ARマーカを用いたプログラミング支援タンジブルインタフェース

松崎 智史[1,a)]　滝本 宗宏[1,b)]

概要：近年，社会の情報化に伴い，低年齢からのIT教育についての議論が盛んである．そのようなIT教育の1つとしてプログラミング教育が挙げられる．しかしながら，前提知識を持たない初心者や子どもが，一般的に使用されているプログラミング言語を用いて，プログラミングを学習するのは容易ではない．一方で，子どもが容易にプログラミングを学べるようなプログラミング言語や支援環境が提案されている．本研究では，そのような子供向けプログラミング環境の1つとして，キーボードやマウスを使わずに，複数の正方形のカードを実際に手にもち，それらを自由に並べたりあるいは動かすことによって，ロボットの振舞いを直感的かつ容易にプログミングできる環境を提案する．

キーワード：タンジブルプログラミング，AR，ユーザインタフェース

1. はじめに

情報技術の発展に伴い，低年齢からのIT教育が試みられてきている．そういったIT教育の1つとしてプログラミング教育が挙げられる．初等教育の段階からプログラミングについて学ぶことは，理数・情報分野だけでなく他の様々な分野の技能にも良い影響をあたえることが知られるようになってきた．しかしながら，プログラミング初心者や子どもが通常のプログラミング言語を用いて，プログラミングを学習することは容易ではない．このような理由から，子どもでも習得が容易なプログラミング言語や支援環境の必要性が認識されるようになってきている．

本研究では，株式会社イーケイジャパンが開発した開発環境の1つであるIconWorksを参考に，実際にパズルをするようなプログラミングを可能にする環境を提案する．IconWorksは，ロボットの命令に対応したブロックをマウスを用いてPCの画面上で横方向に並べることで，ロボットの動きをプログラミングできるグラフィカルな開発環境である[7]．本プログラミング環境では，実際に触ることのできるロボットへの命令に対応したカードを，手元で並べたり，動かしたりすることでロボットの振舞いを，容易にプログラミングすることができる．

2. 関連研究

プログラミングの初心者や子どもを対象とした支援環境は数多く存在し，これらは大きくグラフィカルなものとタンジブルなものに分類される．グラフィカルプログラミングのScratchでは，様々な形状のブロックアイコンをマウスを用いてPCの画面上で縦方向に並べることで，画面内の画像

1 東京理科大学
a) j6313630@ed.tus.ac.jp
b) mune@cs.is.noda.tus.ac.jp

の動作をプログラミングできる [6]．また同じグラフィカルプログラミングの IconWorks では正方形の命令アイコンを画面上で横方向につなげていくことで，現実のロボットを動かすプログラムを作成できる [7]．いずれもコードを一切記述する必要がないので，子どもでも容易に理解ができ，プログラミングを実践することが可能である．

八城らは，上述した Scratch の命令ブロックを実態のあるブロックとして再現し，それらのブロックを自身の手で組み合わせて，画面内のロボットの画像とともに，実際のロボットの動作をプログラミングできるタンジブルインタフェースを設計した [4]．

Horn らは，パズルピースのようなものひとつひとつに命令に対応した絵柄を貼り付け，それらを実際に自分の手でつなげていくことでロボットの振舞いをプログラミングできるタンジブルな環境を提案した [1]．また，彼らは自身の開発した環境と他のグラフィカルなプログラミング環境との比較実験を行っており，その結果タンジブルな環境の方が子どもの興味を引きやすく，特に少女にその傾向が強いことを示している．一方で大規模なプログラムの作成には向いていないことを指摘している [2]．

藤田らはロボットを実際に手でつかみ動かすことで，その動きをトレースし記録しながらロボットの振舞いをプログラミングできる環境を提案している [5]．その評価として，仮想のロボットを用いたプログラミング手法と，理解のしやすさや振舞いの想像のしやすさなどの点で比較したところ，２つの環境に有意差は見られなかったとしている．物理的なロボットの場合，振舞いの対応を想像しやすいとの予想もあったが，認識や再現度の精度に関する問題からそうならなかったとしている．

3. プログラミング支援環境

本研究で構築した支援環境の全体を図 1 に示す．ユーザは命令マーカと呼ばれる正方形のカードを実際に手にもち，指定の範囲内で並べたり，動かすことでプログラムを作成する．ユーザによって配置されたマーカは，Web カメラを通して画像として本システムに送られ，システムはその画像からマーカの位置や回転角などの情報を取得する．そして，それらの情報を基にして，マーカの上に別の画像を重ねて表示するとともに，プログラムへの命令の登録や，その結果得られるプログラムを，ユーザに分かりやすいように画面上に表示する．ユーザは，システムが表示した映像がまるで手元に存在しているかのようにプログラミングを行うことができる．最終的に，作成したプログラムは PC の Enter キーを押すことでロボットに送信され，実行が開始される．動作確認用のロボットには iRobot 社の iRobot Create を使用する．

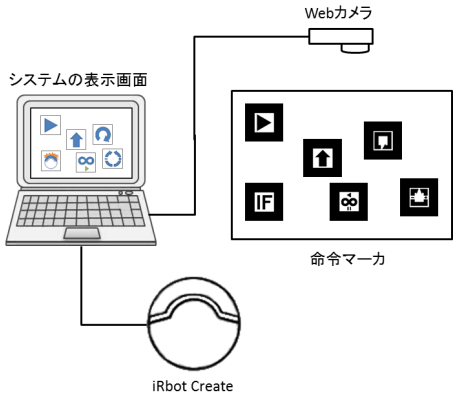
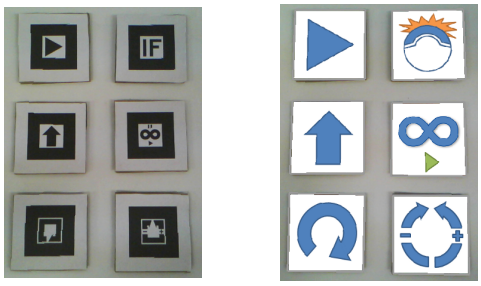


図 1　システムの全体像

3.1 命令マーカ

命令マーカは 1 辺 60mm の正方形のカードであり，各々の内部には 1 辺 40mm のモノクロの絵柄が描かれている（図 2 左）．これらのマーカをシステムが認識すると，それぞれに対応した彩色の絵柄を，システムの画面に重ねて表示する（図 2 右）．本システムが扱うマーカは，次の 6 種類である．

- **開始マーカ（図 2 左，左上）**
  プログラムの開始を示し，命令を登録する際の基準となる．
- **直進マーカ（図 2 左，左中央）**
  矢印が上向きの状態では，ロボットを直進させ，下向きの状態ではロボットを後進させる．
- **回転マーカ（図 2 左，左下）**



実際のマーカ　　画面内に表示されるマーカ

図 2　命令マーカ

ロボットをその場で時計回りに回転させる．

- **衝突分岐マーカ（図 2 左，右上）**
  直進や回転などの命令を実行中にロボットが障害物に衝突したかを判定し，その結果でそのあとに実行する命令を分岐させる．
- **繰返しマーカ（図 2 左，右中央）**
  ロボットに複数の命令を任意の回数だけ繰返して実行させる．
- **調整マーカ（図 2 左，右下）**
  直進の際の距離や回転の際の角度のように，命令固有のパラメータの値を変更する．また編集のやり直しを行う Undo や Redo を実行する．

3.2　プログラミング方法

3.2.1　命令の登録

本システムでは，初期状態でプログラムの開始命令に相当する開始マーカを命令として登録しており，登録済みの命令はその枠を青色の線で，未登録の命令は灰色の線で描画する（図 3 左）．ユーザはロボットに実行させたい命令を，この開始命令を基準として順次登録していく．例えば，直進命令を登録する手順は次のようになる．ユーザは登録したい命令に対応したマーカ（直進マーカ）を開始命令の右方向に配置し，システムがマーカが置かれたことを検知すると，そのマーカを登録の候補として認識し，赤色の線を用いた枠の描画とともに開始命令との接続を表示する（図 3 中央）．ここで右方向に複数のマーカがある場合は開始マーカと最も横軸方向の距離が近いマーカを候補とする．この候補の状態が，3 秒間継続すると，直進マーカを正式に直進命令としてプログラムに登録し，開始命令と緑色の線で接続する（図 3 右）．ここで，直進命令の上に，ロボットを進ませる距離として，0cm という値を表示する．この値の変更方法は 3.2.4 で示す．また次にユーザが新しい命令を登録する場合は，今登録された直進命令のさらに右側に新たな命令マーカを配置し，同様の流れで登録を行う．このように任意の命令を右に向かってつなげていくことで，プログラムを作成していく．一旦命令を登録すると，開始命令との相対距離を記録するので，開始マーカさえ検知できれば，すべての登録した命令を描画できる．すなわち，一旦登録した命令については，マーカ不要なので，1 枚のマーカで，順次命令を登録していくことができる．

3.2.2　衝突分岐命令の登録

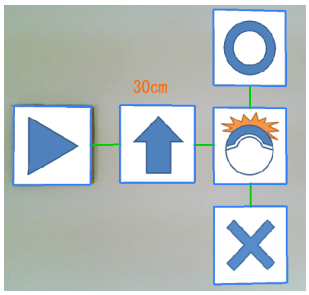


図 4　衝突分岐命令の登録

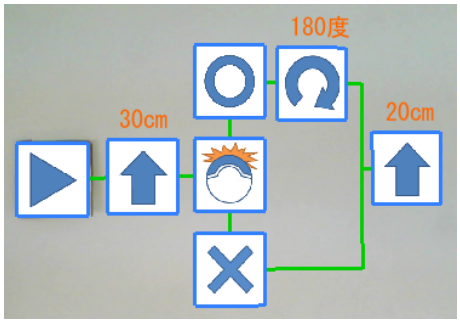
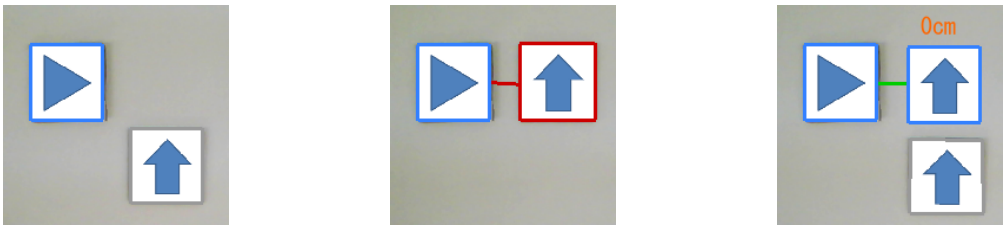


図 5　分岐後の合流

衝突分岐マーカを用いることで，ロボットが障害物に衝突したかを判定し，その結果で 2 通りにプログラムを分岐させることができる．衝突を判定するタイミングは衝突分岐命令の 1 つ前の命令



1. 初期状態　2. 直進マーカを検知　3. 直進マーカを登録

図 3　直進命令の登録の流れ

の実行開始から衝突分岐命令の実行開始までである．図 4 は 30cm 直進する命令の後に分岐命令を登録した直後の状態を示しており，分岐命令を登録すると分岐命令の上方向と下方向に，○の絵柄と×の絵柄の命令が，それぞれ自動的に登録されり．ユーザは衝突判定が真の場合に実行したい命令を○の命令の右方向に，偽の場合に実行したい命令を×の命令の右方向に登録していく．また上下に分岐した 2 列の間，分岐命令の右方向に任意のマーカを置くことで，元の 1 列に合流させることができる．図 5 は 30cm 直進し，障害物に衝突した場合だけ 180 度向きを変え，さらに 20cm 直進するプログラムである．また分岐したいずれかの先でさらに分岐を行う場合，2 回目以降で分岐した列から順次合流させる必要がある．

3.2.3　繰返し命令の登録

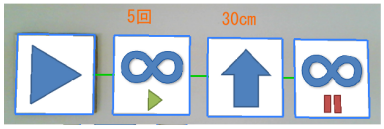
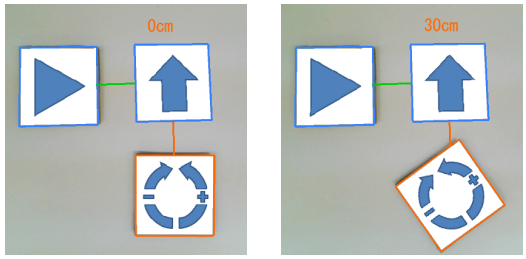


図 6　繰返し命令

繰返しマーカを用いることで，ロボットに複数の命令を任意の回数繰り返させることができる．繰返しマーカには上下の向きが設定してあり，図 2 では，繰返しの開始命令の状態を表示している．このマーカを 180 度回転させると繰返しの終了を示す命令として認識される．図 6 は 30cm 直進する命令を 5 回繰返すプログラムを作成した状態である．ユーザは繰返し実行したい命令列を繰返し開始命令と終了命令で挟み込む形で登録し，開始命令のもつパラメータとして繰返し回数を設定することで任意の回数の繰返し処理を実現できる．

3.2.4　パラメータの変更と Undo と Redo の実行



調整マーカの接続　回転による値の設定

図 7　パラメータの変更

調整マーカを用いることで，特定の命令の持つパラメータの変更と直前に設定した命令を元に戻す Undo や Redo を実行することができる．本システムで，パラメータを持つ命令は直進（後進）命令，回転命令，繰返し開始命令の 4 つである．ユーザは，パラメータを変更したい命令の下方向に調整マーカを配置し，変更対象の命令（直進命令）との接続を行う（図 7 左）．そして，接続された状態で調整マーカを反時計回りに回転させることでその回転角に応じてパラメータを変更することができる（図 7 右）．距離は 10cm ごと，角度は 5 度単位で設定可能である．また，開始命令に調整マーカを接続し，時計回りに 10 度回転させるごとに直前に設定した命令を元に戻す Undo 処理を，また反時計周りに回すことで Undo で戻した命令を再び登録する Redo 処理を実行できる．

4.　まとめと評価

本稿では，実際に触れることができる正方形のカードをインタフェースとし，直感的かつ容易にプ

ログラミンができる環境を提案した．また今回実装した環境をオープンキャンパスや学園祭で訪れた一般の人々や，研究室訪問で訪れた近隣の小学校高学年の児童たちに体験してもらった．その様子を観察した結果，小学校高学年以降の年齢ならば本環境を比較的容易に習得することができ，個人差はあるももの複数の命令を組み合わせた様々なプログラムを作成できることが分かった．また，同様のプログラムばかりを作成している人も，特定の動きを問題として出題することによって，試行錯誤をしながら，プログラミングを上達させていく様子が見受けられた．また，小学校低学年辺りの年齢になると，理解や操作にいくらかの難しさを感じている様子が伺えたものの，カードを並べることで PC の画面上で命令がつながっていき，意味は分からなくともロボットが動くだけで楽しんでいる様子が見られた．

5. 今後の課題

本システムでは，パラメータの変更や操作のやり直しなどの処理を調整マーカの回転によって行うが，各マーカは上方の Web カメラによって撮影され認識されるので，マーカを回転させる際に絵柄を手で覆ってしまうと認識が途中で途切れてしまい，操作をうまく行えない状況が頻出した．これはマーカの絵柄を小さくするか，マーカの四隅に別々の絵柄を印字することによって改善できると考える．

また，Web カメラによる撮影範囲の点から，プログラムの大きさに制限があり，長く複雑なプログラムの作成が難しいという問題もある．これには複数の命令列をモジュール化する機能や，命令列を折りたたむ機能などを実装することで改善できると考える．また編集操作については，命令の削除が新しいものからしか行えないという問題があり，任意の位置の命令の削除や上書き等の機能が必要であると考える．

今後はシステムの改良を行うとともに，実際にプログラム初心者や子どもに体験してもらうことを通して，アンケート等による具体的な評価を行い，既存の他の支援環境との比較を行う予定である．

参考文献

[1] Michael S. Horn, Erin Treacy Solovey, and Robert J.K. Jacob: **Tangible Programming and Informal Science Learning: Making TUIs Work for Museums**.Proceedings of 7th International Conference on Interaction Design & Children, 2008.

[2] Michael S. Horn, Erin T. Solovey, R. Jordan Crouser, Robert J.K. Jacob: **Comparing the Use of Tangible and Graphical Programming Languages for Informal Science Education**. ACM Conference on Human Factors in Computing Systems, 2009.

[3] Hiroshi Ishii, Brygg Ullmer: **Tangible Bits: Towards Seamless Interfaces between People, Bits and Atoms**. ACM Conference on Human Factors in Computing Systems, 1997.

[4] 八城朋仁, 迎山和司: **物質プログラミング-物質によるプロックプログラミング-**. 情報処理学会インタラクション, 2014.

[5] 藤田智樹, 米海鵬，杉本雅則: **タンジブルなロボットを導入したテーブルトップ環境における直観的なプログラミング手法の提案**. 情報処理学会インタラクション, 2011.

[6] Mitchel Resnick, John Maloney, et al: **Scratch: Programming for All**. Communications of the ACM, 2009.

[7] 株式会社イーケイジャパン: **IconWorks**. http://www.elekit.co.jp/download/software/00007 , retrieved 2014/10/30.

質疑・応答

齋藤 (九工大)　対象年齢はいくつぐらいを考えているのか？

松崎　距離や角度の単位から３・４年生あたりを考えています．

齋藤（九工大）　アルゴロジックというものでは１・２年生でも出来ているので，教え方を工夫すればなんとかなるかもしれません．

松崎　なるほど，参考になります．

平石（京大）　直進や回転のパラメータは最初０ではなく適当な値を設定して，とりあえず作ったら動く，みたいな仕様のほうが良いのでは？

松崎　ありがとうございます，参考にします．

不明　IF 分岐の後で更に分岐した場合、合流はどうするのか？

松崎　その場合２回目の分岐を閉じからでないと、最初の分岐は閉じれない仕様になっています．

不明　パラメータの設定が反時計周りというのには違和感がある．

松崎　確かに実際に色々な人に体験してもらった際には，値を増やそうと時計回りに回す人が多かったので，改良したいと思います．

中山（(株)DoBoken）　AR を使うならばもっと色々な動きを取り入れて UX を工夫したほうが良いのでは？例えば、包丁のような動きで命令を切り取ったりとか．

松崎　ありがとうございます．命令の削除は実装したいと思っていたので，参考にさせて頂きます．