Design research of subjective-interaction for 3DCG tools

藤木, 淳

https://doi.org/10.15017/459602

第４章

OLE Coordinate System：３次元ＣＧアニメーションツールのための主観指向インタラクションデザイン

4.1 はじめに

本章では３次元ＣＧアニメーションツールのための主観指向インタラクションデザインについて述べる。ユーザが視点を変更することで、様々なだまし絵のようなアニメーションを楽しく簡単に作成できるインタラクションデザインを開発する。

「OLE Coordinate System」は、ユーザが配置したブロックや階段上を複数のキャラクタが実世界では有り得ない徘徊動作を可能にする３次元ＣＧアニメーションツールである。ユーザは視点を変更することで、キャラクタの振る舞いを操作できる。このような表現の例として、３次元空間では接続されていないブロック間に対する２次元解釈に基づいたキャラクタの移動、同一平面上での落下運動等がある。

図 4.1 に「OLE Coordinate System」の画面表示例を示す。カンバス内の画面右には、ブロック（図 4.2(a)）を配置するためのブロックツール、キャラクタ（図 4.2(b)）を配置するためのキャラクタツール、落とし穴（図 4.2(c)）を配置するための落とし穴ツール、階段（図 4.2(d)）を配置する

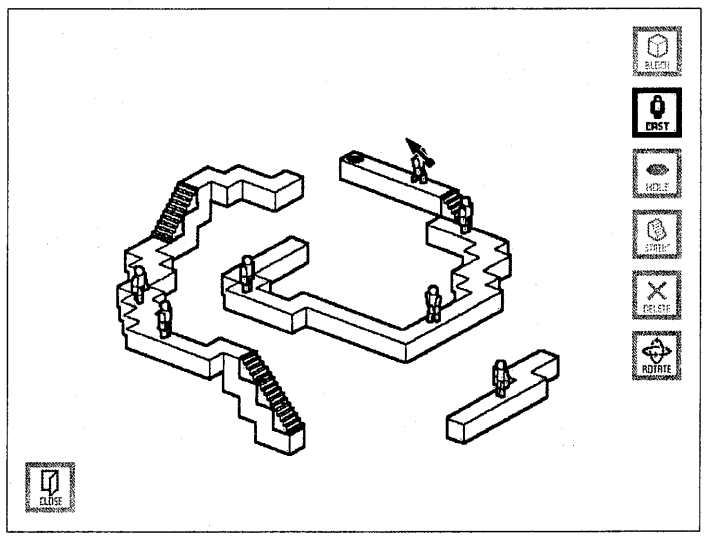


図 4.1 「OLE Coordinate System」の画面

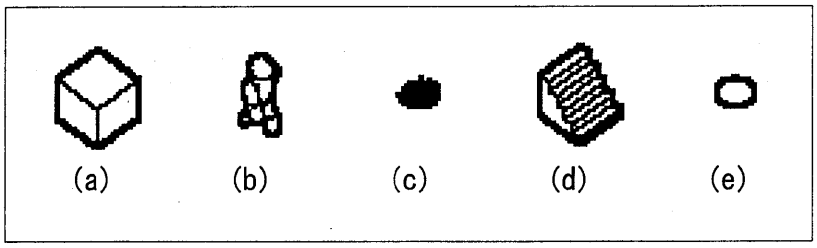


図 4.2 登場するオブジェクト

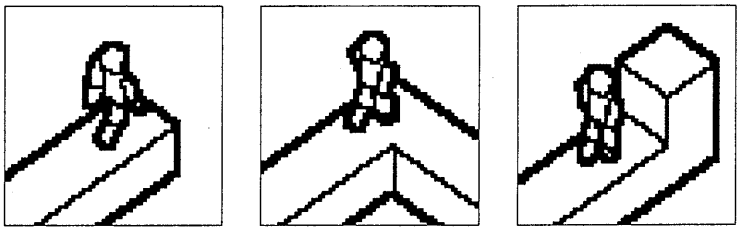
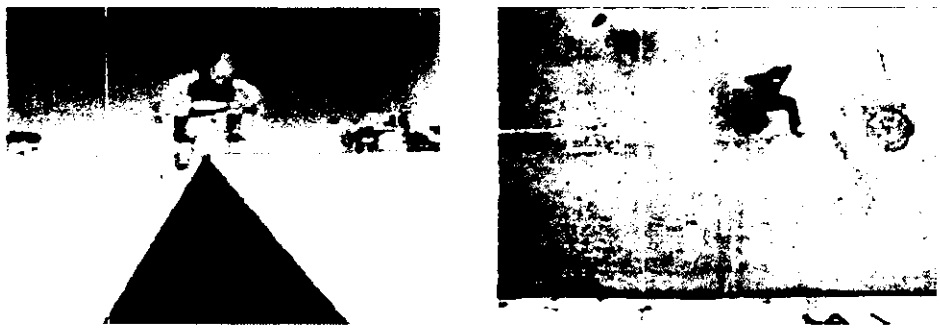


図 4.3 キャラクタが進路を変えるパターン

ための階段ツール、配置したオブジェクトを削除する削除ツール、空間を回転させるための回転ツールが並ぶ。落とし穴ツールはアイコンをマウスの右ボタンをクリックして、ジャンプ台（図 4.2(e)）を配置するためのジャンプ台ツールに切り替え可能である。画面左下のドアツールでセッションを終了する。

オブジェクトは、マウスクリックにより配置場所を指定する。マウスクリック位置にブロックがある場合、ブロックや階段はこのブロックの側面に接するように配置され、キャラクタ、落とし穴、ジャンプ台はブロックの上面をマウスクリックした時にこれに載るように配置される。それ以外は、原点を通り視線方向ベクトルに垂直な平面とマウスクリック位置から視線方向に向かうベクトルとの交点上に置かれる。キャラクタを除く、各オブジェクトはグリッドに沿って配置される。

ユーザが配置したキャラクタは、自動的にブロック上を歩きだし、移動先に階段がある場合は上り下りし、ブロック上に落とし穴がある時は落下、ジャンプ台がある時は飛び跳ねる。図 4.3 のように進路に道が無い場合や分か



(a)フリート・ハルンの「おとぎ話」　　(b)Nike 社のＴＶＣＭ

図 4.4 だまし絵アニメーションの例

お面や障害物となるブロックに当て当たった時は進行方向を変更する。このような実世界を模倣した動作に加え、「OLE Coordinate System」は実世界では起こり得ないが２次元描画されたイメージでは有り得る動作を可能とするようなアルゴリズムを用いた。ユーザは視点変更やオブジェクトの配置により、これらの振る舞いを変更することができる。

近年、３次元ＣＧアニメーションはＴＶＣＭや映画、ゲーム等において欠かせないものとなっており、そのためのツールも重要視されている。しかし、キャラクタに動きをつけるためのインタフェースは熟練したユーザであっても困難で味気ない作業となっている。アニメーション表現は、実世界を模倣した表現のみならず、誇張した表現や意外性を持つ表現などが用いられている。そのような表現を行うツールの開発を行う研究もあるが、ユーザが入力する項目が多く負担は大きい[22]。だまし絵はカートゥーンアニメーション（図4.4(a)）やＴＶＣＭ（図4.4(b)）に用いられる表現である。本研究では、アニメーション表現としてだまし絵に焦点を当て、だまし絵表現を持つアニメーションを楽しく簡単に作成可能とするインタラクションデザインを考案し「OLE Coordinate System」として開発した[44, 45]。

「OLE Coordinate System」は Microsoft 社の Visual C++ .32]を同社の Microsoft DirectX 9SDK[35]を用いて開発した。

4.2 インタラクションデザイン

視点変更から複数のだまし絵表現を操作できるインタラクションデザインを考案した。本研究では図 1.4 の M.C.エッシャーの「滝」や図 1.5 (a) の「ANIMATION OF M.C.ESCHER'S BELVEDERE」のように、閲覧者に実世界では有り得ないと解釈させる表現を対象とし、視点変更に伴う３次元形状の見え方が複数の解釈を持つことを用いる。キャラクタの振る舞いはスクリーン上に描画された２次元イメージから推測される３次元形状に基づいて決定する。３次元形状を平行投影によって描画した場合、奥行きに応じて見た目の大きさが変化することはない。この時、１つの２次元イメージから推測される３次元形状が一意に定まらず、その結果、その振る舞いが実世界では有り得ない違和感のある動作を作りだすことがある。本研究では、このようなだまし絵表現を行うためのインラクションデザインを考案した。

4.2.1 主観的移動

ユーザにより配置されたキャラクタは、不連続なオブジェクト間での移動ができる。３次元空間上でブロックや階段が実際は不連続であっても、２次元描画されたイメージでは繋がっているように視点変更した場合に連続的な動作を行う（図 4.5)。このように実際はあり得ないが描画イメージからは有り得ると解釈できる移動動作を「主観的移動」と呼ぶことにする。主観

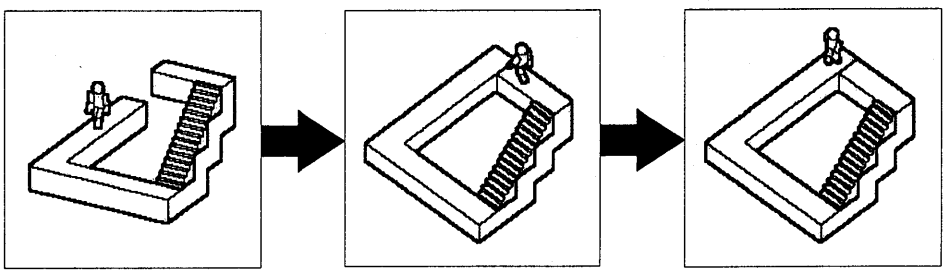


図 4.5 不連続なブロック間を渡るキャラクタ

的移動により、ユーザはキャクラタが移動する形状を不可能物体として認識する場合もある。また、後方にあるオブジェクトに前進することもできる。主観的移動を行った後に視点を回転移動して見ると、実際に離れたブロック上に移動していることが確認できる。移動中に主観的移動が成立しなくなった場合は、通常の３次元空間に対する振る舞いを行うようにした。

4.2.2 主観的着地

配置した落とし穴に到達した場合や、キャラクタが歩いているブロックが消去された場合にキャラクタは落下し、ブロックや階段に着地する。３次元空間内で実際は載っていなくても、２次元イメージにおいてキャラクタがブロックや階段に重なる位置にある場合に、ユーザは「この上に載っている」と認識することがあるように、キャラクタは実際にこれらのオブジェクト上に載ることができる（図 4.6)。ここでは、このような有り得ないが２次元イメージからは有り得ると解釈できる着地動作を「主観的着地」と呼ぶことにする。主観的着地は、同平面上の落下位置だけでなく、高さの異なる平面上への落下も可能である。これにより、落下開始位置よりも高い位置への着地も可能とする。なお、落下により画面外に出たキャラクタは消滅する。主観的着地成立後も、視点移動により着地したオブジェクト上に実際に配置されていることが確認できる。

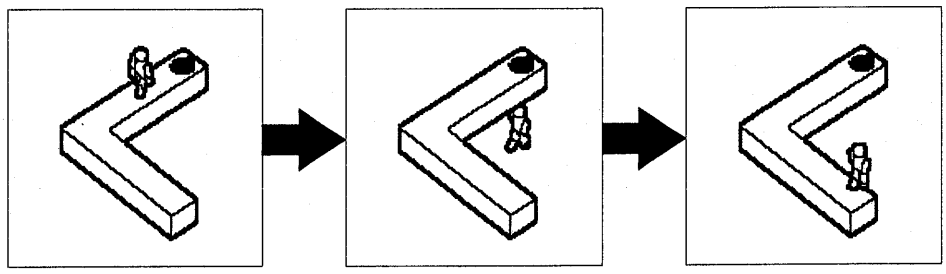


図 4.6 足元にないブロックに着地するキャラクタ

4.2.3 主観的存在

ブロックが不連続であっても、2次元イメージにおいて他のオブジェクトが不連続部分に重なって描画される場合、ユーザが「見えていない場所にブロックが存在する」と認識するようにキャラクタは振舞う（図 4.7)。このような実際は存在しないが、描画イメージからは存在しているような認識を反映した振る舞いを「主観的存在」と呼ぶことにする。主観的移動との区別として、主観的移動が移動元と移動先との2つのオブジェクトによる見かけ上の連続性を利用したものであるのに対し、主観的存在では移動元と移動先の間に第3のオブジェクトの影響により、見かけ上の連続性が成立していることがあげられる。移動中に主観的存在が成立しなくなった場合は通常の3次元空間に対する振る舞いを行うようにした。

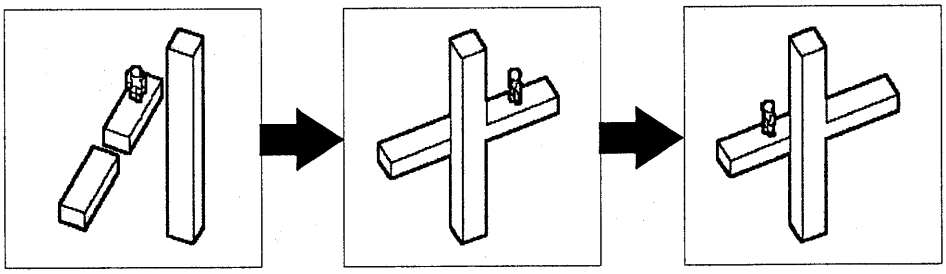


図 4.7 ブロックがあるように振る舞うキャラクタ

4.2.4 主観的不在

キャラクタは落とし穴に到達すると落下し、ジャンプ台では飛び上がるが、実際に仮想3次元空間内の落とし穴やジャンプ台に到達しても、これらが描画されない視点にある時に、ユーザが「これらのオブジェクトが存在しない」と認識するようにキャラクタは振舞う（図 4.8)。このように描画されないものの影響を無視する振る舞いを「主観的不在」と呼ぶことにする。落とし穴やジャンプ台が見えない状況には、ブロックや階段等の背後に隠れている場合の他に、下方向からの視点にある場合がある。これは、落とし穴やジャンプ台はオブジェクトとしてブロックの上面にのみ配置可能であるためで

ある。移動中に主観的不在が成立しなくなった場合は通常の３次元空間に対する振る舞いを行うようにした。

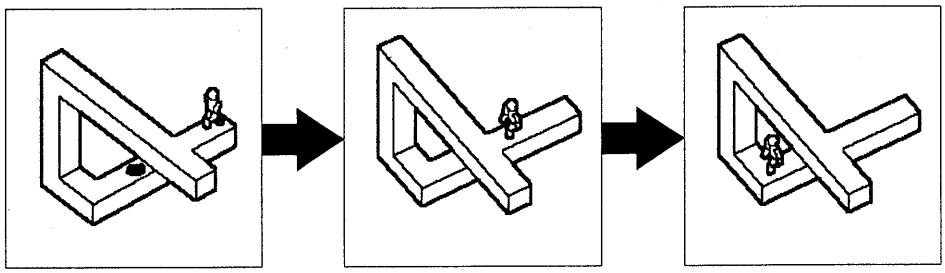


図 4.8 見えない落とし穴を無視するキャラクタ

4.2.5 主観的跳躍

キャラクタはジャンプ台に到達すると放物線を描くように飛び上がる。キャラクタの背面にオブジェクトが存在する視点にある時、キャラクタはオブジェクトの手前に位置するように描画される（図 4.9)。これにより、オブジェクトとの位置関係によっては前後に奥行きを感じさせるものとする。このようにオブジェクトとの位置関係からユーザに遠近感を与える跳躍動作を「主観的跳躍」と呼ぶことにする。他の４つの振る舞いが視点変更後の見えに対する認識を反映させるのに対し、主観的跳躍は意図してユーザに３次元として認識させる状況を作り出す点が異なる。

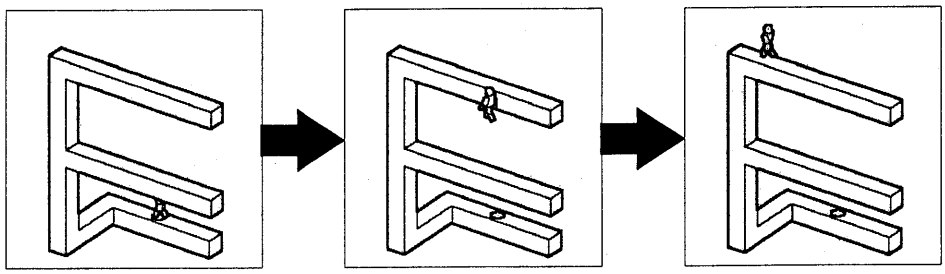


図 4.9 遠近を無視した跳躍を行うキャラクタ

4.3 実装

以下、それぞれの主観指向インタラクティブデザインを実現する技法を述べる。全体の処理の流れは「はいぱーぺいんと」と同様であり、それについては「2.3.1 システム全体の処理の流れ」を参照されたい。また、以下の全ての処理は、ステップ 8)で行われる。

4.3.1 主観的移動

キャラクタの基本動作は１ブロック分先のブロックや階段の上面の中央位置を目指して前進する。生成時のキャラクタは、キャラクタに最も近いオブジェクトを目標とする。目標に到着した場合、キャラクタの周囲４方向のオブジェクトから新しい目標を決定する。この決定には左手法[46]を用いた。左手法はロボット制御で用いられることの多い手法で、下記に示す条件分岐で記述できる。

```
IF 左方向に進行可能 THEN
        左に向く
ELSE IF 前方向に進行可能 THEN
        前に進む
ELSE IF 右方向に進行可能 THEN
        右に向く
```

キャラクタが進行できない状況として、進路上に障害物が存在する場合や道となるオブジェクトがない場合がある。ここでは、接続しているように見えるオブジェクトに対しても、この進路決定を適用可能とする仕組みをとった。具体的なプロセスを順に示す。

1) 周囲4方向のオブジェクト情報の初期化

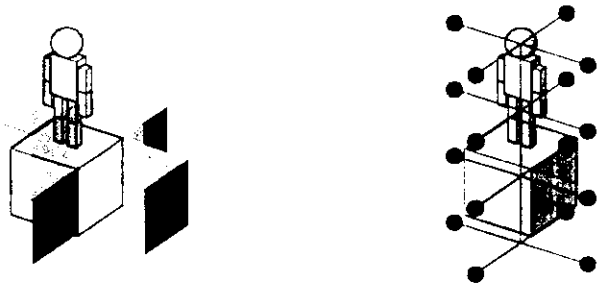
図 4.10(a)に示すようにキャラクタの前後左右4方向に高さ4ブロック分のオブジェクト情報を保持できる空間を用意する。このオブジェクトの情報を格納する空間を「主観バッファ」と呼ぶことにする。1段目の主観バッファは階段を下った先のための情報、2段目はキャラクタが載っているブロック周囲のオブジェクト情報である。本システムでは、キャラクタの高さをブロック2つ分としており、3、4段目はキャラクタの周囲情報のために用いる。初期状態として、各主観バッファに3次元位置が一致するオブジェクトの情報を格納する。

2) 隣接するオブジェクトの描画位置の算出

各主観バッファの重心のスクリーン座標を座標変換により算出する。図4.10(b)の黒丸は主観バッファの重心のスクリーン座標を示す。

3) オブジェクト情報の入れ替え

キャラクタ周囲にある可視オブジェクトの重心のスクリーン座標とステップ2)で算出した主観バッファの重心のスクリーン座標を比較し、この位置がほぼ一致する場合、主観バッファのオブジェクト情報を一致したオブジェクトの情報に書き換える。



(a) 主観バッファ　　(b) 重心のスクリーン位置

図 4.10 主観バッファのイメージ図

4) 進路決定

すべてのキャラクタ周囲のオブジェクトに処理 3)を施した主観バッファに対し、左手法を適用し進路の決定を行う。進路が決定されると、進路方向に1ブロック離れたオブジェクトを次の目標とする。

5) 目的地への移動

キャラクタの移動時、キャラクタが目標とするオブジェクト情報を持つ主観バッファに移った際に、そのオブジェクトの存在する空間側にキャラクタを移動させる（図 4.11)。この位置はキャラクタの足元位置 P1 を通る視線方向ベクトル Vc と移動目的対象とするオブジェクトの上面 T との交点 Pc である。ただし、この処理だけでは、図 4,12(a)のように主観バッファに入る直前に、手前にある移動目的対象オブジェクトに一部描画が遮られる場合がある。これを回避するため、主観バッファ横断処理を行った後で、キャラクタと前方にあるオブジェクトとの描画イメージを比較し、交差する場合、前述の処理により前方のオブジェクトの方へ移動させる（図 4.12(b)）。なお、この処理のみでは側面方向からの視点にある場合に交差領域が見つからないため、主観バッファ横断時の判定も必要である。下方向からのアングルの場合は、後方のオブジェトに遮られる方が好ましいため、後方のオブジェクトと比較し、交差している場合は後方オブジェクト側に移動させる。

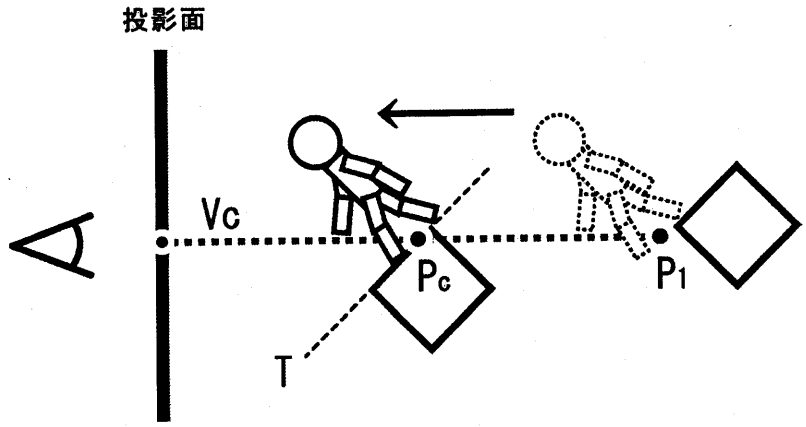
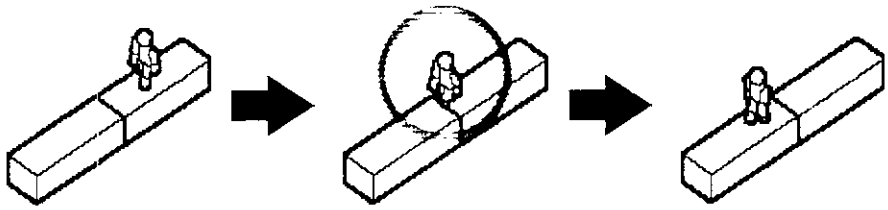


図 4.11 不連続なオブジェクトに対する移動



(a)前方の立方体に描画が遮られるキャラクタ

(b)前方の立方体との交差判定

図 4.12 不正な描画とその修正

4.3.2 主観的着地

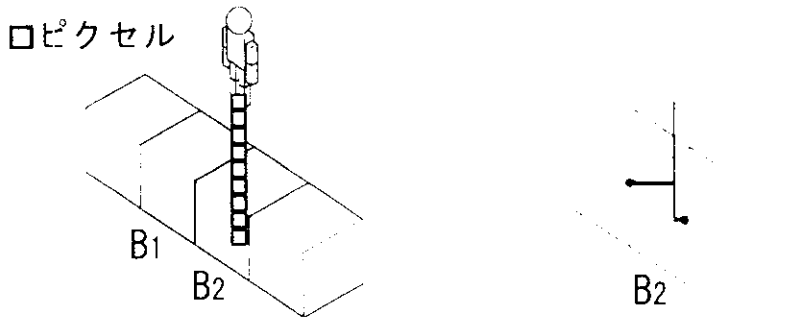
主観的着地は、可能な限りオブジェクトの中央付近に落下して見えることが好ましい。このような最も中央付近に落下可能なオブジェクトにキャラクタを着地させる手順を次に説明する。

1) 着地可能オブジェクトの探索

キャラクタの重心のスクリーン座標から下方向に数ピクセル走査して、各ピクセルを描画したオブジェクトを見つける。図 4.13(a)では2つの灰色のブロックB1、B2がこの走査により見つかったことを示している。ただし、この処理では、落下直後、直前に載っていたオブジェクトも候補として選ばれ、落下しない場合がある。そこで、キャラクタの描画が障害物により遮られる場合は、通常の落下を行うことでこれを回避する。

2) 理想的な着地可能オブジェクトの選択

着地すべきオブジェクトは中央スクリーン座標のX座標からキャラクタのスクリーン座標のX座標との距離が短い方のオブジェクトとする。図



(a) ピクセル走査　　(b) X 軸方向の距離の比較

図 4.13 理想的な着地可能オブジェクトの選定

4.13(b) では、オブジェクト B1 よりもオブジェクト B2 の方が短いため、オブジェクト B2 が着地すべきオブジェクトとなる。

3) 着地対象オブジェクト上への移動

図 4.14 に示すように、キャラクタの足元位置 P_1 を通り視線方向ベクトル Vc と着地対象オブジェクトの中央 P2 を通りオブジェクトに対して垂直且つカメラ方向を向く平面 T との交点 Pc にキャラクタを移動させた後、通常の落下移動を適用する。

以下、落下中はステップ 1) からの処理を繰り返す

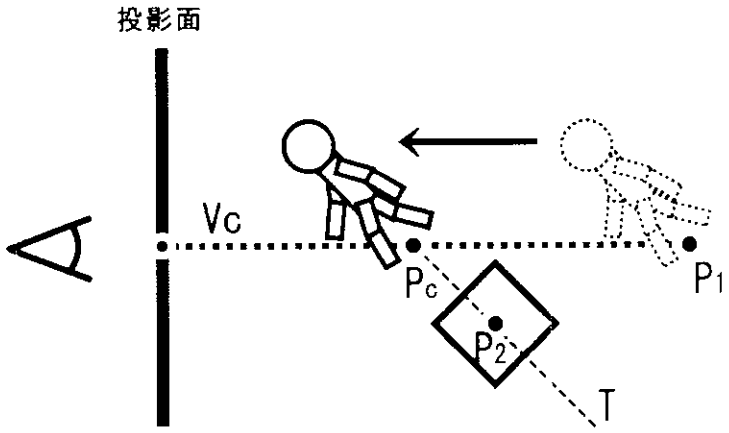


図 4.14 着地対象オブジェクトへの移動

4.3.3 主観的存在

主観的存在は、ユーザに見えない不連続な部分がオブジェクトの背後にあるように認識される場所を特定し、不連続の部分が連続であるとみなしてキャラクタを動作させる。ここでは、離れている距離をブロック１つ分、対象をブロックに限定して解決する。特定処理と移動処理を次に述べる。

1) 繋がっているように見える場所の特定

キャラクタ直下のブロックから１ブロック空いた先にブロックが存在する場合に、空いている空間にブロックがあると仮定して、仮想ブロックを描画処理する。この処理は実際には描画しない。この仮想ブロックが最前面となるピクセルを持たない時、位置対応する主観バッファに仮想ブロック情報を持たせる。進路決定時には仮想ブロックも通常のブロックとして処理する。

2) 足場のない場所に対する移動

位置対応する主観バッファ内の足場に仮想ブロックがある時、通常のブロックと同様に処理する。

4.3.4 主観的不在

落とし穴やジャンプ台の可視状態の判定は、そのオブジェクトを描画したピクセルの有無を調べることで可能である。描画ピクセルを持たない落とし穴やジャンプ台に対しては処理を無視する。

4.3.5 主観的跳躍

主観的跳躍は、上昇時に前後の奥行き判定に関係なく常にキャラクタを描画する。ただし、ジャンプ開始直前に他のオブジェクトの背後に隠れていたキャラクタが突然前面に現れるのは好ましくない。このことを考慮して、ジャンプ開始時に一部でもキャラクタの前面に描画されるオブジェクトが存在する時は、キャラクタ全体が描画される状態になるまで、この処理をしない。

4.4 評価

上記の手法を用いて実装した「OLE Coordinate System」のプロダクトを制作し、本インタラクション表現が実現可能となったが、対処できない状況も存在することが確認された。

検証として、学内外の展示で7歳から50歳までの鑑賞者に1分弱の概要説明をした後「OLE Coordinate System」を使用してもらい、行動観察及びインタビューにより本インタラクション表現の有効性を検証した（図4.15）[E14,E15]。その結果、立体形状の生成は困難なことが分かったが、その表現に多くのユーザが興味を示した。形状生成は難しいが夢中になれる、と意見したユーザもいた。もっともユーザの興味を引いた表現は主観的着地であった。一方対比主観的跳躍はあまり関心を示さなかった。「Incompatible BLOCK」との比較検証では、多くのユーザが「OLE Coordinate System」を好む結果となった。一方、3次元CG作成ツールに興味のあるユーザは「Incompatible BLOCK」を選ぶ傾向にあった。インターネットで「OLE Coordinate System」をダウンロードしたユーザからも形状生成は困難だがやりがいがあり楽しい、他のバリエーションも楽しみたい、保存機能が欲し

図4.15 「OLE Coordinate System」を使用する様子

いという意見が得られた。

また、「OLE Coordinate System」は専門家の評価を得ることができた。以下に得られたコメントをあげる。

- 複数の人形が歩道で延々と歩き回る様子をアニメーションで楽しめる。“だまし絵”を作成するうえで、歩道を見る角度も重要な要素になるので、工夫しながら全体のデザインを考える必要がある[C8]。
- ルールが決まっているようで、現実では出来ないような状況が生まれるので、やればやるほど意外なことが起こって、どんどんはまってしまいました[C9]。
- 形状の生成は難しい[C9]。
- インタラクティブにだまし絵が作れるのがすごい[C9]。
- 完成度が高い[C10]。
- 頭の中にインスタレーションが出来上がってしまうところが面白い。そして何よりシンプルなデザインに惹かれる[C10]。
- アニメーションとしてもゲームとしても評価できる[C10]。

4.5 まとめ・考察

本章では、視点を変更することで、様々なだまし絵のようなアニメーションを楽しく簡単に作成できる主観指向インタラクションデザインを考案し、プロトタイプとして「OLE Coordinate System」を制作したことを述べた。本章で用いたアルゴリズムで本インタラクション表現を実現することができたが、まだいくつかの問題がある。

５つの表現の中では、主観的跳躍がユーザの関心が低かったことが分かった。このことは他の４つの表現が視点変更インタラクションにより操作できるのに対し、主観的跳躍は視点変更によらず常に実行可能であることから、偶然性の欠如とインタラクション性の欠如が原因として考えられる。「OLE

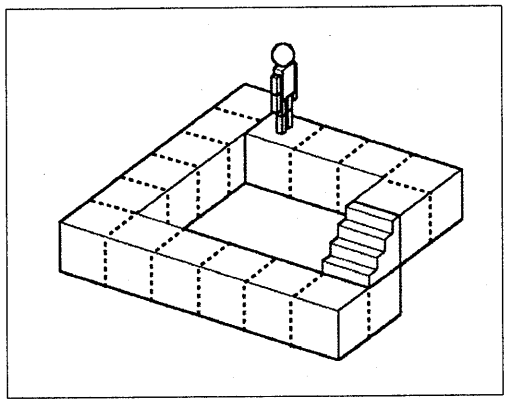


図 4.16 主観的移動の対処できない場合

Coordinate System」はプロトタイプとして形状生成操作は安易な方法を取った。「Incompatible BLOCK」のモデリングのための主観指向インタラクションデザインを用いることで改善可能となるだろう。しかし、形状生成は困難だったが、ユーザが満足を示したことにより、本インタラクション表現を用いることで楽しさによりユーザビリティ品質が向上したことを示すことが出来た。

専門家による評価は、「OLE Coordinate System」の完成度、だまし絵表現の面白さとインタラクティブにだまし絵が作れる楽しさは評価されたが、形状生成の操作性を指摘された。前述のように「Incompatible BLOCK」で用いたモデリングのための主観指向インタラクションデザインを用いることで改善したい。

また、本章で用いたアルゴリズムは問題点や改善すべき点がある。以下にそれらをあげる。

- 主観的移動

  図 4.16 に示すような主観バッファの中心のスクリーン位置とオブジェクトの中心のスクリーン位置が一致しなくても成立すべき状況が存在するが、本手法では対処できない。これを解決するアルゴリズムを考案する。

- 主観的着地

  より精度の高い方法でオブジェクトの中央位置に着地できるようにする。
- 主観的存在

  現段階では１ブロック離れたブロック間のみの対応だが、これをより離れた距離で、階段にも対応できるようにする。
- 主観的跳躍後

  落下時には好ましくない着地が起きる場合があることが確認されているので、この不具合を解決する。

全体として、今回用いた技法はアドホックな対策を取った感は否めない。本研究で最も重要とするところはインタラクションデザインとしての新規性であるが、これらの振る舞いを統一的に扱える技法は、認知工学分野に貢献できることが期待できる。これらの問題を解決していくと共に、更なる表現の追及を行っていきたい。また、本インタラクション表現と技法をアニメーション表現のみならず様々な方面へ応用していきたい。

本章の終わりに「OLE Coordinate System」を用いて制作されたモデルを図 4.16 に示す。

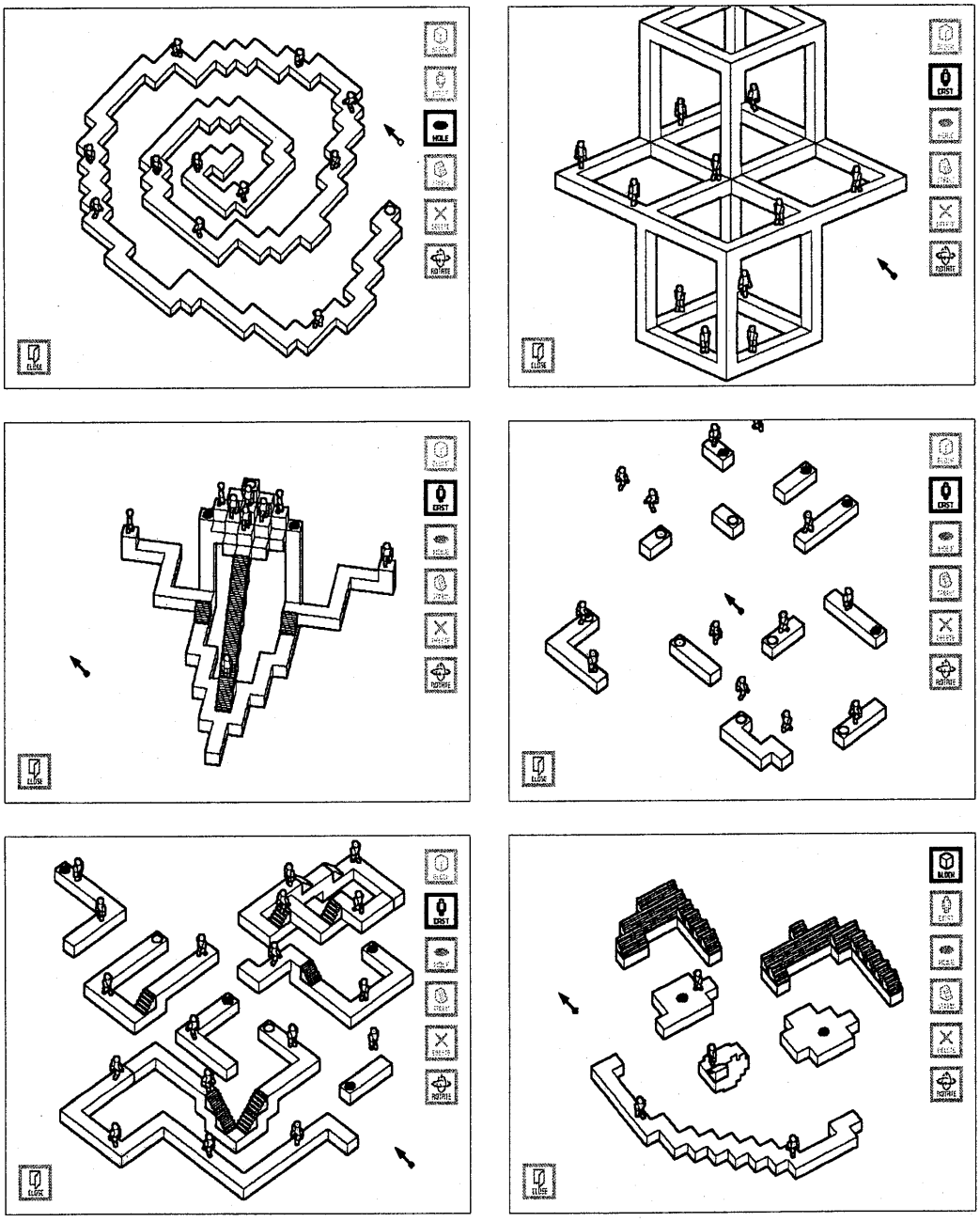


図 4.16 「OLE Coordinate System」の制作例