筑波大学大学院博士課程

システム情報工学研究科修士論文

円筒型マルチタッチインタフェースを用いた 3D ポインティング手法

内藤 真樹

(コンピュータサイエンス専攻)

指導教員　田中 二郎

2010 年 3 月

概要

本研究では、立体ディスプレイに表示された 3D コンテンツを操作するときに基本となる、ポインティング手法を開発した。

立体ディスプレイに表示された 3D コンテンツを操作する場合には、既存の入力デバイスよりも適切な入力デバイスや操作手法があると考え、3D 操作を行うためのデバイスとして円筒側面を操作面に備える円筒型マルチタッチインタフェースを入力デバイスとして採用した。

また、3D ポインティングを行う利用シーンを一人で操作する場合と、複数人で操作する場合にわけた。そして、一人で操作する場合として両手で 3D ポインティングを行う Two-handed 3D ポインティング手法と複数人で操作する場合として片手で 3D ポインティングを行う One-handed 3D ポインティング手法を開発し、その操作性を評価するための評価アプリケーションを実装した。また、評価実験によって得られたコメントより、本手法が立体ディスプレイにおける 3D ポインティング手法として有効であることが分かった。

目 次

第 1 章　はじめに **1**
1.1　背景 1
1.2　既存のポインティング手法の問題点 2
1.3　本研究の目的とアプローチ 3
1.4　本論文の構成 3

第 2 章　関連研究 **4**
2.1　レイキャスティングによる 3D ポインティング 4
2.2　その他の 3D ポインティング 5
2.3　タッチパネルを用いた 3D インタラクション 6
2.4　本研究の位置づけ 6

第 3 章　円筒型マルチタッチインタフェース **7**
3.1　システム構成 7
3.1.1　タッチパネル部 7
3.1.2　情報提示手法部 7
3.2　特徴 10
3.3　利用アプリケーション 11

第 4 章　タッチジェスチャの検討 **14**
4.1　タッチイベント 14
4.2　既存ジェスチャの分類 15
4.3　円筒型マルチタッチインタフェースにおける 3D 操作 18

第 5 章　3D ポインティングジェスチャの設計 **21**
5.1　設計の方針 21
5.2　Two-Handed 3D ポインティングの操作手法 23
5.3　操作手法の妥当性に関する議論 26
5.4　One-Handed 3D ポインティングの操作手法 26
5.5　操作手法の妥当性に関する議論 28

第 6 章　ジェスチャ認識の実装 **29**
6.1　認識に利用するデータ 29

6.2 ジェスチャの認識 30
6.2.1 Two-Handed 3D ポインティングのジェスチャ認識 30
6.2.2 One-Handed 3D ポインティングのジェスチャ認識 30

第 7 章 評価実験と考察 **33**
7.1 評価実験 33
7.1.1 実験の目的 33
7.1.2 実験用アプリケーション 33
7.1.3 実験概要 33
7.1.4 被験者 34
7.1.5 手順 34
7.2 結果と考察 35
7.2.1 アンケート結果 35
7.2.2 考察 35

第 8 章 まとめ **37**

謝辞 **38**

参考文献 **39**

図 目 次

1.1 Web ページで閲覧できる製品の 3D モデル 2
1.2 ディスプレイの表示方向とデバイスの操作方向 2

3.1 円筒型マルチタッチインタフェースの AR 利用時のシステム構成 8
3.2 AR による情報提示に利用した HMD と情報提示の様子 9
3.3 円筒型マルチタッチインタフェースの背面投影利用時のシステム構成 9
3.4 背面投影のモデル 9
3.5 背面投影の様子 10
3.6 平面と円筒曲面によって異なる操作対象への操作 11
3.7 3D モデルビューアーの操作 12
3.8 ウォークスルーアプリケーションの操作 12
3.9 平面と円筒曲面によって異なる操作対象への操作 13

4.1 タップジェスチャ 15
4.2 移動ジェスチャ 16
4.3 ストロークジェスチャ 16
4.4 ピンチインジェスチャ 17
4.5 回転ジェスチャ 17
4.6 平面上に 2 点を指定した際に生成される直線の例 18
4.7 円筒上に 2 点を指定した際に生成される直線の例 19
4.8 平面上に 3 点を指定した際に生成される平面の例 19
4.9 円筒上に 3 点を指定した際に生成される平面の例 20

5.1 円筒型マルチタッチインタフェースの操作面とその仮想空間へのマッピング . 23
5.2 Two-Handed 3D ポインティングによるスフィアカーソルの初期表示位置 . . . 24
5.3 Two-Handed 3D ポインティングによる上方向へのスフィアカーソルの移動 . . 24
5.4 Two-Handed 3D ポインティングによる手前方向へのスフィアカーソルの移動 24
5.5 Two-Handed 3D ポインティングによる左方向へのスフィアカーソルの移動 . . 24
5.6 Two-Handed 3D ポインティングによるスフィアカーソルの拡大 25
5.7 円筒型マルチタッチインタフェースを操作する際の両手の稼働域 25
5.8 両手の中点のみを使って水平移動を行う場合の手の位置 25
5.9 One-Handed 3D ポインティングによるスフィアカーソルの初期表示位置 . . . 27

5.10 奥方向へのスフィアカーソルの移動 27
5.11 One-Handed 3D ポインティングによるスフィアカーソルの移動方向の変更 . . 27
5.12 One-Handed 3D ポインティングによるスフィアカーソルの拡大 27

6.1 イベントとしてアプリケーションに渡されるデータ 29

7.1 実験の様子 34

第1章　はじめに

1.1　背景

3D 技術の発展に伴い、3D コンテンツは我々にとって非常に身近なものとなっている。たとえば、従来の Web ページにおける自動車や携帯電話といった製品の紹介には、詳細な製品の様子を示すためにさまざまなアングルから撮影した製品の写真を掲載していた。しかし昨今では図 1.1 のように製品の 3D モデルを表示し、閲覧者が自由にその 3D モデルを閲覧することができる Web ページも増えてきている。他にも、3D コンテンツを利用した映画やゲームなども増えてきており、3D コンテンツは日常的に利用するコンテンツとなっている。

また、このような 3D コンテンツの増加に合わせて、これらを表示するためのデバイスとして、一般的な平面ディスプレイに 3D コンテンツを表示するのではなく、実際に立体的にオブジェクトを表示する立体ディスプレイが各社電機メーカー [OHH06, son] から発表されている。これにより 3D コンテンツが今以上に普及し、一般的な家庭やオフィスなどにおいて 3D コンテンツとインタラクションを行う機会はより一層増加すると考えられる。

ここで挙げた 3D インタラクションとは、大きく「選択」、「ナビゲーション」、「操作」、「データ入力」の 4 種類に分類される [FB09]。「選択」とは表示されている 3D オブジェクトを選ぶことであり、「ナビゲーション」とは 3D 空間を表示するためのカメラ移動のことである。また、「操作」とは 3D オブジェクトに変更を加えることであり、「データ入力」とは数値やパラメータなどの入力を行うことである。

これらのインタラクションを行うにあたり、3D 空間の座標を指定するという基本動作が必要となる。例えば「選択」を行うには、表示されている 3D コンテンツの中にある選択対象の座標を指定する必要がある。また、「ナビゲーション」を行うためには、3D 空間中の移動先の座標を指定したり、カメラの注視点を指示するために座標を指定する必要がある。「操作」について考えれば、まず操作対象の 3D オブジェクトを「選択」する必要がある。また、「データ入力」についても、何もない 3D 空間にデータを入力することはなく、データを入力する入力先の「選択」を行う必要がある。このように 3D 空間の座標を指定するという操作は、3D インタラクションを行うために重要であると考えられる。以降では、3D 空間の座標や、一部の空間を指定することを 3D ポインティングと呼ぶことにする。

[1]2009 嵐× au 秋冬のニューモデル新登場! ARASHI × NEW COLLECTION 2009 AUTUMN & WINTER au by KDDI - http://www.au.kddi.com/collection/09autumn/?cid=66201#/ca003/3d/

[2]TCV: TOYOTA CAR VIEWER - http://toyota.jp/tcv/camry/200901/index.html

図 1.1: Web ページで閲覧できる製品の 3D モデル
（左：携帯電話の例 [1]、右：自動車の例 [2]）

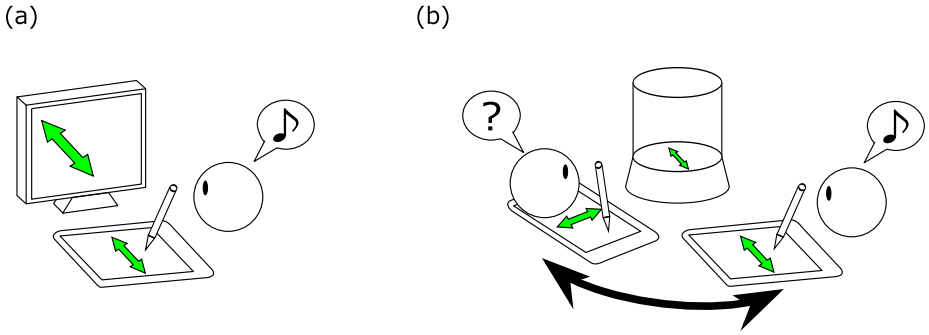


図 1.2: ディスプレイの表示方向とデバイスの操作方向

1.2 既存のポインティング手法の問題点

既存のマウスやスタイラスのような 2 次元入力デバイスを用いた 3D ポインティングについては、すでに一般的なアプリケーションで実用化されている。たとえば単純な 3D ポインティング手法として、「マウスカーソルを使って平面ディスプレイの表示面に対応する X-Y 座標を指定する。その後、奥行きをマウススクロールなどを使い指定し、3D 座標を指定する」や、「平面ディスプレイに 3 面図を表示し、複数の図面の位置を指定することで 3D 座標を指定する」といった手法が採られている。

しかし、このような操作手法において 2 次元入力デバイスによって入力できる座標軸は常に固定されている。これは、図 1.2 左のように位置を固定した平面ディスプレイを操作者が移動せずに操作を行う場合においては、問題なく利用できる。しかし、立体ディスプレイにおいては 3D コンテンツを好きな方向から閲覧できるといった特徴があり、操作者がある一定の方向からディスプレイを閲覧しているとは考えにくい。このため、図 1.2 右のように 2 次元入力デバイスを用いた既存の操作方法は操作者の閲覧方向とデバイスの操作方向がずれ、立体ディスプレイには向いていないと考えられる。

1.3　本研究の目的とアプローチ

本研究では立体ディスプレイにおける 3D ポインティング手法を開発することを目的とする。

立体ディスプレイにおいては、既存の入力デバイスよりも適切な入力デバイスや操作手法があると考え、3D インタラクションを行うためのデバイスとして円筒側面を操作面に備える円筒型マルチタッチインタフェースを入力デバイスとして採用した。円筒型マルチタッチインタフェースを用いることで、側面から立体ディスプレイに向けて操作することができるようになる。

本研究では、立体ディスプレイと円筒型 を組み合わせて利用する際に、一人で利用する場合と、複数人で利用する場合の二つのシーンがあると考えた。それぞれに有効だと考えられる Two-handed 3D ポインティングと One-handed 3D ポインティングの二つの手法を開発する。

1.4　本論文の構成

第2章　関連研究

本章では、関連研究について紹介し、本研究の位置付けを明らかにする。2.1 節では 3D ポインティングのもっとも主要なグループであるレイキャスティング手法についての研究を述べ、2.2 節では、レイキャスティング手法に分類されないポインティング手法について述べる。また、2.3 節では本研究において用いる操作デバイスと同様に、タッチインタフェースを用いて 3D インタラクションを行っている研究について述べる。

2.1　レイキャスティングによる 3D ポインティング

3D オブジェクトを選択するにあたり、操作者が 3D 空間に線分を定義し、その線分と選択対象となるオブジェクトを交差させることで 3D オブジェクトを選択するといった手法がいくつかある。

もっとも単純なポインティング手法として Mine らのレイキャスティングと呼ばれる手法がある [Min95]。この手法において操作者は 3D 位置と向きを取得できる 6 自由度センサを利用して 3D 空間内の位置と向きを入力し、3D 空間中に線分を定義する。そしてその線分と交差する 3D オブジェクトを選択対象とするものである。しかし、この方法では 3D 空間中に存在する複数のオブジェクトと線分が交差することがあるため、一般的には操作者にもっとも近い 3D オブジェクトが選択されるように設計されている。

また、レイキャスティング手法はレイの傾きに 6 自由度センサの向きを用いているが、手ぶれや傾き検知の精度によって正確にポインティングできないことがある。そこで Mine らはレイキャスティングを 3D 座標を取得できる 3 自由度のセンサを 2 つ利用する手法を提案し、レイキャスティング手法を改良している [MBS97]。傾きよりも 3D 位置のほうが精度よくセンシングできるため、操作者が両手にそれぞれ取り付けたセンサから 2 点の 3D 座標を取得してベクトルの定義を行い、精度の向上が行われている。

これらの 2 つの方法は選択に線分を利用するため、3D オブジェクトと正確に交差させる必要がある。そこで、Liang らは選択に利用する形状として線分ではなく円錐を利用するフラッシュライトと呼ばれる手法を開発した [LG94]。フラッシュライト手法ではレイキャスティング手法と同様に 6 自由度のセンサを利用し、円錐の頂点座標と、円錐の伸びる方向を入力する。レイキャスティング手法においてはレイと 3D オブジェクトの交差に正確性が求められたが、フラッシュライト手法においては 3D オブジェクトの選択は円錐型のボリュームとの交差によって行う。これにより、操作者は 3D オブジェクトの選択を大まかに行うことが可能となり、レイキャスティング手法よりも利便性を向上させている。しかし、この手法においても

複数の 3D オブジェクトが円錐と交差する可能性がある。その場合は円錐の中心線により近い 3D オブジェクトが選択対象となり、複数の 3D オブジェクトが中心線と同じ距離にある場合は操作者からもっとも近いものを選択する。

しかし、フラッシュライト手法は 3D 空間中に 3D オブジェクトが大量にあるときに円錐と多くのオブジェクトが交差し、選択ミスが多発する。そこで、Forsberg らはフラッシュライト手法におけるこの問題を解決するためにアパーチャテクニックを提案している [FHZ96]。アパーチャテクニックでは、フラッシュライト手法で用いる円錐の広がりをインタラクティブに変化させることでレイキャスティング手法における細かい選択とフラッシュライト手法における大まかな選択を操作者が動的に選択することができる。

また、Grossman らのボリュームディスプレイに対してレイキャスティングを行っている研究がある [GWB04]。彼らは、3D 位置と向きを取得できる 6 自由度センサを指に装着している。そして、その指をつかって立体ディスプレイの内部に指さしを行うことによって、レイを定義し、立体ディスプレイに表示された 3D コンテンツの選択を行っている。さらに、3D 位置と向きを取得できる 6 自由度センサを指ではなく、ボタン付きのデバイスに取り付けてレイキャスティングを行っている研究もある [GB06, GB08]。この研究では、レイキャスティングにおいて問題となっている、3D オブジェクトとレイが複数交差した場合に、ジェスチャや交差オブジェクトの再配置によって選択対象のオブジェクトを選択するための手法について述べられている。

2.2 その他の 3D ポインティング

レイキャスティングをベースにしたポインティングの他にも 3D ポインティングを行う手法がある。

仮想空間にマッピングされた実空間を指示具を用いて、指し示すことでポインティングを行う手法として、清川らのシステムがある。このシステムにおいて操作者は 6 自由度のセンサつきコントローラを両手に持ち、このコントローラを用いて仮想オブジェクトであるブロックの組み立てを行う [清川 25]。また、KATO らの研究では AR 技術を用いることで仮想空間と実空間の対応付けを行っており、操作者が実空間上で paddle と呼ばれる指示具を移動させることによってポインティングを行っている [KBP$^+$]。

また、Benko らは紐につながった浮力を持つ風船のメタファを用いたポインティング手法を開発している [BF]。この手法においては、操作者はタッチパネルを使って X-Y 座標を片手を用いて入力し、Z 座標をもう一方の手との距離の変化量として入力を行っている。

Lapides らは水平に設置されたタブレットを上下に並行移動するシステムを用いており、タブレットによるポインティングで x-y 座標を入力し、タブレットの上下位置で z 軸の入力を行うデバイスを開発している [LSSS]。

2.3 タッチパネルを用いた 3D インタラクション

平面型のタッチパネルを用いた 3D 閲覧手法について Reisman らの研究がある [RDH]。この研究では、手で物を持ちながら回転させるというメタファを用いて、複数の点でタッチパネルに表示されている 3D オブジェクトを押さえて動かすことで 3 次元的な回転を行っている。また、Gingold らは、平面型タッチパネルを用いて 3D オブジェクトへのテクスチャマッピングを行う手法を開発している [GDHZ06]。この研究もタッチパネルに表示されている 3D オブジェクトを押さえることによって、マッピングをするテクスチャの位置の決定を行っている。

また、平面型タッチパネルを箱状に配置することで 3D インタラクションを行うシステムもある [dlRKOD08]。このシステムでは、正面のタッチパネルの操作が x-y 平面の入力に制限されるが、左右や上部にあるタッチパネルから入力を行うことによって X-Z 平面や、Z-Y 平面の入力を行うことができ、3D 操作を行っている。

また、非平面型のタッチパネルを用いた研究がある [BWB08, CCY+07]。これらの研究は球状のタッチパネルを用いた 3D インタラクションについて述べている。

2.4 本研究の位置づけ

2.1 節、2.2 節で述べた 3D ポインティングおよび 3D インタラクション手法は操作を行うために 6 自由度センサのついたデバイスや指示具を用いている。このようなデバイスを必要とする操作方法は、複数人で利用するような場合においては、操作者の人数分のセンサが必要となる。これは本研究で立体ディスプレイを操作すると想定している利用シーンの「複数人で利用する場合」おいては不適切であると考えられる。しかし、本研究においては入力デバイスとして円筒型マルチタッチインタフェースというデバイスを用いることでこの問題を解決する。

また、2.1 節で述べたようなポインティング手法は、すでにあるオブジェクトに対して選択を行うことを目的としている。しかし、この方法では選択した 3D オブジェクトを新しい 3D 座標に移動するような操作を行う場合において任意の座標を選択することができない。本研究では 3D 空間中の任意の座標をポインティングすることを可能とし、この問題を解決する。

また、2.2 節で述べたような仮想空間を実空間にマッピングし、実空間上で直接ポインティングを行う手法は、操作者にとってもっとも直感的に 3D 空間のオブジェクトを選択することが可能な手法である。しかし、現在の多くの立体ディスプレイは体積を用いて立体を表示する手法をとっており、実際の表示空間は直接触ることができないため、これらの手法を適用することはできない。また、これらのディスプレイと同程度のスペースを確保したうえで、新たに操作空間をマッピングすることも可能ではあるが、空間効率の面で有効ではないと考えられる。そこで本研究では立体ディスプレイに表示された 3D コンテンツを包み込むようなタッチインタフェースである円筒型マルチタッチインタフェースを用いることで解決している。

2.3 節で述べたインタラクション手法は、タッチインタフェースを用いて 3D インタラクションを行ってはいるが、3D ポインティングは行っていないので本研究と異なる。

第3章　円筒型マルチタッチインタフェース

本研究において3D操作を可能とするマルチタッチインタフェースとして円筒曲面を操作面とする円筒型マルチタッチインタフェースを用いる。本章では、円筒型マルチタッチインタフェースのシステム構成と、その特徴を述べる。

3.1　システム構成

3.1.1　タッチパネル部

円筒型マルチタッチインタフェースのタッチパネルは、操作者に自然な操作を提供するために以下のように設計されている。

- 操作者が特殊な装置を身に着けずに操作が行える
- 複数の接触点のクラスタリングを行える

これらの特徴は、Hanによって示されたFTIR (frustrated total internal reflection)[Han05]に基づき設計し、実現した。これにより操作者は何も装着することなくタッチインタフェースを利用することができる。タッチパネルの大きさは、直径600mm、高さ700mmの円筒型である。

タッチパネルへの接触点は、距離尺度を基にしてクラスタリングされる。ある程度の範囲にまとまって存在する接触点を1つのクラスタとして扱うことにより、ある手の複数の接触点は1つの手による接触点として認識される。よってクラスタに含まれる接触点の移動方向などを解析することで、手ごとの操作を識別することができる。

タッチパネルが円筒型なので、平面型のマルチタッチインタフェースと異なり、左右へ作業領域が途切れない。これにより、操作者はデバイスの周りを360度自由に動き、任意の場所から操作を行うことができる。

3.1.2　情報提示手法部

操作者にHMD（Head Mounted Display）などの情報表示デバイスを装着させることなく全方位から視認可能な3次元表示を行うために、立体ディスプレイを円筒中心に設置することが望ましい。しかし立体ディスプレイは各種電機メーカから発表されているが、まだ販売されているものは少なく入手することは難しい。そこでこのインタフェースでは、将来的に立

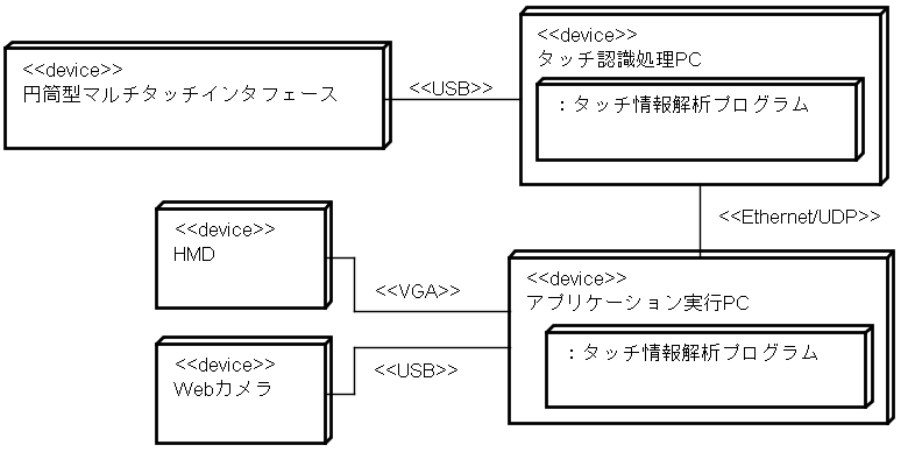


図 3.1: 円筒型マルチタッチインタフェースの AR 利用時のシステム構成

体ディスプレイが安価で高性能になることを前提に、立体ディスプレイを内部に置いた状態をシミュレートするために 2 種類の情報提示方法を実現している。

AR 技術による情報提示

インタフェースの内部に立体ディスプレイを設置した状態を想定し、3D 環境における情報提示をシミュレートするために AR(Augmented Reality) 技術を用いた。HMD に Web カメラを取り付け、取り付けた Web カメラの撮影画像に 3D オブジェクトを合成し HMD に表示をおこなっている。これにより、3D オブジェクトが円筒内に存在するかのように 3D オブジェクトを重畳表示できる。3D オブジェクトの描画位置の調整は、円筒中心底部に置いたマーカーを HMD に搭載した Web カメラで撮影し、認識することによって行う。本研究ではインタフェースの操作者が操作を行う際にマーカーを手で隠してしまうことがあるため、Web カメラがマーカーを見失った際は、最後にマーカーを認識した位置に 3D オブジェクトを表示している。図 3.1 に、AR による情報提示を行う際のシステム構成図を示し、図 3.2 に実際に利用した HMD と、操作者に提示している情報の様子を示す。

背面投影による情報提示

立体ディスプレイによる表示をシミュレートするには上述した AR 環境を用いる手法のほうがより想定した環境に近づくが、複数人で同時に利用する状態をテストするには、操作者一人ずつに HMD が必要となってしまう。そこで簡易な情報提示手法として円筒の内部にスクリーンとしてトレーシングペーパーを貼り、円筒後方に設置したプロジェクタから背面投影を行った（図 3.4）。実際に投影を行って操作を行っている様子を図 3.5 に示す。

図 3.2: AR による情報提示に利用した HMD と情報提示の様子

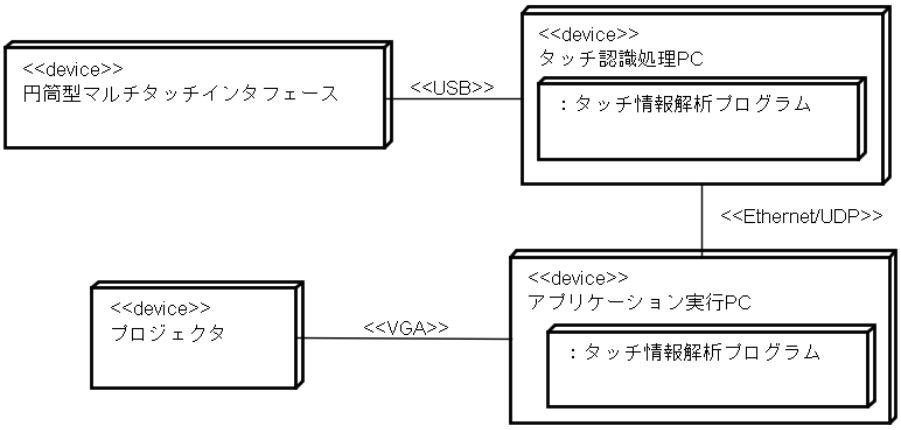


図 3.3: 円筒型マルチタッチインタフェースの背面投影利用時のシステム構成

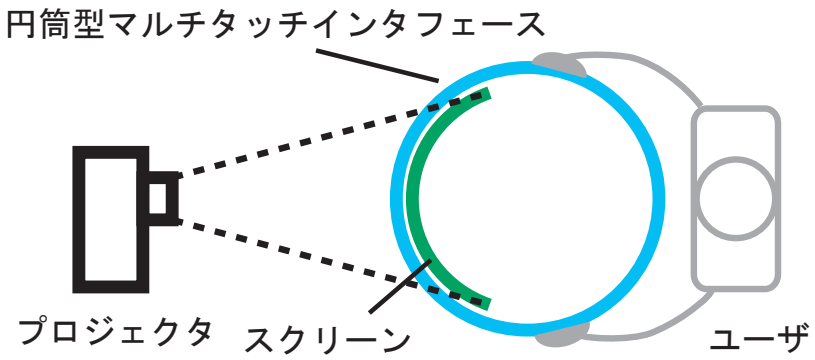


図 3.4: 背面投影のモデル

図 3.5: 背面投影の様子

3.2 特徴

上述のシステム構成により、円筒型マルチタッチインタフェースは以下の特徴を持つ。

- 奥行きを利用した 3 次元的な操作性

  入力に使用する操作面は曲面という 2 次元であるが、操作面自体が曲がっているため、操作者は操作対象への任意の方向から操作を行うことができる。例えば操作面が平面形状であれば、どの位置で操作を行っても 2 次元平面に対する操作となる (図 3.6a)。しかし、円筒型の操作面から操作を行えば、その操作位置に対しての方向から操作を行うことができる (図 3.6b)。また、このインタフェース自体が奥行きを持っていることにより、従来の平面型マルチタッチインタフェースにおいては “触る” と表現される動作に加えて両手で “はさむ” といった操作も可能になる。

- 作業領域の連続性

  円筒型であることから作業領域が左右方向に途切れない。これにより、立体ディスプレイを使用する際に操作者がデバイスの周りを 360 度自由に動く場合においても、操作者は全方位からインタフェースを操作することが可能になる。

これらの特徴を利用することにより、平面型のマルチタッチインタフェースよりも、3D オブジェクトの閲覧操作や編集操作を容易にすることができると考えられる。

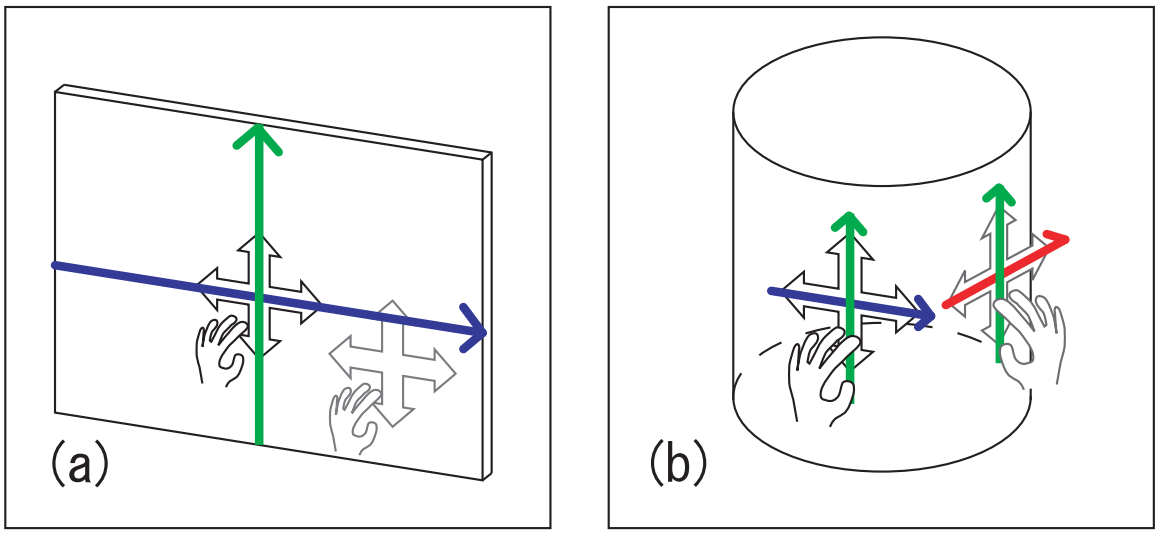


図 3.6: 平面と円筒曲面によって異なる操作対象への操作

3.3 利用アプリケーション

円筒型マルチタッチインタフェースを用いて操作を行うアプリケーションとしていくつか実装を行っている [内藤 08a, 内藤 08b, SNT]。3D モデルを閲覧する 3D モデルビューアを図 3.7 に示し、3D 空間のウォークスルーを行うアプリケーションを図 3.8 示す。また、伊藤らの大規模データの可視化手法 [IMT09] により表示されたスフィアアンカーマップを円筒型マルチタッチインタフェースを用いて操作を行っている様子を図 3.9 に示す。

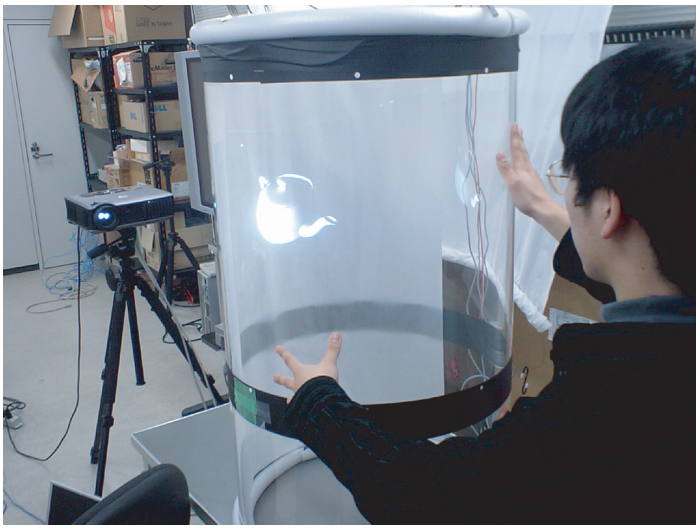


図 3.7: 3D モデルビューアーの操作

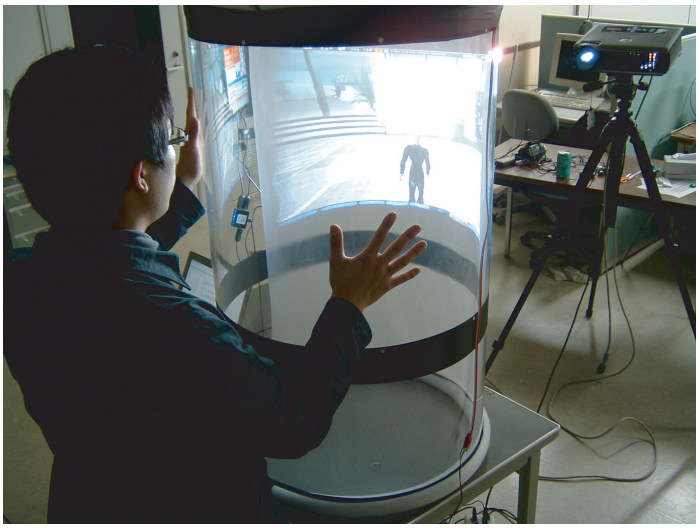


図 3.8: ウォークスルーアプリケーションの操作

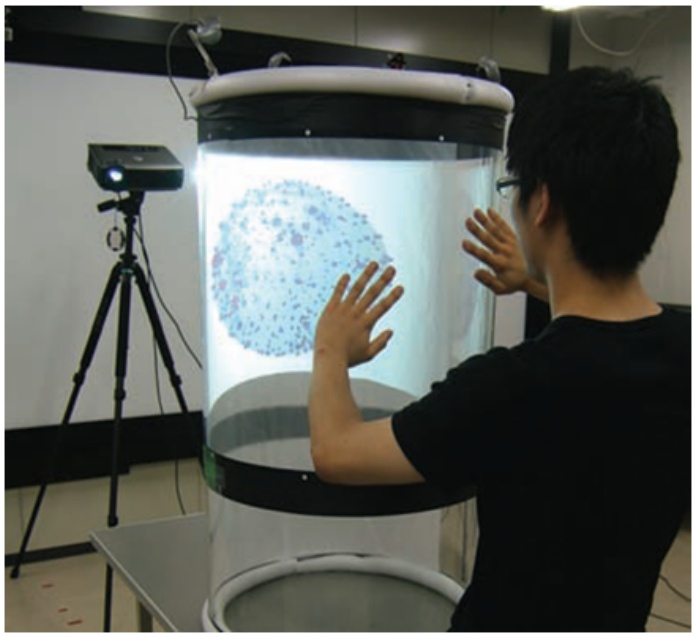


図 3.9: 平面と円筒曲面によって異なる操作対象への操作

第4章　タッチジェスチャの検討

ジェスチャとは、身体を利用した日常生活中の非言語的コミュニケーションの一つである。操作者が指や手などを用いて日常的に行う動作を入力に用いることによって、操作者は自然な操作を行うことができる。本論文ではタッチインタフェースにおけるジェスチャ操作をタッチジェスチャと呼ぶ。

本章では既存のタッチインタフェースに用いられているタッチジェスチャの分類を行うとともに、円筒型マルチタッチインタフェースにおける 3D タッチジェスチャについて述べる。

4.1　タッチイベント

タッチインタフェースを利用するシステムがタッチジェスチャを認識するために、タッチインタフェースは操作者のタッチインタフェースへの操作をタッチイベントと呼ばれるイベントとして処理する。ここではタッチジェスチャを認識するために必要となるタッチイベントについて説明する。

基本的なタッチインタフェースには以下に挙げる 3 つのイベントが処理される。

開始イベント

タッチインタフェースを指で触った時に発行される。

終了イベント

タッチインタフェースから指を離した時に発行される。

移動イベント

タッチインタフェースを触っている指をスクリーン上で移動したときに発行される。

また、デバイスによってはタッチインタフェースに指を近づけたことを認識し、ホバーイベントを発行するものがあるが、すべてのタッチインタフェースにおいて認識できるわけではないので、今回は考慮しない。

これらのイベントの組み合わせを元にタッチジェスチャは認識される。

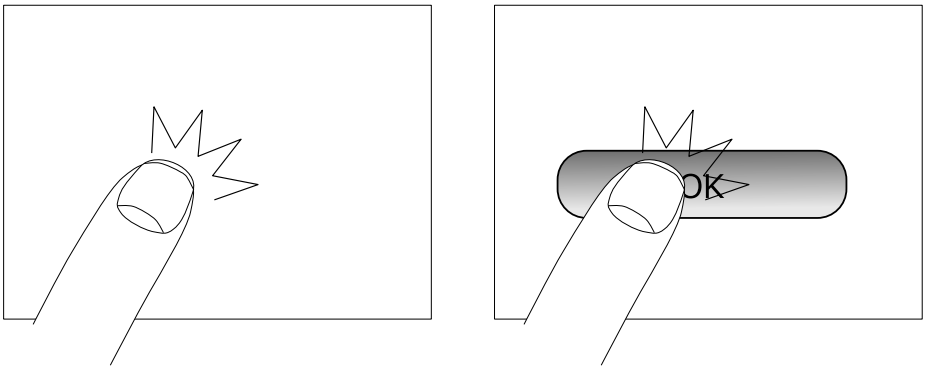


図 4.1: タップジェスチャ

4.2 既存ジェスチャの分類

円筒型マルチタッチインタフェースにおいて 3D インタラクションを行うためのタッチジェスチャを設計するにあたり、現在普及している平面型タッチインタフェース上でのタッチジェスチャと大きくかけ離れたものである場合、操作者に受けいれられにくくなると考えられる。そこで、円筒型マルチタッチインタフェースで利用するジェスチャを決定する前に、平面型マルチタッチインタフェースにおいて用いられているジェスチャの調査を行った。既存のマルチタッチインタフェースで利用されているジェスチャは次に示す 5 つに大きく分類することができる。

タップジェスチャ

タップジェスチャは、指でタッチインタフェースを触り、瞬時に指で離すことによって行う（図 4.1）。

既存の 2D システムにおいては、このジェスチャは主にマウスによるクリックイベントを発行するために利用される。また、タップジェスチャの応用として同じ点を複数回タップするダブルタップ、トリプルタップと呼ばれるジェスチャや、指を長時間触ってから離すロングタップと呼ばれるジェスチャも存在する。

移動ジェスチャ

移動ジェスチャは、指でタッチインタフェースのある点を触り、タッチインタフェースから指を離さず任意の点まで指を動かし、指を離すことによって行う（図 4.2）。このジェスチャはタッチインタフェースだけでなく、マウスやスタイラスなどでも広く利用されている。

このジェスチャは主に始点から終点までの移動経路を示すものであり、既存の 2D システムにおいては、対象とするアイコンの移動などに利用される。また、ジェスチャによる入力は始点と終点なので、操作者がどのような移動経路を示してもシステムは同じ挙動を示す。

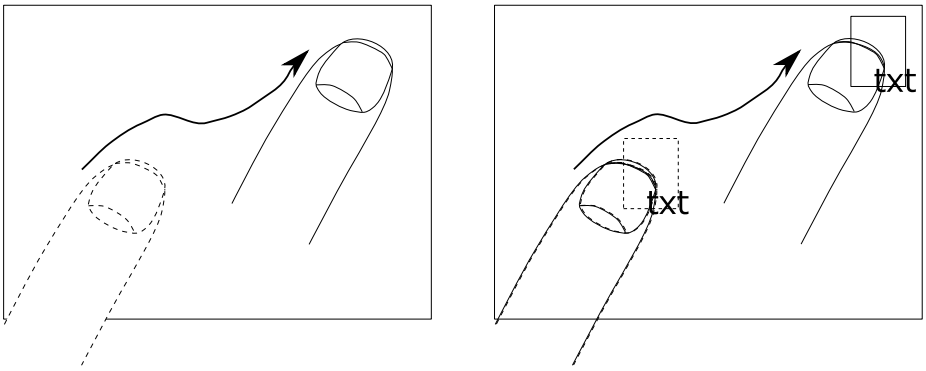


図 4.2: 移動ジェスチャ

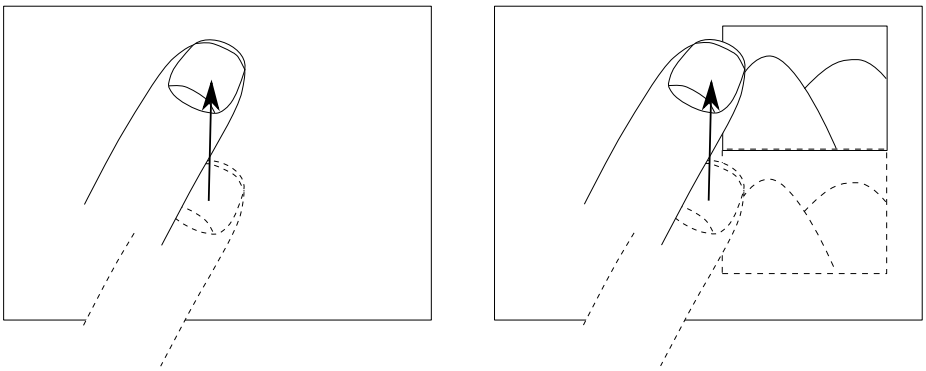


図 4.3: ストロークジェスチャ

ストロークジェスチャ

ストロークジェスチャは移動ジェスチャとは異なり、指でタッチインタフェースを触り、タッチインタフェースから指を離さずある特定の軌道を指でなぞり、指を離すことによって行う（図 4.3）。このジェスチャも既存の 2 次元入力デバイスで広く利用されているジェスチャである。

入力する軌道は「上方向に指を動かす」といった単純なものから、「記号を描く」といった複雑なものなど利用するシステムによって定義されている。操作者がこのジェスチャをシステムに入力することによってシステムは登録してある対応するコマンドの発行を行う。既存の 2D システムでは Web ブラウザ上でのページ送り機能などで利用されている。

また、利用するシステム毎に同じストロークでも違うコマンドが発行する場合や、同じコマンドであってもストロークが違う場合などがあり、システムにとって自然なジェスチャが選ばれる。

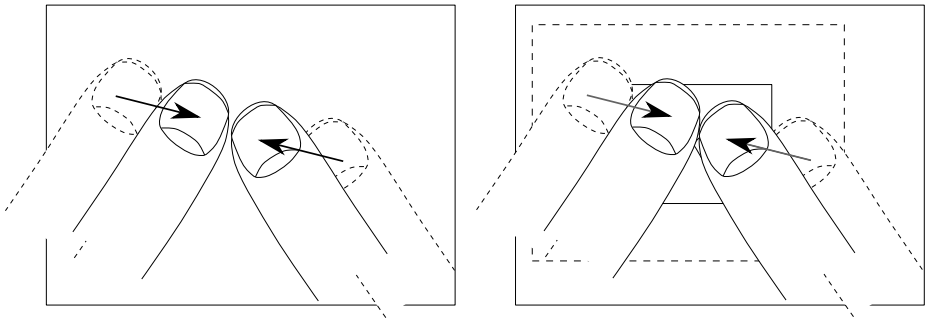


図 4.4: ピンチインジェスチャ

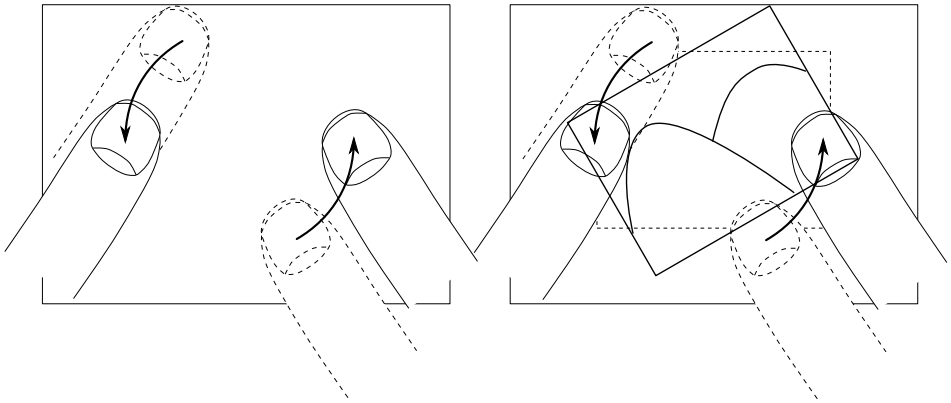


図 4.5: 回転ジェスチャ

ピンチイン/ピンチアウトジェスチャ

ピンチイン/ピンチアウトジェスチャは 2 本の指でタッチインタフェースを触り、その指の間隔を縮める/広げることによって行う（図 4.4）。このジェスチャはマルチタッチインタフェースにおいて実現できるジェスチャである。

このジェスチャはオブジェクトの拡大縮小を行うために利用され、既存の 2D システムでは表示されているオブジェクトを操作者の望む任意のサイズにリサイズする時に利用される。特に control-display ratio を 1:1 にすることで、操作者は実世界の距離感覚でオブジェクトの大きさを変更することができる。

回転ジェスチャ

回転ジェスチャは 2 本の指でタッチインタフェースを触り、2 本の指の中点を基準とし 2 本の指を回転することによって行う（図 4.5）。このジェスチャもマルチタッチインタフェースにおいて実現できるジェスチャである。

このジェスチャはオブジェクトの回転を行うために利用され、既存の 2D システムでは表示

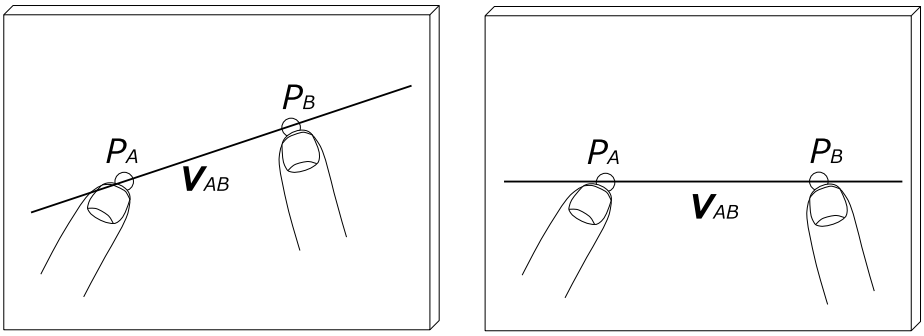


図 4.6: 平面上に 2 点を指定した際に生成される直線の例

されているオブジェクトを操作者の望む任意の角度に回転する時に利用される。このジェスチャもピンチイン/ピンチアウトジェスチャと同様に control-display ratio を 1:1 にすることによって、操作者は物理的なオブジェクトを回転するように操作することができる。

この 5 つのジェスチャ以外にもこれらをさらに複数組み合わせることによってマルチタッチインタフェースにおいてはより自由度の高い入力を操作者に提供している。たとえば、「複数の指で同時にタップする」や、「ロングタップ中にストロークを行う」「2 点を自由に動かすことでピンチイン/ピンチアウト、回転を同時に行う」といった組み合わせなどができ、既存のマルチタッチインタフェースを利用した地図アプリケーションや、ドローアプリケーションでも取り入れられている。

4.3 円筒型マルチタッチインタフェースにおける 3D 操作

本研究で利用した円筒型マルチタッチインタフェースにおいても、上記で挙げたイベント、ジェスチャをすべて実現することができる。しかし、円筒型マルチタッチインタフェースは 3.2 節で述べたように操作対象となる 3D 空間を包み込むような操作面をもっている。これを活用することで、円筒型マルチタッチインタフェースでは平面型のマルチタッチインタフェースにはない 3D インタラクションを実現することができる。

例えば直線を定義するには、3D 空間に点を 2 つ定義すればよい。もっとも単純な点の指定方法としてタッチインタフェースを触った点を 3D 空間にマッピングするという方法が考えられる。たとえば、X-Y 平面にマッピングされた平面型マルチタッチインタフェースにおいて 2 点 P_A、P_B を指定したとき、指定した 2 点によって生成されるベクトル V_{AB} は図 4.6 に示すような X-Y 平面上の直線となり、Z 軸方向への傾きを指定することはできない。

しかし、中心線が Y 軸となるような円筒にマッピングされた円筒型マルチタッチインタフェースにおいて 2 点 P_A、P_B を指定すれば、2 点間を結ぶ直線 V_{AB} は図 4.7 に示すように奥行方向への傾きをもつことができる。

また、同様に 3D 空間に点を 3 つ定義すれば平面を定義することができる。たとえば、x-y 平面にマッピングされた平面型マルチタッチインタフェースにおいて 3 点 P_A、P_B、P_C を指

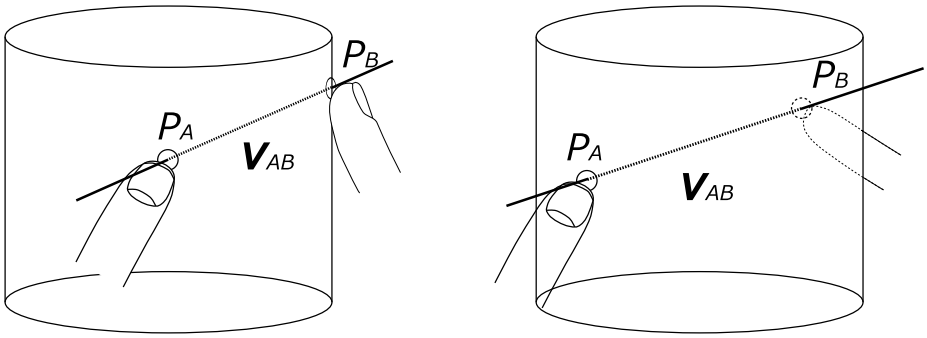


図 4.7: 円筒上に 2 点を指定した際に生成される直線の例

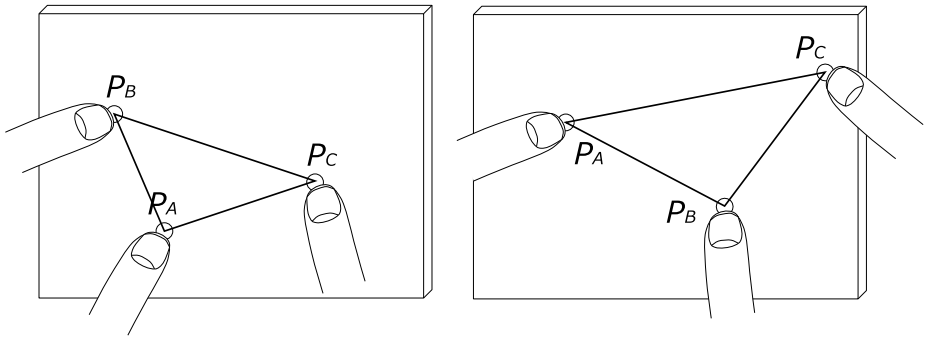


図 4.8: 平面上に 3 点を指定した際に生成される平面の例

定したとき、指定した 3 点によって生成される平面 α は図 4.8 に示すような X-Y 平面と同一の平面となる。

中心線が Y 軸となるような円筒にマッピングされた円筒型マルチタッチインタフェースにおいて 3 点 P_A、P_B、P_C を指定すれば、3 点によって定義される平面 α は図 4.9 に示すように自由に傾きをもつことができる。

このように、円筒型マルチタッチインタフェースにおいて平面型マルチタッチインタフェースと同じようにジェスチャを行っても、その操作内容は 2D 平面ではなく 3D 空間に対応付けることができる。

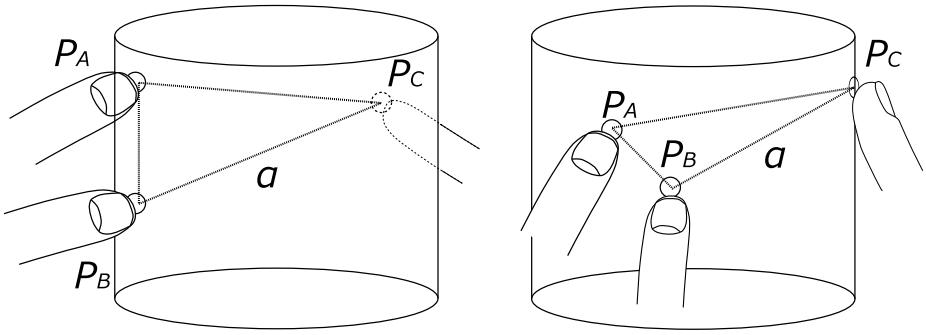


図 4.9: 円筒上に 3 点を指定した際に生成される平面の例

第5章　3Dポインティングジェスチャの設計

5.1　設計の方針

立体ディスプレイと円筒型マルチタッチインタフェースを組み合わせて、3D ポインティングを行うにあたり、一人で操作する場合と、複数人で操作する場合の 2 つの場合が考えられる。これらの利用シーンに置いてそれぞれどのような操作体系をとるべきかを考えたので以下にその利用シーンとその操作体系について述べる。

一人で操作する場合

一人で操作する場合とは、現在の一般的なコンピューティング環境のように一人だけでコンピュータを操作する場合である。現在、我々の扱うデータ量は日々増加しており、これらの情報を閲覧するために 3D 空間にデータを配置し、それを閲覧するといった方法がある。このような 3D 空間を閲覧するために立体ディスプレイが使用される場合を考える。

一般的にディスプレイと一体のタッチインタフェースを利用する際には、タッチ操作をおこなっている個所の表示が隠れてしまうという問題がある。特に、複数人で同時に利用する場合には、一人の操作者が操作領域を多く取り、大きな面積を使って操作を行ってしまうと他の利用者がディスプレイを見れないといった問題が発生する。この問題は立体ディスプレイと円筒型マルチタッチインタフェースを組み合わせた場合にも同様に起こる。

しかし、この利用シーンにおいては、操作者は一人であるため、円筒型マルチタッチインタフェースの操作領域を隠してしまうことによる他の操作者への考慮は必要ない。また、円筒型マルチタッチインタフェースを操作者が両手で挟みこむように操作を行うことで、操作者の手によってディスプレイの表示面を隠してしまうといった問題が発生しなくなる。

これらのことから、一人で円筒型マルチタッチインタフェースを利用する際には両手を使って円筒型マルチタッチインタフェースを利用することが有効であると考えられる。

複数人で操作する場合

複数人で操作する場合とは、公共の場に立体ディスプレイを設置し、それを閲覧するような場合である。全方向からの操作が可能であることを利用し、公共施設で利用する地図案内システムなどに利用される場合を考える。

一人で操作する場合にも述べたが、ディスプレイと一体のタッチインタフェースを操作する際には、その操作領域が操作者の手によって隠れるという問題が発生し、複数人で利用する場合には、一人が多くの操作領域を使って操作を行うことは現実的ではない。

そこで各操作者に割り当てる操作領域を小さくするために、操作者が片手のみを使って円筒型マルチタッチインタフェースを利用することが有効であると考えられる。

これらの設計方針をとり、一人で操作する場合の両手を用いた 3D ポインティング手法を Two-handed 3D ポインティングと名付け、複数人で操作する場合の片手を用いた 3D ポインティング手法を One-handed 3D ポインティングと名付けた。

また、3D ポインティングにあたり、そのポインティングについて用いるカーソルと、円筒型マルチタッチインタフェースにおける操作面と立体ディスプレイの操作空間のマッピングについて検討したので以下に記す。

3D カーソル

3D ポインティングを行うに当たり、3D 座標を指定すればその目的は達成される。しかし、2.1 節で述べた関連研究にあるように、3D ポインティングを行うに当たり、正確に座標の指定を行うことは難しく、フラッシュライト手法のようにボリュームオブジェクトを利用して、目的とするポインティグ座標を含むように大きく選択するほうが効率的であることが示されている。

そこで本研究では 3D 座標および 3D 空間を指定するための 3D ポインティングカーソルとしてスフィアカーソル [NSTH] を用いることにした。スフィアカーソルとは、3D 座標と半径をパラメータとし、3D 空間を指定する半透明の球状のカーソルである。操作者は、ジェスチャを用いてこの座標と半径を変化させることによってスフィアカーソルを操作してポインティングを行う。

3D 座標のマッピング

タッチインタフェースを利用するにあたり、その入力量をどのように用いるかを設定する必要がある。たとえば平面型タッチインタフェースである液晶タブレットにおいては、タブレット表面に表示されている座標と、操作者が触る座標は control-display ratio が 1 対 1 に対応している。しかし、ノートパソコンに取り付けられているトラックパッドのような平面型タッチインタフェースでは、指の移動量が入力に利用されている。

本研究において円筒型マルチタッチインタフェースを用いて行う 3D インタラクションは、4.3 節で述べたような円筒表面を触ることによって円筒内部の 3D 空間を指定する方法をとる。そのため、円筒型マルチタッチインタフェースの操作面とその内部で表示される 3D 空間は図 5.1 で示すように control-display ratio が 1 対 1 になるようにマッピングした。

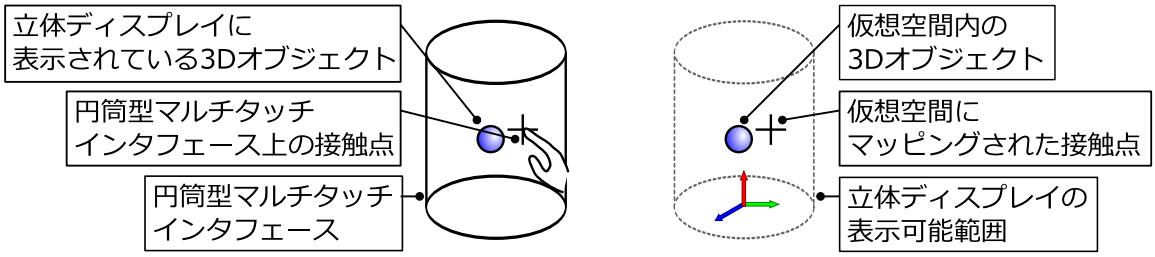


図 5.1: 円筒型マルチタッチインタフェースの操作面とその仮想空間へのマッピング

5.2 Two-Handed 3D ポインティングの操作手法

スフィアカーソルの座標と大きさを決定するには両手で円筒型マルチタッチインタフェースを触ることによって行う。またそれぞれの手の指の本数は 2 本のみで行う。

位置の指定

スフィアカーソルの位置の決定は、初期座標の決定と座標位置の変更の 2 段階の操作手法で決定する。

1. 初期座標の決定

   スフィアカーソルの初期座標の決定は両手で円筒型マルチタッチインタフェースを触ることによって開始する。このときスフィアカーソルの位置は図 5.2 に示すように、両手の接触点の座標を結ぶ線分の中点座標になる。

2. 座標の修正

   スフィアカーソルの上下の移動と前後の移動については図 5.3 と図 5.4 に示すように両手をそれぞれの方向に動かすことで実現する。左右への移動は図 5.5 に示すように円筒型マルチタッチインタフェースを触っている手を用いて回転ジェスチャを行うことで実現する。右手で右（左）回転ジェスチャを行うことで、スフィアカーソルは両手の接触点を結ぶ線上を左（右）手側に等速で移動する。また、同様に左手で右（左）回転ジェスチャを行うことで、スフィアカーソルは両手の接触点を結ぶ線上を右（左）手側に等速で移動する。この回転と移動方向は、右ねじの法則を模して決定した。

大きさの指定

スフィアカーソルの大きさの変更は円筒型マルチタッチインタフェースを触っている手を用いてピンチアウト（イン）ジェスチャを行うことで実現する。どちらかの指をピンチアウト（イン）させると、スフィアカーソルの大きさは大きく（小さく）なる（図 5.6）。

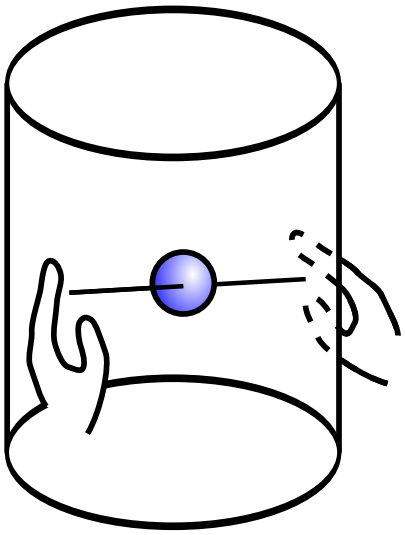


図 5.2: Two-Handed 3D ポインティングによるスフィアカーソルの初期表示位置

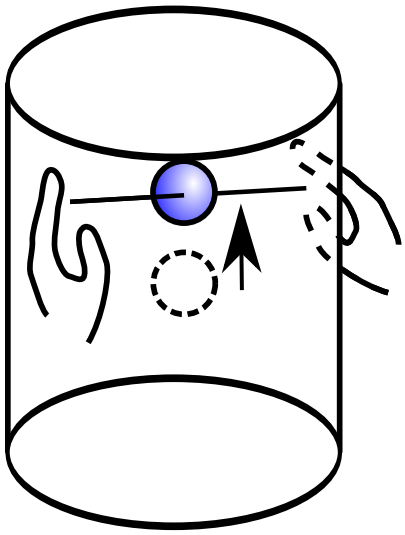


図 5.3: Two-Handed 3D ポインティングによる上方向へのスフィアカーソルの移動

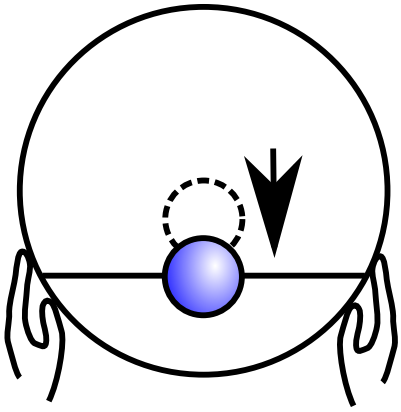


図 5.4: Two-Handed 3D ポインティングによる手前方向へのスフィアカーソルの移動

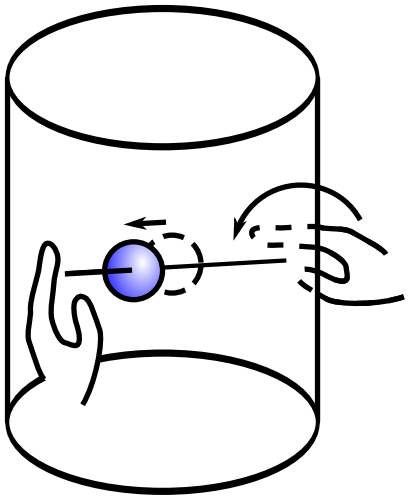


図 5.5: Two-Handed 3D ポインティングによる左方向へのスフィアカーソルの移動

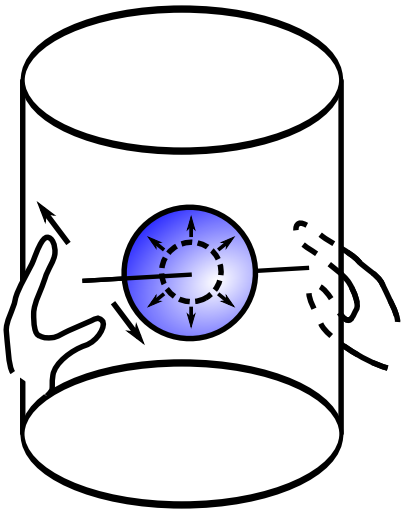


図 5.6: Two-Handed 3D ポインティングによる
スフィアカーソルの拡大

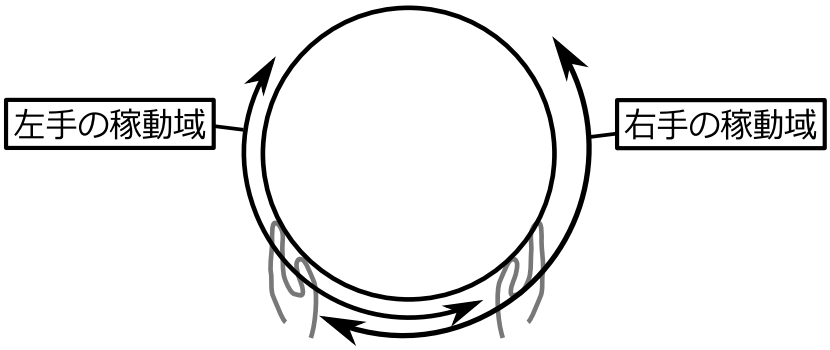


図 5.7: 円筒型マルチタッチインタフェースを操作する際の両手の稼働域

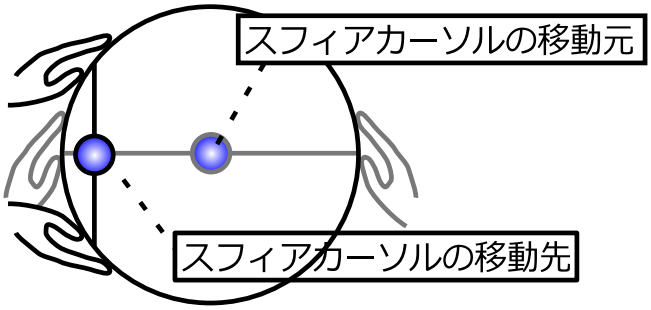


図 5.8: 両手の中点のみを使って水平移動を行う場合の手の位置

5.3　操作手法の妥当性に関する議論

両手を円筒型マルチタッチインタフェースから離さず動かせばスフィアカーソルは両手の中点に追従して移動する。前後上下方向への移動の修正は手の動きだけで簡単に行える。しかし、左右方向への修正は、図 5.7、図 5.8 に示すように手の稼働域が制限されるため、利用者が大きく動かなければポインティングできない。そこで今回は左右方向へカーソルを移動させるために、回転ジェスチャを用いて左右方向への移動ジェスチャとした。これにより、利用者が大きく動くことなく立体ディスプレイの表示範囲をスムーズにポインティングすることができる。

また、大きさの変更についてはタッチインタフェースを用いる一般的な 2D アプリケーションでサイズ変更を行うためのジェスチャであるピンチアウト（ピンチイン）ジェスチャを用いた。

5.4　One-Handed 3D ポインティングの操作手法

スフィアカーソルの座標と大きさを決定するには片手の 2 本の指で円筒型マルチタッチインタフェースを触ることによって行う。

位置の指定

スフィアカーソルの位置の決定は初期座標の決定と、座標位置の変更の 2 段階の操作手法で決定する。

1. 初期座標の決定

   スフィアカーソルの初期座標の決定は円筒型マルチタッチインタフェースを片手で触ることによって開始する。このときスフィアカーソルの位置は図 5.9 に示すように、円筒内の 2 本の指の中点に接するように表示される。

2. 座標の修正

   スフィアカーソルの座標の修正は図 5.10 に示すように、触れている 2 本の指による回転ジェスチャを使う。右（左）回転ジェスチャを行うと、スフィアカーソルは 2 本の指の中心からスフィアカーソルの中心を結ぶ線上を奥（手前）に等速で移動する。この回転と移動方向は、Two-Handed に右ねじの法則を模して決定した。

   カーソルの移動方向を変更する場合は、図 5.11 に示すように 2 本の指を移動させて「2 本の指の中点とスフィアカーソルの中心とを結ぶ線」の向きを変えることで対応する。

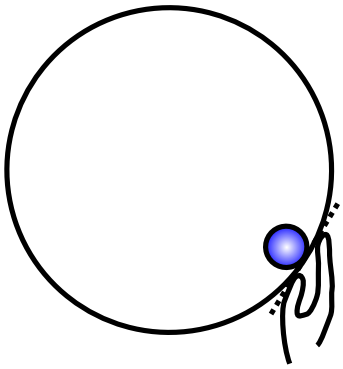


図 5.9: One-Handed 3D ポインティングによるスフィアカーソルの初期表示位置

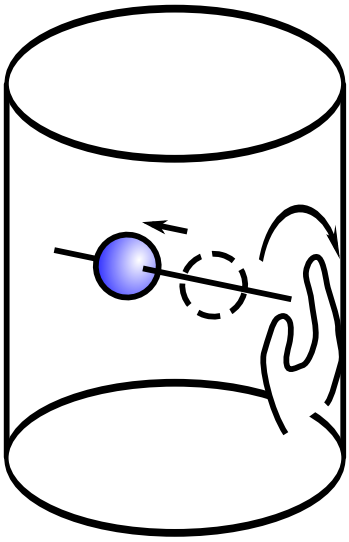


図 5.10: 奥方向へのスフィアカーソルの移動

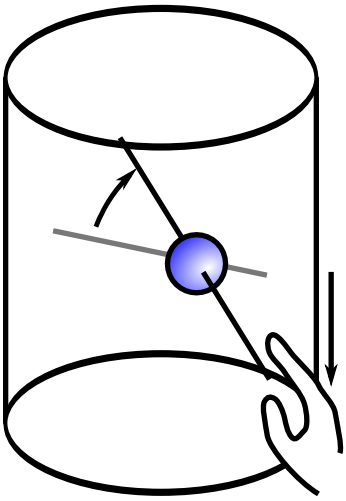


図 5.11: One-Handed 3D ポインティングによるスフィアカーソルの移動方向の変更

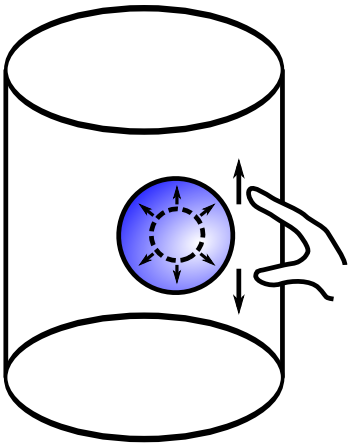


図 5.12: One-Handed 3D ポインティングによるスフィアカーソルの拡大

大きさの指定

スフィアカーソルの大きさの変更は円筒型マルチタッチインタフェースを触っている手を用いてピンチアウト（イン）ジェスチャを行うことで実現する。図 5.12 に示すように指をピンチアウト（イン）させると、スフィアカーソルの大きさは大きく（小さく）なる。

5.5　操作手法の妥当性に関する議論

One-handed 3D ポインティングはポインティング中に利用する操作空間を小さくする必要があった。大きさの変更については Two-Handed 3D ポインティングと同様にピンチアウト（ピンチイン）ジェスチャを用いることで、操作者は指二本のみを使ってタッチインタフェース上で操作を行うことができる。位置の修正についても、操作を行っている手をスライドさせることによって移動方向を変更することができるので、一人の操作者がタッチインタフェースの表面を隠す面積は小さくなっている。

第6章　ジェスチャ認識の実装

ジェスチャ認識の実装はWindows上でC#を用いて行った。これは、円筒型マルチタッチインタフェースを用いて操作するアプリケーションが、円筒型マルチタッチインタフェースを触った時の接触点のデータを受け取るためのインタフェースの実装が.Net FrameworkのDLL形式で行われているためである。

6.1　認識に利用するデータ

円筒型マルチタッチインタフェースから、操作者の円筒型マルチタッチインタフェースを触った接触点の座標を図6.1に示す形でデータを受け取ることができる。このデータはC#上で*TouchEvent*オブジェクトとして受け取ることになる。*TouchEvent*オブジェクトは、円筒型マルチタッチインタフェースから1秒間に約30回通知される。

*Touch*オブジェクトは、操作者が円筒型マルチタッチインタフェースを触っているときの各接触点のデータである。*Touch*オブジェクトは接触点の位置を$\theta - y$の円柱座標系として*position*に持っている。また、各*Touch*オブジェクトは*id*を持っており、操作者が円筒型マルチタッチインタフェースを触り続けている場合, その接触点は同じ*id*が与えられ、アプリケーションは同一の*id*を持つ*Touch*オブジェクトを解析することで、接触点の移動を知ることができる。

*Cluster*オブジェクトは、*Touch*オブジェクトを距離尺度でクラスタリングした結果のクラスタを管理するためのオブジェクトである。クラスタに含まれる*Touch*オブジェクトはList形式で格納されている。*Cluster*オブジェクトの位置は*position*に格納され、その値はクラスタが持つ*Touch*オブジェクトの重心位置とした。また、クラスタの大きさの*size*は、クラスタの重心位置からクラスタに含まれるもっとも遠い*Touch*オブジェクトの位置までの距離とした。*Cluster*オブジェクトも*Touch*オブジェクトと同様に*id*を持っており、同一の*id*を持つ

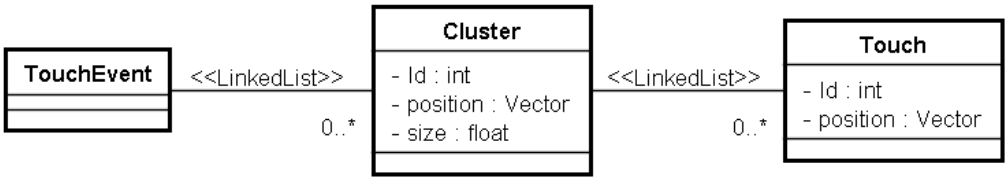


図6.1: イベントとしてアプリケーションに渡されるデータ

表 6.1: Two-Handed 3D ポインティングのジェスチャ解析に用いる変数

変数名	保持する値
$baseSlope[i]$	操作開始時の [i] 番目の手の指と指の傾き
$baseDist[i]$	操作開始時の [i] 番目の手の指と指の距離
$currentPos[i]$	現在の [i] 番目の手の座標
$currentSlope[i]$	現在の [i] 番目の手の指と指の傾き
$currentDist[i]$	現在の [i] 番目の手の指と指の距離
$cusorPos$	スフィアカーソルの位置
$cursorSize$	スフィアカーソルの大きさ

Cluster オブジェクトを解析することで、クラスタの移動を知ることができる。

TouchEvent オブジェクトは *Cluster* オブジェクトを List 形式で格納しており、アプリケーションは *TouchEvent* オブジェクトから、*Cluster* オブジェクト、*Touch* オブジェクトと辿ることによって、接触点の情報を得ることができる。

6.2 ジェスチャの認識

6.2.1 Two-Handed 3D ポインティングのジェスチャ認識

Two-Handed 3D ポインティングのジェスチャを認識するために用いたアルゴリズムを Algorithm 1 に示す。また、Algorithm 1 で用いたアルゴリズムの各変数を表 6.1 に示す。

ジェスチャの認識のプログラムはシステムが起動している間は常に動作している。しかし、ジェスチャの判定処理は、円筒型マルチタッチインタフェースから受け取った *Cluster* オブジェクトや、*Touch* オブジェクトが利用するジェスチャに適したものであるときのみ行う。今回のジェスチャの場合は両手を用いて操作を行い、それぞれの手は指を 2 本使うように設計しているため、*TouchEvnet* オブジェクトに含まれる *Cluster* オブジェクトが 2 つで、それぞれの *Cluster* オブジェクトに *Touch* オブジェクトが 2 つ含まれているときにジェスチャの判定が行われる。

6.2.2 One-Handed 3D ポインティングのジェスチャ認識

One-Handed 3D ポインティングのジェスチャを認識するために用いたアルゴリズムを Algorithm 2 に示す。また、Algorithm 2 で用いたアルゴリズムの各変数を表 6.2 に示す。

One-Handed 3D ポインティングも Two-Handed 3D ポインティングと同様にジェスチャの認識のプログラムはシステムが起動している間は常に動作している。One-Handed 3D ポインティングのジェスチャの場合は、片手を用いて操作を行い、指を 2 本使うように設計しているため、*TouchEvnet* オブジェクトに含まれる *Cluster* オブジェクトが 1 つで、その *Cluster* オブジェクトに *Touch* オブジェクトが 2 つ含まれているときにジェスチャの判定が行われる。

Algorithm 1 Two-Handed 3D ポインティングのジェスチャ解析アルゴリズム

loop
  if 新しくジェスチャが始まった **then**
    for $i = 0$ to 1 **do**
      $baseSlope[i] \leftarrow currentSlope[i]$
      $baseDist[i] \leftarrow currentDist[i]$
    end for
    $cusorPos \leftarrow (currentClusterPos[0] + currentClusterPos[1]) * 0.5$
    $cursorSize \leftarrow basicSize$
  else
    for $i = 0$ to 1 **do**
      if $|currentSlope[i]$ - $baseSlope[i]|$ が $\frac{\pi}{6}$ より大きい **then**
        if $baseSlope[i]$ から $currentSlope[i]$ への変化が右回転である **then**
          $cursorPos \leftarrow movePosition(currentClusterPos[i], cusorPos)$
        else
          $cursorPos \leftarrow cursorPos + 0.4 \times (currentPos[i] - currentPos)$
        end if
      end if
      if $currentDist[i]$ が $baseDist[i] * 1.4$ より大きい **then**
        $cursorSize \leftarrow cursorSize * 1.4$
      else if $currentDist[i]$ が $baseDist[i] * 0.6$ より小さい **then**
        $cursorSize \leftarrow cursorSize * 0.6$
      end if
    end for
  end if
end loop

表 6.2: One-Handed 3D ポインティングのジェスチャ解析に用いる変数

変数名	保持する値
$currentClusterPos$	現在のクラスタの座標
$baseSlope$	操作開始時の指と指の傾き
$baseDist$	操作開始時の指と指の距離
$currentSlope$	操作中の指と指の傾き
$currentDist$	操作中の指と指の距離
$cusorPos$	スフィアカーソルの位置
$cursorSize$	スフィアカーソルの大きさ

Algorithm 2 One-Handed 3D ポインティングのジェスチャ解析アルゴリズム

loop
  if 新しくジェスチャが始まった **then**
    $baseSlope \leftarrow currentSlope$
    $baseDist \leftarrow currentDist$
    $cusorPos \leftarrow currentClusterPos$
    $cursorSize \leftarrow basicSize$
  else
    if $currentSlope$ と $baseSlope$ の差の絶対値が $\frac{\pi}{6}$ より大きい **then**
      if $baseSlope$ から $currentSlope$ への変化が右回転である **then**
        $cursorPos \leftarrow movePosition(currentClusterPos, cusorPos)$
      else
        $cursorPos \leftarrow cursorPos + 0.4 \times (currentPos - currentPos)$
      end if
    end if
    if $currentDist$ が $baseDist * 1.4$ より大きい **then**
      $cursorSize \leftarrow resizeLarger(cursorSize)$
    else if $currentDist$ が $baseDist * 0.6$ より小さい **then**
      $cursorSize \leftarrow resizeSmaller(cursorSize)$
    end if
  end if
end loop

第7章　評価実験と考察

7.1　評価実験

7.1.1　実験の目的

実験の目的は以下の 2 つである。

- 本研究で述べた操作手法について操作性を検証する
- 本研究で述べた操作手法について改善点を調査する

7.1.2　実験用アプリケーション

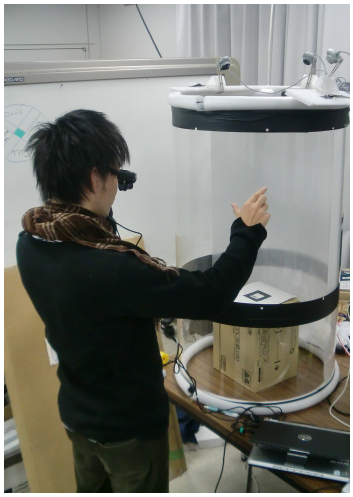
提案手法の操作感を確かめるために、3D 空間中に球状の赤いターゲットオブジェクトを 1 つだけ表示し、それをスフィアカーソルを用いて選択するというアプリケーションを実装した。このアプリケーションは、ターゲットオブジェクトをスフィアカーソルによって選択するものである。ターゲットオブジェクトは選択されると、ランダムで新しい位置に表示される。

アプリケーションの表示にはヘッドマウンテッドディスプレイを利用し、AR 技術を用いた。円筒内部にターゲットオブジェクトおよびスフィアカーソルが存在するかのように重畳表示を行った。また、3D ポインティング操作中のスフィアカーソルの回転ジェスチャによる移動方向については、緑色の線分を表示して示した。

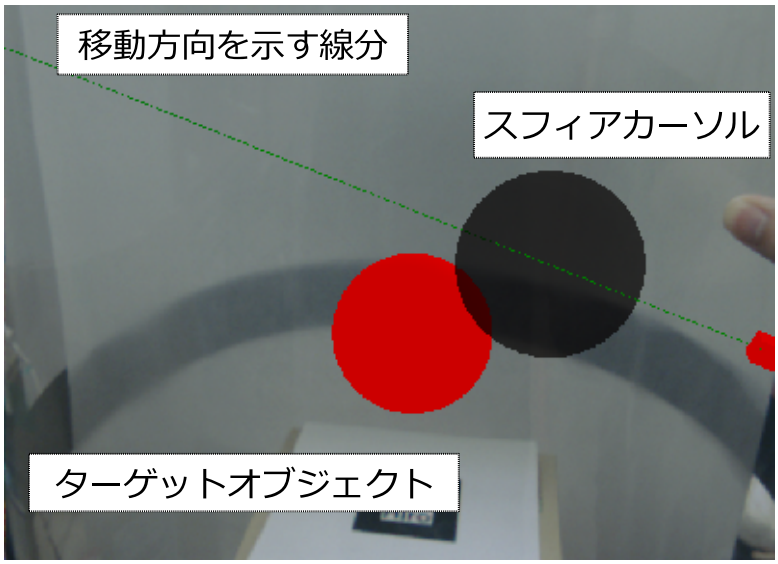
選択の確定は操作者がスフィアカーソルを移動させ、操作に用いていた手をインタフェースから離した瞬間とし、成否の判定はターゲットオブジェクトとスフィアカーソルが交差していた場合に成功とした。また、スフィアカーソル移動中の選択の成否について視覚的フィードバックを操作者に与えるために、ターゲットオブジェクトとスフィアカーソルが交差しているときは、ターゲットオブジェクトを青色に変化させた。

7.1.3　実験概要

被験者には、Two-Handed 3D ポインティングと One-Handed 3D ポインティングの両方の手法で実験用アプリケーションを操作してもらった。ただし、本実験において選択を指示したターゲットオブジェクトは 3D 空間中に 1 つのみとして、スフィアカーソルの移動操作についてのみ操作してもらった。



(a) 操作中の様子



(b) 実行中のアプリケーション

図 7.1: 実験の様子

その後、それぞれの操作手法の感想についてアンケート調査と、その内容を元にしたヒアリングを行った。

実験中の被験者の様子と、AR を用いて提示したアプリケーションの実行画面を図 7.1 に示す。

7.1.4 被験者

コンピュータサイエンスを専攻する大学生および大学院生 6 名を対象に実験を行った。被験者はいずれもヒューマンインタフェースに関する研究室に所属する学生であり、いずれの被験者もタッチインタフェースを操作したことがあった。4.2 節に述べたタッチジェスチャについては全員が操作したことがあり、そのうちの 4 名が日常的にそれらのタッチジェスチャを利用していた。

7.1.5 手順

まず、実験の目的について説明した。その後、Two-Handed 3D ポインティングと One-Handed 3D ポインティングをそれぞれ用いて順に実験用アプリケーションを操作してもらった。操作タスクとして操作者にはスフィアカーソルを用いてランダムに表示されるターゲットオブジェクトを 15 回選択してもらった。

7.2 結果と考察

7.2.1 アンケート結果

Two-Handed 3D ポインティング

Two-Handed 3D ポインティングについて肯定的な意見として、「回転ジェスチャを行わなくても、両手によるタッチジェスチャのみで操作を行うことができるため簡単に座標を指定することができる」、「タッチジェスチャを奥で行うことで奥にスフィアカーソルが表示されるので、直感的である」といったコメントがあった。否定的な意見として「遠くにターゲットがあるときに、操作を行うために両手を伸ばして行う場合があり、腕が疲れる」や、「意図せず傾きジェスチャを入力してしまう」といった意見があった。

One-Handed 3D ポインティング

One-Handed 3D ポインティングについて肯定的な意見として、被験者から「片手で 3D 空間を自由に移動させることができた」、「奥行方向への操作が直感的である」というコメントが得られた。また、「片手のみを利用して操作ができるので楽にポインティングできる」というコメントもあった。否定的な意見として「移動方向が把握しづらい」、「奥行方向に移動をさせる際に時間がかかる」といったコメントがあった。

7.2.2 考察

操作性について

被験者らのコメントから、どちらの手法についても 3D ポインティングを行うに当たり、奥行方向へのポインティングが既存のマウスを利用する 3D ポインティング手法よりも直感的で行いやすいといった好意的な意見が得られた。また、One-Handed 3D ポインティング手法よりも Two-Handed 3D ポインティングのほうが操作が行いやすい、速く操作ができそうであるといった意見が得られた。これは、ジェスチャの設計時に立てた仮説を支持しているものと考えられる。

Two-Handed 3D ポインティング

被験者からのコメントより Two-Handed 3D ポインティングのカーソルの移動位置となる両手の中間地点は操作者にとって 3D 座標を把握しやすく、ポインティングをスムーズに行えるということが分かった。また、ターゲットオブジェクトが操作者に対して横方向にある場合に、回転ジェスチャを用いてスフィアカーソルを左右に動かして選択を行うよりも、操作者本人が円筒型マルチタッチインタフェース側面に移動し、選択を行うほうを好むことが分かった。

しかし、被験者の操作を観察したところ、ターゲットオブジェクトが奥に表示されたときは、操作者は円筒型マルチタッチインタフェースの周りを移動するよりも腕を伸ばして円筒型マルチタッチインタフェースの奥をタッチすることによって選択を行うことが多かった。このようなときに、腕を伸ばす操作が大変であるという意見が被験者からあった。この点は、Two-Handed 3D ポインティングの新たな操作方法として奥行方向への操作を追加することで改善できると考えられる。

また、移動ジェスチャを行う途中に手を傾かせてしまい回転ジェスチャが意図せずに起きてしまうという意見があった。

One-Handed 3D ポインティング

One-Handed 3D ポインティングは Two-Handed 3D ポインティングに比べて、回転ジェスチャによるスフィアカーソルの平行移動操作が好印象だった。これは、片手によるジェスチャにおいては右ねじの法則に則った回転方向と移動方向の対応付けが把握しやすいためだとコメントが得られた。

しかし、回転ジェスチャの感想として、Two-Handed 3D ポインティングの時よりも、腕を伸ばした時に腕がねじれてしまい、回転ジェスチャを行いにくいという意見もあった。One-Handed 3D ポインティングの操作方法としてあまり手の位置を動かさないように設計を行っていたが、

ジェスチャの改善案

両操作法に見られた回転ジェスチャに対する問題点は、操作に慣れた著者が同じように操作を行った時には発生しないため、慣れにより改善されると考えられる。また、問題の原因として移動ジェスチャによる腕の傾きの変化が回転ジェスチャへ影響を与えていることから、両方のジェスチャを同時に行うという設計に問題があったと考えられる。そこで、移動ジェスチャの後に回転ジェスチャを行う前に、3 本目の指をタップするといったタップジェスチャの改良も考えられる。

しかし、ジェスチャの複雑化は、操作者にジェスチャを覚えてもらう必要があり、操作者に負担がかかる場合があるため、検討を行う必要がある。

第8章　まとめ

本研究では円筒型マルチタッチインタフェースを用いて立体ディスプレイに表示された3D座標を指定するためのポインティング手法として、Two-handed 3Dポインティングと One-handed 3Dポインティングを開発し、その操作性を評価するための評価アプリケーションを実装した。また、被験者実験によって得られたコメントより、本手法が立体ディスプレイにおける3Dポインティング手法として有効であることが分かった。

今後は、本研究で開発したポインティング手法の改良を行うとともに、3Dアプリケーションの操作を行うために必要となる操作手法の開発を行う。

謝辞

本論文の執筆にあたり、指導教員である田中二郎教授、志築文太郎講師をはじめ、三末和男准教授、高橋伸講師には多くのご助言やご指導をいただきました。心より感謝申し上げます。また、IPLAB の皆様にも大変お世話になりました。特に WAVE チームの皆様にはチームゼミだけでなく日常的にご意見を頂いており、ここに深く感謝いたします。

共同研究者である国立情報学研究所の細部博史氏には研究活動においてご助言をいただき感謝を申し上げたいと思います。

高度 IT 人材育成のための実践的スペシャリスト専修プログラムにおいて、ご指導頂きました菊池純男教授、駒谷昇一教授の両名にはこの場を借りて深く感謝を申し上げたいと思います。

株式会社 NTT データには奨学金による経済的な支援していただくとともに、研究発表と議論の場を提供していただきました。また、大友様、福田様には発表の場に置いて的確なご助言と指導をいただき、心から御礼申し上げます。

最後に日頃より私を支えてくれました家族や友人たちに心より感謝いたします。

参考文献

[BF] H. Benko and S. Feiner. Balloon selection: A multi-finger technique for accurate low-fatigue 3d selection. *3D User Interfaces*.

[BWB08] Hrvoje Benko, Andrew D. Wilson, and Ravin Balakrishnan. Sphere: multi-touch interactions on a spherical display. In *UIST '08: Proceedings of the 21st annual ACM symposium on User interface software and technology*, pp. 77–86, 2008.

[CCY+07] Li-Wei Chan, Yi-Fan Chuang, Meng-Chieh Yu, Yi-Liu Chao, Ming-Sui Lee, Yi-Ping Hung, and Jane Hsu. Gesture-based interaction for a magic crystal ball. In *VRST '07: Proceedings of the 2007 ACM symposium on Virtual reality software and technology*, pp. 157–164, New York, NY, USA, 2007. ACM.

[dlRKOD08] Jean-Baptiste de la Rivière, Cédric Kervégant, Emmanuel Orvain, and Nicolas Dittlo. Cubtile: a multi-touch cubic interface. In *VRST '08: Proceedings of the 2008 ACM symposium on Virtual reality software and technology*, pp. 69–72, 2008.

[FB09] Bernd Froehlich and Doug Bowman. 3d user interfaces. *IEEE Computer Graphics and Applications*, Vol. 29, pp. 20–21, 2009.

[FHZ96] Andrew Forsberg, Kenneth Herndon, and Robert Zeleznik. Aperture based selection for immersive virtual environments. In *UIST '96: Proceedings of the 9th annual ACM symposium on User interface software and technology*, pp. 95–96, 1996.

[GB06] Tovi Grossman and Ravin Balakrishnan. The design and evaluation of selection techniques for 3d volumetric displays. In *UIST '06: Proceedings of the 19th annual ACM symposium on User interface software and technology*, pp. 3–12, 2006.

[GB08] Tovi Grossman and Ravin Balakrishnan. Collaborative interaction with volumetric displays. In *CHI '08: Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, pp. 383–392, 2008.

[GDHZ06] Yotam I. Gingold, Philip L. Davidson, Jefferson Y. Han, and Denis Zorin. A direct texture placement and editing interface. In *UIST '06: Proceedings of the 19th annual ACM symposium on User interface software and technology*, pp. 23–32, 2006.

[GWB04] Tovi Grossman, Daniel Wigdor, and Ravin Balakrishnan. Multi-finger gestural interaction with 3d volumetric displays. In *UIST '04: Proceedings of the 17th annual ACM symposium on User interface software and technology*, pp. 61–70, 2004.

[Han05] Jefferson Y. Han. Low-cost multi-touch sensing through frustrated total internal reflection. In *UIST '05: Proceedings of the 18th annual ACM symposium on User interface software and technology*, pp. 115–118, New York, NY, USA, October 2005. ACM Press.

[IMT09] Takao Ito, Kazuo Misue, and Jiro Tanaka. Sphere anchored map: A visualization technique for bipartite graphs in 3d. In *Proceedings of the 13th International Conference on Human-Computer Interaction. Part II*, pp. 811–820, Berlin, Heidelberg, 2009. Springer-Verlag.

[KBP+] H. Kato, M. Billinghurst, I. Poupyrev, K. Imamoto, and K. Tachibana. Virtual object manipulation on a table-top ar environment. *International Symposium on Augmented Reality*.

[LG94] Jiandong Liang and Mark Green. Jdcad: A highly interactive 3d modeling system. *Computers & Graphics*, Vol. 18, No. 4, pp. 499–506, 1994.

[LSSS] Paul Lapides, Ehud Sharlin, Mario Costa Sousa, and Lisa Streit. The 3d tractus: A three-dimensional drawing board. *International Workshop on Horizontal Interactive Human-Computer Systems*.

[MBS97] Mark R. Mine, Frederick P. Brooks, Jr., and Carlo H. Sequin. Moving objects in space: exploiting proprioception in virtual-environment interaction. In *SIGGRAPH '97: Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pp. 19–26, 1997.

[Min95] Mark R. Mine. Virtual environment interaction techniques. Technical report, 1995.

[NSTH] Masaki Naito, Buntarou Shizuki, Jiro Tanaka, and Hiroshi Hosobe. Interaction techniques using a spherical cursor for 3d targets acquisition and indicating in volumetric displays. *Information Visualisation, International Conference on*.

[OHH06] Rieko Otsuka, Takeshi Hoshino, and Youichi Horry. Transpost: A novel approach to the display and transmission of 360 degrees-viewable 3d solid images. *IEEE Transactions on Visualization and Computer Graphics*, Vol. 12, pp. 178–185, 2006.

[RDH] Jason L. Reisman, Philip L. Davidson, and Jefferson Y. Han. In *UIST '09: Proceedings of the 22nd annual ACM symposium on User interface software and technology*, pp. 69–78, New York, NY, USA.

[SNT] Buntarou Shizuki, Masaki Naito, and Jiro Tanaka. Browsing 3d media using cylindrical multi-touch interface. *Multimedia, International Symposium on*.

[son] Sony Japan — プレスリリース — 「デジタルコンテンツ EXPO2009」出展のご案内. http://www.sony.co.jp/SonyInfo/News/Press/200910/09-123/.

[清川 25] 清川清, 竹村治雄, 片山喜章, 岩佐英彦, 横矢直和. 両手操作を用いた仮想物体モデラ vlego. 電子情報通信学会論文誌. A, 基礎・境界, Vol. 80, No. 9, pp. 1517–1526, 19970925.

[内藤 08a] 内藤真樹, 小林敦友, 志築文太郎, 田中二郎. 円筒型マルチタッチインタフェース. 研究報告「ヒューマンコンピュータインタラクション」No.2008-HCI-127, pp. 37–43. 社団法人情報処理学会, January 2008.

[内藤 08b] 内藤真樹, 志築文太郎, 田中二郎. 円筒型マルチタッチインタフェースによるインタラクション. 全国大会講演論文集 = / Information Processing Society of Japan, Vol. 70, No. 4, pp. 4–209, 3 2008.