

## 機能オブジェクトの3次元化とその機能合成について (IntelligentPadの3次元化)

岡田 義広      田中 譲  
北海道大学工学部

計算機上で種々のメディアを統一的に扱うためのシステムとして、**IntelligentPad**と呼ばれるシンセティックメディア・システムが現在北海道大学で研究、開発されている。著者らは、**IntelligentPad**システムの基本概念を継承し、3次元仮想現実感のアプリケーションを開発するためのシンセティック・システムの研究、開発を行っている。今回、2次元形状をもつパッドを3次元形状をもつオブジェクト(ボックス(仮称)とよぶ)に拡張したシステムを開発した。これを著者らは**IntelligentBox**(仮称)と呼んでいる。本稿では、**IntelligentBox**システムの基本機構であるMVCモデリング、ボックス間構造とそのメッセージ送信プロトコルについて述べる。また、**IntelligentBox**システムの応用例を挙げてこれらの妥当性を述べる。

### 3-D REPRESENTATIONS and SYNTHESSES of FUNCTIONAL OBJECTS : A 3-D extension of IntelligentPad

Yoshihiro Okada and Yuzuru Tanaka

Electrical Engineering Department, Faculty of Engineering, Hokkaido University  
N. 13, W.8, Kita-ku, Sapporo, Japan

The synthetic media system called **IntelligentPad** is researched and developed at Hokkaido University. It treats various media as pads. A pad is a functional object manipulated like a sheet of paper. As an extension of **IntelligentPad**, we have developed a prototype system called **IntelligentBox** for constructive development of VIRTUAL REALITY (VR) applications through the combination of 3-D functional objects. In **IntelligentBox**, all functional objects are called boxes. A box can be regarded as a 3-D extension of a pad. **IntelligentBox** inherits its fundamental mechanisms from **IntelligentPad**. Among them are the MVC modeling, hierarchical structures of boxes, and the standardized message passing protocols. This paper describes them and shows their ability to construct VR applications.

## 1. はじめに

近年のグラフィックス・ワークステーションの性能向上に伴い、仮想現実感の研究が行える環境が整いつつある。著者らは、固有の機能をもち3次元表示されたオブジェクトを組合せ機能合成することにより、仮想現実感を構築可能なセンセティック・システムの研究を行っている。今回 **IntelligentBox** とよぶ(仮称)プロトタイプ・システムを開発したので、この設計概念と基本機構について述べる。

すでに、**IntelligentPad** とよばれるセンセティックメディア・システム<sup>[1~4]</sup>の研究・開発が北海道大学で行われている。**IntelligentPad** は、すべてのメディアを電子的な紙として扱う。この紙をパッドとよんでいる。各パッドは固有の機能を持ち、このパッドを画面上で貼り合わせることにより機能が合成され、複雑な機能をもつパッドを実現できる。2次元表示されたオブジェクトであるパッドを3次元表示されたオブジェクトであるボックス(多面体を総称してボックス(仮称)とよぶことにする)へ拡張することで、**IntelligentPad** の基本機構を仮想現実感を構築するためのセンセティック・システムへ応用できると考えられる。そこで、**IntelligentPad** の基本機構を変更することなしに、**IntelligentBox** の基本機構として実現した。

従来の仮想現実感では、3次元で表された仮想世界上の対象物が個々に内部状態と反応をもっているのではない。その個々の対象物を他の仮想世界へ持ち込むことはできない。使用者が仮想世界の中で対象物の機能の合成操作を対話的に行えるものは著者の知る限りない。

**IntelligentBox** では、各ボックスは3次元表示され内部状態と機能あるいは反応をもっている。それは、現実世界での電卓のボタンや液晶表示器、ラジオのボリュームに対応する。電卓の計算機能をもったボックスに、ボタンやディスプレイのボックスを配置し合成することにより、3次元形状をもつ電卓がつくられる。このように、現実世界にある対象物を固有の機能をもった部品として扱い、計算機上でこれらの部品を組合せ機能合成することにより、複雑な機能をもった対象物を構築できる。

3次元表示機能をもつ応用プログラムを開発するための部品を提供するツールキットとして、すでに **3D Widgets**<sup>[5]</sup> がある。これでは、使用者の部品への操作に対する応答が対話的に行われる。だが、3次元表示された部品の合成を画面上での直接操作により行うことはできない。**IntelligentBox** では、部品の配置と機能合成は計算機の画面上での直接操作により行える。

以下では、2節で、**IntelligentPad** の概要を述べる。**IntelligentPad** の基本機構と機能合成における概念を簡単に説明する。3節では、**IntelligentBox** の実現機構について述べる。**IntelligentPad** から継承した **IntelligentBox** の基本機構を具体的に説明する。4節では、**IntelligentBox** の具体的なボックスの機能合成例を挙げてこれらの実現機構の妥当性を述べる。最後に、5節でまとめと今後の課題を述べる。

## 2. IntelligentPad

本節では、**IntelligentPad** システムに関して、その基本機構を簡単に説明する。

**IntelligentPad** は、すべてのメディアを電子的な紙として扱う。この紙はパッドと呼ばれる。各パッドは固有の機能をもつ部品である。パッドを他のパッドの内部へ貼り合わせることにより、個々のパッドがもつ機能が合成された複雑な機能をもつパッドとなる。これを合成パッドとよぶ。この過程はプログラミングに外ならない。**IntelligentPad** の基本機構は、以下の3点にまとめられる。

### 1) MVCモデリング

1つのパッドは、**Model**、**View**、**Controller** とよばれる3つのオブジェクトで構成される。これは、オブジェクト指向言語 **Smalltalk-80**<sup>[6]</sup> のウィンドウ・システムに採用されている **MVC** モデリングと同様の方式である。**Model** はパッドの内部状態を保持するオブジェクトである。**View** はパッドの画面上での視覚的な形状を表すオブジェクトである。**Controller** はパッドに対する使用者からのマウスイベントを受け取り、**View** へ送るためのオブジェクトである。このように、1つのパッドを3つのオブジェクトへ機能分化することにより、**Model**、**View**、**Controller** が相互に再

利用可能となる。また、後述するモデル共有によるデータ送受機構が実現される。

## 2) 貼り合わせ構造

貼り合わせによるパッドの合成は、貼るパッドと貼られるパッド間に親子関係を与えることである。貼るパッドは貼られるパッドの子パッドとよばれる。逆に、貼られるパッドは貼るパッドの親パッドとよばれる。各パッドは複数の子パッドをもつことができる。だが、複数の親パッドをもつことはできない。したがって、その階層構造は、最下段のパッドを根とする木構造となる。これらの階層構造は、各パッドのViewにおける関係として決められる。

## 3) パッド間データ授受のためのプロトコル

パッド間でのデータ授受は、貼り合わされたパッド間でのみ可能である。つまり、各パッドは、その親パッドあるいはその子パッドとの間でのみデータの授受が行える。親パッドと子パッド間のデータ送受は、パッドのViewがもつset、gimme、updateとよばれるフラグにより制御される。各パッドのModelは、そのパッドの内部状態を保持するためのスロットをもつ。setフラグは、子パッドがもつスロットの値を親パッドがもつスロットへ代入するか否かを決めるためのフラグである。gimmeフラグは、子パッドがもつスロットへ親パッドがもつスロットの値を読み出し代入するか否かを決めるためのフラグである。親パッドのスロットの値が更新された際に、そのすべての子パッドへupdateメッセージが送られる。updateフラグは、このメッセージを受けるか否かを決めるためのフラグである。親パッドから更新されたことが知らされた子パッド(updateフラグが立てられていなければならない)は、setフラグあるいはgimmeフラグによりスロットの値の送受を行う。

IntelligentBoxでは、上記1),2),3)の基本機能を継承している。また、IntelligentPadでは、モデル共有によるデータ送受機構がある。IntelligentBoxでも採用している。以下では、これらについて詳しく説明を加える。

## 3. IntelligentBox

本節では、IntelligentBoxの実現機構について述べる。特に、IntelligentPadがもつ上述した1),2),3)の基本機能との違いを述べる。

### 3.1 MPCモデリング

IntelligentPadのMVCモデリングに対して、IntelligentBoxでは、MPCモデリングを採用している。すなわち、1つのボックスは、Model、Polyhedron、Controllerとよばれる3つのオブジェクトで構成される。これは、MVCのView部分を視覚的な3次元形状を表すPolyhedronに置き換えたものである。IntelligentPadと同じように、図1に示される方式でメッセージ送信と更新報告が行われる。更新報告は、Modelがもつスロット値が変更された際に、ModelからPolyhedronに対してupdate-selfというメッセージが送られることで行われる。

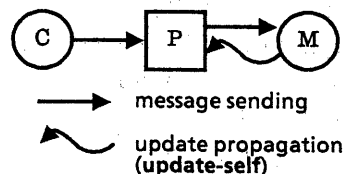


図1 ボックスのMPC構造

### 3.2 親子関係

IntelligentPadでパッドのView間に親子関係があるように、IntelligentBoxではボックスのPolyhedron間に親子関係を定義する。IntelligentPadでは、パッドの親子関係は貼り合わせるという画面上での直接操作により定義される。IntelligentBoxでは、ボックスの親子関係はメニュー選択により決められる。ボックスのPolyhedron間に空間的な交わりがない場合にも、親子関係を定義できる。IntelligentBoxにおける親子関係は、子ボックスはその親ボックスに従属するという関係である。子ボックスの空間的な位置は、その親ボックスの位置に対する相対的な座標で表される。つまり、親ボックスを平行移動、回転させた場合には、そのすべての子ボックスはそれに追従して平行移動、回転をする。逆に、子ボックスを平行移動、回転させた場合には、その親ボックスに対する相対的な位置が変化するのみである。

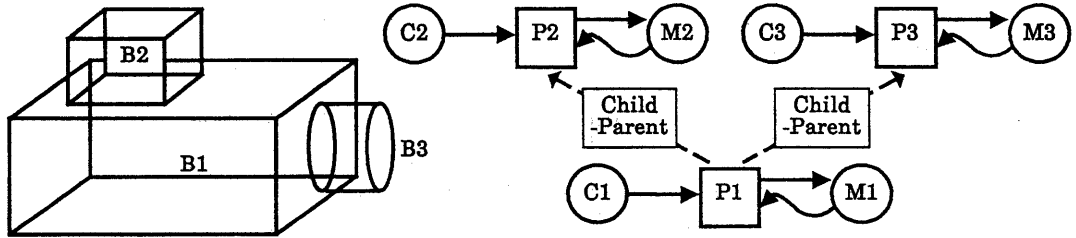


図2 ボックスのPolyhedron間における親子関係

図2の左図に示されるように、3つのボックス B1、B2、B3においてボックスB1がボックスB2、B3の親ボックスである場合、各ボックスの Polyhedron間には、図2の右図に示されるような親子関係が存在する。

以下で説明するスロット結合によるボックス間データ授受プロトコルでは、この親子関係がある間でのみ各ボックスがもつデータの授受が可能となる。ただし、モデル共有によって、親子関係がない場合にもデータ授受が可能である。

### 3.3 ボックス間データ授受プロトコル

IntelligentBoxでは、IntelligentPadと同じプロトコルにより、親子関係があるボックス間でデータの授受が行える。

#### 1) メッセージの種類と動作

図3に示されるように、`set <slotname>`、`update`、`gimme <slotname>` メッセージがある。`set`と`gimme`はB1からその親ボックスであるB0へ、`update`はB0からその子ボックスであるB1へ送られる。`set <slotname>`、`gimme <slotname>`は、各ボックスが自分の親ボックスが持つ<slotname>で指定されるスロットに対して値<value>を送るためのメッセージである。<value>は、プライマリ・スロットとよばれるスロットの値である。プライマリ・スロットは、そのボックスの内部状態を代表する値が格納されるスロットである。このスロットは、ボックスの種

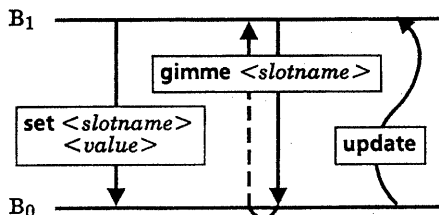


図3 ボックス間標準メッセージ

類ごとに異なる。`gimme <slotname>`は、自分の親ボックスが持つ<slotname>で指定されたスロットから値を読み出してくるためのメッセージである。<slotname>で指定されるスロットから読み出された値は、自身のプライマリ・スロットに書き込まれる。`update`は、各ボックスが自分のすべての子ボックスに対して、自分の内部状態が変化したことを知らせて状態更新を要求するためのメッセージである。`set`メッセージにより自分のどのスロットの値を親ボックスのどのスロットに代入するか、および`gimme`メッセージにより自分のどのスロットに親ボックスのどのスロットの値を代入するかはメニュー選択により一意に決められる。これをスロット結合とよんでいる。

#### 2) メッセージの送信許可フラグとその起点

上記のメッセージの送信許可は、`set`、`update`、`gimme`フラグによって与えられる。ボックスのPolyhedronが`set`、`update`、`gimme`フラグをもち、それぞれ独立に許可・禁止をすることができる。

[1] `set`メッセージが親ボックスに対して送られるのは、自分のもつプライマリ・スロット値が変化して、ModelからPolyhedronへ`update-self`メッセージが送信され、さらに`set`フラグが立っている場合である。

[2] `gimme`メッセージが親ボックスに対して送られるのは、親ボックスから`update`メッセージが送信され、さらに`gimme`フラグが立っている場合である。

[3] `update`メッセージが子ボックスに対して送られるのは、自分のもつプライマリ・スロット値が変化して、ModelからPolyhedronへ`update-self`メッセージが送信され、さらに`update`フラグが立っている場合か、親ボックスから`update`メッセージが送信され、さらに`update`フラグが立っ

ている場合である。このupdateフラグの使い方は、IntelligentPadでの使い方とは異なる。

### 3) メッセージの意味

ボックスがsetメッセージの送受を許可されると、その親ボックスに対するスロット値の入力が行われる。すなわち、setフラグが立っているボックスは、その親ボックスに対する入力器として機能する。ボックスがgimmeメッセージの送受を許可されると、その親ボックスがもつスロット値の読み出しが行われる。すなわち、gimmeフラグが立っているボックスは、その親ボックスに対する出力器として機能する。ボックスがupdateメッセージの送受を許可されると、そのすべての子ボックスへ状態更新を要求する。自分もつプライマリ・スロット値が変更されない場合や、出力器としての子ボックスがない場合には、updateフラグを立てる必要はない。

### 3.4 モデル共有によるデータ送受機構

上述したデータ送受機構は、親子関係が存在するボックス間でのみ可能である。親子関係がないボックス間でのデータ送受は、モデル共有により可能である。先述したように、それぞれのボックスはModel、Polyhedron、Controllerの3者で構成される。ひとつのモデルは、複数のPolyhedronとControllerの組をもつことができる。この際、複数のPolyhedronは、モデルを共有しているという。図4の左図に示されるボックス3(図中B3)とボックス3'(図中B3')は、モデルを共有しているとする。また、ボックス3はボックス1(図中B1)の子ボックスであり、ボックス3'はボックス2(図中B2)の子ボックスであるとする。この場合の、各ボックスのModel、Polyhedron間の関係は、図4の右図に示されるようになる。この際、ボックス3とボックス3'のPolyhedron(図中P3と

P3')には、それぞれsetフラグとgimmeフラグが立てられているとする。ボックス2のPolyhedron(図中P2)には、updateフラグが立てられているとする。いま、ボックス2のModel(図中M2)がもつプライマリ・スロット値が変更された場合、M2からP2へupdate-selfメッセージが送信され、さらに、updateメッセージがその子ボックスであるボックス3'へ送信される。実際には、P3'へ送られる。P3'は、gimmeフラグが立てられており、M2の該当するプライマリ・スロットの値をgimmeメッセージにより読み出しM3の該当するスロットへ代入する。M3は、P3とP3'に共有されているModelである。この値が元のスロット値と同じ場合には、代入は行われない。この値が元のスロット値と異なる場合には、M3は、P3とP3'へupdate-selfメッセージを送信する。update-selfメッセージを受けたP3とP3'は、setフラグ、gimmeフラグにしたがって、それぞれset、gimmeメッセージを送信する。setメッセージを受けたP1は、M1の該当するプライマリ・スロットへ値の代入を行う。このようなメッセージの流れにより、M2のもつプライマリ・スロットの値がM1のもつプライマリ・スロットへ代入される。図中の破線がデータの流れを示している。このように、モデル共有により親子関係のないボックス間でもデータ送受が可能である。

## 4. 機能合成例

3次元表示されたオブジェクト(部品)は、個々に機能を持ち、それらを組み合わせることにより、さらに、複雑な機能をもった部品を作製できる。それは、あたかも使用者が現実の部品を組み合わせる一つの大きな道具や機械を作製する過程と同様と考えられる。これを1)構成の定義機能とよぶことに

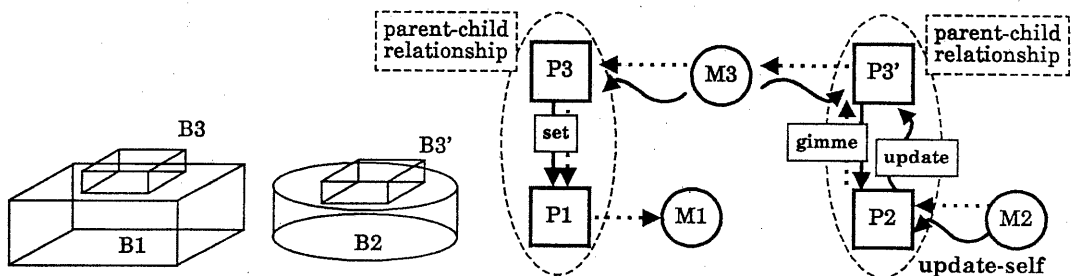


図4 モデル共有によるデータ送受機構

する。また、図5に示されるように、回転のみをするオブジェクト(これを回転部品とよぶ)と伸縮のみをするオブジェクト(これを伸縮部品とよぶ)をジオメトリ情報をもつ部品として扱い、これらジオメトリ部品の合成により、複雑なジオメトリ情報を扱うことが可能となる(現在はボックスの座標情報のみ扱い可能)。これはアニメーションへ応用することができる。これを2)アニメーション機能とよぶことにする。

以下では、1)構成の定義機能と2)アニメーション機能の2つについて、それぞれ機能合成の例を挙げる。

#### 4.1 構成の定義機能

(a) スロット結合によるデータ授受を説明するために、非常に単純な部品の合成例を示す。

図6は、ボリューム(Volume)とスライダー(Slider)が、*io-value*とよぶプライマリ・スロットをもつボックス(IO Box)の子ボックスとして合成されている例である。ボリュームは、*ratio*とよぶプライマリ・スロットをもち、スライダーも*ratio*とよぶプライマリ・スロットをもち、画面上でマウスを用いてボリュームを直接回すことにより、そのプライマリ・スロット*ratio*の値が0.0から1.0の間で変化する。同様に、スライダーの上面をスライドさせることにより、そのプライマリ・スロット*ratio*の値が0.0から1.0の間で変化する。ボリュームもスライダーも、*set*と*gimme*フラグが立てられている。IO Boxは、*update*フラグが立てられている。ボリュームのプライマリ・スロット*ratio*は、親ボックスのプライマリ・スロット*io-value*と結合されている。同様に、スライダーのプライマリ・スロット*ratio*も親ボックスのプライマリ・スロット*io-value*と結合されている。ボリュームを回すことにより、ボリュームがもつスロット*ratio*の値が変化する。*set*フラグが立てられているので、親ボックスのもつスロット*io-value*へその値が代入される。親ボックスは、*io-value*の値が変更され、しかも*update*フラグが立てられているので、すべての子ボックスに対して*update*メッセージを送る。*update*メッセージを受けたスライダーは、*gimme*フラグが立てられているので、親ボックスのもつスロット*io-value*の値を読み出して、自身のスロット*ratio*へ代入する。その値にしたがって、上面をスライドさせる。逆に、

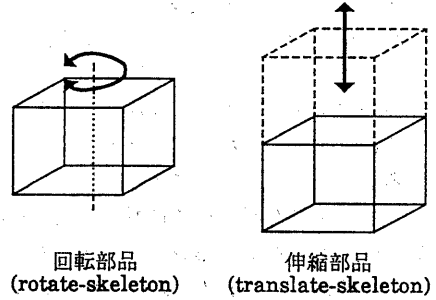


図5 回転部品と伸縮部品

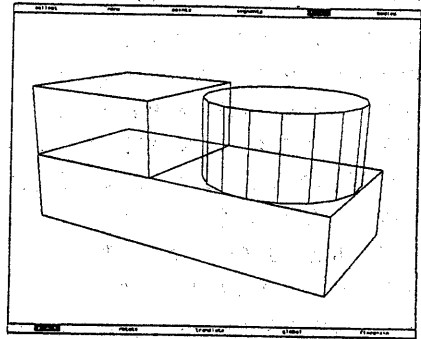


図6 ボリュームとスライダーによるデータ授受  
スライダーの上面をスライドさせることにより、同様のメッセージとデータの流れによりボリュームが回転する。各ボックス間の親子関係およびメッセージとデータの流れを表したものが図7である。

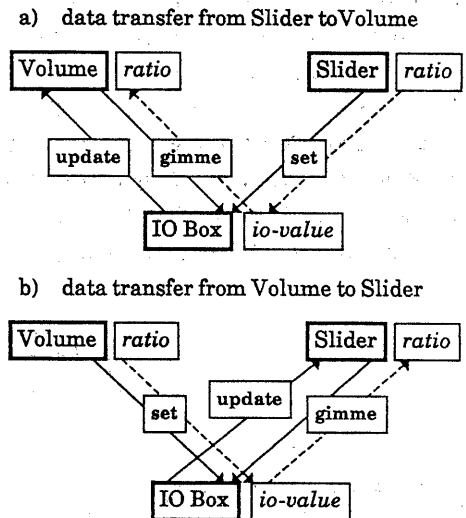


図7 図6の例の構造とメッセージとデータの流れ

(b) 図8は、モータと歯車を組み合わせてタイヤを駆動させる例の画面ハードコピーである。

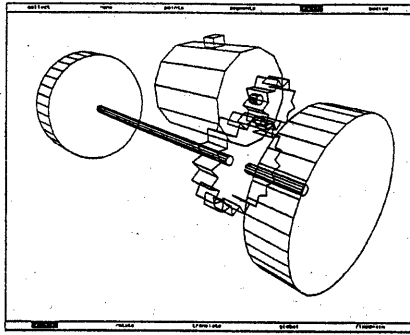


図8 モータ駆動によるタイヤの回転

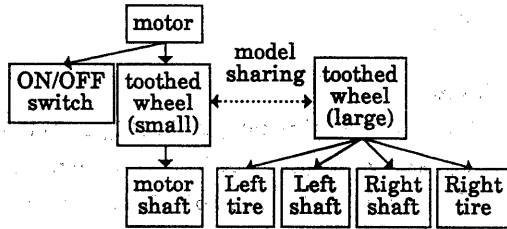


図9 図8の例の階層構造

モータ(motor)はプロセスにより自律的に駆動する。モータの本体に付けられたスイッチのオン/オフ(ON/OFF switch)により、モータのプロセスが起動しモータが駆動する。モータの軸(motor shaft)には歯車(toothed wheel (small))が付けられ、モデル共有により、タイヤの軸に付けられた歯車(toothed wheel (large))に回転が伝わりタイヤが回転する。

図9は、各ボックスの親子関係とモデル共有関係を表したものである。親子関係は、親ボックスから子ボックスへ向かう有向グラフで表している。モデル共有を破線の双方向グラフで表している。モータ軸は、モータの歯車の子ボックスであり、モータの歯車はモータ本体の子ボックスである。モータ本体は、プロセス起動により自動的に値が変化するカウンタである。プライマリ・スロットとしてcount-valueをもち、その値は0.0から1.0までの間を適当な増分値で変化する。モータの歯車は、gimmeフラグが立てられており、親ボックスであるモータのcount-valueの値を読み出す。モータの歯車は、その読み出した値に対応する角度(1.0を360°としている)まで回転する。タイヤ軸に付けられた歯車は、タイヤ(左右で2個)とその軸(左右で2個)の子ボックスとする。ただし、これらの親子関係にはスロット結合はない。タイヤ軸の

歯車は、モータの歯車とモデル共有しているので、モータの歯車が回転するとタイヤ軸の歯車も回転する。タイヤ軸の歯車は、タイヤとその軸の親ボックスであるから、タイヤ軸の歯車が回転することにより、タイヤとその軸も回転する。

モータとその軸と歯車は、合成されたひとつの部品として扱われる。これを他の部品として使用することができる。同様に、タイヤと軸と歯車を合成されたひとつの部品として使用することができる。

#### 4.2 アニメーション機能

図10は、シャトルがローリングしながら上昇するアニメーションを実現した画面ハードコピーである。先に述べた回転部品(rotate-skeleton)と伸縮部品(translate-skeleton)を組み合わせることで、ローリングしながら上昇する動作を実現している。

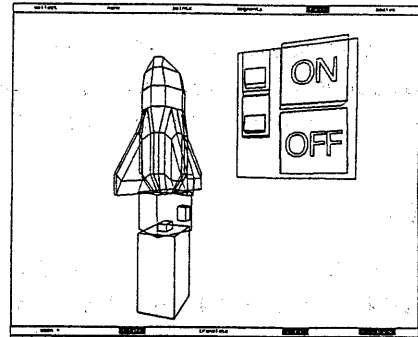


図10 シャトルのアニメーション

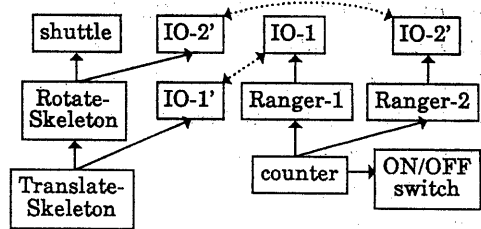


図11 図10の例の階層構造

図11は、各ボックスの親子関係とモデル共有関係を表したものである。カウンタ(counter)をプロセスにより自律的に駆動させ、カウンタの値を適当なレンジ(Ranger-1とRanger-2)で増幅し、これらの値を回転部品(rotate-skeleton)と伸縮部品(translate-skeleton)へ送り動作させる。io-valueというプライマリ・スロットをもつボックスを2組(IO-1とIO-1'およびIO-2とIO-2')用いて、Ranger-1

とRanger-2で増幅された値を、モデル共有により、rotate-skeletonとtranslate-skeletonへ転送している。rotate-skeletonとtranslate-skeletonは、送られた値にしたがって、それぞれ回転と伸縮動作をする。シャトル(shuttle)はrotate-skeletonの子ボックスであり、rotate-skeletonはtranslate-skeletonの子ボックスである。すなわち、シャトルは回転しながら上昇する。

この例では、カウンタの値をアニメーションの時間軸と考えることができる。カウンタの増分値は、1秒間のフレーム数を決定するパラメータとなる。カウンタの増分値を0.05とした場合には、1秒間に20回その値が更新される。すなわち、秒20フレームのアニメーションとなる。カウンタの最大値をTとし、増分値をDとすると、秒(1/D)フレームでT秒分のアニメーションが可能となる。ただし、このカウンタの値は実時間とは無関係である。

以上、3つの機能合成例を挙げて、3節で説明したIntelligentBoxの4つの基本機構が仮想現実感のアプリケーション開発に有効であることを示した。

## 5. まとめ

仮想現実感のアプリケーションを3次元表示された機能オブジェクトの組合せにより開発するためのシンセティック・システムとしてIntelligentBoxを開発した。IntelligentBoxの基本機構の大部分は、IntelligentPadのもつ基本機構である。仮想現実感のアプリケーションを3次元表示された部品の合成により開発する場合にも、これらの機構が変更なしに採用可能であることが分かった。これらは以下の4点である。

- 1) MPCモデリング
- 2) 親子関係
- 3) スロット結合によるデータ授受
- 4) モデル共有によるデータ送受

本稿では、これらの基本機構について解説した。また、実際に実現されている部品の機能合成例を挙げてこれらの有効性を示した。IntelligentBoxには、固有の機能をもつ部品の合成による1)構成の定義機能と、ジオメトリ情報をもつ部品の合成による2)アニメーション機能の2つの応用があることを例を挙げて示した。

IntelligentBoxでは、3次元表示されたボックス間に空間的な交わりがない場合にも親子関係を定義できる。よって、親子関係の視覚的な把握は困難である。また、モデル共有によるデータ送受の場合にも、モデルを共有しているボックス間の把握が困難である。これらは、支援ツールを開発することにより解決されると思われる。

今後は、種々の機能をもつボックスを開発・利用し、仮想現実感アプリケーションを構築する上で、問題点を検討するつもりである。また、IntelligentPadシステムとの融合を試みるつもりである。

## 参考文献

- [1] Tanaka, Y. and Imataki, T.: IntelligentPad: A Hypermedia System Allowing Functional Composition of Active Media Objects through Direct Manipulations, Proc. of the IFIP 11th World Computer Congress, San Francisco (1989), pp.541-546.
- [2] Tanaka, Y.: A Toolkit System for the Synthesis and Management of Active Media Objects, Proc. of the 1st Int'l Conf. on Deductive and Object-Oriented Databases, Kyoto (1989), pp. 259-277.
- [3] Tanaka, Y.: A Synthetic Dynamic-Media System, Proc. of International Conference on Multimedia Information Systems, Singapore (1991), pp. 299-310.
- [4] Tanaka, Y., Nagasaki, A., Akaishi, M. and Noguchi, T.: A Synthetic Media Architecture for an Object-Oriented Open Platform, Proc. of the IFIP 12th World Computer Congress, Madrid (1992), pp.104-110.
- [5] Conner, D.B., Snibbe, S.S., Herndon, K.P., Robbins, D.C. and Zeleznik, R. C. : 3D Widgets, ACM SIGGRAPH-1992 Symposium on Interactive 3D Graphics, pp.183-188.
- [6] Goldberg, A. and Robson, D.: Smalltalk-80: The Language and its Implementation, Addison Wesley, 1983.