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

紙の配置や書き込みを活用したプログラミング環境の開発

夛田 一貴

修士（工学）

（コンピュータサイエンス専攻）

指導教員　高橋 伸

2017 年 3 月

概要

本研究では，初学者やエンドユーザを対象とする紙媒体を用いたプログラミング環境の実装を行い，被験者実験を経て紙媒体をプログラミングのインタフェースとして採用することの有用性を明らかにした．提案手法では，紙媒体の命令カードを規則通りに配置したり，紙に対する書き込みを行う等のインタラクションを行うことにより，簡単なプログラムを作り上げることができる．本研究の目的は，初学者やエンドユーザが扱うプログラミング手法に紙へのインタラクションを取り入れることの有用性を示すことである．

本研究ではまず，一般的なプログラミングに用意されている逐次実行や条件分岐，繰り返しといった基本シンタックスを踏襲し，簡単なグラフィックスの操作が可能な環境を紙インタフェースを用いて実装した．この環境を用いて被験者実験を行い，紙という媒体や，配置や書き込みといった身近な動作をプログラミングの過程に取り込むことが，プログラミングを身近に感じる要素になり得るという結果を確認した．

次に，紙媒体を用いたプログラミングをより一般的な環境に浸透させるために，照明やエアコンといった実世界のデバイスを制御することが可能なプログラミング手法の提案とプロトタイプ実装を行った．この環境では，命令カードの配置に加え，制御するデバイスのパラメータ (照明の色やエアコンの温度など) を色付けや塗りつぶしによって指定することが可能である．この環境を用いた被験者実験では，同等の操作が可能な GUI アプリケーションとの操作性比較の結果や，被験者が行う紙ならではの操作を観察することにより，エンドユーザがスマートデバイスを操作するにあたり本環境が有用に利用できるということを明らかにした．

以上の 2 種類の環境実装と被験者実験を通して，プログラミングに紙媒体を用いることが有効な手段として成立する可能性があることを確認した．

目 次

図 目 次

第1章　序論

1.1　背景

プログラミングは論理的な思考力を養うと共に様々な動作を自動化し，コンピュータの能力を真に引き出すことができるツールであるが，エンドユーザに対しては未だその効果が発揮されていない．多くのエンドユーザや初学者はプログラミングに対しての苦手意識を拭きれずにおり，Mitchel ら [9] の分析によると,「プログラミング言語やツールが難しすぎて文法が取得できず，使いこなせない」「計算や簡単な線や図形を引くだけといった，ユーザの興味とは直接結びつかない事柄が結果として提供されている」といったことが，プログラミングが普及しない原因とされている．こうした分析の通り，現状のプログラミング環境には「取っつきにくさ，扱いづらさ」といったユーザインタフェース的側面の問題と,「プログラミング結果が貧相である」といったプログラミングを行うユーザを楽しませるユーザエクスペリエンス的側面の問題が生じていると考えられる．

こうした背景のもと，プログラミングの基盤となる論理的思考要素や基本的なシンタックスを残したまま，あらかじめ用意された命令を組み合わせることにより本格的な実行結果が得られるエンドユーザ向けの導入環境の実現について，これまでに多くの検討がなされてきた．

Graphical User Interface(GUI) を用いて操作を簡略化したアプローチが提案されている．例えば，Mitchel らの Scratch[9] は，マウスによるドラッグ&ドロップによる命令の組み合わせによってプログラミングが可能な環境である．また，プログラミング動作に実世界の物体を用いる Tangible User Interface(TUI) を採用した環境の提案も多くなされている．Horn らは木製のブロックを命令オブジェクトとし，これを組み合わせることにより積み木遊びのような感覚でプログラミングが可能な環境を提案した [21]．こうした TUI を用いたプログラミング環境は，GUI を用いた環境よりも学習効果や親しみやすさが有意に高いという結果が報告されている [23].

1.2 本研究の目的とアプローチ

1.2.1 本研究の目的

本研究では，紙をプログラミングの入力インタフェースとして扱うことに着目し，本提案がエンドユーザが扱うプログラミング環境として有用性があることを示す．紙という媒体は多くのユーザが日常的に触れる身近なインタフェースであり，これをプログラミングの過程に取り組むことによってどのような効用が得られるかを明らかにする．具体的には，プログラミングの過程に紙への操作を取り入れることによって，エンドユーザのプログラミングに対する親しみやさすが向上するか，プログラミングの過程にどのような変化が生じるのであるかを検証する．

1.2.2 本研究のアプローチ

エンドユーザが扱うタンジブルなプログラミング環境のインタフェースとして紙に着目した．多くのユーザにとって紙は慣れ親しんだ媒体であり，配置や書き込みといった様々なメンタルモデルを既に持ち合わせている．この紙に対する一般的なインタラクション (配置や書き込み) をプログラミング手法に取り込んだ環境を開発し，被験者実験を行うことによって本研究の目的である，プログラミングのインタフェースに紙媒体を採用することによる変化，既存環境との違いを示す．

本研究ではプログラミングに紙媒体を採用することの有用性，既存環境と比較した際のメリットを調査するために，2 段階の環境開発と被験者実験を行った．まず，一般的なプログラミングに用意されている逐次実行や条件分岐，繰り返しといった基本シンタックスを踏襲し，簡単なグラフィックスの操作が可能な環境を構築した．この環境を用いて被験者実験を行い，紙という媒体や，配置や書き込みといった身近な動作をプログラミングの過程に取り込むことが，プログラミングを身近に感じる要素になり得るかどうかを調査した．

次に，紙媒体を用いたプログラミングを，エンドユーザが使用する一般的な環境に浸透させるために，照明やエアコンといった実世界のデバイスを制御することが可能なプログラミング手法の提案とプロトタイプ実装を行った．この環境を用いた被験者実験では，同等の操作が可能な GUI アプリケーションとの操作性比較の結果や，被験者が行う紙ならではの操作を観察することにより，エンドユーザがスマートデバイスを操作するにあたり本環境が有用に利用できるかどうかを調査した．

以上 2 種類の被験者実験を通して，プログラミングに紙媒体を用いることが有効な手段として成立する可能性があることを確認することが，前説で述べた目的を達成するための本研究のアプローチである．

本研究の貢献は，ユーザが慣れ親しんだ紙媒体を用いたプログラミング環境を実装し，紙ならではの操作がユーザに与える影響やメリットを見出し，既存環境に比べて有用な側面を明らかにしたことである．

1.2.3 本論文の構成

第 1 章以降の本論文の構成は以下の通りである．はじめに第 2 章にて関連研究の提示と分析を行うことによって，本研究の位置づけを明らかにする．3 章では前項で述べた 1 つめのアプローチのために実装した環境の概要，構成，使用方法について述べ，4 章にて被験者実験の概要や結果について述べる．5 章では前項で述べた 2 つめのアプローチのために実装した環境の概要，構成，使用方法について述べ，6 章にて再び別の被験者実験の概要や結果について述べる．7 章では 2 種類のアプローチやそれを用いた被験者実験の結果についての考察，議論について述べ，最後に 8 章にて結論を述べる．

第2章　関連研究

本研究と関連している研究として，まずエンドユーザを対象とするプログラミング環境の研究があげられる．また，中でもプログラミングの操作手法についてタンジブルな側面から考察を行っている研究については，本研究と近い位置づけにあり関連性付けをする必要があるため，このテーマに関する研究についても述べる．また本研究の 2 つめのアプローチは簡単な操作で実世界操作を行うプログラミング手法について実装及び考察を行っているため，実世界操作プログラミングの先行研究について述べる．最後に本研究の中心となる紙媒体について，同様に紙を入力インタフェースとして採用している研究を述べる．本章では，それぞれの研究について述べた後，本研究との関連性や相違点を示す．

2.1　エンドユーザ向けプログラミング環境の研究

これまでにエンドユーザに優しいプログラミング環境について様々な研究が行われてきた．HMMMML[1] では，プログラミングに対して苦手意識を持つ学生の学習モチベーションを高めることを目的に，最低 1 行から実行できる非常に簡潔な構文によって，音声合成や web 検索，MIDI の出力を実行を可能にした新言語をデザインした．この研究の後継となる HMMMML2[2]，HMMMML3[3] では，プログラミングにおけるソースコードのスペルミスや宣言抜けなどを許容し，中途半端なコードであっても意味解釈を行うことによりエンドユーザがプログラミングに苦手意識を持つ要素の一つであるコンパイルエラーが極力起きない環境を示した．Nigrai[4] はプログラムを自動的に可視化することにより，処理の流れの直感的理解を促した．コードの可視化については Alex らの研究 [5] でも言及されており，ライブコーディングの様子を可視化することによってライブコーディングを見る視聴者に対する視覚的面白さのアプローチを提案した．また，加藤ら [6] はプログラミングに時間軸を取り入れ，ソースコードの過去の状態と実行結果を視覚的に表示することによってプログラミングエラーを少なくする手法を示した．佐藤ら [7] は統合環境 eclipse 上で動作するプログラム実行部分の可視化システムを示した．これらの研究はプログラミング環境そのものを自らデザインしたものであるが，伊藤ら [8] は一般的なプログラミング言語の使用の際にメモ用紙というアナログな要素を取り入れることにより，プログラミングを行うユーザの思考の見える化とそれに伴う理解の深化手法を提案した．Scratch[9] はマウスによる操作によって，画面上に表示される命令ブロックのドラッグ&ドロップ動作を行い，これによって簡易的なプログラミングができる環境である．VISCUIT[10] ではメガネと呼ばれる命令オブジェクトに対して絵や数字を書き込み，その組み合わせによって様々な命令を作ることができる．これらのような GUI 操

作を取り入れたプログラミング環境を用いたワークショップを通した学習効果やプログラミングに対する意識変化に関する調査が行われており [11][12]，その結果プログラミング手法をGUI でより簡単にすることがエンドユーザのプログラミングに対する姿勢や意識に有効に働くという報告がなされている．

これらの研究は主に CUI や GUI を用いたプログラミング環境についての提案がなされている．本研究では，これらの研究と同様にエンドユーザが扱いやすいプログラミング環境デザインについて考察すると共に，紙というタンジブルな媒体を用いることによって，よりユーザフレンドリなインタフェースの提案を行う．

2.2 タンジブルなプログラミング環境の研究

エンドユーザに対するプログラミングの導入を容易にするために，TUI をプログラミングのインタフェースとして取り入れている環境がこれまでにいくつか提案されてきた．タンジブルインタフェースは Hiroshi[13] らによって提唱された，現実世界の有形オブジェクトを用いてコンピュータの操作を行う概念であり，プログラミング環境に関する研究でも，この直感的なインタフェースを取り入れた提案がなされている．

E-block[14] や T-Maze[15] はブロックの中に LED やセンサ，アクチュエータを取り入れ，これを並べることによって，LED 制御やセンシングを取り入れたプログラムが作成可能な環境を示した．こうした環境を用いた子供を対象としたワークショップでは，子供に対して問題の抽象化，自動化，分析，分解といった論理的思考意識を育成する可能性を持っているという報告が挙げられている [16]．八城ら [17] は，前節で述べた Scratch の物質化を実現し，パズル状のオブジェクトを組み合わせて Scratch のプログラムを動作させる環境を構築した．Turtan[18] や Andrew らの研究 [19] では，テーブルトップディスプレイの上にタンジブルなオブジェクトを組み合わせて置くことによって，画面に表示されるビジュアライズを制御できる環境を提案している．また，tactusLogic[20] や Tern[21] は木製の命令ブロックを組み合わせることによってプログラミングが可能な環境を提示した．また， Michael ら [22] は，木製のブロックを用いたプログラミング手法と，GUI を用いたプログラミングを組み合わせたハイブリッドな環境についての研究も行っている．この環境ではブロックによって基本的な振る舞いを決めるタンジブルな操作と，置かれたブロックの詳細な振る舞いを設定するグイラフィカルな操作が共存している．

これまでにあげたようなタンジブルインタフェースを採用したプログラミング環境は，GUI のみのプログラミング環境と比較して学習効果が高いという報告がなされている [23]．

本研究では，タンジブルなインタフェースとして紙媒体を利用する．これにより，有形オブジェクトをただ配置するだけではなく，紙に対する書き込みや塗りつぶしといったアナログの動作をプログラミングの過程に取り組むことができる．

2.3 実世界操作プログラミング環境の研究

ユーザが能動的に，またはプログラムによって自動的に身近なデバイスを制御する，スマートホームという考え方が社会に受け入れらている [24]．こうしたユーザが任意に家電等の実世界デバイスの制御をプログラムする環境は，実世界指向プログラミングと称して提案され [25]，これを実現する多くのエンドユーザツールが提案されている．Jan ら [26] や Fahim ら [27] は，あらかじめ家庭のいたる所にセンサやアクチュエータを設置し，GUI 操作や RFID カードをかざす操作によってユーザがコンテキストに合わせたスマートホーム制御プログラムを構築できる環境を提案した．Colin ら [28] は，家庭にあるテレビやビデオを，*GetImage();* や *Play();* といった非常に簡潔な命令文で制御する関数ベースのプログラミングシステムを構築した．

その他に，前節で述べた Scratch のような命令のピースを繋ぎ合わせるような GUI 操作によってロジックを組み立て，ユーザが設定したい任意のコンテキスト (時間が経過したら，扉を開けたら，照明がついたら，など) を設定して，家電を操作するプログラミング環境が提案されている [29][30][31].

既存のスマートホーム製品を用いたユーザ調査を行った研究もある [32]．この研究はユーザのスマートデバイスに対する利用状況を観察するものであり，すべての被験者がデバイスを操作するアプリケーションを日常的に利用している様子が観測された．

また，IFTTT はユーザが GUI 操作を用いてスマートデバイスの動作要件と動作条件を指定してプログラムを組むことができる環境である．このような,「環境がどのような状態になったら (動作要件)，どのようにデバイスを制御するか (動作条件)」を定めてプログラミングを行う 手法は Trigger-action programming と呼はれている．Trigger-action programming を用いた実世界制御環境は，他に提案されているプログラミング手法と比較して，エンドユーザに取って扱いやすいものであるという研究報告が挙げられている [33].

本研究で行ったアプローチ 2 では，実世界指向プログラミングに有用な手法とされる Trigger-action programming を紙インタフェースによって実現し，プログラミング環境そのものが実世界に溶け込むデザインの提案を行う．

2.4 紙を用いた操作手法の研究

紙をコンピュータへの入力として採用したり，紙を用いることによって操作性を拡張する手法が研究されている．紙コントローラパネル [34] は，紙に描いた図形をボタンやスライダーとして扱うことができ，画像処理によって図形に対するタッチ入力を検出することによりコンピュータの操作を行う．紙窓 [35] は，カードに対して複数の導電部を取り付け，タッチパネルディスプレイに重ねることで対応したアプリケーションが起動する．タッチ点はカードのデザインによって様々であるので，生じたタッチ点から重ねられたカードがどのようなものであるかを識別することができる．Greg ら [36] はユーザが自らデザインすることができる紙製のロボットを提案した．Sketching in Circuits[37] では，導電性のあるテープを用いることにより電子回路を紙の上に実現する手法を示した．また Robert[38] は本にシール状の命令オブ

ジェクトを貼り付けることによって物語の展開をプログラムする環境である．Daisuke[39] らは，タッチ入力が検出可能な紙シートを導電性インクを用いて実現し，導電パターンをユーザが任意にデザイン可能なインタフェースを提案した．また，竹川ら [40] はユーザが紙の上に描いた絵を楽器に見立てるインタフェースを提案した．

これらの研究は，紙が持つアフォーダンスである，配置，書き込み，貼り付け，つなげるといった特徴を入力インタフェースとして活用している．こうした操作はユーザにとって親しみがある身近な動作であり，システム操作に対する敷居が低くなると考えられる．また，入力インタフェースそのもののコストを低価格化できると共に，印刷するだけで準備ができるといった整備の容易さに繋がることも多い．

本研究では，こうした紙ならではの特徴をプログラミングの過程に取り込み，エンドユーザのプログラミングに対する操作性，親しみやすさに寄与することを目的としている．

第3章　図形や音の操作が可能な紙媒体を用いたプログラミング環境

本章では，1 章で述べた本研究のアプローチの第一段階である，一般的なプログラミングシンタックスを備えた紙媒体の命令オブジェクトを用いた環境の提案についての概要や機能，実装及び被験者実験とその結果について述べる．

3.1　概要

提案する環境は，図 3.1，3.2 に示すような紙媒体の命令オブジェクトを並べて組み合わせることによってプログラミングを行うことができる．紙媒体を命令オブジェクトとして採用することで，プログラミングの過程に紙に対する手書き入力や自由なスケッチを行うといったアナログな動作を組み込むことが可能である．ユーザは命令オブジェクトを組み合わせてプログラムを作り，画面上における図形の描画・移動等を制御する．命令は，図や音，アクチュエータの操作，繰り返し，条件分岐，変数操作などが用意されている．ユーザは本環境を通して，プログラミングに必要な逐次実行や条件分岐，繰り替えしといった概念について触れることができる．

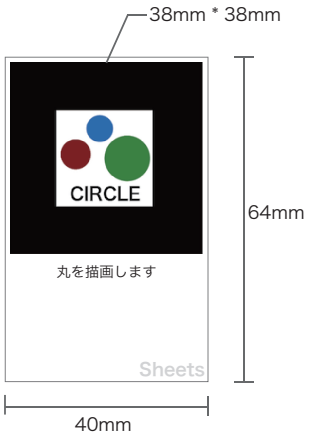


図 3.1: 図形や音の操作が可能な紙媒体を用いたプログラミング環境で用いる命令オブジェクトの概要

図 3.2: 図形や音の操作が可能な紙媒体を用いたプログラミング環境で用いる命令オブジェクトの例

本章で提案する環境の全体図を図 3.3 に，システム構成を図 3.4 に示す．本環境はプログラミングを行うツールとなる紙媒体の命令オブジェクト，並べられた命令オブジェクトを認識するための web カメラ，命令オブジェクトの意味を解釈し実行結果の表示を行うためのコンピュータ及びソフトウェアからなる．

図 3.3: 図形や音の操作が可能な紙媒体を用いたプログラミング環境で用いるシステムの全体像

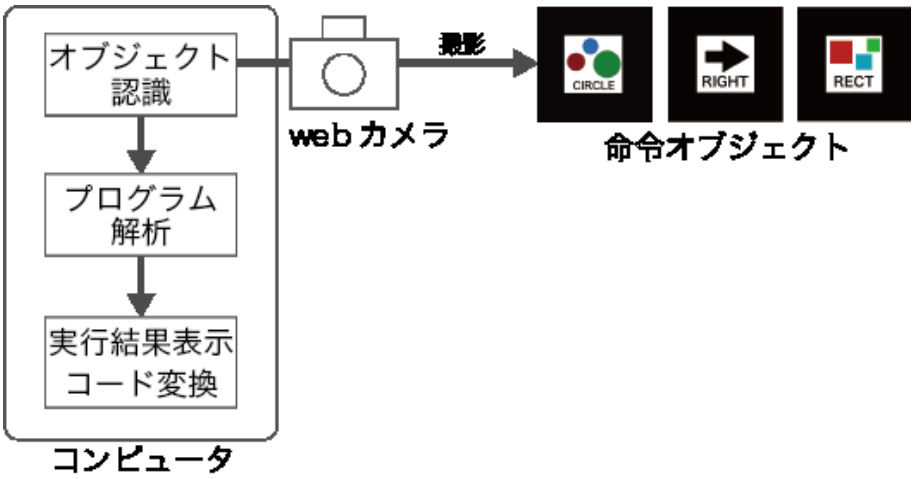


図 3.4: 図形や音の操作が可能な紙媒体を用いたプログラミング環境で用いるシステム構成

3.2 実装

本環境では，紙媒体の命令オブジェクト，web カメラおよびプログラムの認識と処理を行う PC を使用する．紙媒体の命令オブジェクトには命令の内容を表すと共に認識のために使われるマーカと，命令の説明文が記述されている．並べられた命令オブジェクトは web カメラ (ZiggiHD[1]) で撮影され，実行結果が PC の画面上の図形の変化によって表れる．命令を認識し解釈を行った上で実行結果を反映するソフトウェアは ActionScript 及び openFramworks によって実装した．ソフトウェアの実行画面を図 3.5 に示す．

図 3.5: 図形や音の操作が可能な紙媒体を用いたプログラミング環境におけるソフトウェア実行画面

3.2.1 命令オブジェクトの認識

命令オブジェクトには AR マーカが描かれているため，これを認識することによってどの命令オブジェクトがユーザによって設置されたかを判断することができる．AR マーカとは正方形の黒枠で囲まれた中に何かしらのパターンが描かれた画像である．パターン画像を数値化して，パターンファイルとしてあらかじめソフトウェアに登録することにより，web カメラによって撮影される画像の中に登録されたパターンが存在するかどうかを判断することができる．本システムで用いた AR マーカは解像度を 16*16 の 256 分割とした．ソフトウェアで登録されるパターン画像は，任意の ID とカメラ空間状における位置座標を持つため，これらの情報を使用することによって，実世界でどのような命令オブジェクトがどのように並べ

[1]ZiggiHD http://www.latex-cmd.com/struct/footnote.html

られているかを把握することができる．AR マーカの認識には ActionScript3.0 の AR ライブラリである FLARToolkit[2]を用いた．

また，本手法ではマーカに対する塗りつぶしを行うことによって命令の内容が変化することがある．登録されたマーカと認識されたマーカの差分を出し，残った画像 (塗りつぶし部分) の位置や面積を計算することによって，ユーザの意図通りの命令として認識するように処理している．

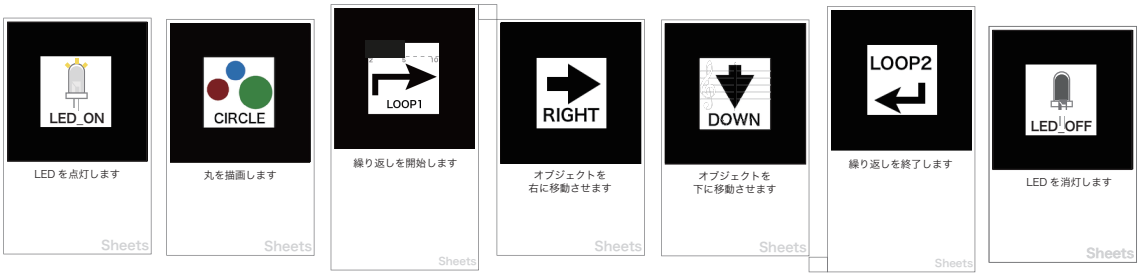
3.3 紙プログラミングに用いる命令カード

本環境は基本的に図形の描画や音の再生とその制御を組み合わせることによってプログラミングを行う．そのため，各種図形を描画する命令である「描画命令」，任意の図形を移動させる「移動命令」，サウンドを発生させる「音命令」，変数を操作する「変数命令」，繰り返しや条件分岐といった基本的なシンタックスを記述するための「構文命令」」等が用意されている (図 3.6)．これらを組み合わせたプログラムの一例を図 3.7 に示す．

命令オブジェクトによって作られたプログラムの実行結果は図 3.5 に示した実行結果画面部に表示される．描画命令に対応した結果は，四角や丸などの図形が実際に描画される．変数命令に対応した結果は，実行結果画面下部に値として表示される．命令はユーザ視点から向かって左から順に逐次実行されるため，描画命令は置く順番によって実行結果の見た目が異なることがある．

[2]FLARToolkit http://www.libspark.org/wiki/saqoosha/FLARToolKit

図 3.6: 図形や音の操作が可能な紙媒体を用いたプログラミング環境で実装されている命令カード群



LED を点灯し，丸を右下に 5 回移動させた後，LED を消灯する

図 3.7: 図形や音の操作が可能な紙媒体を用いたプログラミング環境を用いたプログラムの例

3.3.1 ソースコードへの変換

ユーザが命令オブジェクトを並べて作るプログラムは，実際に存在するプログラミング言語のソースコードにリアルタイムに変換して表示される．変換される言語は，描画やサウンドの制御が容易に行える Processing 言語[3]を用いた．例えば，四角を描画する命令オブジェクトが置かれた場合，ソースコード出力部には Processing にて四角を描画する命令である *rect(x,y,w,h);* が出力される．

3.4 紙による命令オブジェクト

命令オブジェクトとして紙を用いることにより，紙がもつアナログな動作をプログラミングの過程に取り入れることができる．本環境では，命令オブジェクトへの書き込みによって繰り返し文の回数や変数の値を変更することができる (図 3.8(a))．ユーザが書いた自由な絵を描画命令として扱うといったことも可能である (図 3.8(b))．

その他大きな特徴として，本環境では作成したプログラムをノートや本に貼付けることができるため，周りに注釈や解説を書きながらの学習が可能である (図 3.9)．これは一般的なプログラミングにおけるコメント文に相当するものであり，ユーザの試行錯誤の過程をアーカイブすることができるため，学習の大きな支えとなりえると考える．

[3]Processing https://processing.org/

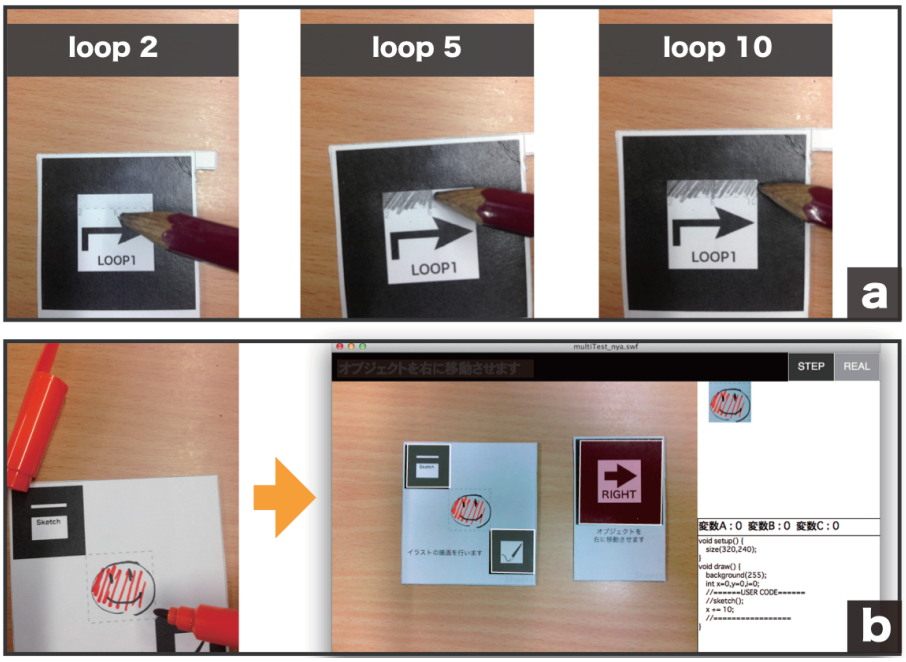


図 3.8: 図形や音の操作が可能な紙媒体を用いたプログラミングの環境における書き込みによるインタラクションの例

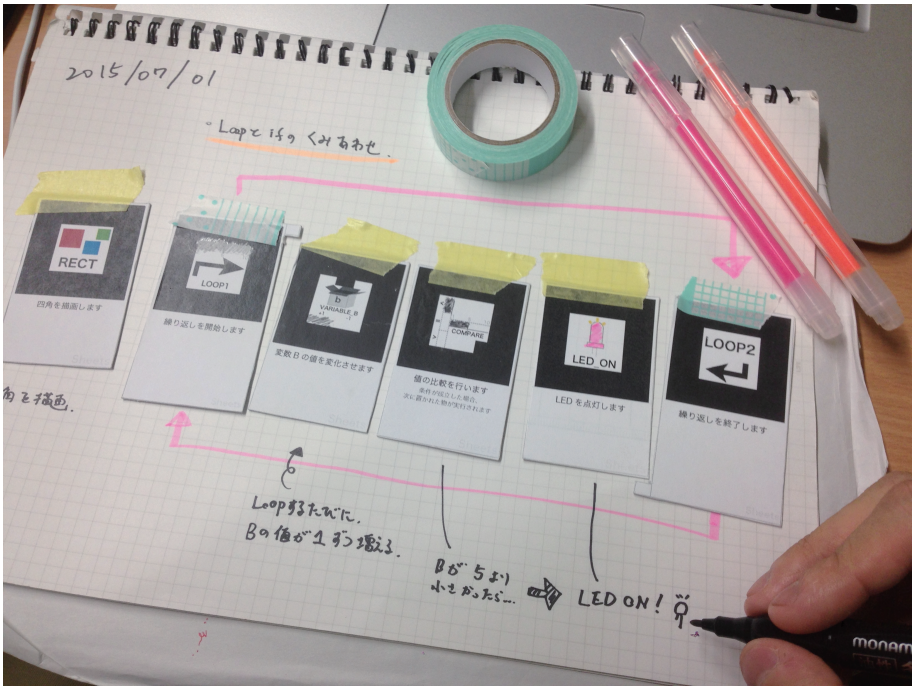


図 3.9: プログラムをメモとともに保存する例

第4章　図形や音の操作が可能な紙媒体を用いたプログラミング環境を用いた被験者実験と結果

本章では，前章で提案した紙媒体を用いたプログラミング環境を使用した被験者実験について述べる．被験者実験の目的は，提案手法のユーザビリティ調査と，紙に対する書き込み等のインタラクションがユーザにとって有意な操作として成立するかを調査するためである．

4.1　実験内容

提案手法のユーザビリティ及び学習効果を評価するために，被験者実験を行った．被験者は 18 歳から 23 歳までの合計 12 名 (女性 3 名) であり，全員が大学生もしくは大学院生である．プログラミングを日頃から使用する情報系の学生が 7 名，授業等で多少の経験がある学生が 3 名，全く経験のない学生が 2 名であった．被験者は本環境を用いたプログラミング方法や特徴の説明を受けた後，いくつかの問題が提示され，それに回答する形でプログラミングを行う．提示する問題の例を以下に示す．

- 例題１：四角形を描画し，5 回下に移動させつつ，ドの音を発生させるプログラム
- 例題２：プログラム終了後に，変数 B が 5，変数 A が 10 になっているプログラム

また，問題回答形式の実験の後，自由にプログラミングを行う時間を設けた．実験終了後，被験者に対し，環境についての操作性，思った通りのプログラミングが行えたか，紙媒体を用いていることへの感想，総合的な好感度等について 7 段階のリッカート尺度を用いたアンケートを行い，その集計を行った．

実験内容は紙へのインタラクションが含まれた環境のものと，含まれていない環境の 2 通りを用意し，被験者はどちらか片方のみを体験した．提示される問題は同じ物であるが，紙へのインタラクションが含まれていない環境では，図 3.8(a) で示したようなループ回数や変数値，条件分岐などの指定を行う際に，あらかじめ塗りつぶして用意されたものの中から適切なものを選ぶといった，紙に対する塗りつぶしや書き込みなどの動作が制限された条件で実験が行われた．

この２つの異なる条件下における被験者の環境への評価を比べることにより，紙へのインタラクションがプログラミングの過程に取り組まれることが，プログラミングへのモチベーションや環境への好感度に影響するかどうかを調査することがこの被験者実験の目的である．

4.2 結果

図 4.1 に実験の様子を示す．紙へのインタラクションが含まれた環境 (GroupA 7 人) と，含まれていない環境 (GroupB 5 人) における，被験者の実験後アンケート結果を示したものが表 4.1 となる．またその平均及び分散を算出したものが表 4.2 となる．

表 4.1: 図形や音の操作が可能な紙媒体を用いたプログラミング環境におけるアンケート調査の結果

	Group A							Group B				
	被験者1	被験者2	被験者4	被験者6	被験者7	被験者10	被験者12	被験者3	被験者5	被験者8	被験者9	被験者11
環境の操作性	7	6	5	4	7	6	7	7	5	6	6	6
紙媒体使用の印象	7	6	7	7	7	6	7	7	4	5	4	5
思い通りの操作	7	6	5	7	7	6	6	5	4	6	6	6
環境への好感度	7	7	6	7	7	7	7	7	5	5	5	6
使用モチベーション	7	7	6	7	6	6	7	7	4	5	5	6

表 4.2: 図形や音の操作が可能な紙媒体を用いたプログラミング環境におけるアンケート調査の平均及び分散

	被験者 1(A)	被験者 2(A)
環境の操作性	6 (SD=1.06)	6 (SD=0.63)
紙媒体使用の印象	6.71 (SD=0.45)	5 (SD=1.09)
思い通りの操作	6.28 (SD=0.70)	5.4 (SD=0.8)
環境への好感度	6.86 (SD=0.35)	5.6 (SD=0.8)
使用モチベーション	6.57 (SD=0.50)	5.4 (SD=1.02)

評価値は 7 段階のリッカート尺度によるアンケート調査によって集計された物である．項目は，環境の操作性 (使いやすいかどうか)，紙媒体使用の印象 (親しみやすいかどうか)，思い通りの操作 (自分の意図した通りのプログラミングができるかどうか)，環境への好感度 (この環境が好きになるかどうか)，使用モチベーション (使用していて楽しさやわくわく感が生まれるかどうか) の 5 つである．

すべての項目にて，紙へのインタラクションが含まれた環境 (GroupA) への評価値が，紙へのインタラクションが含まれていない環境 (GroupB) への評価値と同等かそれ以上のものとなった．また，被験者毎のデータを用いた u 検定では，紙媒体使用の印象と環境への好感度について，紙へのインタラクションが含まれた環境への評価値の方が有意に大きいという結果となった (紙媒体使用の印象 $p = 0.023 < 0.05$, 環境への好感度 $p = 0.017 < 0.05$). 以上の結果から，タンジブルなプログラミング環境では，タンジブルなオブジェクトを整列して命令を組み上げること以外に，ユーザにとって身近な動作をプログラミングの過程に取り込むといったインタラクティブ性を付与することで，ユーザのプログラミング環境に対する印象が明るくなり，親しみ易く感じるようになることが分かった．

その他，アンケートの他に実施した口頭インタビューからは,「鉛筆などアナログなものを使ったため，プログラミングを身近に感じることができた」,「塗り潰しなどの動きがあり面

白い」といった感想を得られた．しかし一方で,「初学者はこれで十分かもしれないが慣れた人には物足りなく感じる」,「実行結果はもっと大きく, 激しいほうが楽しい」といった意見も得られたため，用意する命令群や出力される実行結果，作成可能なプログラムの規模等に改善が必要であると考えられる．

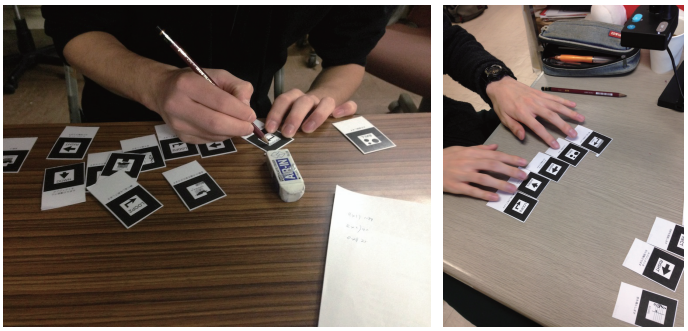


図 4.1: アプローチ 1 で実施した被験者実験の様子

4.3 考察

前節にて述べた実験結果から，紙に対するアナログな動作が，ユーザの操作性，操作意欲に好意的に働いているということが分かった．グループ B の被験者が一貫して適切な命令オブジェクトを探して配置する操作だけを行なっている一方，グループ A の被験者は「ループの回数」,「条件分岐のパラメータ」,「変数の値」,「音符」といった要素について，自ら紙に対して書き込むことによって振る舞いを定めることができる．グループ A の被験者の操作の大きな特徴は，あらかじめ必要な書き込みを全て行なってしまうのではなく，逐次実行の順番通りに命令を一つずつ,「見つける」→「書き込みが必要ならば行う」といった流れでプログラムを作っていた点である．このように，オブジェクトを並べるという単調な作業の間に，書き込みというユーザの個性 (塗りつぶしの濃度や面積) が現れる行為を取り入れることによって，操作に対する退屈さがグループ B のユーザよりも拭えていたのではないかと考える．

また，口頭インタビューで得られた,「鉛筆などのアナログなものを使ったため，プログラミングを身近に感じることができた」といった意見の通り，紙や鉛筆といった多くのユーザが日常的に使用するツールをプログラミングの過程に取り入れることにより，プログラミングという未知な存在のものに対する不安感や嫌悪感のようなものを軽減する効果が生じているのではないかと考える．

しかし，一方で,「初学者はこれで十分かもしれないが慣れた人には物足りなく感じる」,「実行結果はもっと大きく, 激しいほうが楽しい」といった意見が出た通り，本環境に触れる時間が長くなるほど，必ず「飽き」の感情が芽生えてしまうことが懸念されるため，紙に対する操作の多様性や実行結果の規模について改良の余地が存在することが分かった．

第5章　実世界制御が可能な紙媒体を用いたプログラミング環境

前章までの実装および考察を経て，紙を用いたプログラミング環境を見直し，文法やプログラミング対象，システムデザインに変更を行った．以前の環境は既存のプログラミング言語に搭載されている逐次実行や条件分岐，繰り返し，変数などの基本的なシンタックスを持ち，エンドユーザの中でもプログラミングの学習をこれから始める入門者を対象とする導入環境としての側面が強かった．本章で提案する環境は，文法的な要素をより少なくし，プログラミングはもちろんコンピュータの操作にも不慣れなユーザであっても扱えるものを目指した．また，以前の環境ではユーザが作り上げたプログラミングの実行結果が画面のオブジェクトの変化として反映されていたが，本章で提案する環境はユーザの身の回りにある実環境のデバイス制御が可能である．

本章では以上のような改変を加えた，紙媒体をインタフェースとするスマートデバイスを制御可能なプログラミング環境の概要について述べる．

5.1　紙の特徴を活用した実世界プログラミング環境

5.1.1　システムデザイン

本章で述べるような，ユーザが任意に家電等の実世界デバイスの制御をプログラムする環境は，実世界指向プログラミングと称して提案され [25]，これを実現する多くのエンドユーザツールが提案されている．

スマートデバイスを制御する代表的なツールとして，IFTTT という環境が挙げられる．IFTTT はユーザが GUI 操作を用いてスマートデバイスの動作要件と動作条件を指定してプログラムを組むことができる環境である．このような,「環境がどのような状態になったら (動作要件)，どのようにデバイスを制御するか (動作条件)」を定めてプログラミングを行う 手法は Trigger-action programming と呼はれている．

Trigger-action programming を用いた実世界制御環境は，他に提案されているプログラミング手法と比較して，エンドユーザに取って扱いやすいものであるという 研究報告が挙げられている [33]．こうした先行研究の流れに則り，Trigger-action プログラミング形式に特化した紙プログラミング環境を構築した．

ユーザは図 5.1，5.2 に示す紙媒体の命令カードを図 5.3 のように if-then 形式で並べることによってプログラミングを行うことができる．

図 5.1: 実世界制御が可能な紙媒体を用いたプログラミング環境で使用される紙製命令カード

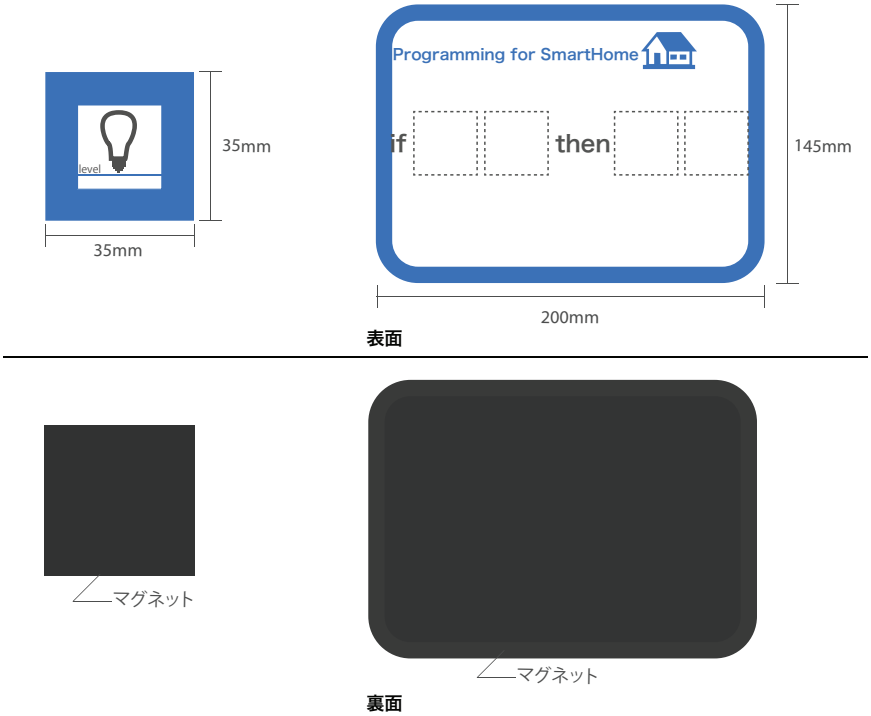


図 5.2: 実世界制御が可能な紙媒体を用いたプログラミング環境で使用される紙製命令カードの設計

図 5.3: 実世界制御が可能な紙媒体を用いたプログラミング環境におけるプログラミング例

if に相当する命令群を Trigger 命令と呼び，これは「任意の時間，気温，SNS 通知」等をもとに後述する Action 命令を実行するためのイベントとなる．また，then に相当する命令群を Action 命令と呼び，これは「音楽を再生する，照明を点灯する，エアコンをつける」等実世界のデバイスを任意に制御することができるものである．

ユーザはこの Tirgger と Action の命令を組み合わせ，「20 時になったらエアコンをつける」といった実世界制御プログラムを組むことができる．現在の設計では，Trigger と Action の命令はそれぞれ 2 枚ずつまで設置することができ，Trigger が 2 枚置かれた場合は AND 文として解釈されるようになっている．

5.1.2　命令一覧

図 5.4 に本環境で提供する Trigger および Action 命令の一覧を示す．ユーザはこれらのコマンドを扱い，Trigger として「任意の時間，気温，SNS 通知」を設定でき，また Action として「照明，エアコン，テレビおよび音楽の制御」を設定することができる．

図 5.4: if-then 形式で指定する Trigger と Action の命令例

5.1.3　書き込み動作

本環境を使用するユーザは if-then 形式に命令カードを並べることによってプログラミングを行うことができる．この際，具体的な日時や制御する照明の色や照度といった具体的なパラメータを，紙に対する書き込み動作によって指定することができる (図 5.5,5.6)．各命令カードには様々な書き込み可能な領域が用意されており，例えば照明制御カードであれば，電球のアイコン内部を色ペンで塗りつぶすことで照明の色を制御したり，カード下部のスライダの塗りつぶし具合で照度を制御することが可能である．

こうした紙に対するアナログな動作をプログラミングの過程に取り組むことは，単純に必要な命令カードを組み合わせるだけの場合よりプログラミングに対する親しみや扱いやすさが増すという，前章の被験者実験によって得られた結果の元設計されている．

前章で扱った環境と比較し，照度や温度といったパラメータの数値を塗りつぶすといった抽象的な動作に加え，照明の色をユーザが選択したペンの色に対応するという，直感的な動作を取り入れた．

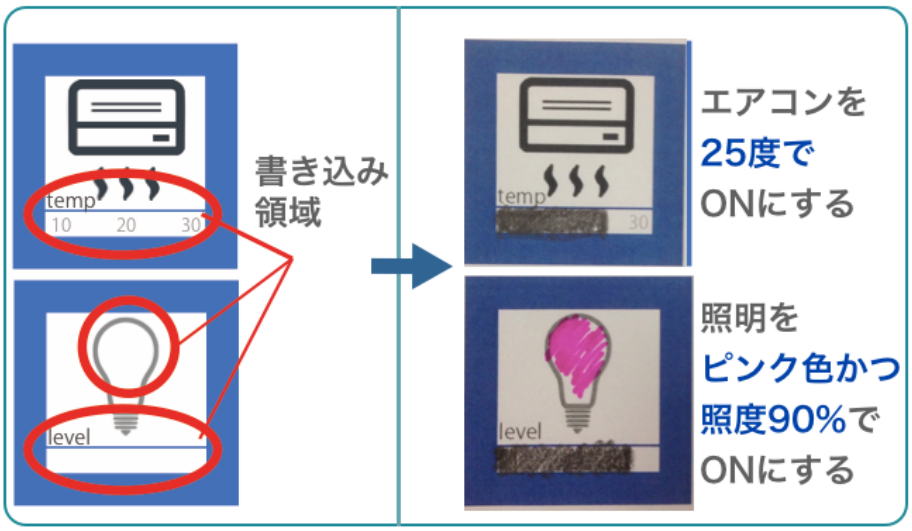


図 5.5: 実世界制御が可能な紙媒体を用いたプログラミング環境におけるユーザの書き込み動作による命令の詳細を定める例

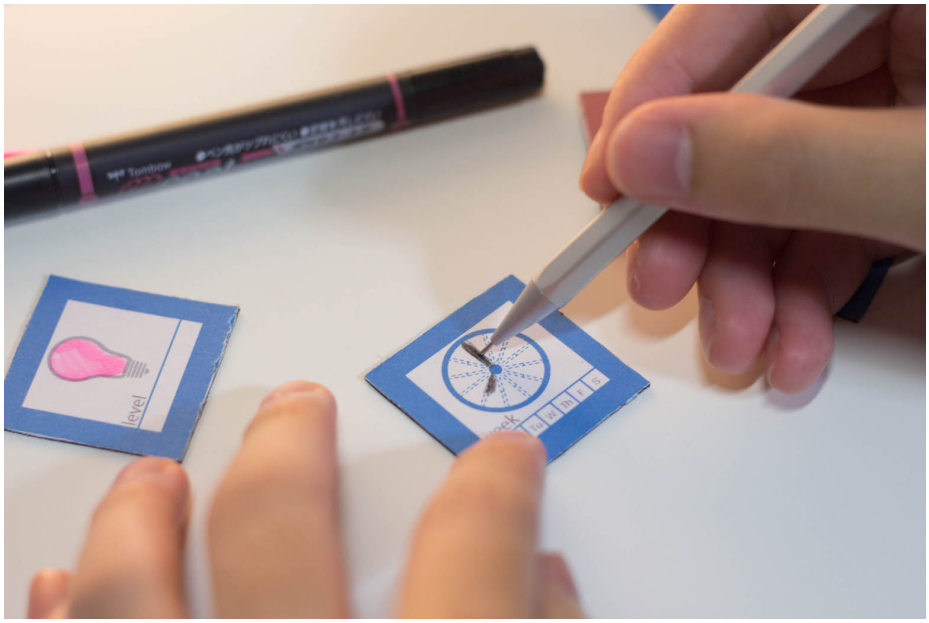


図 5.6: ユーザが実際に書き込み動作を行なっている例

以下に本環境で提供している命令とそれに対応する書き込み動作の一覧を示す.

- Trigger 命令
  - 時間指定命令
    - アナログ針の書き込み : 具体的な時刻の指定
    - AM/PM エリアの塗りつぶし : 午前/午後の指定
  - 気温指定命令
    - 数値の塗りつぶし : 具体的な気温の指定
  - Twitter 通知命令
    - マークの塗りつぶし : お気に入り，返信，DM のどのタイミングを Trigger とするか指定
  - Facebook 通知命令
    - マークの塗りつぶし : お気に入り，返信，DM のどのタイミングを Trigger とするか指定
- Action 命令
  - 照明制御命令
    - 電球の中への色付け : 照明の色の指定
    - 数値の塗りつぶし : 照度の指定
  - エアコン制御命令
    - 数値の塗りつぶし : 温度の指定
  - テレビ制御命令
    - 書き込み動作無し
  - 音楽制御命令
    - 書き込み動作無し

5.2　実装

5.2.1　概要

図 5.7 に本環境のシステム構成を示す．システムは主にソフトウェア部とハードウェア部の 2 つから構成されている．ソフトウェア部では，ユーザが用いる紙媒体の命令カード (35mm*35mm 裏面にマグネットシートを接着) と，それを認識する web カメラ (IPEVO Ziggi-HD Plus)，解析を行うアプリケーションによって構成される．ハードウェア部では，実世界環境のセンシングと制御をおこなうための Arduino マイコンとそれに繋がれる各種環境センサ，学習リモコン，照明制御のための Philips Hue によって構成される．

これらはラップトップ PC 上で動作し，ActionScript3.0 と Adobe Animate を用いて実装を行った．

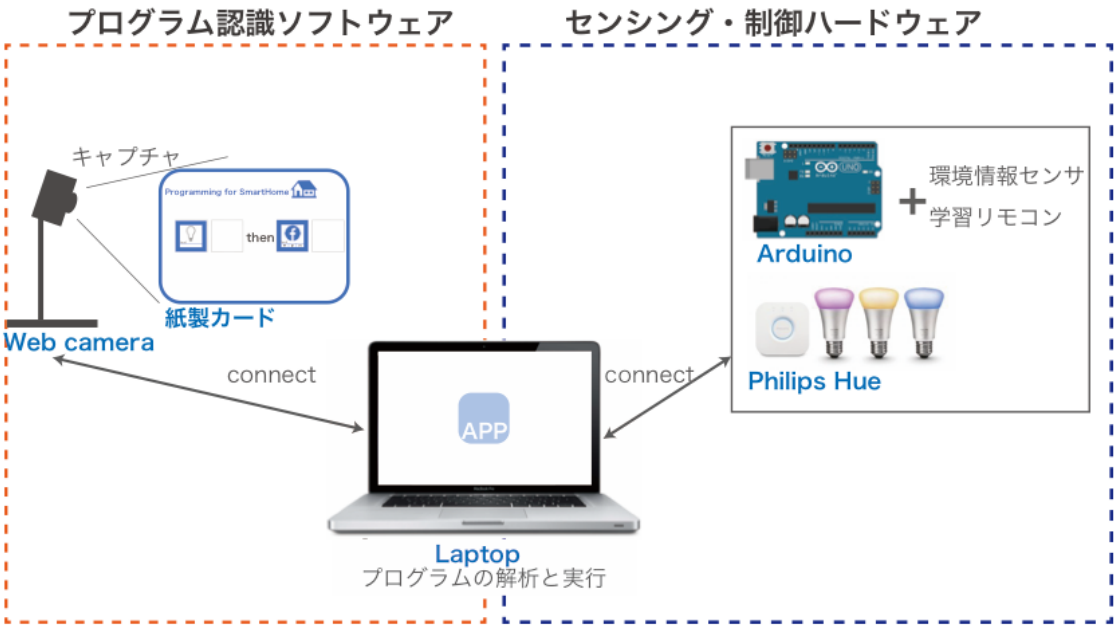


図 5.7: 本システムのアーキテクチャ

5.2.2　ソフトウェア実装

概要

ユーザが並べる紙媒体の命令カードの認識には ARToolkit を用いた．ARToolkit はユーザが手軽に拡張現実感 (Augmented Reality：以下 AR) を用いたアプリケーションを構築することを可能とするソフトウェアライブラリであり，これを使うことにより，web カメラに写り込んだ映像の中から特定の画像パターンの検出することを容易に実現することができる．本手法ではユーザが取り扱うプログラムの構成要素である紙媒体の命令カードのパターンをあらかじめマーカとして認識させておき，映像のどの位置にどのような角度でマーカが検出されたかを判断することで，ユーザがどのようなプログラムを作り上げたのかを推測している．実際に登録されたマーカがソフトウェアによって認識されている様子を図 5.8 に示す．

図 5.8: ARToolkit によってマーカがソフトウェア上で認識されている様子

また，紙に対する書き込み情報は画像処理によって書き込み領域を抜き出している．本環境で提供しているカードは，枠色を薄紺，イラストの線色を薄灰に設定した．ソフトウェア上でマーカと認識された領域を抜き出し，これらの色を除いた部分をユーザが新たに書き込みを行った色や線の情報として認識する．

この新たに加えられた色や線の長さを計測し，ユーザが意図する照明の色や照度といったパラメータを推測することにより，書き込みによる詳細な動作制御を実現した．

マーカに書き込まれた情報の取得

マーカとして登録する画像は 5.1.2 項で述べた 8 個の命令群の書き込み情報がないものである．使用したライブラリでは，あらかじめ登録したマーカ情報に一番近いもの該当マーカとして検出されるため，多少の汚れや書き込みがあっても想定通りのものが認識される．本システムでは，この性質を利用し検出されたマーカ領域から，あらかじめ登録されている書き込みなどの情報が無い素のマーカ画像の差分を取り，ユーザがマーカに付与した色や塗りつぶしを測定している．

マーカの書き込み情報を取得する流れを，照明を制御する命令カードを例に述べる．また，処理の例を図 5.9 に示す．

1. 照明制御カードのマーカ ID を 1 とする．
2. ユーザが照明マーカに対して，ピンク色に点灯させたいという色の塗りつぶしと，半分程度の明るさで点灯させたいという照度情報の塗りつぶしを行う．
3. ユーザがマーカを設置すると，ソフトウェアが ID1 のマーカが写っていると認識する．
4. この際，ID1 のマーカに対する書き込み検出処理を行う．ID1 に対する書き込み検出処理は，マーカ中心部から色を検出することと，マーカ下部から塗りつぶされた長さを検出することである．
5. ソフトウェアはまず ID1 のマーカに使われている基本色 (黒，紺，白) を検出された領域から削除する．
6. 残った領域において，中心部分にある塊が色を示す塗りつぶしで，下部分にある塊が明るさの強さを示す塗りつぶしである．
7. 色の検出については，中心部分にある塊に使われている色の中から使用頻度が高い 3 色を選び，更にその平均値を算出することにより，ユーザが意図した色を推測する．
8. 明るさの強さを示す塗りつぶしについては，検出されているマーカの大きさの横幅の長さと，下部分に残されている塗りつぶし領域の横幅の長さを比較し，割合を算出する．

以上が照明を制御するマーカに対する書き込み情報の取得処理である．このような塗りつぶし領域の長さやその座標の取得に関する処理は，検出されたマーカによって少しずつ異なる (例えば，Twitter 通知カードであれば，塗りつぶし領域が単に左寄り，中央，右寄りのいずれかであるかのみを判断する，など)．

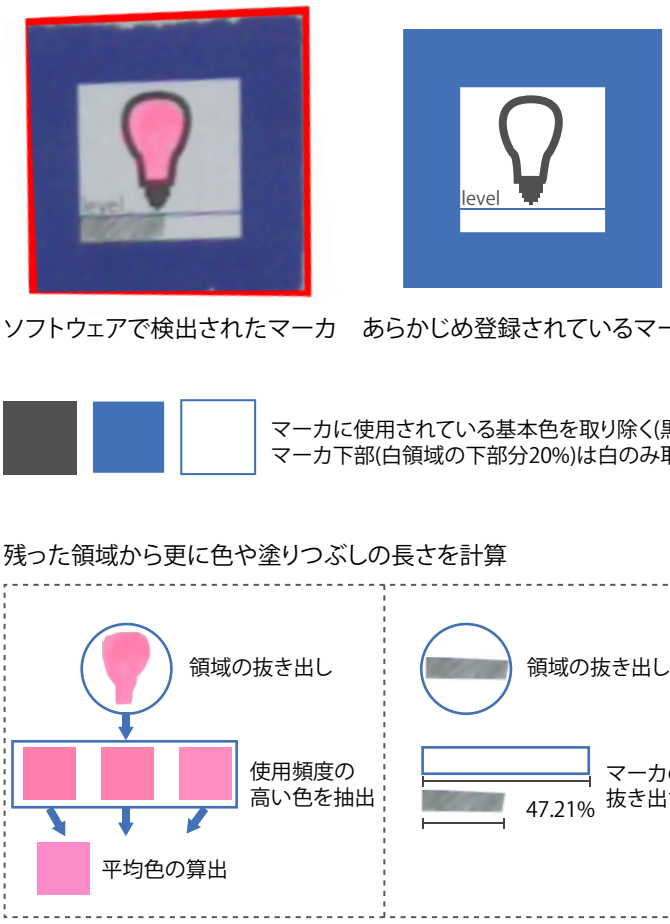


図 5.9: 検出されたマーカから書き込み情報を抜き出す処理の例

SNS 情報の取得

Trigger 命令の中に，Twitter および Facebook の通知を取得し，その動作をトリガーにアクション命令を発火させるものがある．この Twitter 及び Facebook 情報の取得は，それぞれのサービスで公式に提供されている API を使用している．執筆時現在の使用している Twitter の API バージョンは 1.1，Facebook の API バージョンは 2.8 である．Twitter API は REST な方法で取得できるため，必要な情報を URL パラメータとして渡し，ユーザの返信やだれいくとメッセージ，お気に入りの最新情報を監視する．また，Facebook API は garphAPI という方式で情報を取得することができる．こちらも基本的には URL パラメータで必要な情報が何かを指定し，新たな更新があることを監視する．また，どちらも json 形式にて情報が返却される．

この 2 種類の SNS 情報をトリガーとする命令を使用するには，事前にユーザが自分のアカウントで本システムが提供するアプリケーションの Oauth 認証をあらかじめ通しておく必要がある．

5.2.3　ハードウェア実装

概要

ユーザが Trigger や Action として指定する実世界情報の取得および制御のために，Arduino と気温センサ，学習リモコン等を使用した．照明の制御には API を通して制御可能な Philips Hue[1] を採用した．

Arduino による環境情報の取得と学習リモコン機能

Trigger 命令として提供している気温監視および Action 命令として提供してるエアコン制御，テレビ制御は命令カードの認識を行うソフトウェアが動作する PC に接続されている Arduino マイコン経由による家電やセンサの制御によって実現している．Arduino の制御はソフトウェアがシリアル通信を通して命令カードの認識と並行して実行する．Arduino には，家電のリモコンから対象の操作の赤外線点灯パターンを記録するための赤外線受信モジュール (GP1UXC41QS[2])，実際に家電の操作を行う赤外線 LED(OSI1CA5111A[3])，気温を取得する温度センサ (LM35DZ[4]) を搭載している．

ユーザはあらかじめエアコンやテレビのリモコンの ON/OFF ボタン及び温度変化ボタンを赤外線受光部にあてて学習させることにより，これらの命令を使用することができる．

Arduino と各種センサが接続されている様子を図 5.10 に，回路図を図 5.11 にそれぞれ示す．

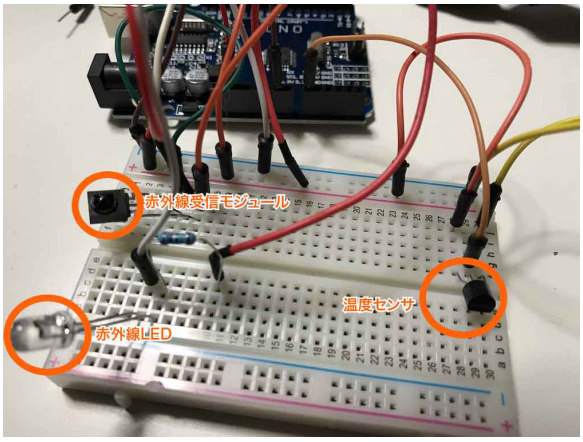


図 5.10: Arduino を用いた環境情報の取得と制御を行う回路

[1] Philips Hue http://www2.meethue.com/
[2] GP1UXC41QS http://akizukidenshi.com/catalog/g/gI-06487/
[3] OSI1CA5111A http://akizukidenshi.com/catalog/g/gI-04779/
[4] OSI1CA5111A http://akizukidenshi.com/catalog/g/gI-00116/

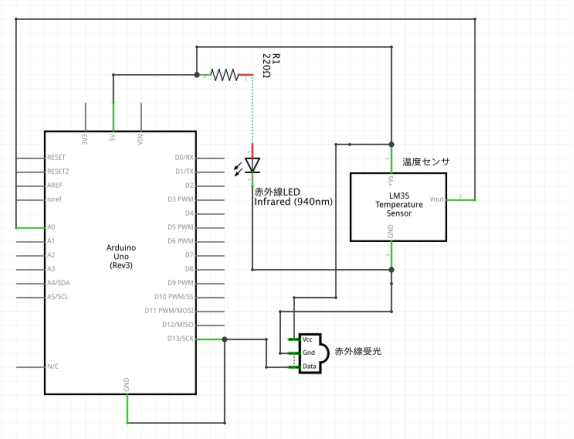


図 5.11: Arduino を用いた環境情報の取得と制御を行う回路図

Philips Hue を用いた照明制御

照明制御カードの実装には，Philips Hue を使用した．Philips Hue はユーザがスマートフォンなどから照明の色や明るさを制御できるスマートデバイスであり，開発用の SDK も提供されている．Philips Hue はブリッジと呼ばれる一つ一つの電球を制御するための母体と，ブリッジから送られてきた信号を元に振る舞いを変更する電球によって構成される．ブリッジは有線 LAN にてインターネットに接続され，同じネットワークに存在する無線 wifi からブリッジ自体に接続/通信を行うことができる．Philips Hue を制御するデベロッパーは SDK を通してこのブリッジに対して制御信号を送り，ブリッジがそれぞれの電球に対して信号を送ることによって電球を操作することができる．

Philips Hue は前述した通り，家庭内のネットワークに接続されることになるため，固有の ip アドレスが割り振られ，ユーザはこれにアクセスすることによって Hue とのやりとりが可能になる．通信は URL パラメータを用いた REST なものになっており，Philips Hue の状態の取得または制御は json 形式のデータを用いてやりとりされる．例えば，Philips Hue の ip アドレスが *192.168.11.1* であった場合，*http://192.168.11.1/api/username/lights/id/state* というアドレスに GET 形式で制御パラメータを付与することによって制御することができる．URL に含まれる *id* という部分はブリッジが管理している Hue の数に応じたユニークな ID(1 から始まる連番) に置き換えることによって，一意な電球に対するアクセスが可能となる．色や明るさを制御する具体的なパラメータは以下の通りである．

- ”パラメータ名” : 具体的な値 (説明)
- ”on” : true / false (照明の on/off の設定)
- ”bri” : 0∼255 (brightness 明度の設定)
- ”hue” : 0∼65535 (hue 色相の設定)
- ”sat” : 0∼255 (saturation 彩度の設定)

5.3 デモ展示による試用評価

前節までに述べた実装によって動作するシステムを用いて，学会や研究室のオープンハウスの際にデモ展示を行い，使用したユーザからフィードバックを得た．デモ展示は 2016 年 9 月 14 日から 16 日に渡り，ドイツで開催された Ubicomp という学会にて約 40 名に対して行なった．また，2016 年 10 月 20 日に著者が所属する研究室にて 5 名に対してもデモ展示を行なった．デモの際には，実際の家電を操作することが困難なため，センサやアクチュエータを搭載した模型や家電の制御が切り替わる動画をユーザのプログラムに応じて制御することによって実行結果を表現した．

デモ展示を行なっている様子を図 5.12 に示す．デモ展示を通してユーザから得られたフィードバックは以下の通りである．

- 紙のカードで操作するのが非常に面白い．ただ，なぜ紙を用いたのか？これによるメリットは何か．
- コンピュータを感じさせない入力インタフェースなため，子供や専門性が無い人でも扱いやすいと思う．
- 電球などを OFF にする機能はないのか．
- カメラを用いているので環境によって認識率が相当変化するのではないか．
- どのようなプログラムが設定されたかのフィードバックや，それを確認する手立てが欲しい．
- GUI を用いて実装しない理由は何か．

紙というインタフェースを肯定する意見の一方，ユーザが作ったプログラムへのフィードバックや家電等を OFF にする命令が無いなど機能不足を指摘される面もあった．デモ展示では，時間や温度といった変化に長時間を要するものを短縮していたため (指定時間や温度の数秒，数度前後からカウントダウンする演出を設けた)，ユーザが自分で作ったプログラムの結果がすぐに反映されていたが，実際にタイマー等をセットする際には数時間後にならないと結果がわからない．それまで自分が作ったプログラムが正しく認識されて登録されたのかがユーザに伝わらないのは，使用するユーザを不安にさせる要素になるため，この点については改良が必要であることが明らかになった．また，電球やエアコンを OFF にしたい場面も場合によっては確かに存在するため，この機能を搭載することも必要である．

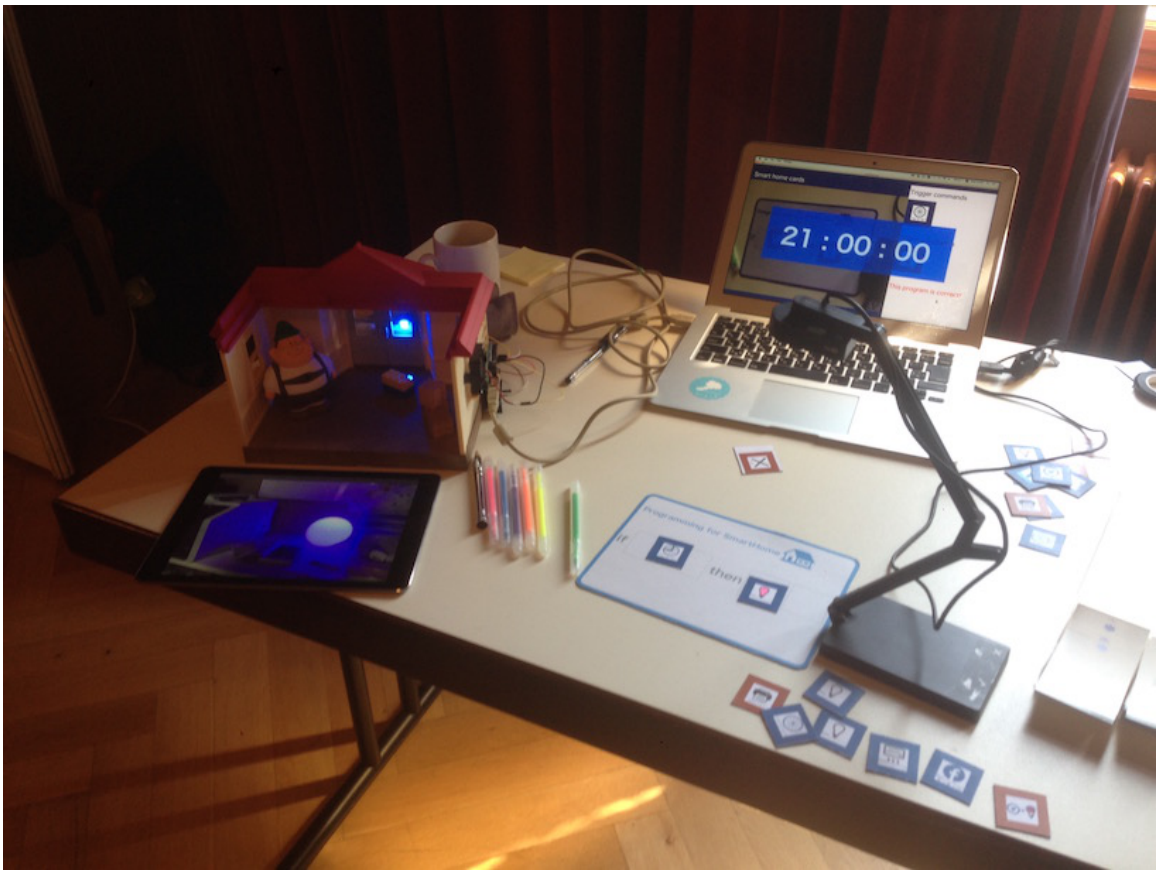


図 5.12: 紙媒体を用いた実世界制御プログラム環境のデモ展示の様子

5.4 試用をうけての改良

5.4.1 音声フィードバック

前節で述べた試用ユーザから得られたフィードバックをもとに，システムに対していくつかの改良を行なった．

ひとつは音声によるフィードバックの搭載である．ユーザが Trigger と Action の命令カードをそれぞれセットし，プログラムとして正しいことが判断されてから 5 秒間認識されている命令に変化がなければプログラムが確定されたとし，日本語の音声によって，ユーザがセットしたプログラムの意味を読み上げる．例えば，Trigger 命令として「21 時になったら」という時間監視のカードをセットし，Action 命令として「照明をピンクにする」という照明制御のカードをセットした場合，*21* 時になったら，照明をピンクに点灯させますという文章を読み上げる．

これはシステムを動作させている Mac OS に搭載されている say コマンドを利用して実現している．say コマンドは *say (message)* といった形式で実行すると，message の部分の読み上げを行う．*-v name* というオプションを付与することによって，読み上げ音声の声優を変更す

ることができる．今回は *kyoko* という日本語対応の女性音声を使用した．そのため，例えば *say* 音声読み上げテスト *-v kyoko* のように実行すると，女性の声で音声読み上げテストという音声が流れる．

各コマンドに対して，音声テンプレート文章を用意しておき，ユーザの書き込み動作によって変化するパラメータを後から文章に付け加えることによってユーザが意図する命令の読み上げ音声を生成している．例えば，時間指定命令であれば，時になったら，というテンプレート文章が用意されており，ユーザが書き込んだ時間に応じて文章の先頭に数字を付け加えて，何時になったらという文章を生成している．

say コマンドの実行には shell スクリプトを使用しており，メインのソフトウェアから読み上げる文章を引数に shell スクリプトを実行して音声を再生している (図 5.13).

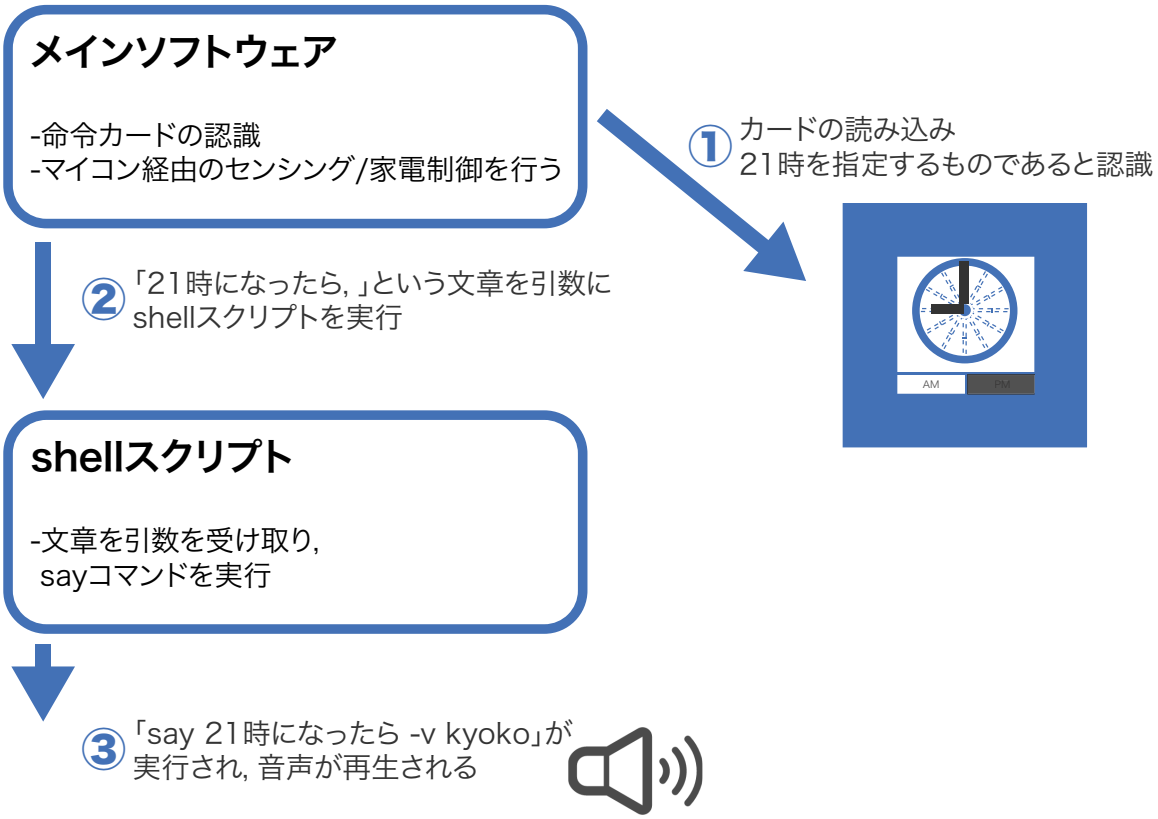


図 5.13: 音声フィードバックの実装について

5.4.2　OFF 制御コマンドの追加

試用ユーザから得られたフィードバックとして，家電等の電源 OFF 機能が欲しいとの意見があった．これを受け，各種家電に対する OFF にする Action カードを追加した．これらの命令がセットされた場合，Trigger にセットされている条件が満たされた際に該当の家電の電源が OFF になる．新たに追加した命令カードを図 5.14 に示す．

図 5.14: ユーザのフィードバックを受けて新たに追加した OFF 機能命令カード

これらの機能は，それぞれ該当する家電のリモコンの OFF 信号をあらかじめ学習させておくことによって実現した．Philips Hue に関しては以前に述べたような API を用いて，*”on”*パラメータを*”false”*にすることによって消灯の動きを実現した．

5.5　利用シナリオ

前節までで述べた本システムを用いた利用シナリオについて述べる．ユーザおよびシステムは以下の手順を経て環境のセットアップおよびプログラミング，実世界デバイスの制御を行う．

1. SNS 情報の登録 (API 認証)，Philips Hue の準備 (ネットワークへの接続) および，各種家電のリモコンを用いた信号等の事前情報を登録する．
2. 命令カードを貼り付けるボードが画角に映るような位置に web カメラを設置し，ソフトウェアを起動する．
3. 用意されている命令カードに対して書き込み等を経てプログラミングを行う (紙を貼り付ける)．
4. この際，ソフトウェアが web カメラが紙を認識し，書き込み情報の解析を行う．
5. Trigger,Action それぞれの命令が正しく設定されていた場合，ソフトウェアがプログラム内容の解釈を行う．

6. ソフトウェアが設定されたプログラムの内容を読み上げる音声フィードバックを行う．

7. ソフトウェアが指定された Trigger イベントをセンサや API にて監視する．

8. Trigger に設定された条件が満たされた場合，Action に指定された内容を Philips Hue や Arduino に搭載されたセンサを用いて実行する．

本環境は，事前設定である手順 1,2 を一度行ってしまえば，基本的にはコンピュータに触れることなく実世界デバイスの制御を行うプログラミングを任意に行うことができる．そのためユーザはセットアップ後はコンピュータの存在を意識することなく，紙媒体の操作のみでプログラミングを行うことが可能となる．

想定される利用シーンとしては，以下のようなものが考えられる．

- 勉強机の前にあるホワイトボード等に命令カードを貼り付けるボードを設置し，就寝前などに朝テレビや照明が ON になるようにプログラムを行う．
- 玄関の扉に命令カードを貼り付けるボードを設置し，外出前にプログラミングを行い，帰宅時に照明やエアコンが付いているようにする．
- 長期間家を空ける際に，観葉植物やペットのために部屋の温度管理を行うために，エアコンの制御を部屋の室温に合わせて調整するプログラムを作る．

ユーザが実際に本システムを利用する際に作られる環境の外観を図 5.15 に示す．この例では玄関に本システムを設置し，外出時/帰宅時等に紙媒体を用いた実世界制御プログラミングを行うことが想定される．

図 5.15: 想定される利用シーンの環境外観図

第6章　実世界制御が可能な紙媒体を用いたプログラミング環境を用いた被験者実験と結果

本章では，前章にて述べた紙媒体を用いた実世界制御プログラミング環境を使用した被験者実験について述べる．被験者実験の目的は，提案手法のユーザビリティ調査と，実世界制御プログラミングに紙媒体を入力インタフェースとして扱うことの優位性の確認，およびユーザの動向を観察することによる紙媒体プログラミング環境ならではの特徴を見出すことである．

6.1　実験概要

提案手法のユーザビリティ及び紙インタフェースの優位性を評価するために，被験者実験を行った．今回の実験では，提案手法のユーザビリティを調査するために，System Userbility Scake(SUS)[41] を使用した．SUS は 10 個の質問に対して 5 段階の評価を用いて被験者に回答してもらうことにより，対象となるシステムの使いやすさを定量的に評価することができるアンケートである．ユーザには紙媒体を用いた環境と実験用に用意した GUI で操作可能な環境をユーザに使用してもらい，それぞれの使用タスク後この SUS を用いたアンケートに答えてもらうことによって，紙と GUI のユーザビリティ比較を行った．

6.1.1　実験環境と被験者の内訳

被験者は 22 際から 24 歳までの合計 7 名 (女性 1 名) であり，全員が大学生もしくは大学院生である．プログラミングを日頃から使用する情報系の学生が 5 名，授業等で多少の経験がある学生が 2 名であった．被験者は本環境を用いたプログラミング手法や特徴の説明を受けた後，いくつかの問題が提示されるのでそれに回答する形でプログラミングを行う．実験は周りに人がいない落ち着いた環境で行われ，実世界制御の様子は前章で述べたデモ展示と同じ手法で模型や動画によって与えらえた．

6.1.2　実験内容

実験の初めには著者から環境についての説明を行った．その後ユーザに対して問題を提示し，それを実現するプログラムを作成してもらう．ユーザが GUI，紙どちらの環境を先に扱

うかはランダムに決められた．ユーザは指定したタスクをこなした後，自由に環境を操作する時間が与えられ，その後 SUS 調査および各種アンケートに回答する．ユーザに与えられるタスクの例は以下の通りである．

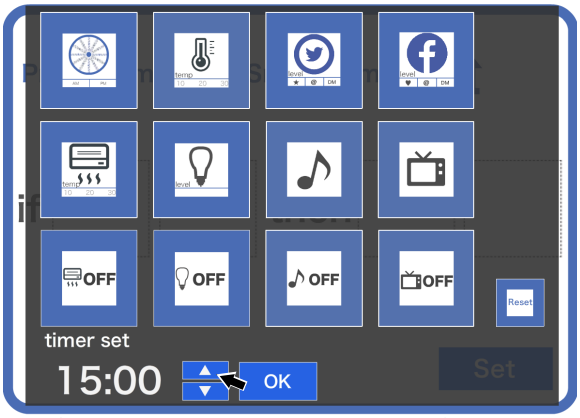
- 例題１：室温が 30℃になったら，エアコンを 20℃で ON にするプログラム
- 例題２：Twitter の Mention がきたら，テレビを ON，照明をピンクに点灯させるプログラム
- 例題３：21 時になったら，照明を OFF，エアコンを OFF にするプログラム

実験の所要時間は約 30 分であった．

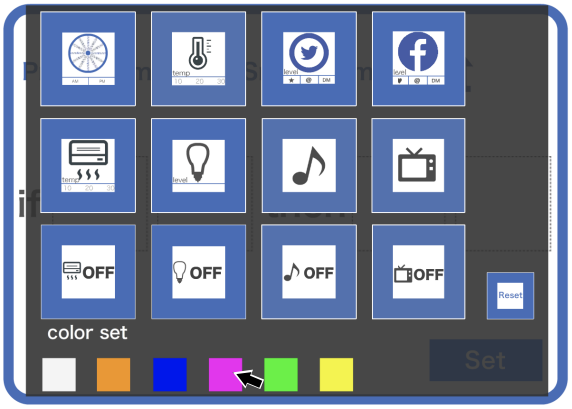
GUI アプリケーションについて

本実験を行うため，紙媒体の環境との使用感を比較するために GUI 操作によって同等のプログラミングが可能な環境を用意した．環境が動作している様子を図 6.1 に示す．

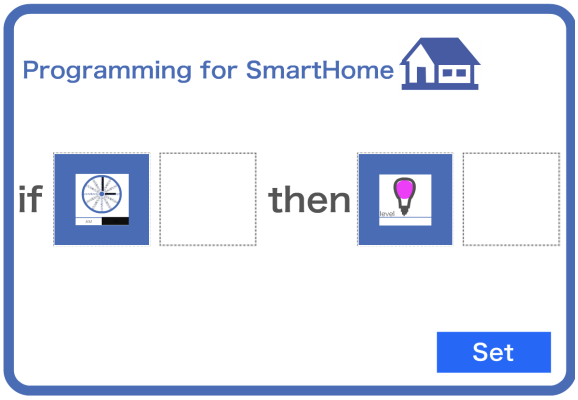
GUI アプリケーションでは，起動時に命令カードをセットするボードの画像が表示されており，Trigger/Action 命令をセットする点枠線をクリックすると，セットする命令カードを洗濯する画面に切り替わる．この時表示される命令は 5.1.2 項で述べた全てと，5.4.2 項で述べた追加された命令を合わせた 12 種類である．ユーザはこの中から必要な命令をクリックして選択を行う．その際に，時間や照明などパラメータの設定が必要なものの場合，時間指定や照明の色指定など対象に合わせて更なる選択画面が表示される．この動作を Trigger，Action 命令それぞれの選択において行った後，*set* ボタンをクリックすることによって命令が確定される．この時文法的に正しければ，その通りに実行され，間違っていればシンタックスエラーですという警告文が表示される．



時間指定命令を選択し, タイマーをセットしている様子



照明制御命令を選択し, 色を選択している様子



Trigger,Actionそれぞれの命令がセットされている様子

図 6.1: 被験者実験に用いる GUI アプリケーションの概要

6.2 被験者に対する調査

被験者に対して実験後に行った SUS およびアンケート調査の内容は以下の通りである．

- SUS10 項目 (回答方法：5 段階評価)
  - Q1. このシステムをしばしば利用したいと思う
  - Q2. このシステムを利用するには説明が必要となるほど複雑であると感じた
  - Q3. このシステムは容易に使いこなす事ができると思った
  - Q4. このシステムを利用するのに専門家のサポートが必要だと感じる
  - Q5. このシステムにあるコンテンツやナビゲーションは十分に統一感があると感じた
  - Q6. このシステムでは一貫性のないところが多々あったと感じた
  - Q7. たいていの人は、このシステムの利用方法をすぐに理解すると思う
  - Q8. このシステムはとても操作しづらいと感じた
  - Q9. このシステムを利用できる自信がある
  - Q10. このシステムを利用し始める前に知っておくべきことが多くあると思う
- アンケート調査 (回答方法：記述式)
  1. 年齢，性別，利き手についての質問
  2. プログラミングの経験度についての質問
  3. 本環境を用いたプログラミング (紙に書き込みを行って配置するまで) はやさしかったか (1 から 5 までの選択評価．1 に近いほど好感)
  4. 本環境の操作感はどうであったか (1 から 5 までの選択評価．1 に近いほど好感)
  5. 本環境を使用していて良かった点についての質問
  6. 本環境を使用していて悪かった点についての質問
  7. 本環境を用いてみたい利用シナリオについての質問
  8. 本環境に追加したい他の実世界制御命令があるかについての質問
  9. 紙媒体ならではの操作について何かアイデアがあるかについての質問

6.3 結果

6.3.1 SUS 調査の結果

被験者実験にて行われた SUS 調査の結果を操作手法ごとに表 6.1，表 6.2 それぞれに示す．また平均点を図 6.2 に示す．

表 6.1: SUS 調査の結果 (紙インタフェース)

	被験者1	被験者2	被験者3	被験者4	被験者5	被験者6	被験者7
Q1	4	4	4	4	3	4	4
Q2	1	2	0	1	1	0	1
Q3	4	4	4	4	4	4	4
Q4	0	1	1	2	2	0	0
Q5	3	3	4	4	4	4	4
Q6	1	0	0	1	2	0	0
Q7	4	4	4	3	4	3	3
Q8	0	2	0	1	1	0	0
Q9	4	4	4	4	4	3	4
Q10	2	1	0	1	2	1	1
SUS合計点	83.16	82.11	97.89	82.11	74.74	90.53	90.53

表 6.2: SUS 調査の結果 (GUI)

	被験者1	被験者2	被験者3	被験者4	被験者5	被験者6	被験者7
Q1	3	4	4	2	3	2	3
Q2	2	1	1	2	1	0	0
Q3	3	4	3	3	3	4	4
Q4	1	1	1	2	2	0	0
Q5	3	3	3	3	4	4	4
Q6	2	0	2	3	3	0	2
Q7	4	3	4	3	4	4	4
Q8	1	1	1	2	2	0	0
Q9	4	4	4	4	4	4	3
Q10	3	1	1	2	2	0	1
SUS合計点	65.26	84.21	80.00	60.00	68.42	95.79	86.32

SUS は 10 項目の質問に対して 0 から 4 までのいずれかの数字を選択するもので，数字が大きいほど質問に対しての同意が大きくなるというものである．10 個の質問は，ポジティブな質問とネガティブな質問が交互に提示されている．ポジティブな質問に対しては付けられた値から 1 を引いたものを点数とし，ネガティブな質問に対しては 5 から付けられた値を引いたものを点数とする．これらを全て足し合わせて，最高点の合計が 100 点になるように正規化したものが，ユーザのシステムに対する評価点数となる．

今回行った実験では，1 人の被験者を除いて紙媒体をインタフェースとする手法の方が評価点が高い結果となった．また，図 6.2 に示した通り，紙媒体をインタフェースとする手法の方

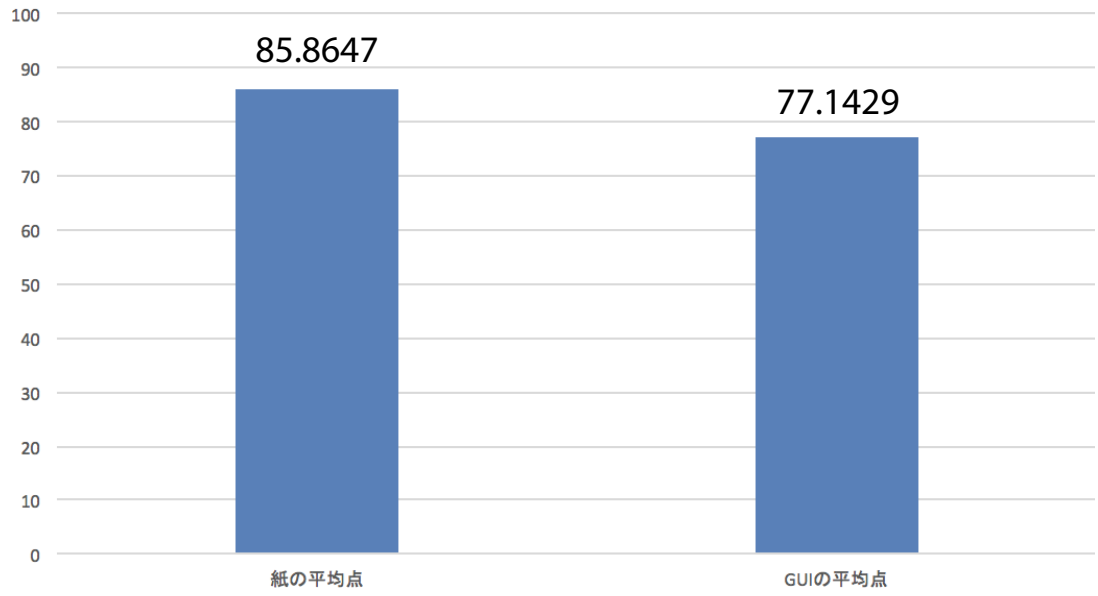


図 6.2: SUS 調査の平均点

が，GUI を用いた手法よりも高い点数を得た．

6.3.2 アンケート調査の結果

アンケート調査にて集計した本環境に対する印象について,「本環境を用いたプログラミング (紙に書き込みを行って配置をするまで) はやさしかったか」という質問に対しては 7 人中 6 人が「1. 非常にやさしい」と回答し，残りの 1 人も「2. やさしい」という回答をした.「本環境の操作感はどうであったか」という質問に対しては，7 人中 5 人が「1. 非常に使いやすい」と回答し，残りの 2 人も「2. 使いやすい」という回答をした．

上記の通り，本環境は GUI を用いた環境に比べてユーザビリティ面において高い評価を得ており，扱いの難易度や操作性についても好感的な意見がユーザから得られた．

また，記述式の各質問に対するユーザのコメントは以下の通りである．

- 本環境を使用していて良かった点についての質問
  - 書き込みで時間や色を指定する点が斬新だと思った
  - 非常に容易に家電を操作できて良い
  - デザインが可愛いと思った

  - 書き込み動作が楽しかった
  - if-then の形式がすぐ理解できた
  - 紙だと手軽で身近に感じた
- 本環境を使用していて悪かった点についての質問
  - 出来ることが少ないと思った
  - 色ぬりを間違えるとカードが無駄になる点
  - 認識の精度が悪い時があった
  - 自分でコマンド追加などの機能がなく拡張性がない点
- 本環境を用いてみたい利用シナリオについての質問
  - 予定のリマインドなどをしてほしい
  - 家から出かけるときや，帰ってきたときなど
  - 長期間家を留守にするとき
  - センシングを増やせば利用シナリオが広がりそうだが，今のままでは外出前くらいしか思い浮かばない
- 本環境に追加したい他の実世界制御命令があるかについての質問
  - コーヒーを入れる
  - 換気扇などの消し忘れ防止
  - 加速度や圧力などのセンシングの充実による，ドアが開いたら... しまったら... の実現
  - カレンダーと同期したアラーム機能（音だけではなく家電の動作で伝えてくれると現状のサービスと差別化できそう)
  - 汎用的な I/O モジュールの追加
  - 何かの在庫が少なくなったらの検知 (圧力センサを用いる)
- 紙媒体ならではの操作について何かアイデアがあるかについての質問
  - 角を曲げて NOT にする
  - シールを貼る
  - 切り取ることによって命令が変化する
  - 曲げ具合によって命令が変化する

6.4 被験者の動きの観察

実験中に観察した被験者の動作について述べる．今回の実験に参加した被験者 7 名を被験者 1–7 として述べる．

被験者 2 は紙媒体の環境に触れていた際，照明制御の命令カードの明るさ指定部分の塗りつぶし具合を誤り，その書き込み動作がペンによって行われていたものであったため，修正ペンを用いて塗りつぶしをやり直していた．

被験者 3 は一つ目に与えられたタスクで作成した命令カードの配置をそのままの状態でカメラの画角外にスライドし，そのカードを見ながら二つ目のタスクを完了させるプログラムの作成を行っていた．

被験者 6 が紙媒体の環境に触れていた際，エアコン制御の命令カードの温度指定部分の塗りつぶしをシャーペンで行い，塗りつぶしの濃度が薄かったため，想定した 20 ℃ではなく，塗りつぶしが行われていない状態の命令カードである 10 ℃のものとしてソフトウェアが誤認識してしまった．

被験者 7 は時間指定の命令カードの時針，分針の書き込みを正しく書き込むことができたが，カード下部にある AM/PM の指定塗りつぶしを見落としていたため，正しいプログラムとして認識されずエラーが表示された．

各被験者が紙媒体の環境を用いて作成したプログラムはタスクおよび自由に触れる時間を含めて，それぞれ 3,4 回であり，実験全体を通して合計で 25 個のプログラムが作成された．この中でソフトウェアによる認識エラーは 1 回，被験者の書き込み不足による誤りは 2 回，被験者の文法エラーによる誤りは 1 回であった．

また，以下のような参考になる意見が被験者の実験中の発話から得られた．

- 命令カードを多めに用意しておけば，日常的に使用するプログラムはそのままどこかに貼り付けて保存しておける
- 書き込みを間違えたが，修正ペンの痕が残ることによってどこを間違えたのが視覚的にわかる
- プログラムコードの履歴をそのまま残しておくことができるのでは
- プログラムコードの UNDO/REDO を物理的に行うことができる

6.5 考察

前節にて述べた実験結果から，紙媒体を用いた実世界制御プログラミング環境が使用したユーザから高評価が得られるものであるということが分かった．また，GUI 環境との比較によって，今回用意した環境デザインにおいては紙媒体の方がユーザビリティが高いということが分かった．

被験者の記述式のアンケートには紙媒体の環境における「書き込み動作」に対する好意的な意見が多く観測され，紙という媒体をプログラミングに用いることの有用性を垣間見るこ

とができた．また一方で，ソフトウェアによる命令カードの誤認識や，命令の少なさを指摘される面もあった．

また，その他のアンケート回答や被験者の動きの観察から，紙媒体ならではのユーザの動きを観測することができた．例えば，前節で述べたような「一度作ったプログラムをそのまま別の場所に移して保存する」といった点は特に紙の特徴が現れているように感じる．一般的なプログラミングでも，以前試した実験的なコードを別名のファイルとして保存しておくことができるが，その数が増えれば増えるほど，全てを見通しながら新しいコードを書くことは難しい．しかしプログラムコード自体がタンジブルな形状である場合，コードに相当する実物体をそのまま物理的に保存しておくことができる．先行研究ではよくブロックを使ったプログラミング環境が提案されているが，これと比較すると紙媒体は保存性という点において強い優位性を持っていると感じる．ブロックではその形状から物理的な保存に空間の一部分を占領されることがあっても，紙の場合，図**??**に示したようにノートなどにページ別でレイヤー形状で保存することができる．

また，書き込み間違えの状況がそのまま保存されるというのも紙ならではの特徴であると言える．被験者が扱ったあとの命令カードを眺めてみると，色付けや塗りつぶしの方法に非常に強い個性が現れていた．中には前節で述べたように塗りつぶしが甘く，誤認識が起きていた例もある．プログラムコードのミスとして，こうした自分ならではの筆跡のようなものが残ることは，ユーザが後にその紙を見た際に自分がどのようなミスをしたのかを思い出す強力な手立てとなる．プログラミングや一般的な学習において，一度誤った問題を忘れないということは非常に重要である．こうした書き間違えに対する修正を億劫であると感じる意見も得られたので，命令カードの設計に改良の余地が見られるが，一般的なプログラミングであれば「戻るコマンド」を押すだけで直せてしまう問題を，あえて「修正ペンなどを利用して元に戻し，再び書き込みを行う」といった身体的な動作によって行うことにより，その行為自体が記憶に残りやすくなるのではないかとも考える．プログラミングの経験が豊富なユーザには単に億劫な作業としかなりえないが，まったくの未経験者に対しての学習環境としては，こうしたアナログな煩わしさもプログラミングに対する印象を変化させるきっかけとなり得るのではないか．

第7章　考察と議論

7.1　GUIとの比較

エンドユーザ向けのプログラミング環境について，GUIよりもTUIを使用した方が良いという研究報告 [23] が既にあげられているが，今回2つの被験者実験を通してこのことを再認識することができた．GUIとの比較実験は本研究においても前章で取り扱い，GUIよりも提案手法の方がユーザビリティが高いという結果が得られた．この結果には様々な要因が考えられる．まずGUI操作によるプログラミングは動作が非常に単調になることがあげられる．先行研究として提案されている様々なGUIを用いたプログラミング手法や既存のアプリケーションでは，マウスのクリックおよびドロップ&ドラッグの操作がほとんどであり，プログラミングの始めから終わりまでユーザの操作に大きな変化が無い．対してタンジブルな環境では，ブロックや紙といった実物体に対し，並べる，組み合わせる，重ねる，分解する，書き込みを行うといった様々な動作が行われる．この動作の絶対数の差は使用するユーザのストレスに影響する要因として十分に考えられる．

また，GUIを用いた環境の場合，その操作性上コンピュータに触れることは必須であるが，タンジブルな環境においてはこの限りでは無い．プログラミングに慣れ親しんだユーザではなく，子供や専門知識のないエンドユーザを対象とした環境の場合，複雑な操作や選択肢を極力見せないようにするためには，コンピュータの存在を感じさせない環境づくりが重要である．本手法を含むタンジブルなプログラミング環境では，始めの環境セットアップを除くとユーザがコンピュータの操作を行うことは殆どない．こうしたプログラミング及びコンピュータに対する複雑さを感じさせない環境が，エンドユーザにとって親しみやすい環境となっているのではないか．本手法が提供するような紙という身近に存在する物体を用いることが，より親近感を持たせる環境となっていることが，前章までに述べた被験者実験のアンケートなどからも読み取れる．

7.2　プログラミングのシンタックスについて

3章で述べたアプローチ1では，一般的なプログラミングに搭載されている逐次実行，条件分岐，繰り返し，変数という概念をそのまま取り入れたシンタックスを提供した．また5章で述べたアプローチ2では，実世界制御プログラミングにおいて有効とされるif-then形式のシンプルなシンタックスを提供した．これらの2つの環境はそれぞれ目的が違うため単純にどちらの方が良いと言い切ることはできないが，それぞれ不足している点や改善点が見受け

られる．

5 章で述べた環境については，後の被験者実験によって「文法や出来ることが少ない」という意見が伺えた．紙という物理的な媒体を使用しているため，文法が複雑になり使用するカードが増加すると煩雑さが増すことになる．このようにタンジブルなプログラミング環境において，シンタックスの充実と，エンドユーザが扱いやすいシンプルな環境はトレードオフの関係にある．

3 章で述べた環境については，作成したプログラムによって実現できるものの規模に関する意見は得られたが，シンタックスに関する指摘は少なかった．逆に長いプログラムを作りすぎて，web カメラの画角に入らないケースが見受けられた．

エンドユーザにとっては，プログラミングそのものの学習よりもプログラミングの利便性を活かして生活を豊かにすることがシステムを利用する目的になり得るため，一般的なプログラミングのシンタックスをそのまま持ち込んできた環境では扱いが難しいのではないかと考える．しかしその一方，5 章で述べたような環境では最大で 4 つの命令を配置することしかできないため，最初は簡単で楽しく思えても，次第にその行為に飽きてしまうのではないかという懸念がある．

そのため，命令カードを貼り付けていたボードの種類を充実させ，初めは 4 つしか置けない if-then 形式のシンプルなもの，慣れてきてもう少し複雑な制御がしたくなった場合は条件分岐が追加されたボード，次はループの概念が追加されたボード，というようにユーザが目的に合わせてステップアップ形式に使用させるシンタックスを変化させる環境を提供すると，上記にあげたような問題を解決できる可能性がある．この場合，ユーザは自分が詳細に家電等を制御させたいという欲求を単純に満たす過程で，プログラミングのシンタックスを自然と身につけるといった効用を得ることも可能である．

7.3 紙を使ったならではの良さについて

本研究ではプログラミング環境に紙媒体のインタフェースを取り入れることの効果について，環境の実装と実験を通して検討を行ってきた．被験者実験の結果から，紙に対する書き込み等のインタラクションを提供することでユーザに良い影響があるということが分かった．被験者のアンケートや実験中の様子からは，紙を使うことによってプログラムコードの保存や見直し，やり直し等の記録が物理的に残るため，次なるプログラムの作成や記憶想起に役立つということが明らかとなった．

また，タンジブルな媒体としてブロックなどではなく，紙を用いることによってプログラミングを行う環境整備のコストが大幅に削減される．アプローチ 1 に示した手法を使い子供等の学習環境として利用する場合，ソフトウェアと web カメラ，印刷紙があればすぐにタンジブルなプログラミングを行うことができる．

また，今回の手法では取り入れていなかった紙のメンタルモデルとして,「折る」「切る」「重ねる」などがあげられる．これらを今後プログラミングの一要素として取り入れることにより，エンドユーザにとってより親しみのある環境になる可能性がある．

紙は一般的に身近にある親しみある媒体である．これをプログラミングの一部として取り入れることで，書き込みや貼り付けといった動作が効果的であるということが実験によって明らかとなった．更に紙は物理的なものであるため，プログラムコードの保存や見直しが形として残るという特徴を持つ．

こうした強みのある媒体を使ったプログラミング環境に対し本章で述べたような改善案を加えることにより，子供やプログラミングの経験がないユーザが,「紙遊び」をしているといつの間にかプログラミングの概念が身についているという環境を実現することも可能ではないかと考える．

第8章　終わりに

本研究では紙媒体を用いたプログラミング環境を提案し，その作成と使用を行うためのシステムを 2 つのアプローチとして実装した．ユーザは紙媒体の命令カードに対して書き込みや色付けといった動作を行いつつ，提供されているシンタックスに沿って紙を配置することによりプログラミングを行う．その実行結果として，グラフィックの変化や実世界デバイスの制御が可能となる．2 つの被験者実験を通して，紙媒体をプログラミング環境に持ち込む効果や特徴，GUI と比較した場合の優位性を明らかにした．今後の課題として，プログラミングによって実現する結果の規模やシンタックスの複雑さについて，使用するユーザを見極めた上での再設計があげられる．また，今回取り入れなかった紙が持つメンタルモデル「折る」「切る」「重ねる」といった要素をプログラミングの一環として取り入れることにより，よりアナログな操作が豊富な環境の構築が考えられる．

謝辞

本研究を進めるにあたり，指導教官である高橋伸先生には多くのご指導及びご助言を頂きました．また，システム情報工学研究科，インタラクティブプログラミング研究室の志築文太郎先生，三末和男先生には日頃研究に関する大変たくさんのご指導を頂きました．また元システム情報工学研究科教員の田中二郎先生には，修士 1 年間の短い間ではございましたが，日々の研究活動や論文の執筆，学会参加へのサポート，研究生活における心構えまで，至る所で親身にご指導頂きました．ここまで指導いただきました多くの先生に心より感謝申し上げます．

インタラクティブプログラミング研究室の皆様には研究活動において様々なアドバイスを頂きました．一部の方には評価実験の被験者としてご協力いただきました．誠にありがとうございます．

最後に，自分の生活を支えてくださった両親，日常生活でお世話になった友人，本研究をご支援くださった皆様に心から感謝申し上げます．

参考文献

[1] 宮下芳明．プログラミングに対するモチベーションを向上させる新言語 HMMMML の開発．第 51 回プログラミングシンポジウム論文集，pp.57–64，2010.

[2] 中橋雅弘，宮下芳明．HMMMML2：超好意的に解釈するコンパイラ．情報処理学会夏のプログラミング・シンポジウム報告集，pp.107–110，2011．

[3] 中橋雅弘，宮下芳明．HMMMML3：他人を意識したモチベーション向上を考えたプログラミング環境．インタラクション 2011 論文集，pp.511–514，2011.

[4] 長慎也，甲斐宗徳，川合晶，日野孝昭，前島真一，筧捷彦．Nigari-Java 言語へも移行しやすい初学者向けプログラミング言語．コンピュータと教育研究会報告，pp.13–20，2003．

[5] Alex McLean，Dave Griffiths，Nick Collins，and Geraint Wiggins． Visualisation of live code In Proceeding of the 2010 international conference on Electronic Visualisation and the Arts，EVA'10，pp.26–30，2010．

[6] 加藤邦拓，宮下芳明．時間とのインタラクションによるプログラミング支援．情報処理学会研究報告 HCI，HCI-149，No.2，pp.1–6，2012．

[7] 佐藤竜也，志築文太郎，田中二郎．実行の可視化システムと連動した統合開発環境による GUI ベースプログラムの理解支援．第 15 回インタラクティブシステムとソフトウェアに関するワークショップ (WISS2007) 論文集，pp.25–30，2007．

[8] 伊藤恵，椿本弥生．プログラミング教育における吹き出し導入の試みと分析．教育システム情報学会研究報告，No.30，pp.13–20，2016．

[9] Mitchel Resnick，John Maloney，Andrs Monroy-Hernndez Natalie Rusk，Evelyn Eastmond，Karen Brennan，Amon Millner，Eric Rosenbaum，Jay Silver，Brian Silverma，and Yasmin Kafai． Scratch programming for all. In Communications of the ACM, 52, 11, pp.60–67, 2009.

[10] Yasunori Harada and Richard Potter． Fuzzy rewriting:soft program semantics for children. In IEEE Symposium on Human Centric Computing Languages and Environments，Vol.1，No.1，pp.39–46，2003．

[11] 兼宗進，阿部和弘，原田康徳．プログラミングが好きになる言語環境．情報処理，Vol.50，No.10，pp.986–995，2009．

[12] 森秀樹，杉澤学，張海，前迫孝憲． Scratch を用いた小学校プログラミング授業の実践: 小学生を対象としたプログラミング教育の再考．日本教育工学会論文誌，Vol.34，No.4，pp.387–394，2011．

[13] Hiroshi Ishii． Tangible bits: towards seamless interfaces between people, bits and atoms. In Proceedings of the ACM SIGCHI Conference on Human factors in computing systems，CHI'97，pp.234-241，1997．

[14] Danli Wang，Yang Zhang，Tianyuan Gu，Liang He，and Hongan Wang．E-Block: a tangible programming tool for children． In Proceedings of the 25th annual ACM symposium on User interface software and technology，UIST'12，pp.71–72，2012．

[15] Danli Wang，Cheng Zhang，and Hongan Wang． T-Maze: a tangible programming tool for children． In Proceedings of the 10th International Conference on Interaction Design and Children，IDC'11，pp.127–135，2011．

[16] Danli Wang，Tingting Wang，and Zhen Liu． A Tangible Programming Tool for Children to Cultivate Computational Thinking． The Scientific World Journal，2014．

[17] 八城朋仁，迎山和司．物質プログラミング -物質によるプログラムの可視化と開発環境の制作-．インタラクション 2014 論文集，pp.647–650，2014．

[18] Daniel Gallardo，Carles F. Julia，and Sergi Jorda．TurTan: a Tangible programming language for creative exploration． In Proceedings of the 3rd annual IEEE international workshop on horizontal human computer systems，TABLETOP'08，pp.412–420，2008．

[19] Smith Andrew Cyrus． Cluster-based tangible programming． In Proceedings of the Fourth International Conference on Digital Information and Communication Technology and its Applications，DICTAP'14，pp.405–410，2014．

[20] Andrew Cyrus Smith，Heinrich Springhorn，Steven Bruce Mulligan，Ireyan Weber，and Jackie Norris． tactusLogic: Programming using physical objects． IST-Africa Conference Proceedings，pp.1–9，2011．

[21] Michael S. Horn and Robert J. K. Jacob． Designing tangible programming languages for classroom use． Proceedings of the 1st international conference on Tangible and embedded interaction，TEI'07，pp.159–162，2007．

[22] Michael S. Horn, Crouser Jordan，and Bers Marina．Tangible Interaction and Learning: The Case for a Hybrid Approach． Journal Personal and Ubiquitous Computing，Vol.16，No.4，pp.379–389，2012．

[23] Michael S. Horn, Erin Treacy Solovey, Crouser Jordan, and Robert J. K. Jacob. Comparing the use of tangible and graphical programming languages for informal science education. In Proceedings of the 27th international conference on Human factors in computing system, CHI'09, pp.975–984, 2009.

[24] Sarah Mennicken and Elaine Huang. Hacking the naturalhabitat: An in-the-wild study of smart homes, their development, and the people who live in them. In Proceedings of the 10th international conference on Pervasive Computing, Pervasive'12, pp.143–160, 2012.

[25] Toshiyuki Masui. Real-World Programming. In Proceedings of Designing Augmented Reality Environment, DARE'2000. pp.115–2000. 2000.

[26] Jan.Humble, Andy Crabtree, Terry Hemmings, KarlPetter kesson, Boriana Koleva, Tom Rodden, and Pr Hansson. "Playing with the Bits" UserConfiguration of Ubiquitous Domestic Environments. In Proceedings of the 5th international conference on Ubiquitous computing, UbiComp'03, pp.256–263, 2003.

[27] Fahim Kawsar, Tatsuo Nakajima, and Kaori Fujinami. Deploy spontaneously: supporting end-users in building and enhancing a smart home. In Proceedings of the 10th international conference on Ubiquitous computing, UbiComp'08, pp.282–291, 2008.

[28] Colin Dixon, Ratul Mahajan, Sharad Agarwal, and A.J.Brush. An Operating System for the Home. In Proceedings of the 9th USENIX conference on Networked Systems Design and Implementation, pp.337–352, 2012.

[29] Manuel Garca-Herranz, Pablo A. Haya, and Xavier Alamn. Towards a Ubiquitous End-User Programming System for Smart Spaces. Journal of Universal Computer Science, Vo.16, pp.1633-1649, 2010.

[30] Amy Hwang, Michael Liu, Jesse Hoey, and Alex Mihailidis. DIY Smart Home: Narrowing the gap between users and technology. In Proceedings of the IUI Workshop on Interactive Machine Learning, 2013.

[31] Tao Zhang and Bernd Brgge. Empowering the User to Build Smart Home Applications. In Proceedings of 2nd international conference on smart homes and health telematic, 2004.

[32] Jong-bum Woo and Youn-kyung Lim. User Experience in Do-It-Yourself-Style Smart Homes. In Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing, UbiComp'15, pp.779–790, 2015.

[33] Blase Ur, Elyse McManus, Melwyn Pak Yong Ho, and Michael L. Littman. Practical trigger-action programming in the smart home. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI'14, pp.803–812, 2014.

[34] Masahiro Kaneko and Jiro Tanaka．Paper control panel: Making paper-based touch interface．In Interaction 2014，pp.562-567，2014，(In Japanese)．

[35] 加藤邦拓，宮下芳明． 紙窓: そこに置くだけで操作可能なカードインタフェース． 第 21 回インタラクティブシステムとソフトウェアに関するワークショップ，WISS ’13，pp.163-164，日本ソフトウェア科学会，2013．

[36] Greg Saul，Cheng Xu，and Mark D. Gross． Interactive paper devices: End-user design & fabrication. In Proceedings of the Fourth International Conference on Tangible，Embedded，and Embodied Interaction，TEI’10，pp.205-212，2010．

[37] Jie Qi and Leah Buechley． Sketching in circuits: designing and building electronics on paper． In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems，pp.1713-1722，2014．

[38] Michael S. Horn，Sarah AlSulaiman，and Jaime Koh．Translating Roberto to Omar: Computational Literacy, Stickerbooks, and Cultural Forms． In Proceedings of the 12th International Conference on Interaction Design and Children，pp.120–127，2013．

[39] Daisuke Komoriya，Buntarou Shizuki，and Jiro Tanaka． Task Specific Paper Controller that Can Be Created by Users for a Specific Computer Operation． In Proceedings of 17th International Conference on Human-Computer Interaction，HCII’15，pp.418–428，2015．

[40] 竹川佳成，福司謙一郎，Machover Tod，寺田 努，塚本昌彦． 絵楽器の設計段階におけるプロトタイピング支援システムの設計と実装． 第 19 回インタラクティブシステムとソフトウェアに関するワークショップ，WISS’11，pp.60–65，日本ソフトウェア科学会，2011．

[41] John Brooke． SUS: a retrospective． Journal of Usability Studies，Vol.8，No.2，pp.29–40，2013．