値の受け渡しを行う．

☆ 本論文では，スロットの名前を *#slotname* のように表記することにする．

表 3 空間描画手法の比較
Table 3 The comparison of rendering method between 3D Magic Lenses with WorldMirror/Bottle.

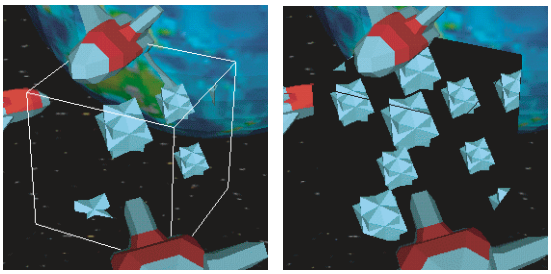
	(a) 3D Magic Lenses（クリップ面の利用）	(b) WorldMirror/Bottle（ステンシルバッファの利用）
1 シーンあたりの描画回数	8 回（クリップ面；6 面の場合）	2 回
描画アルゴリズム	クリップ面の追加により指定オブジェクト範囲を切り抜き，ターゲット空間を描画．さらに，クリップ面を反転しオブジェクト範囲外を切り抜く．	ステンシルバッファによりオブジェクトをマスクと指定し，その範囲にターゲット空間を上書き（図 5）．
効果	ターゲット空間の描画をオブジェクト内だけに限定（図 4 (a)）	オブジェクト内におけるターゲット空間の広がり表現（図 4 (b)）．

3.2 必要なスロット

表 1 に示した機能を WorldMirrorBox と WorldBottleBox で実現するために最低限必要となるスロットを**表 2** にあげる．また，WorldMirror/Bottle に特殊な効果を与えるスロットも同時にあげる．機能をスロットとして持つことで WorldMirror/Bottle と外部コンポーネントとの機能連携が実現できる．機能連携としては 2 種類ある．1 つは，WorldMirror/Bottle と別コンポーネントとの機能連携である．これにより，外部コンポーネントから WorldMirror/Bottle の制御が可能になる．また，外部コンポーネントによる状態の確認も可能になる．他の 1 つは WorldMirror/Bottle どうしの機能連携である．たとえば，複数の WorldMirror/Bottle 間で空間名を保持する *#SpaceName* のスロット結合を行うことで，ある 1 つの WorldMirror/Bottle に行われた空間の変更を，他の連携付けされたすべての WorldMirror/Bottle に同時に反映させることが可能になる．この際，複数の WorldMirror/Bottle に反映された空間の実体は 1 つであり，複数のビューアを用いて同時に見ているにすぎない．

3.3 ターゲット空間の映し込み

空間は空間名 *s-name* と空間の中心座標 *s-coord* および，そこにおいて描画されるオブジェクトのリスト *o-list* の 3 つ組（*s-name*，*s-coord*，*o-list*）で定義される．「中心座標」とは，ターゲット空間のローカル座標系の原点がシステムの基準座標系内でどこに位置しているかを示す位置座表である．本論文では，座標系としてシステムの基準座標系が存在し，各空間の中心座標は，基準座標系が定義する空間内において十分に離れた座標点として与えられるように実装を行っている．この手法に関する問題点とその解決法に関しては 5 章で触れる．IntelligentBox システムによってディスプレイ中に描画された WorldMirrorBox，WorldBottleBox はそれぞれ自身のターゲット空間に関する情報を *#SpaceName* と *#SpacePosition* に保持している．また，IntelligentBox システムはソース空間に関する情報を保持している．そして，システムによる描画のたびに WorldMirrorBox，WorldBottleBox は



(a) クリッピングを用いた描画 (b) ステンシルバッファを用いた描画

図 4 空間描画手法の比較 (2)
Fig. 4 The comparison of two different rendering methods for viewing another space (2).

システムからソース空間に関する情報として座標情報を取り出す．同時に，WorldMirrorBox，WorldBottleBox の持つ独自の描画ルーチンが呼び出され，自身の領域内にターゲット空間を描画する．

3.3.1 描画アルゴリズム

WorldMirror/Bottle が自身の領域内にターゲット空間を描画する手法には 2 通りの方法がある．1 つは，3D Magic Lenses などで用いられたクリップ面を用いる手法である．本論文で用いた OpenGL の環境では，OpenGL がサポートするクリップ面の最大数は 6 面である．OpenGL ではクリップ面として指定できる平面の数は，少なくとも 6 面保証しており，OpenGL の実装によってはそれ以上サポートするものもある．つまり，この動作環境においてクリップ面を用いた手法で描画を行うと，WorldBottle として利用できる形状は面の数が直方体以下の形状に制限されることになる．もう 1 つは，本論文で用いたステンシルバッファを用いる手法である．ステンシルとは「孔版」であり孔の開けてあるところだけに描画を制限する仕組みである．

上記 2 手法の比較を**表 3** に示す．クリップ面を用いる手法の場合，描画が形状内だけに限定されるため，WorldBottle 内に埋め込まれる空間の広がりをうまく表現できない．**図 4** では立方体のオブジェクト内に多数の星型のオブジェクトが配置された空間を描画している．クリップ面を用いる手法の場合，立方体の形状

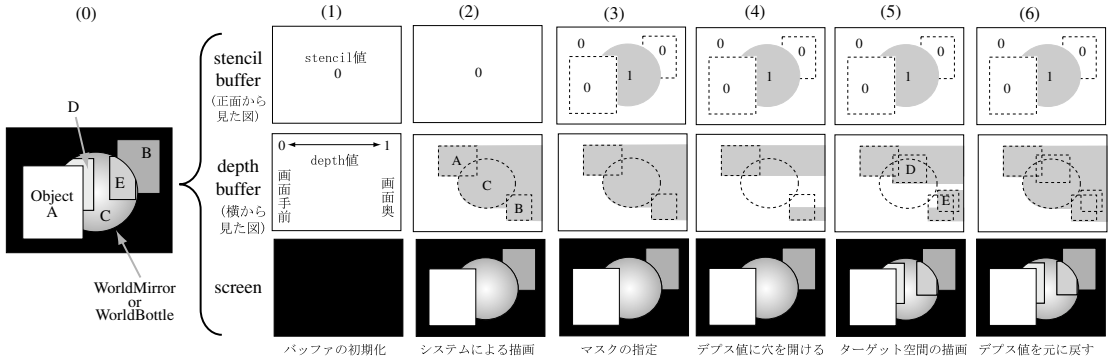
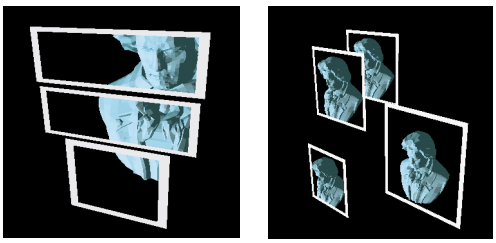


図 5 オブジェクト内へのシーン描画手順
Fig. 5 The procedure for rendering objects in an embedded space.

内に描画が限定されるため，手前と奥の星が切り取られて描画されていない．一方，ステンシルバッファを用いる手法の場合，奥行きのある空間が表現されている．また，クリップ面を用いる手法では，1 シーンあたりの描画回数がクリップ面の数に応じて増えることになりシステム全体に大きな負荷をかける．

図 5 にステンシルバッファを用いた空間コンポーネントの描画手順を示す．ソース空間に Object A，B，C があり，WorldMirror/Bottle である Object C の内部にターゲット空間の Object D と E を描画する．なお，図における横列は上段がステンシルバッファにおけるステンシル値の遷移を示し，中段がデプスバッファにおけるデプス値の最大値の遷移を示し，下段が実際のスクリーン上への描画状態を示す．コンポーネントの描画は図 5 より，(1) バッファが初期化され，(2) ソース空間中のすべてのオブジェクトの描画が行われる．次に (3) ステンシルバッファで WorldMirror/Bottle 内の描画領域を決定するマスクの範囲が指定されるが，指定されるマスクの範囲は，デプス値の小さな（スクリーン手前にある）WorldMirror/Bottle の面が優先的にマスクの範囲として指定される．したがって，WorldMirror や WorldBottle がソース空間中に複数存在し，それらがスクリーン上で重なっている場合，スクリーン手前にある WorldMirror/Bottle のターゲット空間が奥にある WorldMirror/Bottle のターゲット空間より上に描画される．次に (4) ターゲット空間内のオブジェクトの前後関係をソース空間における前後関係と切り離すため，マスク内のデプス値をクリアし，(5) オブジェクトの描画を行う．最後に (6) デプス値をソース空間の値に戻すことで，1 つの WorldMirror/Bottle におけるターゲット空間の描画を完了する．

また，本論文では WorldMirror/Bottle の中にさらに WorldMirror/Bottle が描画されるシーンを描画する場合，(1) ループを避け，(2) システムへの負荷を制限するために別空間の描画は 1 階層までに制限し，問題を単純化して扱っているが，ループ検出を行えば，WorldMirror/Bottle の中に WorldMirror/Bottle が描画されるシーンの描画も可能になる．



(a) 座標系一致方式 (b) センタリング方式

図 6 WorldMirror における視点制御
Fig. 6 The two different types of viewpoint control for a WorldMirror.

3.3.2 視点制御

WorldMirror で他の空間を見る際，問題となるのはユーザがその空間のどこを見たいのかということである．本論文では 2 種類の手法を用意した．

座標系一致方式 座標系一致方式では，ソース空間のローカル座標系とターゲット空間のローカル座標系のいずれにおいてもオブジェクトが同じ相対位置にあるように空間の描画を行う．これにより，**図 6** (a) のように WorldMirror の位置に応じてターゲット空間の見える範囲も変化する．これにより，実世界で窓から外を見るのと同じ視界の変化でターゲット空間を見ることができる．ターゲット空間の描画における視界の決定には，式 (1)，(2) を用いる．

$$\overrightarrow{tc} = \overrightarrow{tp} + (\overrightarrow{sc} - \overrightarrow{sp}) \quad (1)$$

$$\overrightarrow{te} = \overrightarrow{tc} + (\overrightarrow{se} - \overrightarrow{sc}) \quad (2)$$

ここで変数 $\overrightarrow{tp}$，$\overrightarrow{sp}$，$\overrightarrow{sc}$，$\overrightarrow{se}$，$\overrightarrow{tc}$，$\overrightarrow{te}$ は基準座標系で

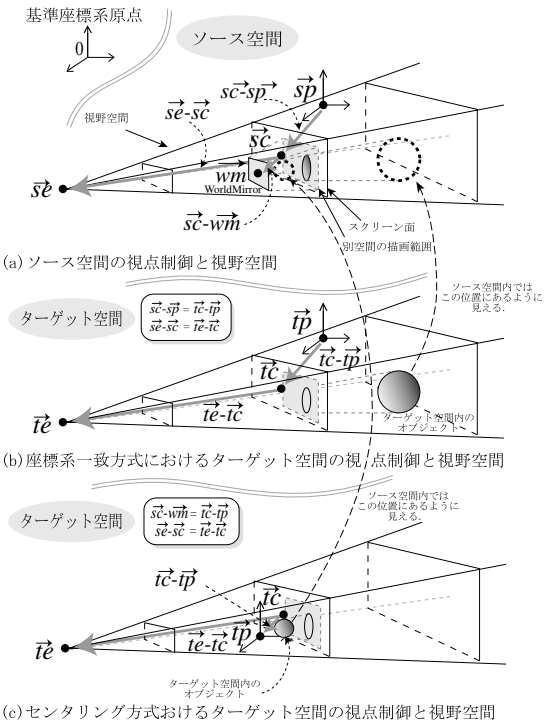


図 7 WorldMirror における視点制御詳細図
Fig. 7 Viewpoint control for a WorldMirror.

与えられ，以下のように定義される（**図 7** (a)，(b)）．

$\overrightarrow{tp}$ ：基準座標系におけるターゲット空間の原点の位置ベクトル．

$\overrightarrow{sp}$ ：基準座標系におけるソース空間の原点の位置ベクトル．

$\overrightarrow{sc}$ ：ソース空間の描画に用いるスクリーン面の中心点の基準座標系における位置ベクトル．

$\overrightarrow{tc}$ ：ターゲット空間の描画に用いるスクリーン面の中心点の基準座標系における位置ベクトル．

$\overrightarrow{se}$ ：ソース空間の描画に用いる視点の基準座標系における位置ベクトル．

$\overrightarrow{te}$ ：ターゲット空間の描画に用いる視点の基準座標系における位置ベクトル．

スクリーンへの空間の描画とは，視点，視線，視野角などから求まる錐台形の視野空間の中身をスクリーン面へ射影する作業である．また，オブジェクト内への別空間の描画とは，スクリーン面に射影された WorldMirror の範囲内に，別空間のための視野空間の中身を射影することである．位置ベクトル $\overrightarrow{se}$，$\overrightarrow{sc}$ は，ソース空間をスクリーン面へ描画するための視野空間を求めるのに用いられる．同様に $\overrightarrow{te}$，$\overrightarrow{tc}$ は，ターゲット空間を WorldMirror 内へ描画するための視野空間を求めるのに用いられる．ターゲット空間内のオブジェクトの描画において，ターゲット空間のローカル座標系とソース空間のローカル座標系のいずれにおいてもオブジェクトが同じ相対位置にあるように描画されるためには $\overrightarrow{sc}$-$\overrightarrow{sp}$=$\overrightarrow{tc}$-$\overrightarrow{tp}$，$\overrightarrow{se}$-$\overrightarrow{sc}$=$\overrightarrow{te}$-$\overrightarrow{tc}$ を満足する必要がある．これらから式 (1)，(2) が求まる．

センタリング方式 センタリング方式では，ソース空間のローカル座標系における WorldMirror の中心点に，ターゲット空間のローカル座標系の原点が一致して見えるように描画が行われる．こうすることにより，図 6 (b) のようにつねに WorldMirror の中心にターゲット空間の中心を見ることができる．WorldMirror を移動させても，WorldMirror 内に見えているオブジェクトは見える角度が変わるだけで位置は変わらない．ターゲット空間の描画における視界の決定には，式 (3)，(4) を用いる．

$$\overrightarrow{tc} = \overrightarrow{tp} + (\overrightarrow{sc} - \overrightarrow{wm}) \quad (3)$$

$$\overrightarrow{te} = \overrightarrow{tc} + (\overrightarrow{se} - \overrightarrow{sc}) \quad (4)$$

ここで変数 $\overrightarrow{wm}$ は WorldMirror の基準座標系における位置ベクトルである（図 7 (a)，(c)）．ターゲット空間内のオブジェクトの描画において，ソース空間のローカル座標系における WorldMirror の中心点が，ターゲット空間のローカル座標系の原点と一致して見えるように描画されるためには $\overrightarrow{sc}$-$\overrightarrow{wm}$ = $\overrightarrow{tc}$-$\overrightarrow{tp}$，$\overrightarrow{se}$-$\overrightarrow{sc}$ = $\overrightarrow{te}$-$\overrightarrow{tc}$ を満足する必要がある．これらから式 (3)，(4) が求まる．

WorldBottle では，縮小したターゲット空間をつねに WorldBottle の中心に見せる．そこで WorldBottle では WorldMirror で用いたセンタリング方式に加えて縮小表示が行えるようにした．WorldBottle における空間のスケール変換では，ターゲット空間のローカル座標系の原点が，WorldBottle の中心に向かう視線に沿って動いて見えるように空間の描画が行われる．空間のスケール変換を行うためにパラメータを用意し，それを WorldBottleBox のスロット $\#Size$ として外部に公開し，他のコンポーネントからスロット結合を介して利用可能にした．これにより，ターゲット空間内の対象物のサイズに合わせて縮小率を自由に変更することが可能になる．ターゲット空間の描画における視界の決定には，式 (5)，(6)，(7)，(8) を用いる．

$$\overrightarrow{tc'} = \overrightarrow{tp} + (\overrightarrow{sc} - \overrightarrow{wb}) \quad (5)$$

$$\overrightarrow{te'} = \overrightarrow{tc'} + (\overrightarrow{se} - \overrightarrow{sc}) \quad (6)$$

$$\overrightarrow{te} = d \times (\overrightarrow{te'} - \overrightarrow{tp}) + \overrightarrow{tp} \quad (7)$$

$$\overrightarrow{tc} = d \times (\overrightarrow{tc'} - \overrightarrow{tp}) + \overrightarrow{tp} \quad (8)$$

ここで変数 $\overrightarrow{wb}$ は基準座標系における WorldBottle の位置ベクトルである．また，d はスロット #Size

の値である．ターゲット空間のローカル座標系の原点が，WorldBottle の中心に向かう視線に沿って動いて見えるように空間の描画を行うには，図 7 (c) において，$|\overrightarrow{te}\text{-}\overrightarrow{tc}|$ を変えることで実現できる．これより，$\overrightarrow{te}\text{-}\overrightarrow{tc} = d \times (\overrightarrow{te}\text{-}\overrightarrow{tc})$ を満足する必要がある．これと式 (3)，(4) から式 (5)，(6)，(7)，(8) が求まる．

3.4 ターゲット空間へのユーザの移動

空間の間の移動に関して，以下の 2 つの方法を実現した．空間の移動は，ターゲット空間の視点の座標およびスクリーン面の中心点の座標をソース空間の視点の座標およびスクリーン面の中心点の座標に切り替えることで行われる．また，ターゲット空間に入ることにより，そのターゲット空間は新しいソース空間として扱われる．

フライ・スルー方式 入るという動作を自然に表すために，ユーザが WorldMirror/Bottle に近づき，WorldMirror/Bottle と衝突したときに異なる空間へと移動する仕組みを導入した．これにより，現在ウェブブラウザなどで一般的に用いられているクリック・アンド・ジャンプとは異なるシームレスな空間移動を実現した．衝突判定は，WorldMirror，WorldBottle の独自の描画ルーチン内において，描画の直前に実行される．

クリック・アンド・ジャンプ方式 現在の視点位置のままターゲット空間の中に入りたいという要求に対処するため，WorldMirrorBox および WorldBottleBox にスロット *#Enter* を用意し，その値が TRUE になったときにターゲット空間へと移動する仕組みを実現した．これをボタン・コンポーネントなどと組み合わせることによりクリック・アンド・ジャンプ方式で空間の間を移動することが実現可能になる．

3.5 ターゲット空間へのオブジェクトの移動

ターゲット空間へオブジェクトを移動する際，2 通りの方法を用意した．1 つは，ソース空間の座標系を維持したまま移動する方法である．これは，WorldMirrorBox および WorldBottleBox にスロット *#ObjEnter* を用意し，その値が TRUE になったときに選択されたオブジェクトがターゲット空間へと移動する仕組みにより実現した．これは，ボタン・コンポーネントなどと組み合わせ，それらからオブジェクトの移動のトリガを WorldMirror/BottleBox に与えることで実行する．たとえば 1 つの空間で人体の表層部，内蔵部，骨格部すべてのオブジェクトの位置を合わせて組み立てた後，それらの位置関係を保ったまま別々の空間に分けて配置する．このさい，位置関係が保たれているので，参照する空間の選択を変えることにより，見えるオブジェクトを切り替えることができる．この手法は後述の 4.3 節において図 9 (c) に示す応用例において用いている．もう 1 つは，ターゲット空間の中心へオブジェクトを移動させる方法である．これは，マウスによるドラッグアンドドロップにより移動させたいオブジェクトを WorldMirror/BottleBox オブジェクトに重なり合うように配置し，WorldMirror/BottleBox をクリックすることでオブジェクト移動のトリガを与え実行する．これにより，たとえば，センタリング方式を用いた WorldMirror/Bottle で写し込んでいる空間にオブジェクトを移動させたい場合，つねにその空間の中心にオブジェクトを保持することが可能となる．

3.6 作業空間の分割と管理

本節では，空間を生成する手法について述べる．独立な空間は，その定義の中に空間名と中心座標を含む．この中心座標の与え方に応じて 3 種類の方法を用意した．まず，(1) ユーザがその空間に対して名前 *s-name* を与え，その空間名からシステムが自動的に，空間の中心座標 *s-coord* を割り振る方法を用意した．この際，特に指定されない限りは，他の空間と十分に離れた位置に新しい空間の中心座標は与えられる．これにより，ある情報に関して関連情報空間を生成するときに，そのカテゴリなどの属性名やオブジェクトに付けられた名前から空間を生成することが可能になる．これにより，属性ごとの空間を生成し，それらの空間をそれぞれ複数の WorldMirror/Bottle のターゲット空間として指定することで，1 つの情報に関する関連情報を属性ごとに同時に見ることが可能になる．この手法は後述の 4.3 節において図 10 に示す応用例において用いている．次に，(2) オブジェクトを指定することにより，オブジェクトの基準座標系における座標を空間の中心座標 *s-coord* として与え，ユーザがその空間に対して名前 *s-name* を与える方法を用意した．これにより，ランドマークとなるオブジェクトをユーザが指定し，そのオブジェクトを中心とした空間を生成することが可能になる．この手法は後述の 4.2 節において図 9 (b) に示す応用例において用いている．最後に，(3) 基準座標系におけるある特定の座標を空間の中心座標 *s-coord* として指定し，ユーザがその空間に対して名前 *s-name* を与える方法を用意した．これにより，オブジェクトの存在しない座標を空間の中心にすることが可能になる．(2)，(3) の手法によって生成された空間の中心座標は，基準座標系において必ずしも十分に離れた座標点としては与えられない．これにより，たとえばソース空間に見えているオブジェクトをそのオブジェクトと並べて WorldMirror/Bottle

にも表 2 で示したような特殊効果を与えて表示させることなどが可能になる．

4. 応 用 例

本章では WorldMirror/Bottle の応用を述べる．まず，移動履歴の可視化を行い，それを用いたバックトラック機能の導入の仕組みを述べる．次に，バーチャルリアリティ空間への応用と情報視覚化への応用について述べる．

4.1 移動履歴の視覚化

WorldMirror/Bottle による複合空間は，ハイパーリンク構造をとる．ハイパーリンク構造をとる空間を移動する際の問題点には，空間の移動を続けるうちに自分がどこにいるのかの認識ができなくなり，元の空間に戻れなくなることがあげられる．この問題の解決法として，バックトラック機能を導入した．**図 8** におけるパネル部分がバックトラック機能の実現部品である．これは，ディスプレイ上のオブジェクトの最手前に貼り付く機能を持つ StickBox，付属部品を不可視状態にする機能を持つ CollapseBox，システムから空間遷移情報を取得しその情報を視覚化する機能を持つ HistoryManagerBox，空間が埋め込まれた WorldBottleBox からなる合成ボックスである．本論文では，以下に示す手順で，空間を移動するたびにその履歴を WorldBottle としてパネルに配置し，それらの WorldBottle をクリックすることで空間移動を行う仕組みを導入した．図 8 (a) においてまず，図 8 (d) 中の① (以下①と記述する) に示すように，ターゲット空間 C へのユーザの移動が行われる．その際に②新しいソース空間 C に関する情報 (*s-name*, *s-coord*) を WorldMirror はシステムに通知する．③システムはソース空間が B から C に変わったことを HistoryManagerBox に通知する．④HistoryManagerBox は新しいソース空間に関する情報 (*s-name*, *s-coord*) をシステムから取得する．図 8 (b) に示すように，⑤HistoryManagerBox は空間 C を指す WorldMirrorBox をパネル上に新規に登録する．⑥ユーザが空間 B を指す WorldBottleBox をクリックすることにより，図 8 (c) に示すように，⑦ユーザは 1 つ前の空間 B へ移動する．このようにすることでウェブブラウザにおける back，forward，go のようなバックトラック機能を実現した．

4.2 バーチャルリアリティ空間への応用

バーチャル空間をデザインする際，幾何的制約がデザインへの制約になることがある．たとえば，建築物などを建てる際に現実と同じ比率で建てるとユーザにとっては大きすぎて全体を把握できないものとなる．

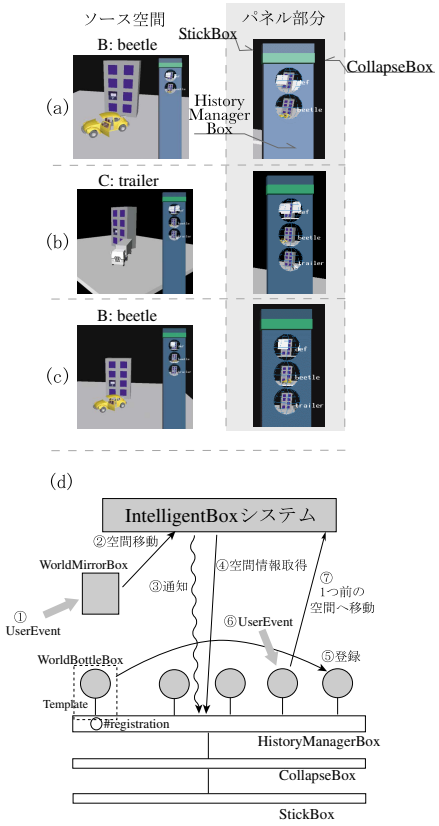


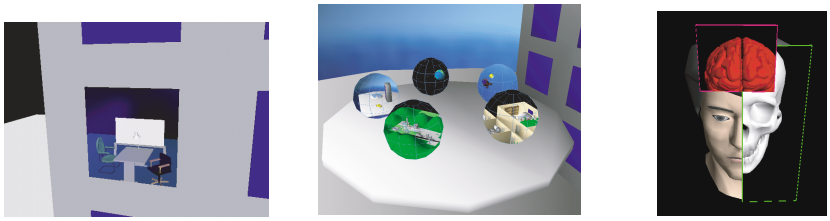
図 8 バックトラック機能
Fig. 8 The mechanism for backtracking.

そこで，建築物を現実より縮小して建てると今度は建物の内部が狭くなるという問題が生じる．また，デザイナが個別に組み立てた構造物を後から組み合わせたいという要求も考えられる．これらの要求は，空間を部分ごとに組み立て，それらを後から WorldMirror/Bottle を用いてつなぎ合わせることで満たされる．**図 9** (a) の例では，ビルの窓のコンポーネントとして WorldMirror を配置し，その窓に内部の部屋として組み立てられた別の空間を割り当てることで，実際にはそこに存在しない部屋をそこに広がっているかのように扱っている．この例の場合，部屋としてビルより大きな部屋を割り当てることも可能である．

また，WorldBottle を用いてバーチャルリアリティ空間における他の空間を集積し，仮想世界におけるガイドブックを実現するという応用が考えられる (図 9 (b))．

4.3 情報視覚化への応用

WorldMirror/Bottle の情報視覚化への応用としては，フィルタ・コンポーネントへの応用が考えられる．フィルタとして考えられる機能としては，詳細情報・関連情報などの付加情報の表示といった情報の重畳表



(a) 自由な空間設計　(b) WorldBottleをアンカーとして用いた空間の集積　(c) フィルターとして利用

図 9 WorldMirror/Bottle を用いた応用例
Fig. 9 An example of using a WorldMirror/Bottle.

図 10 3 次元オブジェクト空間における関連情報へのリンクの実現
Fig. 10 WorldMirror/Bottles used to define an anchor in a 3D hyperspace.

示，現在の表示とは別の観点からの表示，形状や色などの変形表示が考えられる．図 9 (c) は，人体の内部状態の視覚化を試みた例である．顔を表示している空間に骨格を表示している空間と脳を表示している空間を WorldMirror を用いて重畳することで WorldMirror によって限られた範囲内の人体の内部状態を見ることが可能になる．

WorldMirror/Bottle を導入する目的の 1 つに，情報アクセスのための空間構造を WorldMirror/Bottle を用いて組み立てることがあげられる．さまざまな情報をメディア・オブジェクトとして扱うオブジェクト空間において，ある情報を表すオブジェクトに対する関連情報のための空間を関連空間として用意する．そこに関連オブジェクトを配置し，WorldMirror/Bottle により元情報のオブジェクトの空間からリンクを張る．これにより，情報アクセスのための仕組みを構築できる．**図 10** (a) には 3 次元空間中にテキストデータが 3 次元表現で描かれている．この文字列の中の一部の単語 ‘Yesterday’ が WorldBottle によるアンカーとして他の情報へのリンクとなっている．ユーザがこのアンカーに近づくことで 9 つの立体文字 ‘Yesterday’ の中に別空間が見えてくる（図 10 (b)）．これによりユーザは，今いる空間からリンク先の情報を見ることが可

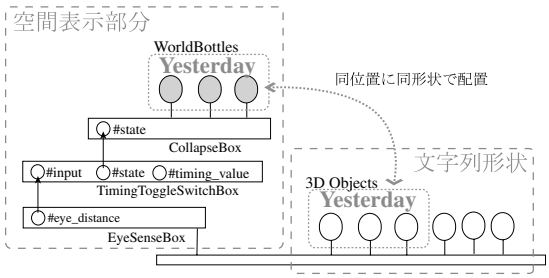


図 11 図 10 の実現機構
Fig. 11 The mechanism of Fig. 10.

能となる．さらに ‘Yesterday’ 中の文字 ‘r’ に近づいて行くようすを（図 10 (c)）に示す．この文字の壁を突き破り別空間に入って行くことでユーザはリンクをたどることになる（図 10 (d)）．さらに，リンク先の空間にも同様のリンクを設定することが可能である．このリンクは，特にテキストに張られている必要はなく，空間中で扱われる任意のオブジェクトに対して同様の仕組みで張ることが可能である．**図 11** に実現機構を示す．アンカー部分は ‘Yesterday’ の形状のみで機能を持たないオブジェクトと，‘Yesterday’ の形状を持つ WorldBottleBox からなる．これらのオブジェクトは，同位置に配置されている．はじめ，アンカーとして指定されている 3 次元形状にはリンク先の空

間は見えていない．これは，‘Yesterday’ の形状を持つ WorldBottleBox を表示しないことで実現している．EyeSenseBox はユーザの視点からの距離を保持する．この距離が *#timing_value* で指定された値になったとき TimingToggleSwitchBox が CollapseBox に WorldBottleBox を表示するようにメッセージを渡す．これにより，アンカー中にリンク先の空間が表示される．

5. おわりに

本論文では，3 次元仮想空間を WorldMirror/Bottle として扱うことにより，限られたディスプレイ空間を効率良く利用し，さらに，3 次元オブジェクト空間を WWW のような情報へのアクセス空間として利用する仕組みの導入について述べた．また，WorldMirror/Bottle を用いることで（1）ユーザによる自由な空間デザイン，(2）実世界には存在しない空間構造の実現，(3）空間中のあらゆるオブジェクトの空間としての利用，(4）それらのオブジェクトに対する関連情報，詳細情報などの同時表示とそれらの情報へのアクセスが可能になることを示した．

情報視覚化へのさらなる応用としては，WorldMirror を 3 次元オブジェクト空間におけるオブジェクトの持つファセット（特性）として用いるフレームワークの導入が考えられる．WorldMirror を通してオブジェクトの異なるファセットを見ることにより，1 つのオブジェクトを異なる側面から見ることが可能になる．

また，本論文では各空間の中心座標が，基準座標系が定義する空間内において，十分に離れた座標点として与えられるように実装を行っているが，与えられた座標点によっては，ある空間から別の空間が WorldMirror/Bottle を介さずに見えることがありうる．現在この問題に対し，それぞれの空間が基準座標系には依存しない独立な座標系を持つようにする拡張が行われている．

参考文献

1) Angus, I.G. and Sowizral, H.A.: Embedding the 2D Interaction Metaphor in a Real 3D Virtual Environment, Technical Report, Bowing Computer Services (1994).
2) Barrus, J.W., Waters, R.C. and Anderson, D.B.: Locales and Beacons: Efficient and Precise Support For Large Multi-User Virtual Environment, *IEEE Computer Graphics and Applications*, Vol.16, No.6, pp.50–57 (1996).
3) Bolt, R.A.: *The Human Interface*, Lifetime Learning Publications (1984).
4) Bolt, R.A.: *Spatial Data Management*, Massachusetts Institute of Technology (1979).
5) Card, S.K., Robertson, G.G. and Mackinlay, J.D.: The Information Visualizer, an information workspace, *Proc. ACM Conference on Human Factors in Computing Systems* (*CHI'91*), pp.181–188 (1991).
6) Elvins, T.T., Nadeau, D.R. and Kirsh, D.: Worldlets—3D Thumbnails for Wayfinding in Virtual Environments, *Proc. ACM User Interface Software and Technology Symposium, UIST'97* (1997).
7) Nelson, T.H.: *Literary Machines* (1981).
8) Okada, Y. and Tanaka, Y.: IntelligentBox: A Constructive Visual Software Development System for Interactive 3D Graphic Applications, *Proc. Computer Animation '95*, pp.114–125 (1994).
9) Stoakley, R., Conway, M.J. and Pausch, R.: Virtual Reality on a WIM: Interactive Worlds in Miniature, *Proc. 1995 ACM SIGCHI Conference* (1995).
10) Viega, J., Conway, M.J., Williams, G. and Pausch, R.: 3D Magic Lenses, *ACM UIST'96*, pp.51–58 (1996).
11) Wloka, M.M. and Greenfield, E.: The Virtual Tricorder: A Uniform Interface for Virtual Reality, *Proc. 1995 ACM UIST Conference*, pp.39–40 (1995).
12) 伊藤正彦，田中　譲：3 次元仮想空間における作業空間の複合化と展覧機能の導入，第 57 回情報処理学会全国大会論文集，No.3, pp.135–136 (1998).
13) 伊藤正彦，田中　譲：3 次元仮想空間中での複数空間に対するアクセス支援機能の導入，第 59 回情報処理学会全国大会論文集，No.3, pp.233–234 (1999).
14) 岡田義広，田中　譲：対話型 3D ソフトウェア構築システム—IntelligentBox，コンピュータソフトウェア，Vol.12, No.4, pp.374–384 (1995).
15) 田中　譲：情報空間のアーキテクチャと建築部品，情報メディアシンポジウム'97 論文集，pp.9–19 (1997).
16) 萩谷昌己：GMW ウィンドウ・システムについて，*bit*, Vol.19, No.3, pp.227–241 (1987).

(平成 12 年 8 月 29 日受付)
(平成 13 年 9 月 12 日採録)

伊藤　正彦（正会員）

1973 年生．1998 年北海道大学工学部電気工学科卒業．2000 年同大学工学研究科電子情報工学専攻博士前期課程修了．情報視覚化，ヒューマンインタフェースの研究に従事．現在，株式会社ビー・ユー・ジーに所属．

田中　　譲（正会員）

1972 年京都大学電気工学科卒業．1974 年同大学院電子工学専攻修士課程修了．工学博士．1974 年北海道大学工学部助手．1977 年同講師．1985 年同助教授を経て，1990 年同教授，現在に至る．1996 年北海道大学知識メディアラボラトリー長．この間，1985 年 10 月より 1 年間，IBM 社 T.J. ワトソン研究所客員研究員．ソフトウェア科学会，人工知能学会，米国 IEEE 各会員．データベース理論，データベース・マシン，並列処理アーキテクチャ，メディア・アーキテクチャ等の研究に従事．コンピュータ・アーキテクチャ等の著書あり．1994 年に IntelligentPad の開発に関して日経 BP 技術賞大賞受賞．