情報処理学会論文誌 Vol.53 No.3 1101–1109 (Mar. 2012)

# テーブルトップ環境における新しいロボットプログラミング手法の提案

藤田 智樹[1] 米 海鵬[1] 杉本 雅則[1,a)]

受付日 2011年6月16日, 採録日 2011年9月12日

**概要**：本稿では，プログラミングの知識や経験が乏しいユーザでも簡単にロボットプログラミングが行える新しい手法を提案する．提案手法は，物理的なロボットをマルチタッチ入力可能なテーブルトップ環境に導入することにより，条件分岐のためのイベントやロボットの振舞いを入力できる．プログラミング経験のない大学生を被験者とし，従来のグラフィカルなプログラミング手法と提案手法の比較実験を行った．その結果，いくつかの評価項目で 2 つの手法に有意差が確認できた．さらに，ユーザ自身によって容易に学習コンテンツやゲームを構築できるかどうかを確認するためのパイロットスタディを実施し，提案手法を用いたプログラミング環境の可能性を検討した．

**キーワード**：プログラミング手法，マルチタッチテーブルトップ環境，タンジブルなロボット

# A Novel Technique for Robot Programming on Tabletop Environments

Tomoki Fujita[1] Haipeng Mi[1] Masanori Sugimoto[1,a)]

Received: June 16, 2011, Accepted: September 12, 2011

***Abstract:*** We propose a novel robot programming technique for supporting users with less programming knowledge or experiences. We use a tabletop environment that can recognize multi-touch input and track physical robots simultaneously. User can easily define events for conditional statements or behaviors of robots. Comparative evaluations between conventional graphical programming techniques and the proposed technique were conducted with university students who did not have programming experiences. Significant differences were found in relation to some of the evaluation items. A pilot study to confirm if the proposed technique allowed users to easily create applications for learning or games, and to explore possibilities of this novel programming environment, was conducted.

***Keywords:*** programming technique, multi-touch tabletop environment, tangible robot

## 1. はじめに

近年，IT（情報技術）の発展にともない，教育の現場にコンピュータを用いた電子教材の導入がさかんに行われている．これまでに提案されている電子教材の 1 つとして，コンピュータを用いたシミュレーションがあげられる．学習者は，シミュレーションを通して環境問題や都市計画など実世界の問題を学ぶことができる．これらの問題に対する解決策をシミュレーション環境上で試行錯誤的に探索することを通して，学習者の知識や理解の深化を促進する効果が報告されている [1]．さらに，物理的なオブジェクトやロボットを導入することにより，学習者の学習への動機づけや興味のレベルを高められることが示されている [3]．著者らは，シミュレーションに代表される学習コンテンツのデザインや構築を，学習者自身で行うことができる物理的な環境を通して，学習者の自発的な学習を促進することを目指している [2]．そのためには，シミュレーション環境で用いる物理的なロボットのプログラミングを学習者自身が行える必要がある．このようなロボットプログラミングでは，コンピュータの操作に加え，周囲の状況に応じた振舞いを実現するための条件分岐や繰返しなどの概念を理解

[1] 東京大学
The University of Tokyo, Bunkyo, Tokyo 113–8656, Japan
[a)] sugi@itl.t.u-tokyo.ac.jp

することが学習者に要求される [5]．よって，プログラミングの知識や経験が乏しい学習者でも，容易にロボットプログラミングを行えるような支援環境が必要となる．

そこで著者らは，物理的かつ直接的な入力のみでロボットの振舞いをプログラミングできる手法を提案する [4]．本稿では，マルチタッチ入力とロボットトラッキング可能なテーブルトップ環境である RoboTable [18] を基に，その拡張版である RoboTable2 を構築した．RoboTable2 では，ユーザはテーブル上でロボットを掴んで動かす，ロボットの周囲に表示されるアイコンを指で触れるなどの直感的な操作で，条件分岐を含むプログラミングを行うことができる．よって，たとえば小学校低学年の子どもやプログラミングを普段行わない大学生など，プログラミング経験の乏しい学習者でも容易にロボットプログラミングを行えると考える．これまで，タッチパネルや TUI（tangible user interface）などのユーザインタフェースを導入した初心者向けのプログラミング手法は多く提案されている [5], [6], [7]．しかし，物理的なロボットを活用し，状況に応じた振舞いをさせるプログラムを容易に構築できるよう支援する手法の提案は，著者らの知る限りほとんど存在しない．

評価実験では，グラフィカルなブロックを操作してプログラミングを行うことができる環境（以下では “グラフィカル手法” と呼ぶ）を構築し，提案手法と比較した．プログラミングの経験がない大学生に被験者として参加してもらい，実験後のアンケートへの回答を依頼するとともに，利用中に撮影したビデオや利用ログの分析を行った．その結果，グラフィカル手法と提案手法の間に，イベントの設定しやすさ，ロボットの振舞いの組み立てやすさ，および作成したプログラムに対する満足度において，有意差が確認できた．さらに，concept-of-proof のためのパイロットスタディにおいて，大学生に交通シミュレータとゲームを自由に作成してもらい，提案するプログラミング手法によるコンテンツ構築の可能性を検討した．

本稿の構成は以下のとおりである．2 章では本研究の関連研究を示す．3 章では RoboTable2 を通して提案されるプログラミング手法について述べる．4 章では実験の詳細および結果について述べる．5 章では評価実験を結果をふまえて考察する．6 章では，コンテンツ構築に関するパイロットスタディについて示す．7 章で，本稿の結論と今後の展開を示す．

## 2. 関連研究

プログラミング経験の乏しい初心者を対象としたプログラミング手法は，数多く提案されている．ここでは本稿と関連の強い研究をいくつか紹介する．Scratch [5] は，グラフィカルなプログラミング環境の 1 つである．Scratch ではブロック状のオブジェクトをマウスで操作し，積み重ねることでコードを記述せずに簡単にプログラミングすることができる．Quetzal [6] は小学校低学年の子供を対象とし，TUI を導入したプログラミング環境である．子どもたちはマーカーが張り付けられたブロックを直接手で掴み，組み立てることでプログラミングを行うことができる．Quetzal では flow-of-control chains という独自のブロック接続手法を用いることで，ループや条件分岐などを含むプログラムを作成することが可能である．Galllardo らは，マルチタッチ入力可能なテーブルトップ環境に TUI を導入した TurTan [7] と呼ばれるプログラミング環境を提案している．TurTan ではテーブル上に配置したタンジブルなブロックを掴んで移動することで，テーブル上に表示されたバーチャルなタートルの経路をプログラミングすることができる．

近年，マルチタッチテーブルを用いたロボット操作手法が多数提案されている．Kato ら [8] は，指ジェスチャでベクトルフィールドを生成し，それに従ってロボットを移動させる手法を提案している．Seifried ら [9] は，掃除ロボットを含む室内の家電製品を操作するためのコントローラを実現している．また，ロボット操作のための自然なジェスチャの設計ガイドラインについても議論されている [10]．Puppet Master [11] は，ユーザがテーブル上で物理的なトークンを平行移動あるいは回転させることで，バーチャルキャラクタの動きを実時間で生成する．Guo ら [12] は，テーブル上の指ジェスチャとタンジブルな入力デバイスを用いることで，離れた場所の複数ロボットを操作する手法を提案している．同様のアイデアは，MuMoMuRo [13] でも議論されている．Tangible Bots [14] では，テーブルトップ上でのタンジブルなロボットとのインタラクション手法について述べられている．これらのシステムでは，指やトークンの動きをロボットやキャラクタの動きに反映させられるが，条件分岐を含むプログラムを作成することはできない．

Horn らは，TUI を導入したプログラミング環境を，コンピュータ上のブロックをマウスで操作するプログラミング環境と比較している [15]．彼らは，博物館に来館した子どもたちが作成したプログラムの数やコード長，および子どもの振舞いの観察を基に分析を行っている．その結果，子どもたちを自発的に協調させるという効果があると述べている．しかし一方で，実体を持つブロックで作成したプログラムとロボットの振舞いが関連付けにくいという問題などが指摘されている．著者らが提案するプログラミング環境では，入出力機能を持つロボットを直接掴んでプログラミングする．よって，このような問題を解決できると考える．

Frei らの提案する curlybot [16] では，ロボットを直接掴んで動かしその動きを再現できる．そのため，ユーザの入力とロボットの出力を関連付けやすいという特徴を持つ．Raffle らの開発した Topobo [17] では，ユーザが様々な形

状のパーツを組み合わせロボットを構築する．ユーザは，ロボットを手で掴んで動かすことにより，その動きを記録，再生できる．しかし，curlybot や Topobo ではロボットが周囲の状況に応じて振舞いを自動的に変化させるプログラムを作成することはできない．

## 3. 提案するプログラミング手法

### 3.1 テーブルトップ環境

RoboTable [18] の拡張版である RoboTable2 では，既存のマルチタッチ入力認識手法である DI（Diffused Illumination）[19] と FTIR（Frustrated Total Internal Refrection）[20] を組み合わせることで，テーブル上の指の接触と，ロボットに取り付けたマーカを同時に認識できる．テーブル上の入力情報は，reacTIVision [21] と呼ばれる画像認識ライブラリにより取得される．テーブル上のロボットには Bluetooth モジュールが実装されており，RoboTable2 は Bluetooth を介してそれぞれのロボットに命令を送ることができる．

入力情報を利用して描画を行うために，RoboTable2 は MT4j [22] と呼ばれるマルチタッチライブラリに加え，オブジェクトの物理的な衝突を表現するために Java JBox2D [23] を組み込んでいる．さらに，異なる駆動機構や無線通信手法を備えたロボットに対応するために，独自のシステム開発 Toolkit を実装している [24]．RoboTable2 のシステム構成を図 **1** に示す．

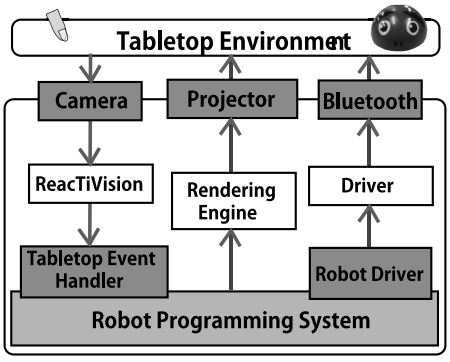



図 1 RoboTable2 のシステム構成
**Fig. 1** System configuration of RoboTable2.

### 3.2 イベント駆動型プログラム

本稿では，プログラムモデルとしてイベント駆動型プログラムを採用する．イベント駆動型プログラムは個々のイベントと，それに対するロボットの振舞いの組合せで構成される．本稿では，この組合せをプログラムブロックと呼ぶことにする．プログラムブロックは RoboTable2 に登録され，RoboTable2 はテーブル上の状況と登録されたプログラムブロックとの照合を行い，個々のロボットに送信する命令を切り替える．たとえば，「正面の障害物を認識（イベント）」と「右に 90° 回転して回避（振舞い）」からなるプログラムブロックが該当すれば，ロボットに障害物を回避させることが可能になる．本稿では，作成したプログラムブロックの集合をプログラムとして扱う．

### 3.3 プログラミング手法

RoboTable2 を用いることで，テーブル上のロボットに対し図 **2** に示す直接的な入力を行うことが可能である．

- ロボットを直接掴んで動かす（“grasp & move”）．
- ロボットの周囲に表示されるボタンなどのバーチャルなオブジェクトに触れて入力する（“touch”）．
- マルチタッチジェスチャで入力する（“resize & reshape”）．

提案するロボットプログラミング手法では，ユーザはこれらの入力操作を組み合わせることによりプログラミングを行う．

#### 3.3.1 認識イベントの登録

テーブル上のロボットは，他のオブジェクトへの接近，衝突を RoboTable2 を介してイベントとして認識することができる．本稿では，これを認識イベントと呼ぶ．認識イベントの取得のためには，あらかじめロボットが認識可能な領域を RoboTable2 に登録しておく必要がある．そこで，認識領域としてロボットの正面に扇型のオブジェクトを配置する．この扇形の領域は，マルチタッチジェスチャ入力（“resize & reshape”）でユーザが半径と角度を自由に調整できる（図 2 (c)）．RoboTable2 は，個々のロボットに割り当てられた認識領域とテーブル上のオブジェクトの接触を定期的に確認する．認識領域にオブジェクトが接触す

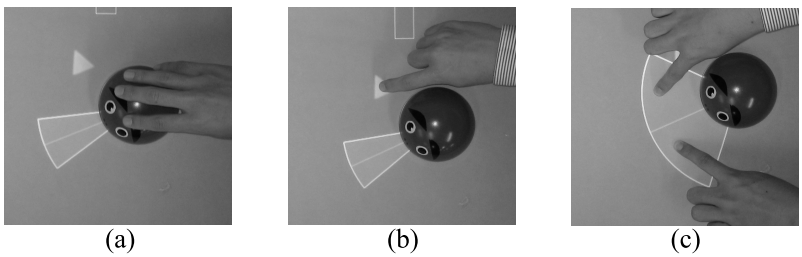



図 2 RoboTable2 上の入力手法：(a) grasp & move (b) touch (c) resize & reshape
**Fig. 2** Input techniques on RoboTable2: (a) grasp & move (b) touch (c) resize & reshape.

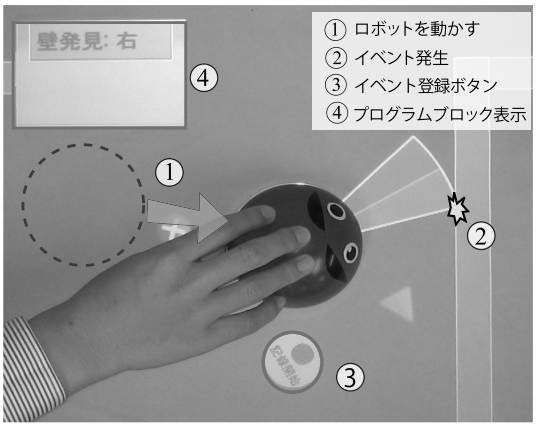



図 3 認識イベントの登録
**Fig. 3** Registration of recognition events.

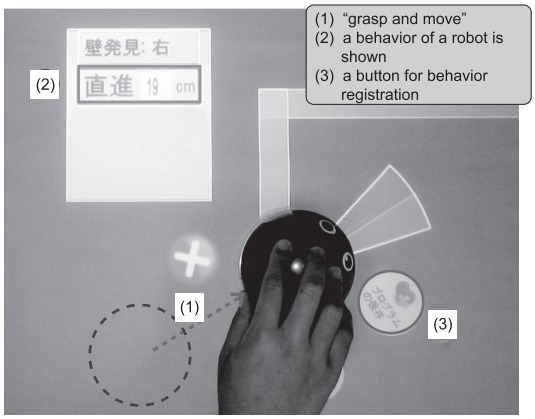



図 4 振舞い情報の登録
**Fig. 4** Registration of robot behaviors.

れば，接触したオブジェクトと，接触した方向（ロボットから見て左，正面，右）の情報を用いて認識イベントを作成する（たとえば，バーチャルな壁をロボットの正面に来るように配置し，そこにロボットを近づけて認識領域と接触させれば「壁：正面」という認識イベントが作られる）．このとき，発生したイベントを登録するためのボタンがロボットの周囲に表示される．ユーザがこのボタンに触れること（“touch”）で，プログラムブロックのテンプレートが表示され，その中に認識イベントを登録できる（**図 3**）．認識イベント発生時に，複数のオブジェクトがロボットの認識領域内に入ることも考えられる．このときは自動的に一番近くにあるオブジェクトのみを選択し，それに対する認識イベントを作成する．なお，複数のオブジェクトを同時に認識して認識イベントを生成することも可能であるが，ユーザのプログラミング経験が乏しいことを考慮して，より単純な機能を実装することとした．

RoboTable2 では，認識可能領域に何もオブジェクトが存在しない状態を通常状態と定義する．通常状態のプログラムブロックもロボットの周囲に表示されるボタンから作成できる．プログラムの実行時には，他のイベントが発生しない限り，通常状態のプログラムブロックが繰り返し呼び出される．

### 3.3.2 振舞い情報の登録

プログラムブロックのテンプレートに認識イベントが登録されると，ロボットの周囲にはそのイベントを含むプログラムブロックが表示される．この状況でユーザがロボットを手で掴んで動かす（“grasp & move”）ことにより，そのイベントに対応するロボットの振舞い情報を作成できる．振舞い情報は，ユーザが動かしているロボットの位置と角度情報を読み取りながら，直進，後退，回転の命令に変換することで RoboTable2 により生成される（たとえば，ロボットを掴んで正面方向に動かした場合，プログラムブロックには直進 30 cm という振舞い情報が生成される）．ユーザはロボットを動かしながら，プログラムブロック内の変換された振舞い情報を確認する．意図したとおりの情報が入力されていれば，ロボット周囲に表示される「振舞い情報保存ボタン」に触れることで，作成された振舞い情報を登録する．振舞い情報が確定すると，RoboTable2 にプログラムブロックとして登録することが可能になる．このとき，ロボットの周囲にはボタンが表示される（**図 4**）．ユーザはこのボタンに触れることで，作成したプログラムブロックを RoboTable2 に登録できる．その後，同様にロボットを動かしてこのボタンを押すという作業を繰り返すことで，新たなプログラムブロックを登録することができる．

### 3.3.3 プログラムブロックの表示と編集

ロボットの周囲に表示されるメニューボタン中のプログラムブロック編集ボタンに指で触れると，RoboTable2 に保存されているすべてのプログラムブロックがロボットの周囲に表示される．ユーザは，各々のプログラムブロックを指で触れて移動させたり，テーブルの隅に表示されるゴミ箱のアイコンにドラッグすることで作成したプログラムブロックを削除したりできる．

### 3.3.4 プログラムの実行

ロボットの周囲に表示されているプログラム実行ボタンに触れると，RoboTable2 は登録されたプログラムブロックを用いて Bluetooth 通信でロボットを操作する．ロボットの認識領域にオブジェクトが接触し認識イベントが発生すると，そのつど登録されているプログラムブロックとの照合を行う．このとき，一致するプログラムブロックが存在すれば，そこから振舞い情報を取り出し，その情報を基にロボットを操作する．一致するイベントが登録されていない場合，通常状態の振舞い情報が用いられる．プログラム実行時には，ロボットの周囲にプログラム停止ボタンが表示される．このボタンに触れることで，プログラムの実行が停止する．

### 3.4 提案手法のシステム構成

図 5 に示すように，提案システムは Tabletop Event Handler からロボットや指などの位置情報を取得し，Robot-Driver 経由でロボットの制御を行う．Tabletop Event Handler からからの入力情報は Event Processor で解析され，認識イベントやボタン入力イベントなどの特定を行う．特定されたイベントに応じて，Program Block Generator，Robot Controller に処理が移される．各々の処理について以下に述べる．

- プログラムブロックの生成

Event Processor がロボットの認識イベントを同定すると，Program Block Generator により「認識イベント保存ボタン」が表示される．ユーザがこのボタンに触れると，Event Generator が起動され認識イベントとして保存される．次に，ユーザはロボットをテーブル上で動かすことで，保存された認識イベントに対応するロボットの振舞いを入力する．入力された振舞いは Behavior Recorder によりロボットの駆動機構（回転，直進など）に応じた命令（振舞い情報）に変換される．意図どおりの動きを入力できたと判断したユーザが「振舞い情報保存ボタン」に触れると，認識イベントと振舞い情報が Program Block Assembler に送られる．ユーザは，ロボットを動かしながら繰り返し振舞い情報の入力を行うことができる．そして「振舞い情報保存ボタン」に触れるたびに，新たな振舞い情報に更新される．

- プログラムブロックの登録

Program Block Assembler は，受け取った認識イベントと振舞い情報からプログラムブロックを構成し，テーブル上に「プログラムブロック登録ボタン」を表示する．このボタンに触れると，Program Block Memory にプログラムブロックが送られ，格納される．その際，すでに格納済みの認識イベントを確認し重複する場合は上書き処理を行う．

- プログラムブロックの実行

Event Processor が「プログラム実行ボタン」からの入力を確認すると，Robot Controller に処理を移行する．Robot Controller は Program Block Memory にアクセスし，プログラムブロックとして格納されている振舞い情報を実行する．Event Processor は新たな認識イベントが発生するたびに，Robot Controller にそのイベントの情報を送信する．Robot Controller は受信したイベントと Program Block Memory に格納されているプログラムブロック中の認識イベントを照合し，一致する認識イベントの振舞い情報を Excecutor に実行させる．ロボットの動きはトラッキングされるため，指定された振舞い情報からの誤差を取得できる．Calibrator は，この誤差を補償するために補正された振舞い情報を Executor に送信する．

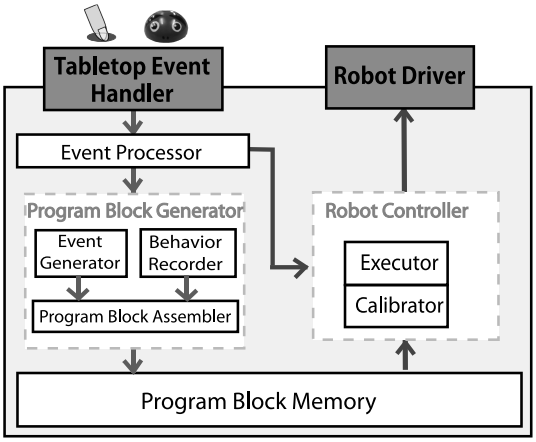



図 5 提案手法のためのシステム構成
**Fig. 5** System configuration of the proposed method.

## 4. 評価実験

### 4.1 実験設定

提案するプログラミング手法を評価するための実験を行った．比較実験を行うため，グラフィカルなブロックを指で操作して組み合わせながらプログラミングを行うグラフィカル手法を，提案手法と同様のテーブル上に構築した．比較するプログラミング環境として，物理的なブロックなどを積み重ねたり連結したりすることによるタンジブルなプログラミング環境を用いることも考えられる．しかし，本稿では RoboTable2 の最初の実験として，マルチタッチ入力が可能なテーブルトップ環境での提案手法の効果を明らかにすることを目標とするため，広く用いられているグラフィカル手法を比較対象とした．

被験者は，グラフィカル手法および提案手法の 2 つのプログラミング手法を利用する．ロボットを掴んでその振舞いを設定できる提案手法とは異なり，グラフィカル手法では振舞いを設定するパラメータを図 6 右下のスライダに触れて設定する．評価実験のタスクとして，ロボットがバーチャルな障害物を回避しながらゴールを探索する maze

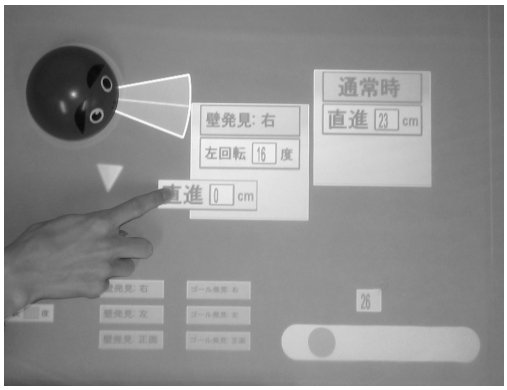



図 6 グラフィカル手法：ユーザは指でブロックを操作し，組み合わせながらプログラミングを行う
**Fig. 6** A graphical programming technique: users conduct programming by manipulating and assembling virtual blocks with their fingers.

（図 7）を採用した．maze には同程度の難易度のマップが 4 つ用意されており，それぞれのマップのスタートとゴールの位置が固定されている．ロボットがゴールに接触すると，自動的にロボットは停止し，画面上に “GOAL” の文字が大きく表示され，タスクが終了する．

評価実験では，プログラミング経験のない大学生 10 名（男性 5 名，女性 5 名，平均 20.3 歳）を著者らの研究室以外から募集した．まず最初に，実験者が被験者に対し 2 種類のプログラミング手法の説明を行った．被験者がタスクの内容を理解しプログラミングに十分慣れた後，実験を開始した．実験では各被験者に対し，2 種類のプログラミング手法と maze のマップをランダムに選択して提供した．ロボットがゴールに到達した時点でタスクを終了し，各プログラミング手法に関するアンケートへの記入を求めた．被験者が回答したアンケート項目を以下に示す．

**項目 1** プログラミング手法の理解しやすさ
**項目 2** 認識イベントの設定しやすさ
**項目 3** 振舞いの入力しやすさ
**項目 4** 作成したプログラムに対する満足度

アンケートでは，以上の 5 項目を 7 段階のリッカート尺度（1：まったくあてはまらない，2：あてはまらない，3：ややあてはまらない，4：どちらでもない，5：ややあてはまる，6：あてはまる，7：非常にあてはまる）で評価を求めた．「作成したプログラムに対する満足度」は，作成されたプログラムによるロボットのイベント認識や振舞いに対し，被験者がどの程度満足できたかを評価するための項目である．アンケートではさらに，プログラミング手法に関する問題点や気づいたことなどについてコメント欄への記入を求めた．実験中はプログラミングに要した時間を測定するとともに，ビデオ撮影を行った．実験中にユーザが行ったすべての操作ログは，システムによって自動的に記録された．

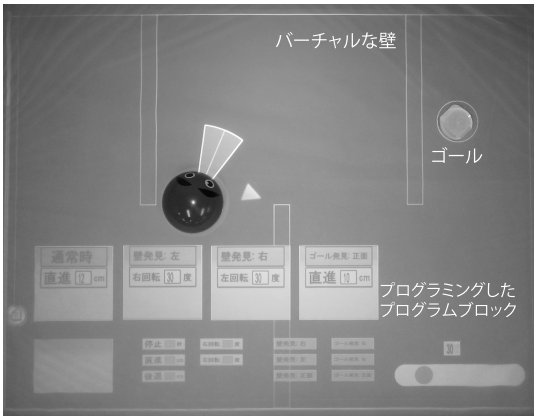



図 7 評価実験に用いた maze の例
**Fig. 7** An example of a maze used in the experiments.

### 4.2 実験結果

評価実験で実施したアンケート結果を，図 8 に示す．順序尺度のため，Wilcoxon の符号付き順位和検定を用いた．その結果，項目 2 ($p < .01$)，項目 3 ($p < .05$)，4 ($p < .01$) において，2 つの手法に有意差が確認された．

2 つの手法において，被験者がプログラミングを完了するのに要した時間の結果を図 9 に示す．Kolmogorov-Smirnov 検定および F 検定を行い，正規分布および等分散の仮説が棄却できないことを確認したうえで t 検定を行った．その結果，2 つの手法に有意差は認められなかった．さらに，作成されるプログラム数についても同様の手順で比較を行った．その結果を，図 10 に示す．t 検定を適用し

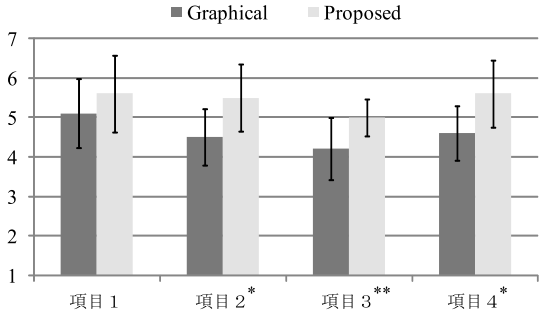



図 8 アンケート結果（*：$p < .01$，**：$p < .05$）
**Fig. 8** Qestionnaire results (* : $p < .01$, ** : $p < .05$).

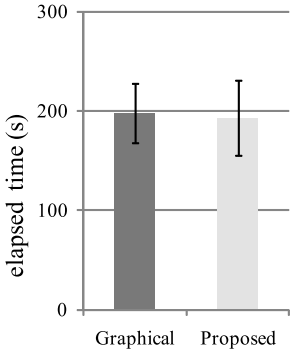



図 9 被験者がプログラミングに要した時間（単位：秒）
**Fig. 9** Time spent for programming by each subject (unit: second).

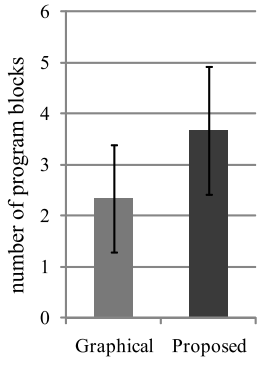



図 10 被験者が作成したプログラムブロック数（$p < .05$）
**Fig. 10** Number of program blocks created by each subject ($p < .05$).

た結果，プログラムブロックの作成数において提案手法の方が有意に多いことが確認された（$t(9) = 2.69$，$p < .05$）．

## 5. 議論

評価実験の結果，提案手法はグラフィカル手法よりも認識イベントの設定，振舞い情報の入力，作成したプログラムの満足度において優れていることが示せた．一方，プログラミング手法の理解のしやすさに関しては，有意な差はなかった．つまり，手順のそのものはいずれの手法についても，ユーザが容易に理解できる一方，それを実際に使ってプログラミングを行う場合に差が現れたことが分かる．さらに，プログラムブロック数に関しては，提案手法の方がより多かった．以上から，提案手法によって，より容易かつ効率的で直感性の高いプログラミングが可能になり，ロボットの動きに対する被験者のアイデアがより多く表現できたと考えることができる．

一方，以下の課題も明らかになった．

- 情報提示法に関する問題点

提案手法を用いた場合，複数の被験者から「角度の細かい調整にストレスを感じた」というコメントを得た．著者らはこの問題の 1 つの原因が，提案手法の情報提示法にあると考えている．提案手法ではロボットを用いて動きの入力を行う際に，ロボットの周囲にプログラムブロックの情報を表示することで，プログラミング中の認識イベントや振舞い情報を確認できる．しかし，この情報提示ではロボットを動かしてパラメータを調整する際に，操作対象のロボットではなくロボットの周囲に表示された情報に視点を移動させなければならない．「細かい調整」の際のこの視点移動の繰返しが，被験者のストレスとなったと考えられる．実験中に撮影したビデオの分析を行ったところ，問題点を指摘した被験者が視点移動を繰り返しつつ，ロボットを何度も左右に回転させながらパラメータの調整を行っている様子が確認できた．よって，微調整を要するロボットの振舞いを設定する際，より効率的かつストレスなく行える情報提示方法を検討する必要がある．

- 入力手法としての得失

グラフィカル手法では，ロボットの移動距離や回転角度を数値で正確に定めることができる．そのため，ロボットを用いた図形描画や，振舞いを正確に定める必要があるシミュレーション環境などに適している．一方で，提案手法では，ユーザがイメージしたロボットの振舞いを容易にプログラムとして生成できる．そのため，シミュレーションやゲームなどで，物理的なロボットの定性的な振舞いをより直感的なやり方でデザインするのに適しているといえる．また，提案手法とグラフィカルなプログラミング手法を組み合わせることにより，両者の利点を生かしたプログラミング環境を実現することも考えられる．

- 操作対象と操作手法

既存研究の多くは，物理的な対象に対しては物理的な操作を行うことで動きの入力を行う方法が提案されている．たとえば，topobo [17] や curlyBot [16] では物理的に関節を回したり，ロボットそのものを移動したりすることで，その動きをプログラムする．一方，本研究で提案する手法は，たとえば物理的な対象，物理的な操作のみでは実現が必ずしも入力が容易ではない場合に，仮想的な対象を物理的な操作で入力している．具体的には，ロボットが認識可能な領域は視覚的に表示されており，それを指で物理的に操作することで入力する．この手法を複雑な機能を持つロボットに適用できれば，従来よりもより細かいレベルでの制御をより直感的にプログラミングできる可能性があると考える．一方，提案システムのように物理的な対象と仮想的な対象が混在する状況で，そのすべてを物理的に操作できるようにするべきかどうか，その場合はどのような操作方法が良いかをさらに検証する必要がある（たとえば，物理的なロボットの速度設定に，仮想的なスライダを用いることが最適かどうかは自明ではない）．今後，物理世界と仮想世界が混在する状況でのプログラミング環境のデザインガイドラインについて，さらなる実験を通して検討を進めたい．

## 6. ユーザ自身によるコンテンツ構築に向けてのパイロットスタディ

提案手法により，ユーザ自身でシミュレーション環境などのコンテンツ構築を行う様子を確認するパイロットスタディを行った．今回は提案手法を用いての最初の実験という位置付けのため，プログラミング経験の有無を問わずに著者らが所属する大学の学生 5 名に，被験者として参加を求めた．パイロットスタディではロボットを用い，1）学習支援用コンテンツとして交通シミュレータおよび，2）エンタテインメント応用としてのゲームのそれぞれについて，被験者が自由にプログラミングを行った．この実験は本研究の proof-of-concept の位置付けであり，またユーザに過度の負荷をかけないようにするため，2 台のロボットが用いられた．各ユーザは最初に 10 分程度システムの使用法についての説明と自身での操作を行った後，各自自由にロボットプログラミングを行った．

以下では，1）交通シミュレータについてのパイロットスタディを示す．提案手法を用いたプログラミング環境では，実世界で使用される信号，各種の交通標識などがバーチャルなオブジェクト（アイコン）として用意された．ユーザは，指で自由に道路を描き，パレット上に配置された上記のアイコンを選んで任意の場所に配置する．被験者はたとえば「注意散漫な運転手」の自動車をシミュレーションするために，ロボットの認識イベント領域を小さく設定することで「止まれ」の標識に気付きにくいようにプログラムブロックを作成したり，あるいは信号機の設定（赤や青の

点灯時間）を変えることで車の流れを制御したりすることができる[*1]．

5名の被験者が交通シミュレーション構築に要した時間は平均16分43秒（最大24分11秒，最小10分36秒）であった．また，作成された平均アイコン数およびプログラムブロック数は，それぞれ6.8個（最大11個，最小4個），16.8個（最大24個，最小11個）であった．

ユーザが作成した交通シミュレーションの例を**図 11** に示す．このユーザは繁華街の交通を想定し，交差点を含む道路地図を作成してロボットプログラミングを行った．本シミュレーション例の完成まで10分36秒を要し，作成されたアイコン数は9個，2台のロボット用に作成されたプログラムブロック数は20個であった．

5名のユーザ全員から，「ロボットプログラミングの過程での困難はなかった」，「あまり複雑でなければ，自分の考えているシミュレーション環境を容易に実現できそう」，「（交通標識の位置を指を触れて変えるなど）シミュレーション設定を容易に変更できるので，異なる条件で簡単に何度でも試せる」，「ロボットを使うことで，よりシミュレーションのリアリティが高まる」「より多くのロボットを配置できれば，より現実的な交通シミュレーションが可能ではないか」との回答を得た．これらの回答は，物理的なロボットに対するテーブルトップ上でのプログラミング手法がユーザ自身によるコンテンツ構築を支援でき，その有用なアプローチとなる可能性を示していると考える．

主な否定的意見として「限られた広さのテーブル上ではロボットを小型にする，またはテーブルをより大きくする必要がある」，「ロボットの認識領域，認識イベント，振舞いの設定や表現が制限されている」があがった．後者の意見に関しては，現在のRoboTable2では単純な機能（移動と回転のみ）のロボットを使用しており，その認識領域は扇型のみ，認識イベントは右，正面，左の3つの方向のみでの物体判定に限定している．ユーザにとってこれらの設定作業は容易であるが，その反面プログラミング言語としての表現力は限られている．たとえば，同じ認識イベントに対して複数の振舞いを定義し，ユーザが与える優先度に応じて適応的に振舞いを変えられるようにすれば，ユーザはコンテンツ構築や表現をより柔軟に行えると考える．一方，表現力を高めるためには，抽象的な概念の理解や煩雑な操作をユーザに要求するシステム設計が必要になる．そのため，たとえば複数のプログラムブロック間の優先度についての理解やその決定および入力に関して，ユーザに適切な支援を与えることができるシステムの実装が求められる．よって，今後はさらなる実験を通して，システム設計と表現力に関するこのようなトレードオフについて検討を進める予定である．

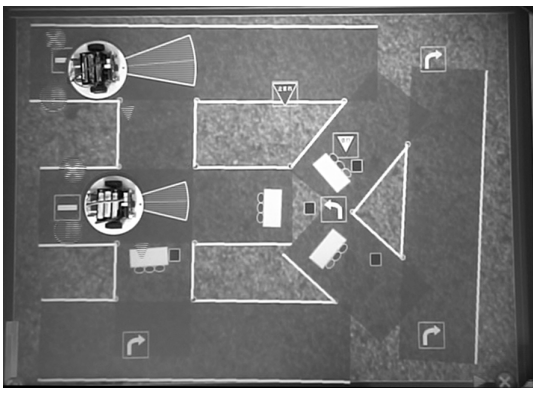


**図 11** ユーザが作成した交通シミュレータの例
**Fig. 11** An example of public transportation simulator programmed by a user.

[*1] ユーザのデザインの自由度を高めるには，配置されるアイコンの追加や変更（色や形状など），聴覚的な情報の追加などを可能にすることも検討されるべきであるが，それらについては今後の研究で行う予定である．

## 7. まとめと今後の課題

本稿では，テーブルトップ環境を用いることによる直感的なロボットプログラミング手法を提案した．さらに，グラフィカルなブロックを組み合わせるプログラミング環境との比較実験を通して，提案手法の評価を行った．その結果，2つの手法との間に有意差があり，提案手法の方が「認識イベントの設定」，「振舞いの設定」，「作成プログラムに対する満足度」などでより高い評価を得た．また，パイロットスタディを通して，ユーザ自身が容易にコンテンツ構築を行えるという目標に関する提案手法の可能性と課題について議論した．今後は，さらなる実験により，より複雑な機能を持つロボットも視野に入れて提案手法の有用性，可能性，限界をより明確にする予定である．また，小学生などプログラミング経験の少ないユーザを対象に，提案手法を用いて構築された学習コンテンツの学習支援効果についての検証を進めたいと考えている．

**謝辞** 本研究は，科学研究費基盤研究（B）「ロボットと拡張現実手法を融合したテーブルトップ協調学習支援環境の構築と評価」（研究課題番号：22300279，2010～2012年度）の支援を受けている．

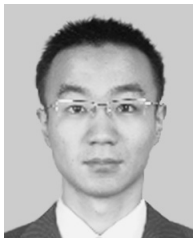
### 参考文献

[1] Yamaguchi, E., Inagaki, S., Sugimoto, M., et al.: Fostering Students' Participation in Face-to-Face Interactions and Deepening Their Understanding by Integrating Personal and Shared Spaces, *Transactions on Edutainment II*, pp.228–245 (2009).
[2] Fischer, G. and Sugimoto, M.: Supporting Self-Directed Learners and Learning Communities with Sociotechnical Environments, *Journal on Research and Practice in Technology Enhanced Learning*, Vol.1, No.1, pp.31–64 (2006).
[3] 神田崇行：コミュニケーションロボットによる学習支援，人工知能学会論文誌，Vol.23, No.2, pp.229–236 (2008).
[4] 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感的なプログラミング手法の提案，インタラクション 2011，pp.155–162 (2011).
[5] Resnick, M. et al.: Scratch: Programming for All, *Comm. ACM*, Vol.52, No.11, pp.60–67 (2009).
[6] Horn, M. and Jacob, R.: Tangible Programming in the Classroom: A Practical Approach, *Proc. CHI 2006*, pp.869–874 (2006).
[7] Gallardo, D., Julia, C. and Jorda, S.: TurTan: A Tangible Programming Language for Creative Exploration, *Proc. TABLETOP 2008*, pp.89–92 (2008).
[8] Kato, J., Sakamoto, D., Inami, M. and Igarashi, T.: Multi-touch Interface for Controlling Multiple Mobile Robots, *Proc. CHI 2009*, Boston, MA, pp.3443–3448 (2009).
[9] Seifried, T., Haller, M., Scott, S., Perteneder, F., Rendl, C., Sakamoto, D. and Inami, M.: CRISTAL: A Collaborative Home Media and Device Controller Based on a Multi-Touch Display, *Proc. ITS 2009*, Banff, Canada, pp.33–40 (2009).
[10] Micire, M., Desai, M., Courtemanche, A., Tsui, K. and Yanco, H.: Analysis of Natural Gestures for Controlling Robot Teams on Multi-touch Tabletop Surfaces, *Proc. ITS 2009*, Banff, Canada, pp.41–48 (2009).
[11] Young, J., Igarashi, T. and Sharlin, E.: Puppet Master: Designing Reactive Character Behavior by Demonstration, *Proc. SCA 2008*, Dublin, Ireland, pp.183–191 (2008).
[12] Guo, C., Young, J. and Sharlin, E.: Touch and Toys, *Proc. ACM CHI2009*, Boston, MA, pp.491–500 (2009).
[13] Tse, J., Chan, S. and Ngai, G.: An Introduction to the Multi-Modal Multi-Robot (MuMoMuRo) Control System, *Proc. IEEE SMC 2010*, Istanbul, Turkey, pp.2941–2947 (2010).
[14] Pedersen, E. and Hornbaek, K.: Tangible Bots: Interaction with Active Tangibles in Tabletop Interfaces, *Proc. ACM CHI 2011*, Vancouver, Canada, pp.2975–2984 (2011).
[15] Horn, M., Solovey, E.T., Crouser, J. and Jacob, R.: Comparing the Use of Tangible and Graphical Programming Languages for Informal Science Education, *Proc. CHI 2009*, pp.975–984 (2009).
[16] Frei, P., Su, V., Mikhak, B. and Ishii, H.: curlybot: Designing a New Class of Computational Toys, *Proc. CHI 2000*, pp.129–136 (2000).
[17] Raffle, H., Parkes, A. and Ishii, H.: Topobo: A Constructive Assembly System with Kinetic Memory, *Proc. CHI 2004*, pp.647–654 (2004).
[18] Krzywinski, A., Mi, H., Chen, W. and Sugimoto, M.: RoboTable: A Tabletop Framework for Tangible Interaction with Robots in a Mixed Reality, *Proc. ACE 2009*, pp.107–114 (2009).
[19] FTIR vs DI, available from ⟨http://iad.projects.zhdk.ch/multitouch/?p=47/⟩ (accessed 2010-11-07).
[20] Han, J.: Low-Cost Multi-Touch Sensing through Frustrated Total Internal Refl ection, *Proc. UIST 2005*, pp.115–118 (2005).
[21] Kaltenbrunner, M. and Bencina, R.: reacTIVision: A Computer-Vision Framework for Table-Based Tangible Interaction, *Proc. TEI 2007*, pp.69–74 (2007).
[22] MT4j Multitouch For Java, available from ⟨http://www.mt4j.org/mediawiki/index.php/Main_Page⟩ (accessed 2010-11-07).
[23] JBox2D Demos, available from ⟨http://www.jbox2d.org/⟩ (accessed 2010-11-07).
[24] 小野島昇吾，米 海鵬，杉本雅則，Krzywinski, A.：テーブルトップヒューマンロボットインタラクションのための Toolkit の設計と評価，情処第 72 回全国大会予稿集 (2010).

**藤田 智樹**

2009 年岡山大学工学部電気電子工学科卒業．現在，東京大学大学院工学系研究科電気系工学修士課程在学中．テーブルトップ環境におけるヒューマンロボットインタラクションに関する研究に従事．



**米 海鵬**

2008 年清華大学大学院電子工学科修了．現在，東京大学大学院工学系研究科電気系専攻博士後期課程在学中．テープルトップインタラクションおよびタンジブルユーザインタフェースに関する研究に従事．

**杉本 雅則** （正会員）

1990 年東京大学工学部航空学科（宇宙工学専修）卒業．1995 年同大学院工学系研究科博士課程修了．博士（工学）．同年より文部省大学共同利用機関学術情報センター（現国立情報学研究所）研究開発部助手．1997 年コロラド大学計算機科学科にて客員研究員．1999 年より東京大学情報基盤センター助教授．2002 年より同大学大学院新領域創成科学研究科基盤情報学専攻助教授となり，工学部電子情報工学科を兼担．2008 年より同大学大学院工学系研究科電気系工学専攻准教授となり，現在に至る．ヒューマンコンピュータインタラクション，モバイル・ユビキタスコンピューティング，音響イメージング等の研究に興味を持っている．