平成 22 年度

筑波大学情報学群情報科学類

卒業研究論文

題目 画像編集ツールにおける
カードを用いたレイヤ操作手法

主専攻　　知能情報メディア主専攻

著者　　棚橋 敬浩

指導教員　高橋伸　志築文太郎　三末和男　田中二郎

要　旨

多くの画像編集ツールにおいて，レイヤと呼ばれる編集機能が搭載されている．レイヤを分けて編集することで，試行錯誤がしやすいといった利点がある．しかしその一方で，現在編集しているレイヤがどこかわからなくなったりするといった問題点が生じる．

本研究では，レイヤ機能を用いることによる利点を生かしつつ，レイヤ機能の問題点を解決するためのシステムを開発する．我々は実物体のカードに着目し，レイヤ一枚一枚を実物体のカードに関連付ける．本システムでは，ユーザはカードを操作するだけでレイヤを操作することが可能になる．そのため，ユーザはカードを並べ変えたり，裏返すことでレイヤを操作でき，より直感的にレイヤを扱うことが出来る．

目 次

図 目 次

第1章　序論

1.1　画像編集ツールの現状

画像編集ツールは，写真や画像を補正する，または特殊効果をつけるなどといった用途に使われ，画像加工，イラストレーション，印刷業界などあらゆる画像分野で欠かせないものとなっている．一般によく使われる画像編集ツールの例としては，Adobe Photoshop[1] や GIMP[2]，Canvas[3]といったものが挙げられる．このようなツールは，画像分野における専門家にとって使いやすいように設計されていることが多い．

1.1.1　初心者にとっての使い心地

情報技術や電子機器が発展し，一般家庭にもパーソナルコンピュータが普及している現在では，コンピュータをあまり得意としない初心者の人であっても，コンピュータ上で図形や画像データを扱う機会が多くなっている．

しかし，初心者が実際に画像編集ツールを使用すると，多くの困難に直面する．まず，上であげたようなツールは多機能で汎用なツールを目指しているため，操作が複雑になりがちである．さらに，そういったツールはそれぞれに独特の操作方法が存在する．例えば，新しいレイヤを作成するという簡単な作業であっても，それぞれのツールによって操作方法が違ってしまうため，初心者の混乱を招く要因となっている．以上の理由から初心者が画像編集ツールを使うのには非常に困難であり，コンピュータを用いた画像編集の難度が非常に高く感じられてしまう．

特に本研究では画像編集ツールでよく使用され，かつ初心者が混乱しやすいと思われるレイヤ編集機能の操作について取り上げる．

1.2　レイヤ編集機能

本研究で扱うレイヤとは，画像編集ツールに搭載されている編集機能の１つであり，画像編集ツールにおいて「描画用の透明なシート」を何枚も重ねたり取り替えたりして，画像に要素を追加したり変化を加えたりするための機能である．図 1.1 のように何枚ものレイヤに分けて描画し，必要に応じてレイヤを切り替えて編集することができる．

[1]URL:http://www.adobe.com/products/photoshop/photoshop/
[2]URL:http://www.gimp.org/
[3]URL:http://www.poladigital.co.jp/canvas/

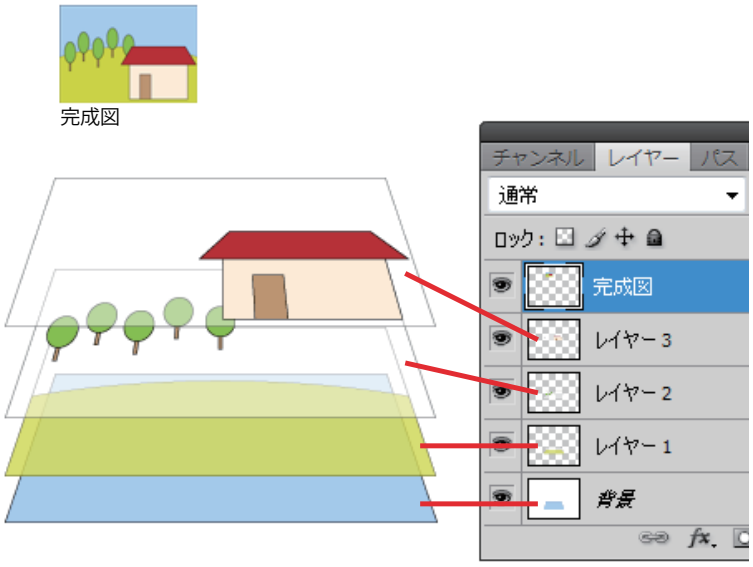


図 1.1: レイヤのイメージ

レイヤを用いる利点として，現在のレイヤで行っている作業が別のレイヤに影響を与えないことがあげられる．このことから，あるレイヤで行った作業が別のレイヤに悪影響を与えるという心配がなくなるため，試行錯誤が容易になる．デザインをしていく際には試行錯誤が多くなるため，このことは重要であると考える．レイヤを用いるもう一つの利点として，画像を配置し直す，移動するといったことがしやすくなる．レイヤを分けていなかった場合は配置の変更が困難だが，レイヤを分けていることにより，回りの物との前後関係を気にせずに配置を変更できる．さらに，レイヤを分けることで，目的のレイヤのみに効果をつけることが出来るといった利点もある．

1.2.1 レイヤ機能の問題点

前述のようにレイヤは確かに便利である．しかし，レイヤ機能を利用し，レイヤを分けて編集することによって生ずる問題点も存在する．

ひとつ目の問題点として，複数のレイヤを使って編集作業を行っている際に，現在自分がどのレイヤを選択しているのかわからなくなることが挙げられる．自分では A のレイヤを編集するつもりだったが，実際には B のレイヤを編集してしまっていた，という状況がまれに起こる．

二つ目の問題点として，レイヤが増えてゆくにつれ，自分の目的とするレイヤに移動することが非常に困難になるという点が挙げられる．

三つ目の問題点として，レイヤの基本的な概念の理解が難しいことが挙げられる．あまり画像編集ツールを扱ったことのない人にとって，最初はレイヤのしくみを理解するのに時間

がかかる．層が何重にもかさなってできているということをわかってはいても，デジタルに画面に描画されるとそれをイメージすることが難しいためである．

1.3　本研究の目的

本研究では，画像編集ツールの機能の中で，レイヤ機能に着目する．画像編集ツールにおけるレイヤ機能のより直感的な利用を可能にすることを目的とする．特に本研究では，画像編集ツールをあまり使用したことのないような初心者にとって使いやすいシステムを設計，実装する．

1.4　本研究のアプローチ

レイヤをイメージしにくいという問題点を解決するために，我々はレイヤ一枚一枚を実物体のカードに関連付ける．レイヤをカードに関連付けることで，ユーザはカードを実際に手を動かして操作できるためより試行錯誤しやすくなり，またカードが机に並ぶことで視覚的にどのレイヤがどこにあるのか理解しやすくなる．また，カードを動かすことでレイヤを変更できるため，より直観的にレイヤを操作できる．

1.5　本論文の構成

本章では，レイヤ編集機能における問題点を挙げ，それを解決する手法の提案と研究の目的について述べた．続いて第 2 章では関連研究について述べる．第 3 章では提案するシステムについて述べる．第 4 章では実装の方法について述べる．第 5 章でそのシステムの評価と議論を行い，最後に第 6 章で結論を述べる．

第2章　関連研究

2.1　タンジブルユーザインタフェース

近年では，情報技術や電子機器が発展し，一般家庭にも PC が普及している．そのため，初心者や子供などコンピュータにあまり興味がない人でも PC を利用しなければならないような場合が多く存在する．同時に PC 上のアプリケーションは多様化し，操作は複雑になっている．

そこで，石井らは，実体をもつデバイスを手で触って，日常的なものを操作するのと同じ感覚でアプリケーションを操作するタンジブルユーザインタフェース (TUI: Tangible User Interface) [1] を提案している．TUI では実際にあるものを操作するかのように入力インターフェースを操作することで，実際に操作して得られた感覚に沿った動作が PC 上でなされるように設計されている．そのため，ユーザにとってわかりやすいインターフェースを実現できる．

既存の TUI としては，瓶のふたを開けることで音楽が流れる musicBottle[2]，砂の形状を変化させることにより地形パターンを変化させることのできる Sandscape[1][3], ブロック型の入力装置を用いることで立体のオブジェクトを画面上で操作できる ActiveCube[4] などが提案されている．

musicBottle では，並べたガラス容器をインタフェースとして用いる．ガラス容器のふたをあけることで音楽が流れ，そしてふたを閉めることで音楽が止まる．ガラス容器の口にコイルが巻いてあり，容器の蓋を空けると磁界に変化が起こるため誘導電流が流れる．その周波数に応じて，それぞれの容器に関連付けられたバイオリンやピアノ，ドラムなどの音が流れる仕組みになっている．また，ユーザはいろいろな楽器を組み合わせることによって演奏をさらに高度な音楽にすることができる．

SandScape では砂をインタフェースとして用いる．光学的に半透明な特殊な砂を用いて，砂丘の高さにより透過する光の量が異なる性質を利用し，赤外線光源と赤外線カメラを用いて，その 3 次元形状を読み取る．ユーザは砂の形状を変化させることで地形の変化を入力する．出力結果は砂に投影され，地形変化による日照や水の流れをシミュレートできる．

ActiveCube はブロック型のインタフェースである．ユーザはブロック型の入力装置を積み木のように組み合わせることで計算機上で 3 次元形状を作成できる．ブロック自体に各種センサやアクチュエータを実装しているため，計算機の画面上で立体のオブジェクトのシミュレーション結果を表示し，さらに計算機上の操作が入力装置に反映される双方向ユーザインタフェースとなっている．

2.2 実物体のカードを用いた研究

本研究ではレイヤの理解を助けるために実物体のカードを用いるが，他にも実物体のカードを用いた研究がある．Nelson らは，プレゼンテーションを助けるシステムとして Palette[5] を提案している．プレゼンテーションのスライド一枚一枚を実物体のカードに印刷し割り当てることで，よりスムーズなプレゼンテーションが可能となる．このシステムでは実際にカードに印刷してしまうため，カードと対応するものを柔軟に切り替えるといったことは出来ない．

ICandy[6] では，音楽とカードを関連付けることが出来る．ICandy システムは iTunes の操作のためのカードを用いたタンジブルインタフェースを提案している．カードの認識には QR コードを利用している．音楽のイメージとして CD のジャケット写真を用いる．一枚のカードにジャケット写真と QR コードを印刷し，物理的なカードという実体を持たせることで，記録されたメディアへの簡単なアクセスを可能とする

2.3 レイヤへのアクセスに関する研究

レイヤを用いることによって生じる問題点である，自分の編集したいレイヤに移動することの難しさを解決するための研究がおこなわれている．Sriram らは，その解決のためにペンの高さを利用して目的のレイヤを選択する手法をとっている [7]．この研究はレイヤ機能を使いなれた人がいかに素早く目的のレイヤを選択できるかが目的となっている．本研究は，あまり使い慣れていない人が使いやすくなることが目的である．

また，Chi-Wing らは 3D の画像編集ツールにおいてレイヤを選択する際に，ユーザのペンの軌道から，ユーザの移動したいであろうレイヤに自動的に移動するシステムを提案している [8]．よって目的のレイヤが表面になくても，ペンの起動からそのレイヤを選択し，編集することが出来る．そのため，この手法ではレイヤ間を自動的に行き来することが出来る．我々のシステムでは，レイヤそれぞれ独立しているという点を重視しており，レイヤを自分の好きなように入れ替えながら使うというレイヤ機能の利点を理解してもらうことを目的としている．このシステムではレイヤ間を自動で移動するため，本研究とは異なる．

2.4 カードメタファを用いた研究

カードメタファを用いた研究として，以下のようなものがある．牧田らは電子単語帳において，カードのメタファを利用することで紙の単語帳特有の操作感を再現する研究を行っている [9]．紙の単語帳には，カードを実際に動かせることから単語のカードをずらしておきいつでも直接参照できるといった利点が存在する．そういった紙の単語帳の利点を PC 上で再現した電子単語帳に取り入れるために，3D デスクトップ上で単語帳を再現している．

また，卓上にあるトランプやタロットカードを指で操作する感覚を用いた研究として，木村らの開発する WATARI システム [10] がある．WATARI システムでは，操作したいデータア

イテムをカード状の視覚オブジェクトに対応させ，ハンドジェスチャによりそのデータアイテムを操作する．

しかし，これらの手法では実物体のカードを実際に動かせるという利点はなくなってしまい，触覚フィードバックもないため，実際に操作している感覚を感じにくい．

David らは，表示部分が柔軟に変形可能なディスプレイであるフレキシブルディスプレイ向けにジェスチャを用いたインタラクションを提案している [11]．例えば，ディスプレイに表示される画像のスクロールは紙を裏返す (flip) ジェスチャでシミュレートしている．また 2 つ以上のディスプレイを使っている時に，あるディスプレイを持つことで，そのディスプレイをアクティブにすることが出来る．これらのジェスチャは基本的には「一枚の紙」に関するジェスチャに基づいた提案であり，最大でも 3 枚の紙を束ねるインタラクションしか存在しない．何枚もの紙と紙との関係性を利用したシステムではないという点で本研究とは異なる．

第3章　レイヤ操作システム

3.1　システムの概要

ユーザが画像編集ツールを用いて画像編集を行う際に，レイヤを実物体のカードに割り当てる．ユーザはカードと直感的なインタラクションを行うことでレイヤの操作が可能となる．

カード一枚が画像編集ソフトにおけるレイヤの一枚と対応する．カードは机の右側に並べ，置く位置を変更したり，カードを裏返すことで画像編集ソフトを操作することが出来る．

また，ユーザはペンタブレットを用いて画像編集を行える．

3.2　利用シナリオ

画像編集ツールをあまり利用したことのないAさんが，このシステムを使い，人の顔を描く状況を想定する．Aさんはまず輪郭のレイヤを作成するために，一枚目のカードを机の右端に置き，一枚目のレイヤを作成する．（図 3.1 - ①）

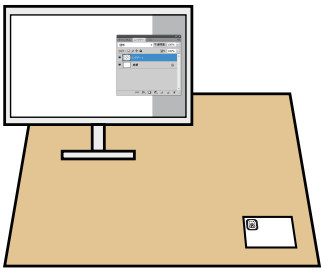
ペンタブレットを用いて，作成したレイヤに描画を行う．輪郭を書き終わったAさんは（図 3.1 - ②），次に髪の毛を描画しようと考え，新しいレイヤを作るために二枚目のカードを机の上に置き，作成されたレイヤに髪の毛を描画した．（図 3.1 - ③）キャラクターに影をつけたくなったAさんは，また新しくカードを机の上に置きレイヤを作成した．（図 3.1 - ④）

このときキャラクターを見ながら編集するために影のレイヤを一番上面にしたくなり，カードの順番を並び替えた．（図 3.1 - ⑤）影を編集し終わったAさんは，影を髪の毛のレイヤの下に戻すためにカードの順番を入れ替えた．（図 3.1 - ⑥）

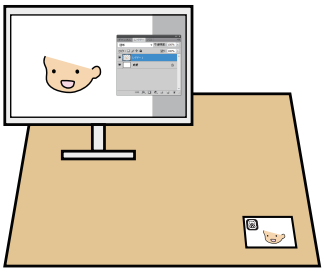
背景を編集しようと思ったAさんは背景用のレイヤを追加した．背景を編集するために顔のレイヤを非表示にしたくなったAさんは3枚のカードをまとめて裏返して非表示にし，背景を編集した．（図 3.1 - ⑦）

背景を編集し終わったAさんは裏返した3枚のカードをもとに戻し，背景と顔の様子を確認した．（図 3.1 - ⑧）

作成した画像を眺めていたAさんだったが，やっぱり背景は白のほうがいいと思ったため，カードを机から片付け背景のレイヤを削除し，完成とした．（図 3.2）



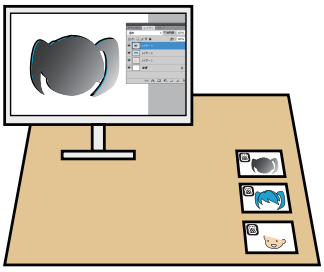
① 一枚目のレイヤを作成



② 輪郭と顔が描き終わる

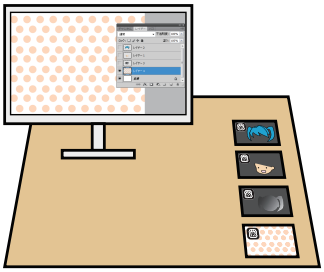
③ 髪の毛のレイヤを作成

④ 影のレイヤを追加



⑤ レイヤの順番を変更し , 影を編集

⑥ レイヤの順番を戻す



⑦ ３枚のレイヤを非表示にし , 背景を編集

⑧ 背景の完成

図 3.1: レイヤ操作システム

図 3.2: 完成図

3.3　使用するカード

今回使用するカードを図 3.3 に示す．カードには片面のみ薄い色が付いており，色のついた方を裏面とする．裏表それぞれの左上にマーカーが付けられている．空白の部分に，現在対応しているレイヤがのサムネイルをプロジェクタにより表示される．

3.4　カードの操作

ここでは，カードを用いて行う操作について説明する．本システムで可能な操作は，新しいレイヤを作成する操作，表示・非表示を切り替える操作，レイヤの順序を入れ替える操作，レイヤをアクティブにする操作，不要になったレイヤを削除する操作の五つである．

3.4.1　新しいレイヤを作成する操作

ユーザはカードを机の上に置きシステムに認識させることで，レイヤを作成することができ，そのレイヤはそのカードと対応付けられる．(図 3.4)

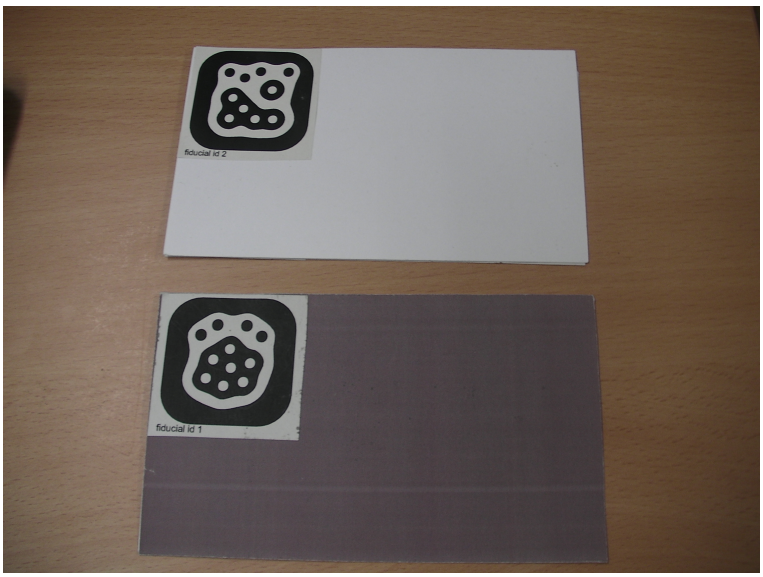


図 3.3: 使用したカード（上：表向き・下：裏向き）

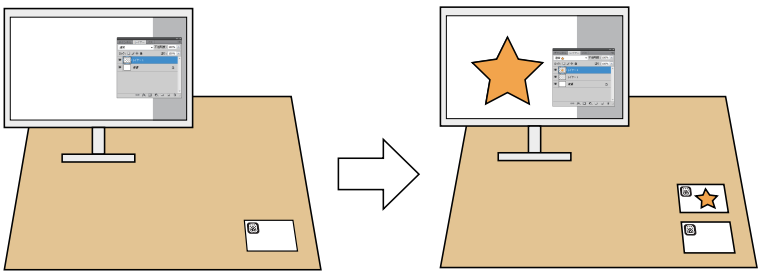


図 3.4: 新しいレイヤを作成する

3.4.2　表示・非表示を切り替える操作

カードを裏返すことで，対応するレイヤの表示非表示を切り替えることが出来る（図 3.5）．トランプにおけるカードでは，表側に数字とマークが印刷されており，裏側には単なる模様が印刷されている．そのためカードを裏返して裏向きにするという動作は，そのカードが何であるかを見えなくする行為であり．逆に裏向きのカードを裏返して表向きにするという行為はそのカードが何であるかわからない状態から，そのカードが何のカードであるか確認する行為である．そのため，ユーザはトランプのカードを確認するのと似たような動作で，表示非表示を切り替えることが出来る．

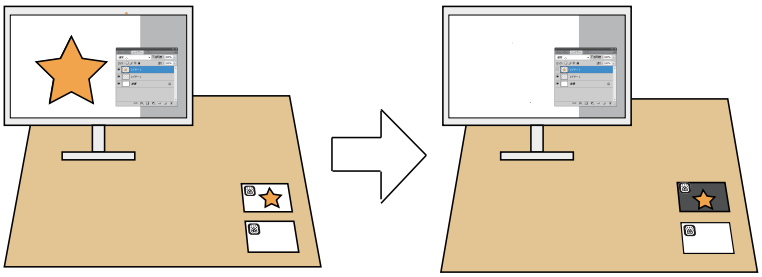


図 3.5: カードを裏返す

3.4.3　レイヤの順序を入れ替える操作

カードを縦に並べることでユーザはレイヤの上下関係を決定する．ほとんどの画像編集ツールにおいて，レイヤは画面の上部に表示されるほど上面のレイヤを表し，下に表示されるほど底面のレイヤを表す．そのイメージと合わせるため，本システムでも，自分から遠い位置に置くほど表面のレイヤ，手前に置くほど底面のレイヤに配置できる．

そのためユーザはカードの順番を並び替えるだけでレイヤの上下関係を変更することが出来る．（図 3.6）

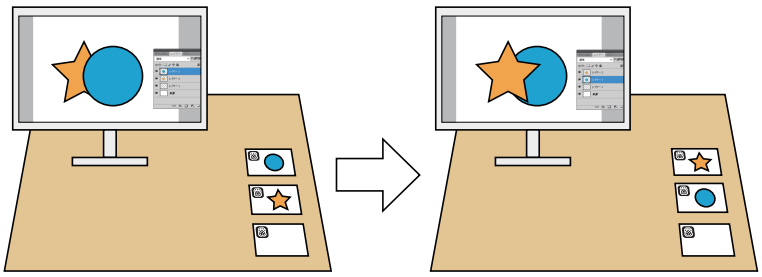


図 3.6: レイヤの順序を入れ替える

3.4.4　レイヤをアクティブにする操作

縦に並んだカードの中から，自分が編集したいカードを左にずらすことで，そのレイヤをアクティブにして，編集可能な状態にすることができる.（図 3.7）これは，自分の編集しているレイヤがわからなくなるという問題点を解決する．カードが実際に動くことで，ユーザは自分の編集しているレイヤが一目でわかり，かつ簡単に編集するレイヤを変更することが出来る.

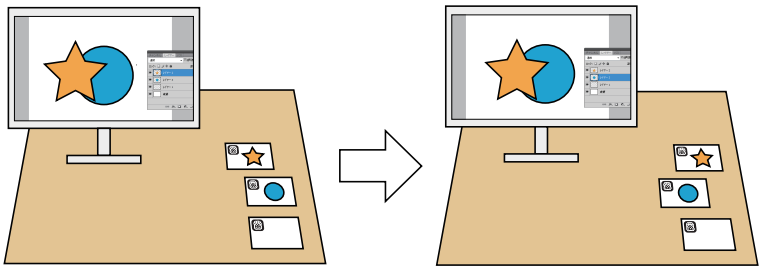


図 3.7: レイヤをアクティブにする

3.4.5　不要になったレイヤを削除する操作

編集を進めていく過程で，あるレイヤを消したくなった場合はそのレイヤと対応するカードをカメラから映らないようにすることで目的のレイヤを削除することが出来る.（図 3.8）そのとき削除されるレイヤの情報はシステムに保存しておき，同じカードがもう一度認識されたときは，そのレイヤを削除した際の状態から編集を再開できる.

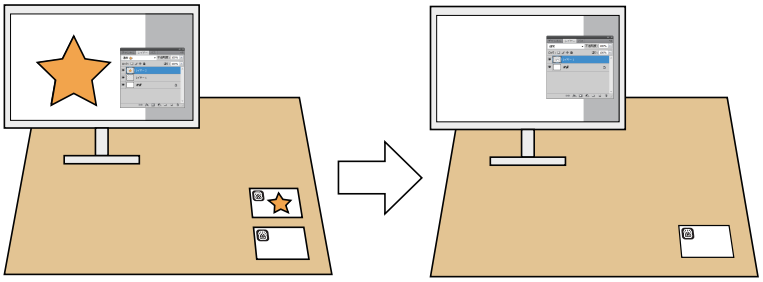


図 3.8: レイヤを削除する

第4章　実装

4.1　開発環境

開発環境として，OS は Windows7，IDE は VisualStudio2008 で実装を行った．開発言語は C++を用いた．ライブラリとして openFrameworks v0.062[16] を用いた．

画像処理のフレームワークとして reacTIVision version1.4[14] を用いた．reacTIVision とは，テーブルトップ型のインタフェースである Reactable[15] のために開発された，オープンソースのコンピュータビジョンフレームワークである．

画像編集ツールとして Adobe Photoshop CS4 を用いた．Photoshop における処理の自動化のために，Photoshop の機能であるスクリプト機能を利用している．スクリプト機能とは Photoshop における操作を自動化するために使われる機能である．スクリプトは Javascript によって実装し，開発には Adobe Extend Script Toolkit を用いた．

4.2　ハードウェア構成

4.2.1　ハードウェア

本システムのハードウェア構成のイメージ図を図 4.1 に示す．カードにレイヤのサムネイルを投影するためのプロジェクタ，カードの動きを認識するための USB カメラを一台ずつ設置する．USB カメラは Logicool Qcam Orbit AF を用いた．解像度は 640x480 で使用した．

それぞれのデバイスを一台の計算機に接続し，その計算機によって処理を行う．

4.2.2　カード

使用したカードの大きさは 75mm × 125mm である．手で持って扱いやすい大きさであり，かつプロジェクタによる表示が十分にできる大きさとしてこのサイズに設定した．

カードの表と裏両方の左上に図 4.2 に示すようなマーカーをつける．マーカーのサイズは縦横ともに 3.7mm である．プロジェクタによる表示がマーカーに重なってしまった際にも，十分に認識できるサイズとして最小のものとしてこのサイズに設定した．

それぞれのカードにカード ID を付与し，それぞれのマーカーにマーカー ID を付与する．カード ID が 1 のカードの表には ID0 のマーカー，裏に ID1 のマーカーが付いている．以降，二枚目に ID2，3．三枚目に ID4，5 と続く．

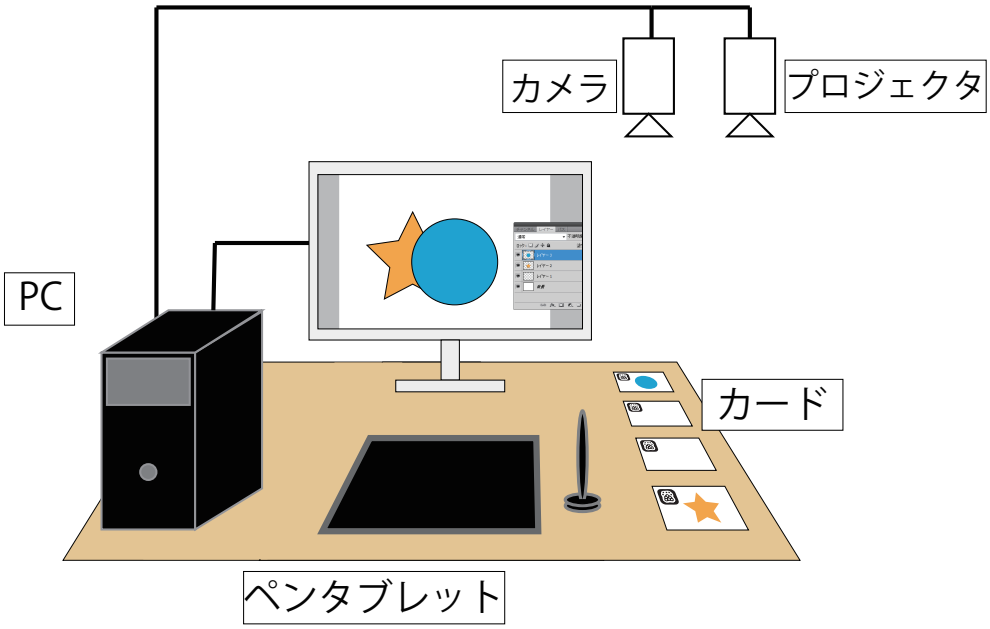


図 4.1: ハードウェア構成

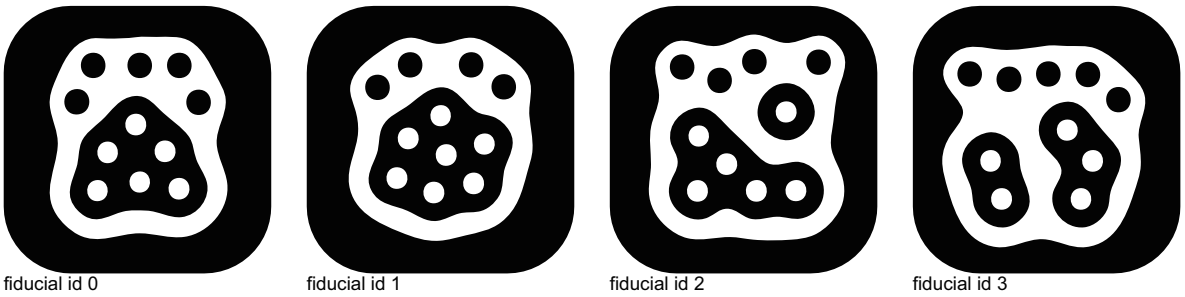


図 4.2: 使用したマーカーの例

4.3　ソフトウェア構成

本システムのソフトウェアは，画像認識コンポーネント，レイヤ管理コンポーネント，表示コンポーネントの 3 つの部分から構成される．ソフトウェア構成のイメージ図を図 4.3 に示す

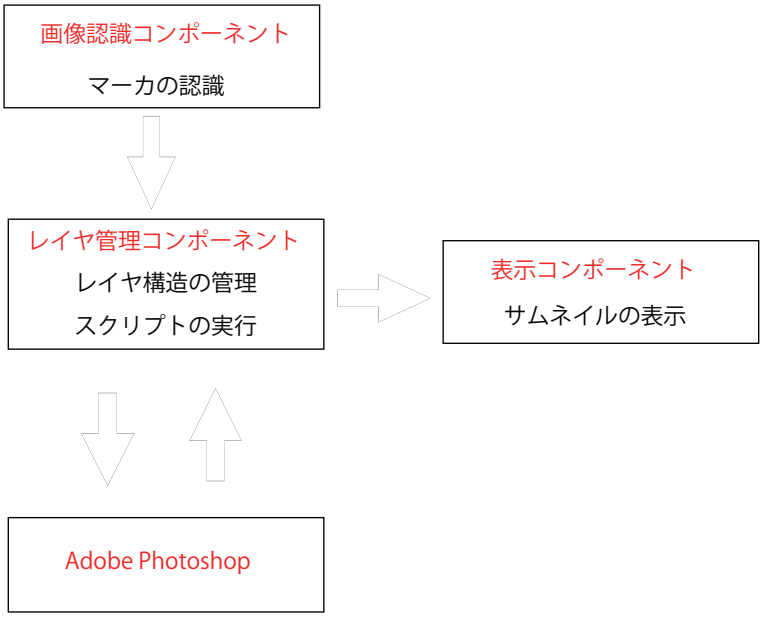


図 4.3: ソフトウェア構成

4.3.1　画像認識コンポーネント

reacTVision を用いて，カメラから得られるカメラ画像からマーカーそれぞれの位置と向きを取得する．取得した情報は TUIO プロトコル [12][13] によってレイヤ管理コンポーネントに送られる．TUIO プロトコルとはマルチタッチのインタフェイスやタンジブルなインタフェイスの情報の送受信に特化した通信プロトコルである．TUIO は UDP によって送信される．

4.3.2　レイヤ管理コンポーネント

システムが独自に保持しているデータ

レイヤ管理コンポーネントでは，以下のデータを内部に保持している．

- カードの位置をもとにしたレイヤ構造
- 編集済みのレイヤと対応するカードのリスト

Photoshop におけるレイヤ構造とは別に，システムが独自にレイヤ構造を持っている．システムはリスト構造としてこの情報を保持しており，より上面のレイヤと対応するカードから順にリストでつながっている．またどのレイヤをアクティブにするのかという情報もここで保持する．またこのレイヤ構造にて，そのカードの対応するレイヤのサムネイル画像を保持している．

レイヤ構造を更新する際には，カメラ画像におけるマーカーの x 座標と y 座標を取得し，y 座標の大きい順に並び替える．また x 座標を元に，そのレイヤをアクティブにするかという情報を更新している．

また，あるカードが追加されたときに，そのカードがすでにレイヤと対応づけられているのかを確認するために，編集済みのレイヤと対応するカード ID のリストを持っている．このデータは，一度削除されたカードがもう一度認識されたときに，以前編集した状態のレイヤから編集を続けるために使用する．

レイヤ管理コンポーネントにおける処理

レイヤ管理コンポーネントは画像認識コンポーネントから送信される情報を受け取り，得られた情報を元に，以下のイベントが起きた時に処理を行っている．

・システムが起動したとき　まず，初期化作業として，Photoshop のウインドウハンドルを検索する．その際に，Photoshop がまだ起動されていない場合は Photoshop を起動する．Photoshop のウインドウハンドルはスクリプトを実行する際に用いられる．

・マーカーの追加を検知したとき　マーカーの追加を検知した場合，まずそのカードが初めて認識されたのかそうでないのかを，編集済みのレイヤと対応するカードのリストと比較して確認する．もし初めて認識されたカードである場合は，新しいレイヤを作成するためのスクリプトを実行し，システム内のレイヤの情報を更新する．一度認識されたことのあるカードだった場合はそのカードと対応するレイヤを復活させるためのスクリプトを実行し，システム内のレイヤの情報を更新する．

さらに，マーカー ID が奇数である場合はカードが裏向き，偶数である場合はカードが表向きであると判断できるため，表向きの場合はレイヤを表示，裏向きの場合はレイヤを非表示にする．

・マーカーの消失を検知したとき　マーカーの消失を検知した場合，そのカードと対応するレイヤを削除するスクリプトを実行する．さらに，システムが持っているレイヤ構造から削除し，そのカードをシステム内の編集済みのカードのリストに登録する．

・マーカーの位置の更新を検知したとき　マーカーの位置の更新を検知した場合，更新されたマーカーの情報をもとにシステム内のレイヤ構造を更新する．その作業によってレイヤ構

造に変化があった場合は，その変化にあわせて，Photoshop でレイヤの上下関係を交換するスクリプトを実行する．

4.3.3 Photoshop の操作

Photoshop における自動処理について

Photoshop の作業を自動化するための機能として，アクション機能とスクリプト機能が存在する．

アクション機能とはマクロ機能とも呼ばれ，Photoshop におけるある一連の作業を記憶しておき，必要な時に利用できるようにする機能である．新しいアクションを作成すると，記録が開始され，記録を中止するまでに使用したコマンドとツールが新規アクションに追加されるという仕組みである．アクションは作成が簡単であり，気軽に使用できることが利点である．しかし，ファイル名の指定などの文字列処理や，画像サイズを計算して求める計算処理，さらに条件分岐といったプログラム特有の処理は，アクション機能では実現できない．

スクリプト機能とは，アクション機能と同様 Photoshop におけるある一連の作業を記憶しておき，必要な時に利用できるようにする機能である．Photoshop では，AppleScript・VBScript・JavaScript の 3 つのスクリプト言語によるスクリプトの作成がサポートされており，Macintosh では AppleScript と JavaScript を，Windows では JavaScript と VBScript を実行することが出来る．スクリプト機能を用いることで，ファイル名の指定などの文字列処理や，画像サイズを計算して求める計算処理，さらに条件分岐といったプログラム特有の処理が可能となる．

スクリプトを実行する為の手順を以下に示す．まず，作成したスクリプトファイルを．jsx という拡張子で，Adobe Photoshop CS4 ディレクトリ内の．Presets/Scripts フォルダに保存する．その後，Photoshop を起動し，ファイル／スクリプトメニューからそのスクリプトを実行することが出来る．

本システムでの Photoshop の自動化

本システムで行う処理は，条件分岐などの作業を用いるため，スクリプト機能を用いて実装を行った．スクリプト言語は JavaScript を使用した．

通常，Photoshop のスクリプト機能による自動処理は，ユーザが自分の実行したいスクリプトを一覧の中から選択し実行する．しかし，本システムでは，カードを動かすなどの操作がおこなわれた際に，システムが自動的に対応したスクリプトを実行しなければならない.，

Photshop の機能として，自分がよく使う処理を自分の好きなショートカットキーに割り当てることのできる機能が存在する．本システムではその機能を利用し，作成したスクリプトそれぞれに個別のショートカットキーを割り当てた．スクリプトを実行する際には，Photoshop のウインドウハンドルに対して実行したいスクリプトと対応するショートカットキーのキーコードを送る．

また，スクリプトは引数を渡して実行することが出来ない．そのため引数 x を受け取って，上から x 番目のレイヤを消すというスクリプトを書くことが出来ない．そこで本システムでは引数としたい番号をいったんファイルに書き出すことで数値をやりとりすることにした．具体的に，上から三枚目に新しいレイヤを作る場合を例に説明する．カードが上から三枚目に追加されると，レイヤ管理コンポーネントにより新しいレイヤを作成するというスクリプトが実行されるが，スクリプトを実行する直前に，”num.dat”というファイルを作成し”3”と書き込む．その後スクリプトを実行する．スクリプトは実行されるとまず”num.dat”を読み込む．その後読み込んだファイルから引数である”3”を確認し，上から三番目に新しいレイヤを作成する．

本システムでは，レイヤの追加，レイヤの復活，レイヤの削除，レイヤの移動という四つのスクリプトを実装した．

・レイヤの追加

このスクリプトはまだ認識されていないカードが追加されたときに実行される．引数として上から何番目に追加したいかという番号と，認識されたマーカー ID を引数として受け取り，引数として受け取った場所に新しいレイヤを作成するスクリプトである．その後，認識されたマーカー ID が奇数である場合は作成したレイヤを非表示にする．

以下に簡単なコードを示す．変数 num には何番目に作成するかという数値が，変数 markerID にはマーカー ID が保存されている．

```
var new_layer = activeDocument.artLayers.add();
new_layer.move(activeDocument.layers[num],ElementPlacement.PLACEAFTER );
if(markerID%2==0){
  new_layer.visible=true;
}else{
  new_layer.visible=false;
}
```

まず，現在編集しているドキュメントに新しくレイヤを作成し，そのレイヤへのリファレンスを変数 new_layer に保存する．このとき，新しいレイヤは現在編集しているレイヤの下に作成されるため，作成したレイヤを目的の位置に移動する．その後，markerID が奇数か偶数かによって，表示と非表示を切り替える．

・レイヤの削除

このスクリプトはカードがカメラから消失した際に実行される．引数として削除したいレイヤが上から何番目かという番号を受け取り，そのレイヤを削除するスクリプトである．しかし，本当に削除してしまうと，もしマーカーの誤認識などの理由でそのレイヤを戻す処理を行うことになった際に非常に時間がかかってしまう．そのため本システムでは，ゴミ箱という名前でフォルダを作り，そのフォルダに移動して非表示にしておくことで対応している．

さらに，ゴミ箱内にあらかじめゴミという名前でレイヤを作成しておく．これは，スクリプトにおいてレイヤを移動するには，あるレイヤを指定してそのレイヤの上下に移動させるという方法しかないためである．そのためあらかじめゴミというレイヤを作成しておき．そのレイヤの上に移動させることで，ゴミ箱に移動している．

以下に簡単なコードを示す．変数 num には何番目のレイヤを削除するのかという数値が，変数 markerID にはマーカー ID が保存されている．

```
var gomibako_Ref = app.activeDocument.layerSets.getByName("ゴミ箱");
var gomi_Ref = gomibako_Ref.artLayers.getByName("ゴミ");
activeDocument.layers[num].move(gomi_Ref,ElementPlacement.PLACEBEFORE );
```

まずゴミ箱というフォルダへのリファレンスを変数 gomibako_Ref に保存する．その後，ゴミ箱内にあらかじめ作ってあるゴミへのリファレンスを変数 gomi_Ref に保存する．そして目的のレイヤをゴミレイヤの上に移動することでゴミ箱のフォルダ中に移動する．ゴミ箱フォルダは非表示になっているため，このフォルダ内に移動してしまえば個別にこのレイヤを非表示にする必要はない．

・レイヤの復活

このスクリプトは，一度認識されたカードがもう一度認識されたときに実行される．削除スクリプトによってゴミ箱に移動されたレイヤをカードと同じ位置に戻すスクリプトである．引数として，復活したいレイヤがゴミ箱の何番目に位置しているかいう数値と何番目のレイヤに復活させるのかという数値，そして認識されたマーカー ID を受け取る．受け取ったレイヤを復活させ，マーカー ID から裏表を判断し表示非表示を切り替える．

以下に簡単なコードを示す．変数 num1 には復活したいレイヤがゴミ箱の上から何番目に位置しているかいう数値，変数 num2 には何番目のレイヤに復活させるのかという数値，変数 markerID にはマーカー ID が保存されている．

```
var gomibako_Ref = app.activeDocument.layerSets.getByName("ゴミ箱");
var restore_Ref = gomibako_Ref.artLayers[num1];
restore_Ref.move(activeDocument.layers[num2],ElementPlacement.PLACEBEFORE );
if(markerID%2==0){
  restore_Ref.visible=true;
}else{
  restore_Ref.visible=false;
}
```

まずゴミ箱というフォルダへのリファレンスを変数 gomibako_Ref に保存する．その後，復活させたいレイヤへのリファレンスを restore_Ref に保存する．復活させたいレイヤを復活させたい場所に移動し，markerID が奇数か偶数かによって，表示と非表示を切り替える．

・レイヤの移動

このスクリプトはシステムが保持しているレイヤ構造が更新されたときに実行される．引数として移動したいレイヤを示す番号と，どこに移動したいかを示す番号を受け取り，レイヤを移動させるスクリプトである．

以下に簡単なコードを示す．変数 num1 には移動したいレイヤを示す数値，変数 num2 にはどこに移動したいかを示す数値が保存されている．

```
layer_ref1 =activeDocument.layers[num1];
layer_ref2=activeDocument.layers[num2];
layer_ref1.move(layer_ref2,ElementPlacement.PLACEBEFORE );
```

・レイヤをアクティブにする

このスクリプトはシステムが保持しているレイヤ構造が更新されたときに実行される．引数としてアクティブにしたいレイヤを示す番号を受け取り，対応するレイヤをアクティブにするスクリプトである．

以下に簡単なコードを示す．変数 num にはアクティブにしたいレイヤが上から何番目がを示す数値が保存されている．

```
activeDocument.activeLayer=activeDocument.layers[num];
```

指定されたレイヤをアクティブレイヤに指定している．

・レイヤごとの画像データの取得

レイヤごとの画像データをファイルに出力するためのスクリプトである．プロジェクタを用いて，レイヤのサムネイルをカードに投影させるために使用する．このスクリプトは 5 秒ごとに実行される．この処理はレイヤ一枚ごとにファイルを出力する処理であることから，非常に時間のかかるスクリプトである．さらに，スクリプトを実行している時間は Photoshop を操作することが出来ないため，時間のかかる処理を行うこのスクリプトはあまり頻繁に呼び出すことは出来ないと考えた．そのため，若干レスポンスが遅く感じてしまうかも知れないが，5 秒ごとに実行とした．

具体的な処理を簡単なコードを示しながら説明する．

```
var visiblelayer:[]
for (var i = 0, j = activeDocument.layers.length; i < j; i++){
  var lay = activeDocument.layers[i];
  if(lay.visible != false){
    push(visiblelayer);
    lay.visible = false;
  }
}
```

すべてのレイヤへのリファレンスを配列 visiblelayers[] に保存し，その後すべてのレイヤを非表示にする．

```
for (var i = 0, j = activeDocument.layers.length; i < j; i++){
  var lay = activeDocument.layers[i];
  lay.visible = true;
  var opt = new JPEGSaveOptions();
  var fileObj = new File(fileName);
  doc.saveAs( fileObj, opt, true, Extension.LOWERCASE );
  lay.visible = false;
}
```

すべてのレイヤに対して，レイヤを表示して保存し，非表示に戻すという作業を行う．

```
for (var i = 0, j = visiblelayer.length; i < j; i++){
  visiblelayer[i].visible = true;
}
```

最後に，スクリプトを実行する前に表示されていたレイヤのみを表示させる．

4.3.4 表示コンポーネント

Photoshop から得られるレイヤごとの画像をそのレイヤと対応するカードにプロジェクションする．その際に，画像認識コンポーネントで取得したマーカーの位置情報をもとに，カードの位置を認識している．

第5章　考察

5.1　試用から得られた知見・課題

今回作成したシステムを試用してもらった．その様子を図 5.1，5.2 に示す．

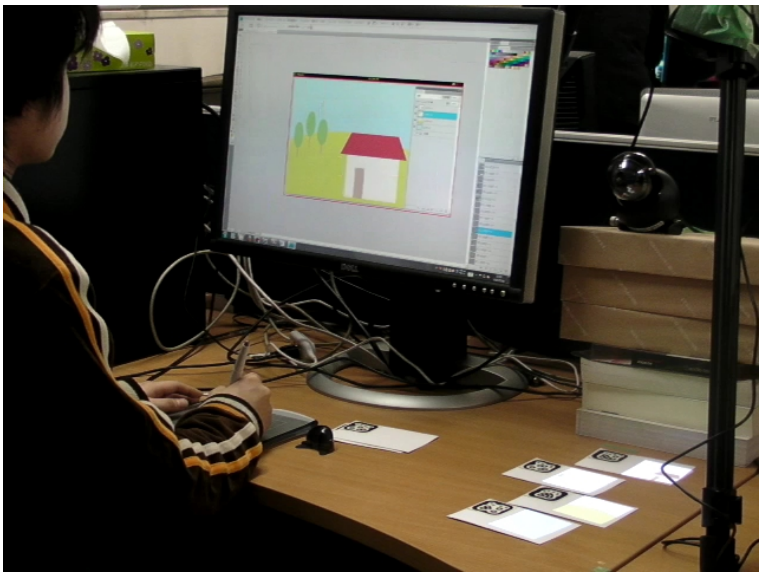


図 5.1: 作成したシステムの試用中

本システムを試用してもらうにあたり，あらかじめ輪郭のレイヤを与えた．試用者はその輪郭を元に色を付け，最後に輪郭のレイヤを消すという作業を行った．

最初に，Photoshop などのツールを用いて画像編集を行ったことの少ない友人 S さんに試用してもらった．システムを試用している際，レイヤを積極的に並び替える様子が観察できた．これはレイヤの上下関係を把握するためであると考えられる．また、レイヤの追加，削除が簡単にできることから，編集に失敗した際に，消しゴムツールを使用して消すのではなく，レイヤごと削除してしまい，新しいカードを用いてまた新しくレイヤを作るといった，普段ではあまりしないような使い方も見受けられた．使用後には，自分がどのレイヤを編集していて，そのレイヤがどこにあるのかということがとても理解しやすかったというコメントを得ることが出来た．

次に，よく Photoshop などのツールを用いて画像編集を行う I さんに試用してもらったところ，複数のカードをまとめて裏返すという動作が便利だという意見が得られた．Photoshop の

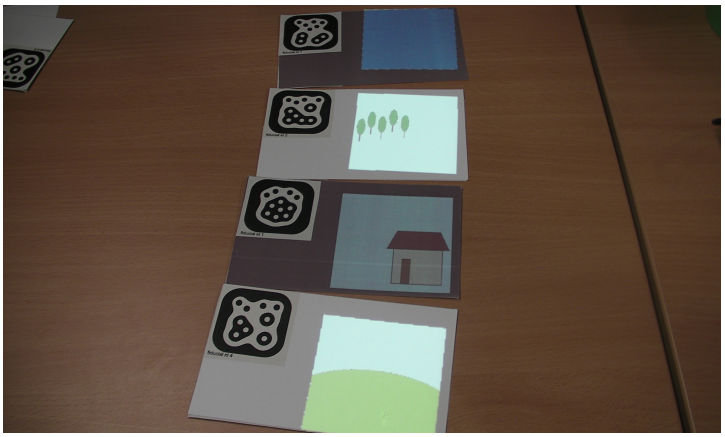


図 5.2: 試用中のカード

ツールパレットを用いて，グループ化されていない複数枚のレイヤを非表示にするには，レイヤの枚数分クリックをしなくてはいけないのに対し，このシステムではカードまとめて裏返すという少ない動作で行うことが出来る．

一方で，問題点も多く残されている．本システムでは，システム内のレイヤ構造に合わせて Photoshop 内のレイヤ構造を変更しているが，Photoshop 内のレイヤ構造が変化した場合に，それに合わせてシステム内のレイヤ構造が変化する仕様になっていない．そのため，ユーザは Photoshop のレイヤパレットを用いてレイヤを変更することは出来ない．つまり，レイヤの変更はすべてカードを使って行わなければならず，そのことで試用中にストレスを感じる場面があった．

また，スクリプトの実行中は画像編集を行うことが出来ないことから，編集作業が一瞬中断してしまう場面があった．特に処理時間の長いスクリプトである,「レイヤごとの画像データの取得スクリプト」が実行される際にこの問題が生じた．

5.2 今後の課題

現在のシステムでは，カードは一種類しか扱っていないが，他とは違う特別なカードを用いることで新しいインタラクションが考えられる．本システムではカードは最も一般的なレイヤであるアートレイヤとしか対応付けられていない．そこで，色の違うカードや大きさの違うカードを，文字を追加し編集するためのテキストレイヤや，四角形や三角形を追加し編集するためのシェイプレイヤなどに対応付けるアプローチが考えられる．

また，今回のシステムではコンピュータビジョンを用いてカードを認識しているため，カードを重ねることが出来ない．そこで，Vogt らによる複数の RFID の認識手法 [17] を用いることで，RFID を用いてカードを重ねられるシステムを実現できると考える．カードを好きなように重ねることが可能になれば，そのことでまた新しいインタラクションが可能となると考える．ひとつの案として，レイヤのグループ化を可能に出来ると考える．一か所に重ねられ

てたカードは一つのグループとして扱うことで，ユーザは直感的にレイヤをグループ化することが可能になる．

第6章　結論

本研究では，カードを用いたレイヤの操作手法を提案し，その手法を実現するシステムの開発を行った．本システムでは，ユーザはカードを操作するだけでレイヤの操作をすることが出来る．また，実物体のカードを用いることでよりレイヤのイメージを理解しやすくなっている．今後は客観的な評価を行い，さらにそのフィードバックから改善を行っていきたい．

謝辞

本研究を進めるにあたり，指導教員の高橋伸准教授をはじめ，田中二郎教授，三末和男准教授，志築文太郎講師には適切なご指導をいただきました．本当にありがとうございました．また，田中研究室の皆様には，ゼミなどを通じて大変貴重なご意見をいただき，研究活動や私生活の両方にわたって大変お世話になりました．ここに心より深く感謝いたします．最後に，研究生活を支えてくださった家族・友人に，心より感謝致します．

参考文献

[1] Ishii, H. and Ullmer, B. Tangible Bits: Towards Seamless Interfaces between People, Bits, and Atoms in Proceedings of CHI '97, pp. 234-241.

[2] H. Ishii, A. Mazalek, and J. Lee. Bottles as a Minimal Interface to Access Digital Information in Extended Abstracts of Conference on Human Factors in Computing systems(CHI 1997), pp.234-241, 1997.

[3] B. Piper, C. Ratti, and H. Ishii. Illuminating Clay: A 3-D Tangible Interface for Landscape Analysis in Proceedings of Conference on Human Factors in Computing Systems (CHI 2002), pp. 355-362, 2002.

[4] 伊藤雄一, 北村喜文, 河合道広, 岸野文郎. リアルタイム 3 次元形状モデリングとインタラクションのための双方向ユーザインタフェース ActiveCube 情報処理学会論文誌, Vol. 42, No. 6, pp. 1338-1347, 2001.

[5] Nelson, Ichimura, Adams, Pedersen. Palette: A Paper Interface for Giving Presentations Proc. ACM Conference on Computer Human Interaction'99 (CHI'99), pp.354-361, 1999.

[6] Graham,J., Hull,J. ICandy: a tangible user interface for itunes Extended abstracts of CHI2008, pp. 2343-2348, 2008.

[7] Sriram Subramanian,Dzimitry Aliakseyeu,Andres Lucero. Multi-layer interaction for digital tables Symposium on User Interface Software and Technology, Proceedings of the 19th annual ACM symposium on User interface software and technology, pp. 269-272, 2006.

[8] Chi-Wing Fu, Jiazhi Xia, and Ying He. LayerPaint: A Multi-Layer Interactive 3D Painting Interface ACM Conference on Human Factors in Computing Systems (CHI 2010) full paper, pp. 811-820, 2010.

[9] 斉藤 朋樹, 牧田 裕喜, 佐々木 整. 3D 電子単語帳の開発教育情報システム学会第 5 回研究会, 22(5):pp. 9-12, 2008.

[10] 木村朝子, 藤田誠司, 岩本和也, 谷津芳樹, 柴田史久, 田村秀行. 壁面と卓上面を併用する電子作業空間 WATARI システムのデザインスキームと実現例 日本バーチャルリアリティ学会論文誌, Vol. 15, No. 2, pp. 191 - 201, 2010.

[11] David Holman, Roel Vertegaal, Mark Altosaar, Nikolaus F. Troje, Derek Johns. Paper windows: interaction techniques for digital paper ACM Conference on Human Factors in Computing Systems（CHI 2005）,pp591-599, 2005.

[12] TUIO http://www.tuio.org/

[13] M. Kaltenbrunner, T. Bovermann, R. Bencina, and E.Costanza. TUIO - A Protocol for Table Based Tangible User Interfaces, in GW '05: Proceedings of the 6th International Workshop on Gesture in Human- Computer Interaction and Simulation, 2005.

[14] reacTIVision http://reactivision.sourceforge.net/

[15] Jorda, S. On Stage: the Reactable and other Musical Tangibles go Real International Journal of Arts and Technology. vol. 1 No.3/4, 268-287, 2008.

[16] openFrameworks http://www.openframeworks.cc/

[17] H. Vogt. Efficient Object Identification with Passive RFID Tags Proceedings of International Conference on Pervasive Computing , Aug. 2002.