

## バーチャル環境における3次元オブジェクト間の動的インタラクシヨンモデル

荒木 義明

慶応義塾大学政策・メディア研究科 〒252 藤沢市遠藤 5322

Email: [aly@mag.keio.ac.jp](mailto:aly@mag.keio.ac.jp) Phone: 03-3728-4359 Fax: 03-3498-9025

バーチャル環境において3次元オブジェクト同士の動的なインタラクシヨンを記述する「ボディとマグネット」モデルを提案する。ここで述べるインタラクシヨンとは、複数の3次元オブジェクト同士がそれらの形状の接触などにより、一体となって振舞うことである。しかし、3次元マルチユーザ・バーチャル環境などのオープンシステムではインタラクシヨンをを行う3次元オブジェクトが予め定めることが出来ない。そこでこのインタラクシヨンをを行う3次元オブジェクト候補を決定し、それらを動的に関連づける必要がある。このモデルにより、3次元オブジェクトの属性にあわせてインタラクシヨンの内容を動的に決定する仕組みを記述することができるようになる。

### 1. はじめに

バーチャル環境における3次元オブジェクト間のインタラクシヨンとは、複数の3次元オブジェクト同士がそれらの形状の接触などにより、一体となって振舞うことである。このインタラクシヨンの例として、人を表す3次元オブジェクトがその手を使って石を表す3次元オブジェクトを握ることや、人を表す3次元オブジェクト同士の握手などが挙げられる。一般に複数の3次元オブジェクト同士のインタラクシヨンを実現するには、キーフレームアニメーションなどによりCGデザイナーが逐一座標を指定するか、特殊なケースに限定した物理シミュレーションにより座標を生成する方法がある。

しかし、複数の3次元オブジェクト同士のインタラクシヨンを実現する状況がオープンシステム[Hewitt 84]である場合、それらの従来手法だけではインタラクシヨンを実現できない。例えば3次元マルチユーザ・バーチャル環境では、参加者を表す3次元オブジェクトが参加者の出入りの度に生成消滅し、それらの3次元オブジェクトの属性は参加者の好みの設定によって絶えず変化する。このためインタラクシヨンをを行う3次元オブジェクトの候補を決めてそれらを動的に関連づけるとともに、お互いの属性にあわせてインタラクシヨンの内容も動的に決定する機構が必要となる。

この論文では、オープンシステムの特徴を持つバーチャル環境において3次元オブジェクト同士の動的なインタラクシヨンを記述する「ボディとマグネット」モデルを提案する。このモデルの設計において、以下の二点を考慮した。第一点は、3次元オブジェクトの種類を人型などの特殊なものに限定せず柔軟な記述ができることである。第二点は、3次元オブジェクト間の動的なインタラクシヨンをういたコンテンツを従来手法を生かして容易に作成できることである。

以下第2節では、3次元オブジェクト間の動的なインタラクシヨ

ンを実現するこれまでの取組みについて紹介する。第3節では、私たちの提案するボディとマグネットのモデルを解説する。第4節では、動的なインタラクシヨンの記述方法について説明する。第5節では、モデルのシンタックスを示す。第6節では、実践的なサンプルを用いて記述実験を行う。第7節では結論と今後の方向性について議論する。

### 2. 背景

これまで動的な3次元オブジェクト間のインタラクシヨンを実現する試みは、特殊なアプリケーションに限定して行われてきた。以下では NPSNET-HUMAN [Zyda 95]、VLNET [Capin 97]、Intelligent Box [Okada 95]における取組みを紹介する。

NPSNET-HUMAN は、軍事医療の訓練を目的としたシミュレーションである。白兵戦などの激戦下での救急治療として、ユーザの仮想的な身体を負傷者の仮想的な体を接触させるインタラクシヨンをを行う。3次元オブジェクトの関係づけはユーザが仮想環境を体験しているランタイムに行う。関係づけの手法として、3次元オブジェクトの形状の衝突シミュレーションを利用している。しかし衝突シミュレーションでは3次元オブジェクトが増えるたびにの計算量が爆発的に増え、拡張性に欠ける。またインタラクシヨンを行える3次元オブジェクトの種類は人間だけに限定している。

VLNET は、エージェント技術を応用して仮想的な人間とゲームを行うバーチャル環境である。エージェントを相手にテニスをする際に、テニスボールを表す3次元オブジェクトに仮想的な人間が握るラケットを表す3次元オブジェクトを当てるインタラクシヨンをを行う。3次元オブジェクト間の関係づけはユーザが仮想環境を体験しているランタイムに行う。関係づけの手法として、人間を表す3次元オブジェクトの仮想的な視覚、聴覚、触覚を表すセンサーを利用している。衝突判定と比べてそれらのセンサー

では効率的に関係づけが行える。しかし、インタラクションできる3次元オブジェクトの種類は人間と予め決められた特定のもののだけに限定している。

Intelligent Box は、ビジュアルな対話的ソフトウェア開発ツールである Intelligent Pad [Nagasaki 94] の3次元版である。Intelligent Box ではユーザがバーチャル環境の作成時に3次元オブジェクト間の関係を定める。3次元オブジェクトはユーザ定義でき、特別なものに限定されない。また、3次元オブジェクトの関係づけするためには、ユーザが GUI を利用して3次元オブジェクトのスロットと呼ばれるインターフェース同士を結合する。しかし、3次元オブジェクト間の関係づけをユーザが仮想環境を体験しているランタイムに行うことはできない。

### 3. 「ボディとマグネット」のモデル

#### 3.1 ボディとマグネット

ボディとマグネットのモデルでは、インタラクションを行う3次元オブジェクトの種類を特殊なものに限定せず、それらを総じて「ボディ」と呼ぶ。また、二つのボディを接続するための機能を「マグネット」と呼ぶ。ただし、マグネットは、予めそれら二つのボディのどちらかの属性として定義されている必要がある。

このモデルの理解を助けるために、3次元オブジェクト同士の動的なインタラクションを、現実世界における例を用いて説明する。例えば、人が石を手を持つことを考える。人と石は別々のものなので、それぞれボディとして捉える。人が石を持つ場合には、石は手の平と密着しており、人が歩いて移動したり上腕を振り上げても石は手の平と一体となって動く。また、人は手で触れることで、石の温度やざらつきなどを知ることができる。このように手はインタラクションにおいて人のボディと石のボディを接続する重要な役割を果たしているため、ここではその手の能力をマグネットとして捉える。そして、このマグネットは人を表すボディの属性である。

#### 3.2 ボディ間の階層関係と通信経路

マグネットは、接続をするボディ間に通信経路と階層関係を与える。

階層関係は、3次元オブジェクト同士が一体となって振舞うための間合いや、位置の依存関係を表現する。人が石を手を持つ例では、石の3次元オブジェクトを人の3次元オブジェクトにおける手の階層下に定義することで、人が歩いて移動したり上腕を振り上げても石は手の平と連動して動くことを表現できる。マグネットを属性に持つ側のボディを「親」と呼び、他方のボディを「子ども」と呼ぶ。

通信経路は、3次元オブジェクト同士が一体になって振舞う内容を決めるための属性の交換や、同期した動作のタイミングや

力加減などを表現する。人が石を持つ例では、石の3次元オブジェクトから人の3次元オブジェクトへ熱や振動などの属性を伝えることで人が石の温度やざらつきを知ること表現できる。

#### 3.3 接続するボディの指定

ボディとマグネットのモデルでは、ユーザが仮想環境を体験しているランタイムに3次元オブジェクトの関係づけを行う。しかし、その関係づけの手法は特殊な3次元オブジェクト間のインタラクションをシミュレーションによるのではなく、ユーザによる指定に委ねている。つまりユーザは、ボディの接続に直接に関与する「親」側のマグネットと「子ども」側のボディをそれぞれ明示的ないしは暗示的に指定する必要がある。

この指定は、マグネットおよびボディを選び順番によって Pick リクエストと Ride リクエストに分けられる。Pick リクエストでは、「親」側のマグネットを先に、「子ども」側のボディを後に選択する。人が石を拾う例では Pick リクエストを利用し先に人の手を選んでおくことで、人が能動的に拾うことを表現できる。Ride リクエストでは「子ども」側のボディを先に、「親」側のマグネットを後に選択する。例えば人が座席を指定して車に乗ることを表現できる。この場合、車が「親」となるボディであり、人が「子ども」となるボディ、座席がマグネットとなる。人が先に選ばれることで、人が能動的に車の座席に座ることを表現できる。

### 4. コンテンツの作成方法

ボディとマグネットのモデルでは、3次元オブジェクト間の動的なインタラクションのコンテンツを従来手法を生かして簡単に作成できることも考慮している。3次元オブジェクト間の静的なインタラクションの記述から拡張することで、コンテンツを簡単に作成することができる。そこで、まず3次元オブジェクト間の静的なインタラクションを記述する言語として VRML を説明し、後にその記述を拡張する方法を説明する。

#### 4.1 VRML の記述

VRML [VRML97]とは、インタラクティブ3次元記述言語の国際標準規格である。VRML では、複数の3次元オブジェクトの静的な関係を、階層関係と通信経路により静的に記述できる。

VRML では3次元オブジェクトをノードの階層的な組み合わせとして記述する。ノードの階層を利用すれば同様に3次元オブジェクト間の階層性を表現できる。ノードには3次元物体の形状や、座標系、ロジック、ユーザからの入力に感応するセンサーなどがある。特にその内で座標系を表すノードは、他のノードを含み階層性を表現する。その階層は全体として見ればツリー状の有効グラフを形成する。図1には、五つの丸がノードを表し、それらを結ぶ線は静的な階層関係を表している。図の左手がツリー構造の親側となる。

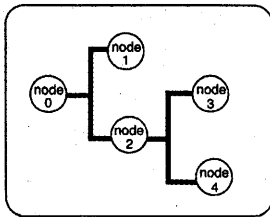


図1 静的な階層関係

また VRML では3次元オブジェクトの挙動をイベント機構により記述する。イベント機構を用いれば同様にして複数の3次元オブジェクト間の一体となった振舞いも表現できる。ノード間でイベントを受け渡しする経路のことをルート(ROUTE)と呼ぶ。ノードは他のノードとのイベントを用いた通信のための入出力インターフェースを表す「イベントイン」、「イベントアウト」を持つ。図2には、ノード1とノード3の間にルートを持つ例を示す。ルートは図中では矢印で表し、矢印の先端側のノードの突起はイベントインを表し、矢印の根元側のノードの突起はイベントアウトを表す。

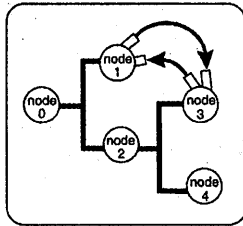


図2 静的な階層関係と通信経路

## 4.2 拡張の記述

3次元オブジェクト間の静的な関係の VRML 記述をもとにして動的な関係を持つ記述への拡張を示す。ボディおよびマグネットを VRML の拡張ノードである Body ノード、Magnet ノードとして導入する。いま、図1が二つの3次元オブジェクトの階層的な関係だけをもつ静的な記述であるとし、ノード0とノード2がそれぞれ、オブジェクトを表すノードとする。

拡張には以下の三つのステップを踏む。まずノード2をノード0の階層から切り離してお互いの関係を断ち切る。第二ステップで、それぞれの3次元オブジェクトをボディとして形式化するためにノード0およびノード2をそれぞれ Body ノード A および B の階層下に設定する。第三ステップで、親ボディ A にマグネットを定義するために、ノード2が元あった場所に Magnet ノード C を定義する。図3には、これらのボディが動的に接続したときの状況を示す。図3の波線は動的な階層関係を表しており、図1と同じようにノード2はノード0の子ども階層として機能する。

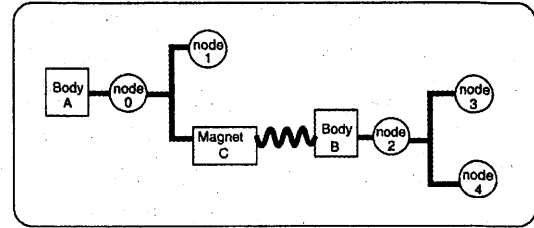


図3 動的な階層関係

次に、図2をもとにして通信経路も備えた VRML 記述の拡張を示す。先ほどと同様にノード0とノード2を分離するが、その際これらにまたがるルートも切断する。次にボディおよびマグネットを定義した後、Magnet ノードCとBody ノードBにそれぞれの切断したルート情報を保管するノード D 及びノード E を定義する。図4には、これらのボディが動的に接続した時の状況を示した。図4の波線は動的な通信経路としての役割も兼ねており、図2同様にイベントのやり取りができる。つまり図4では、ノード1で発生したイベントは、ノード D、Magnet ノード C、Body ノード B、ノード E と通ってノード3へ伝わる。

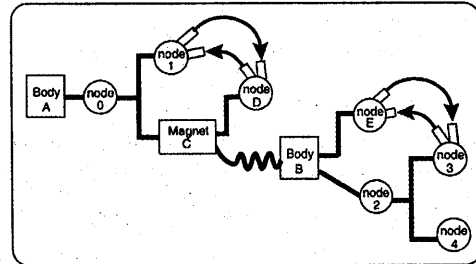


図4 動的な階層関係と通信経路

## 5. シンタックスとセマンティクス

### 5.1 Body ノードと Magnet ノード

ボディを表すための Body ノード及び、マグネットを表す Magnet ノードを示す。

コード1 Body ノードのシンタックス

#インターフェース型	データ型	属性名	値
field	MFNode	looks	NULL
field	MFNode	magnets	NULL
field	MFNode	routesWithParent	NULL
#インターフェース型	データ型	インターフェース名	
eventOut	SFBool	isChild	
eventIn	SFTime	request_pickUp	
eventIn	SFTime	request_throwAway	
eventIn	SFBool	set_bind	
eventOut	SFBool	isBound	
exposedField	SFBool	separate	

Body ノードには、ボディの属性を定義する三つのフィールド

がある。looks フィールドには、ボディの外見を表す3次元形状を定義する。magnets フィールドには、ボディの属性であるマグネットを指定する。指定されたマグネットの本体の定義は、このボディが「親」となった場合に「子ども」のボディを相対位置に配置するために looks フィールド内にある。routesWithParent フィールドには、このボディが「子ども」となった場合に「親」のボディとの間でイベント通信経路を確立するための情報を定義する。このフィールドには、後で述べるダミーノードと呼ばれるノードを設定する。

コード 2 Magnet ノードのシンタックス

#インターフェース型	データ型	属性名	値
field	SFNode	body	NULL
field	MNode	routesWithChild	NULL
#インターフェース型	データ型	インターフェース名	
eventOut	SFBool	isParent	
eventIn	SFTime	request_rideOn	
eventIn	SFTime	request_rideOff	
eventIn	SFBool	set_bind	
eventOut	SFBool	isBound	

Magnet ノードには、マグネットの属性を定義する二つのフィールドがある。body フィールドにはそのマグネットを属性に持つボディを指定する。routesWithChildren フィールドには、このマグネットを属性に持つボディが「親」となった場合にその「子ども」のボディとの間でイベント通信経路を確立するための情報を定義する。このフィールドには、後で述べるダミーノードを設定する。

## 5.2 接続するボディの指定

接続するボディの指定方法は、Pick リクエストと Ride リクエストの二種類があり、それぞれの方法に対して Magnet ノードおよび Body ノードは異なるイベントインを持つ。これらのイベントインに対して、センサーノードからのイベントを送信することでユーザによる指定を記述できる。センサーノードには、ユーザが3次元形状に触れたことに感応する TouchSensor ノードや3次元形状がユーザの目に見える場所へ現れたことに感応する VisibilitySensor ノードなどがあり、使い分けによってユーザによる指定を明示的にも暗示的なものとしても記述できる。

### 5.2.1 Pick リクエスト

Pick リクエストでは、始めにマグネットを確保し、その後でボディの選択を行い、後者のボディを「子ども」とする動的関係の確立を要求する。つまり、ある Magnet ノードの set\_bind イベントインに対して TRUE 値のイベントを送り、そのマグネットを確保する。そして、ある Body ノードの request\_pickUp イベントインに対してイベントを送り、そのボディと確保していたマグネットとを関係づける。その関係を解消する場合には、先の Body ノードの request\_throwAway イベントインに対してイベントを送る。

### 5.2.2 Ride リクエスト

Ride リクエストは、始めにボディを確保し、後でマグネットの選択を行い、前者のボディを「子ども」とする動的関係を確立する。つまり、まずある Body ノードの set\_bind イベントインに対して TRUE 値のイベントを送り、ボディを確保する。そして、ある Magnet ノードの request\_rideOn イベントインに対してイベントを送り、そのマグネットと確保しておいたボディとを関係づける。関係を解除する場合には、先の Magnet ノードの request\_rideOff イベントインに対してイベントを送る。

## 5.3 ボディ間の階層関係と通信経路の確立

ユーザによる任意の Magnet ノードと任意の Body ノードへのイベント送信によって、接続するボディを指定できる。このため最悪の事態では、あるボディがそれ自身を「子ども」とする階層関係を持つ接続を要求できる。もしも全ての指定を要求通りに許可すれば、ボディの階層関係が成す全体のグラフは、そのグラフの中の幾つかのボディの位置を決定出来ない図 5-b、5-c のような構造をもつ可能性がある。このような事態を防ぐため、常に図 5-a のようなツリー構造を保つような制約を受ける。

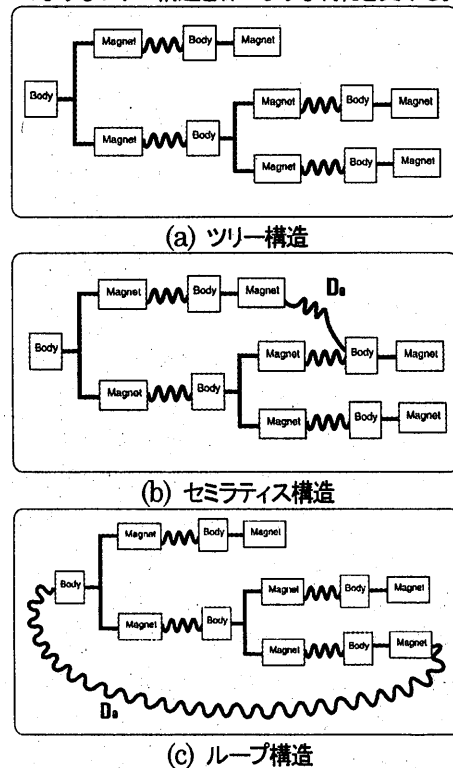


図5 動的階層関係が成すツリー形状 (a) と不正な形状 (b) (c)

接続するボディの指定が上の制約条件を満たす場合、そのボディ間で階層関係と通信経路が動的に確立する。

階層関係の確立では、「子ども」となったボディが「親」となったボディの属性の指定されたマグネットの座標系の下に瞬時に配置される。このとき、その Magnet ノードの isParent イベントア

ウトの値が TRUE をなり、「子ども」のボディを表す Body ノードの isChild イベントアウトの値が TRUE となる。これらのイベントアウトは動的関係に利用されていないときには FALSE の値を示す。また「子ども」となったボディが瞬時に移動することを防ぐためにはその Body ノードの separate フィールドを TRUE に設定する。このフィールドを巧みに利用して「子ども」のボディのスムーズな移動を記述できるだろう。

通信路の確立では、子どものボディと親側のマグネットを介してイベントのやり取りが可能となる。Magnet ノードの routesWithChild フィールドと Body ノードの routesWithParent フィールドに同型のダミーノードが設定されている場合のみ通信経路を確立する。このダミーノードはユーザ定義ノードであるが、必ず interfaces フィールドを持ち、そのフィールドの値はそのダミーノードのイクスポーズドフィールドの名前を設定する。通信経路が確立した場合、ダミーノードのイクスポーズドフィールドの保持している値が一方で変われば、対応する他方のダミーノードのイクスポーズドフィールドも連動して変わることにより、イベントのやり取りを実現する。ダミーノードはユーザ定義ノードであり、そのサンプルのシンタックスをコード3に示す。

コード3 あるダミーノードのシンタックス

Dummy {			
#インターフェース型	データ型	属性名	値
field	MFString	interfaces	["a","b"]
#インターフェース型	データ型	インターフェース名	
exposedField	SFBool	a	TRUE
exposedField	SFTime	b	0

## 6. 記述実験

実践的なサンプルを用いて「ボディとマグネット」のモデルの記述実験を行う。コンテンツの作成手順は以下の四つのステップを踏む。まずコンテンツをボディとマグネットを用いて形式化する。第二ステップではコンテンツを3次元オブジェクトの静的な関係として VRML で記述する。第三ステップで4.2節の手順に従って VRML 記述を拡張し、複数のボディを記述する。最後のステップでは、接続するボディをユーザが指定する操作手順を記述する。以下では実践的なサンプルとして「皿回し」と「シーソー」の2つのコンテンツを示す。

### 6.1 皿回しの記述

#### 6.1.1 ボディとマグネットを用いた形式化

皿回しとは、人が手を使って棒を操り、棒の先端に乗せた皿を回すパフォーマンスである。ここでは、皿回しをする人を親のボディとし、皿回し道具を子どものボディとする。また、人がその右手を使って道具を持つことができる能力をマグネットとする。また接続するボディを指定するには、Pick リクエストにより人の

手を表すマグネットを先に確保することで人が道具を能動的に持つことを表現する。

#### 6.1.2 3次元オブジェクトの静的な関係の記述

人と道具を表す参次元オブジェクト同士の静的な関係に対する VRML 記述をコード4に、そのツリー表現を図4に示す。この記述では7つの名前の付いたノードがある。HUMAN というノードは皿回しをする人を表すノードである。HUMAN の階層下には LEFTHAND というノードがあり、皿回しをする人の左手を表す。LEFTHAND の階層下には STICK というノードがあり、皿回し道具の柄の部分の部分が左手に握られていることを表す。さらに STICK の階層下には PLATE というノードがあり柄の上に皿が配置されていることを表す。この他、ANIM というノードは、左手や皿を動かす力の源を表し、START というノードは柄の部分ユーザが触れたことの検知する。また STOP というノードは皿の部分ユーザが触れたことの検知する。コード4の 20,21 行目のルートにより、ユーザが柄に触れることが切っ掛けで皿回しを開始し、皿を触ることが切っ掛けで皿回しを止めるために力の源へ命令伝達を行う。また 22,23 行目のルートによって力の源から左手や皿が動く力の伝達を行う。

コード4 VRML による皿回しの静的記述 (PlateSpinningwrl)

```

1 #VRML V2.0 utf8
2 EXTERNPROTO Humanoid [...]....
3 EXTERNPROTO Animation [...]....
4 DEF HUMAN Humanoid [
5   left-hand [
6     DEF LEFTHAND Transform {children [
7       inline {url "left-hand.wrl"}
8     DEF STICK Transform {children [
9       inline {url "stick.wrl"}
10      DEF START TouchSensor {}
11      DEF ANIM Animation {}
12      DEF PLATE Transform {children [
13        DEF STOP TouchSensor {}
14        inline {url "plate.wrl"}
15      ]}
16    ]}
17  ]}
18 ]
19 ]
20 ROUTE START.touchTime TO ANIM.startTime
21 ROUTE STOP.touchTime TO ANIM.stopTime
22 ROUTE ANIM.rotateLeft-hand TO LEFTHAND.rotation
23 ROUTE ANIM.rotatePlate TO PLATE.rotation

```

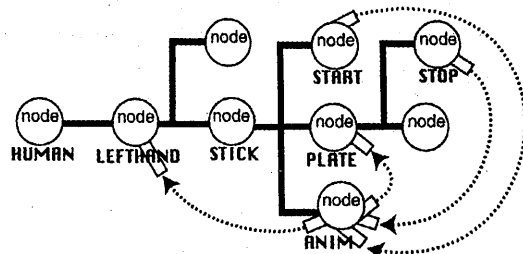


図6 皿回しにおける静的な関係の記述

### 6.1.3 手術による拡張

Body ノードと Magnet ノードを用いてコード4の VRML 記述を拡張し、2つのボディを記述する。皿回しを行う人を表すボディをコード5へ、皿回し道具のボディをコード6へ別々に定義する。

皿回しをする人を表すボディを記述するには、HUMAN というノードとその階層をコード4から調達して Body ノードの属性として定義する。また STICK というノードを Magnet ノードで置換する。その置換した際に切断したルートの情報を保管するために、その Magnet ノードの属性に Dummy ノードを定義して仮接続する。Dummy ノードの interfaces フィールドには "ANIMtoLEFTHAND" という値を設定する。

コード5 皿回しをする人を表すボディの記述 (Avatar.wrl)

```

1 #VRML V2.0 utf8
2 EXTERNPROTO Body [...]""
3 EXTERNPROTO Magnet [...]""
4 EXTERNPROTO Humnoid [...]""
5 DEF BODY_A Body{
6   looks[
7     DEF HUMAN Humnoid{
8       leftHand[
9         DEF LEFTHAND Transform{children[
10          inline{url "leftHand.wrl"}
11          DEF Magnet_C Magnet
12            body USE BODY_A
13            routesWithChild[
14            PROTO Dummy[ ##### begin proto for dummy
15              field MFString interfaces "ANIMtoLEFTHAND"
16              exposedField SFRotation ANIMtoLEFTHAND 0 0 10
17            ]##### end prot for dummy
18              DEF DUMMY_D Dummy{}
19            ]
20          ]
21        ]
22      ]
23    ]
24    DEF PICKBIND ProximitySensor{size 4 4 4}
25  ]
26  magnets USE Magnet_C
27 }
28 ROUTE DUMMY.ANIMtoLEFTHAND TO LEFTHAND.rotation
29 ROUTE PICKBIND.isActive TO MAGNET_C.set_bind

```

皿回し道具を表すボディを記述するには、STICK というノードとその階層をコード4から調達し、Body ノードの属性として定義する。調達の際に切断されたルートの情報を保管するために、その Magnet ノードの属性に Dummy ノードを定義して仮接続する。そのダミーノードの interfaces フィールドにはコード5のダミーノードの interfaces フィールドと同じ値である "ANIMtoLEFTHAND" という値を設定する。

コード6 皿回し道具を表すボディの記述 (Stick.wrl)

```

1 #VRML V2.0 utf8
2 EXTERNPROTO Body [...]""
3 DEF BODY_B Body{
4   looks[
5     DEF PLATE Transform{children[
6       inline{url "stick.wrl"}
7       DEF START TouchSensor {}
8       DEF ANIM Animation {}
9       DEF PLATE Transform{children[
10        DEF STOP TouchSensor {}
11        inline{url "plate.wrl"}
12      ]}
13    ]}
14 ]}
15 DEF PICKREQUEST VisibilitySensor {}
16 DEF BREAKREQUEST ProximitySensor{size 4 4 4}
17 ]
18 routesWithParent [
19   PROTO Dummy[
20     field MFString Interface "ANIMtoLEFTHAND"
21     exposedField SFRotation ANIMtoLEFTHAND 0 0 10
22   ]
23   DEF DUMMY_E Dummy{}
24 ]
25 ]
26 ROUTE START.touchTime TO ANIM.startTime
27 ROUTE STOP.touchTime TO ANIM.stopTime
28 ROUTE ANIM.rotateLeftHand TO DUMMY_E.ANIMtoLEFTHAND
29 ROUTE ANIM.rotatePlate TO PLATE.rotation
30 ROUTE PICKREQUEST.enterTime TO BODY_B.request_pickUp
31 ROUTE BREAKREQUEST.exitTime TO BODY_B.request_throwAway

```

### 6.1.4 動的な関係づけの記述

ユーザが接続するボディを指定する操作手順を Pick リクエストとして定める。まずユーザはナビゲーションにより人を表す3次元オブジェクトの背後に回り、その手を表すマグネットを明示的に確保する。次にユーザはナビゲーションで周りを見渡し皿回し道具を表す3次元オブジェクトがユーザの視覚に入ること、そのボディを暗示的に選択し、Pick リクエストを行う。また、ユーザがナビゲーションにより皿回し道具からある一定距離はなれることで、この接続を暗示的に解消する。

これらの操作手順を記述するために以下のセンサーノードを設置した。コード5の PICKBIND というセンサーノードはユーザと人を表す3次元オブジェクトとの間の距離を監視する。コード6の PICKREQUEST というセンサーノードは皿回し道具を表す3次元オブジェクトが描画されたことを検知する。BREAKREQUEST というセンサーノードはユーザと皿回し道具を表す3次元オブジェクトとの間の距離を監視する。

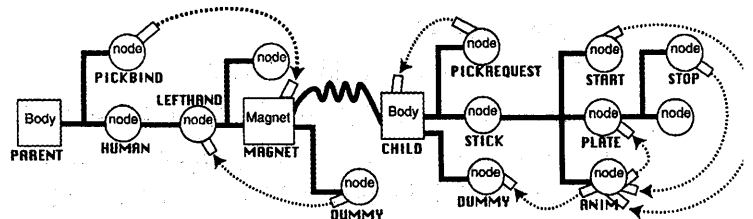


図7 皿回しにおける動的な関係の記述

### 6.1.5 他のオブジェクトとの動的関係

この皿回しは、マグネットの指定位置とダミーノードによる通信経路を正しく設定すれば、体格の異なる人を表す3次元オブジェクトにも適用できる。図8には、異なる複数の3次元オブジェクトに適用した例を示す。

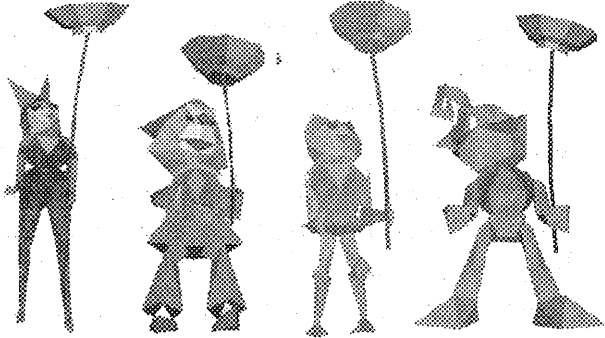


図8 異なる複数の人型3次元オブジェクトによる皿回し

## 6.2 シーソーの例

### 6.2.1 ボディとマグネットを用いた形式化

シーソーとは、二人で同一のシーソー台に乗り体重を移動し合いながら台を回転し、お互いの高さを逆転する遊びである。ここでは、シーソー台とそれに乗る二人をそれぞれボディで表す。またシーソー台の両端に人が乗れることを二つのマグネットで表す。Ride リクエストにより乗る人を表すボディを先に確保することで、人が能動的にシーソー台に乗ることを表現できる。

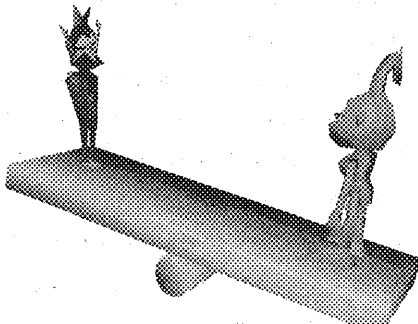


図9 二人でシーソーを漕ぐインタラクションを再現した例

### 6.2.3 次元アニメーションの静的関係の記述

人と道具を表すオブジェクト同士の静的な関係の VRML記述のツリー表現を図10に示す。この記述には7つの名前をついたノードがある。BAR というノードはシーソーの板を表すノードである。BAR の階層下には LEFTRIDE および RIGHTRIDE というノードがあり、それぞれシーソーで漕ぐ二人が乗る場所を表す。さらに LEFTRIDE の階層下には BUNNY というノードがあり、シーソーの左端にバニーガールが乗っていることを表す。また RIGHTRIDE の階層下には BLUE\_GUY というノードがあり、シーソーの右端に青い男が乗っていることを表す。この他、ANIM と

いうノードは、シーソーの板や乗って入り二人を動かす力の源を表し、DOWN\_BUNNY というノードはバニーガールをユーザが触れたことを検知する。また DOWN\_BLUEGUY というノードは青い男をユーザが触れたことを検知する。これらのノードを結ぶ図中の矢印が表すルートにより、コンテンツは以下のように動作する。もしユーザがバニーガールに触れば板はバニーガールを下げるように回転する。また逆にユーザが青い男を触るとが下になるように板は青い男を下げるように回転する。

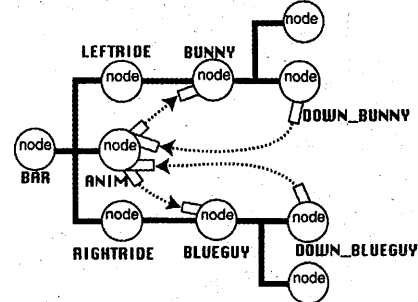


図10 シーソーにおける静的な関係の記述

### 6.2.3 手術による拡張

BodyノードとMagnetノードを利用してVRMLの記述を拡張し、三つのボディを記述する。シーソーの板を表すボディを記述するには、BAR というノードとその階層を図10から調達し、Bodyノードの属性として定義する。またその階層下の BUNNY や BLUE\_GUY というノードをそれぞれ Magnet ノードを置換する。この置換で切断されたルートの情報を保管するために、その Magnetノードの属性にダミーノードを定義して仮接続する。一方、バニーガールや青い男を表すボディを記述するには、BUNNY や BLUE\_GUY という名のノードとその階層を図10から調達し、それぞれ Body ノードの属性として定義する。調達時に切断したルートの情報を保管するためにダミーノードを定義し仮接続する。これらの三つのボディが接続する場合に図10のように通信経路を復旧するために、これらのダミーノードはすべて同型のものを利用している。図11には、それら三つのボディが接続した場合の図を示す。

### 6.2.4 動的な関係づけの記述

ユーザが接続するボディを指定する操作手順を Ride リクエストとして定める。まず、ユーザがナビゲーションにより人を表す3次元オブジェクトの背後につき、そのボディを明示的に確保する。次にユーザがポインティングデバイスによりシーソーを表す3次元オブジェクトの片方の端に触れることで、それを表すマグネットを明示的に選択し、Ride リクエストを行う。また、ユーザがナビゲーションによりシーソーから十分遠くはなれことで、この接続を暗示的に解消する。

これらの操作手順を記述するために、以下のセンサーノードを設置した。図7の RIDEBIND というセンサーノードはユーザと

人を表す3次元オブジェクトとの間の距離を監視する。また、RIDE\_LEFT および RIDE\_RIGHT というセンサーノードはユーザがポインティングデバイスを用いてシーソーの端に触れたことを検知する。RIDEOFF というセンサーノードはユーザとシーソーを表す3次元オブジェクトとの間の距離を監視する。

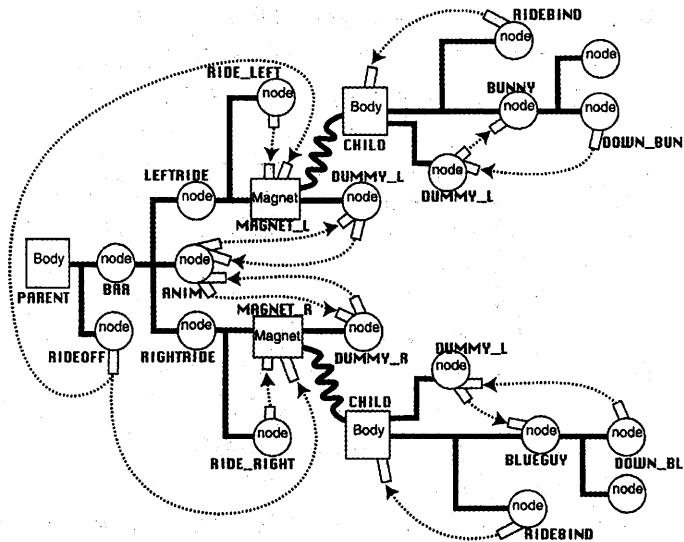


図 11 シーソーにおける動的な関係の記述

## 7. まとめ

この論文では、バーチャル環境における3次元オブジェクト間の動的なインタラクションを記述する「ボディとマグネット」のモデルを提案した。VRML を用いてボディとマグネットのシンタックスを導入し、皿回しやシーソーなどの実践的なサンプルを記述した。これらの記述を組み合わせ、一つのボディに複数のマグネットや通信経路のインターフェースを設置することで多様な3次元オブジェクトを記述できる。さらに、それぞれの3次元オブジェクトの属性に合わせてインタラクションの内容を動的に決定する仕組みを記述できるようになる。

今後の方向性は、以下の三つが挙げられる。

一つ目は、モデルのタクソミーを増やすことで用途別にインタラクションを行う3次元オブジェクトの特殊化を行う。例えばVRML を用いた人型をした3次元オブジェクトの標準化[Bernie 97]を取り入れて、人型の3次元オブジェクトにまつわるインタラクションをライブラリ化する。例えば握手のような人型同士のインタラクションや、道具を持ったり椅子に座るような人型とその環境とのインタラクションが簡単に記述できるようになるだろう。

二つ目は、シミュレーションにより生成されるような3次元オブジェクト間のインタラクションを動的に行う方法について検討する。記述実験では従来手法のうちキーフレームアニメーションで記述できるインタラクションを対象としたが、テニスゲームやキ

ャッチボール、ジャグリングのように3次元オブジェクトの位置の依存関係が動的に変わり、シミュレーションの要素を多分に含む。

三つ目は、3次元マルチユーザ・バーチャル環境における3次元オブジェクト間の動的インタラクションによる効果を研究する。この論文ではローカルなバーチャル環境を想定して行ったが、筆者によるVSPLUS[Araki 98]を用いることで、VRML によるローカルなバーチャル環境の記述を3次元マルチユーザ・バーチャル環境へ拡張できる。3次元マルチユーザ・バーチャル環境では、3次元オブジェクト同士のインタラクションを参加者の化身であるアバタの能動的な行為として位置づけることで、アバタを用いた参加者間の非言語コミュニケーションと捉えることができる。今後「ボディとマグネット」を拡張することで非言語コミュニケーションにまつわる概念を記述することができると考えられる。例えば、皿回し道具のボディの履歴を管理し、特定の参加者が操るボディだけが利用できるように認証を行うことで「所有」の概念を表すことができるだろう。またシーソーのように複数の参加者が操るボディが協調するための制約は「集団規則」の概念を表すことができるだろう。

## 参考文献

- [Zyda 95] Zyda, Michael, Pratt, David R., Pratt, Shirley, Barham, Paul and Falby, John S. "NPSNET-HUMAN: Inserting the Human into the Networked Synthetic Environment," in the Proceedings of the 13th DIS Workshop, 18 - 22 September 1995, Orlando, Florida, pp.103-106.
- [Capin 97] Capin T.K., Pandzic I.S., Noser H., Magnenat Thalmann N., Thalmann D., "Virtual Human Representation and Communication in VLNET Networked Virtual Environments", IEEE Computer Graphics and Applications, Special Issue on Multimedia Highways, March-April 1997.
- [Hewitt 84] Carl Hewitt and Peter de Jong Open Systems, Perspectives from Artificial Intelligence, Databases and Programming Languages, On Conceptual Modeling, Springer, Heidelberg, 1984.
- [VRML 97] The Virtual Reality Modeling Language, <http://www.vrml.org/Specifications/VRML97/DIS/>, Apr 4 1997.
- [Araki 98] Yoshiaki Araki, VSPLUS: A high-level multi-user extension library for interactive VRML worlds, to appear in proc of VRML98 symposium at Monterey USA, Feb 14 1998.
- [Okada 95] 岡田 義広 田中 謙, 対話型3次元ソフトウェア構築システム- Intelligent Box-, コンピュータソフトウェア Vol.12, No.4(1995), pp.88-94
- [Nagasaki 94] 長崎 称 田中 謙, シンセティック・メディア・システム-Intelligent Pad-, コンピュータソフトウェア, Vol. 11 No.1 (1994), pp.36-48
- [Bernie 97] Bernie Roehl, Specification for a Standard VRML Humanoid Version 1.0, <http://ece.uwaterloo.ca/80/~h-anim/spec.html>, August 12, 1997.