Vol.2011-EC-19 No.12
2011/3/26

直感的なロボットプログラミング手法を用いたゲーム構築支援環境とその評価

藤田智樹[†1]　米海鵬[†1]　杉本雅則[†1]

我々は，物理的なロボットをマルチタッチ入力が可能なテーブルトップ環境に導入することで，直感的なロボットプログラミングを行う手法の開発を進めている．本稿では，提案手法を用いることで，ゲームやシミュレーション等のアプリケーション構築を支援する環境について述べる．さらに、ユーザに作成してもらったアプリケーションの多様性やその構築プロセスの分析を通して，提案するゲーム構築支援環境についての評価を試みる．

A Game Construction Environment Using an Intuitive Robot Programming Technique and Its Evaluations

Tomoki Fujita,[†1] Haipeng Mi[†1]
and Masanori Sugimoto[†1]

We have so far been developing an intuitive robot programming technique by introducing a physical robot on a tabletop platform. In this paper we propose an environment that supports users with less programming knowledge or experiences to construct mixed reality games and simulations by using our proposed robot programming technique. We try to evaluate the proposed environment through the diversity of applications that users create and analysis of their design processes.

†1 東京大学
The University of Tokyo

1. はじめに

近年，複合現実環境において物理的なロボットを利用したアプリケーションが提案されている[1)–3)]．表現力の高いバーチャルな空間に物理的なロボットを導入した環境は，リアリティのあるゲームや，学習者の動機付けを高める学習コンテンツの構築[4)]を可能にすることが期待される．

我々は，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[5)]．そのためには物理的なロボットが導入された環境で，そ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る舞いを実現するための条件分岐や繰り返しなどの概念を理解することが学習者に要求される．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．そこで我々は，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[6)] を拡張することで，直感的にロボットプログラミングできる環境を実現した．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，条件分岐や繰り返しを含むプログラミングを行うことができる．プログラミングを普段行わない大学生を対象とした評価実験を通して提案手法の効果を検証した．

これまで，プログラミング初心者を対象とした，バーチャルなコンテンツのデザイン支援環境は多く提案されている[8)]．しかし，バーチャルな環境で振舞う物理的なロボットを含めたコンテンツのデザイン支援環境というものは，我々の知る限り存在しない．そこで本稿では，物理的なロボットの動きを利用した交通シミュレータやゲームなどのコンテンツデザインを支援する環境を構築し，パイロットスタディを行った．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の分析を行った．その結果，ロボットプログラミングに要求される機能と，デザインされるコンテンツにおけるロボットの役割に関連性があることが示唆された．また，ゲーム環境においてロボットを利用した表現の多様性を確認することができた．本稿の構成は以下の通りである．第２節では本研究の関連研究を示す．第３節では提案するコンテンツデザイン支援環境について述べる．第４節ではパイロットスタディの詳細および結果について述べる．第５節では得られたコメントとビデオの分析を基に，提案するデザイン環境について考察する．第６節では本稿の結論と今後の展開について示す．

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ロボット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．

ROBOLAB[9] は LEGO MINDSTORM で製作したロボットを操作するためのプログラミング環境である．ROBOLAB のプログラミングブロックは MINDSTORM で作成されるロボットの入出力ポートやその動作を表現しており，ロボットの振る舞いに加え，条件分岐や繰り返し処理を簡単にプログラミングすることができる．しかし，個々のブロックの意味が一意では無いため，ユーザはプログラミングブロックの意味を理解しなければ意図したとおりにロボットプログラミングすることができない．Quetzal[10] は小学校低学年の子供を対象とし，Tangible User Interface（TUI）を導入したロボット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．Horn らは，TUI を導入したロボット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[11]．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，および子どもの振る舞いの観察を基に分析を行っている．その結果，TUI をプログラミング環境に導入することで，子どもをプログラミング環境に惹きつけ，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る舞いが関連付けにくいという問題や，実体を持つブロックでは長く複雑なプログラムを作ることが難しいという問題が指摘されている．我々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ことができ，さらに作成したプログラムを仮想的なオブジェクトとして表現することができる．そのため，これらの問題を解決できると考える．Frei らの提案する curlybot[12] 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[13] 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る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．

プログラミング初心者を対象としたコンテンツデザイン支援環境の代表的なものとして，Scratch[8] が挙げられる．ユーザはペイントツールでオブジェクトを生成し，画面上に自由に配置することができる．さらに，個々のオブジェクトの挙動をグラフィカルなプログラミング環境を利用して記述することで，ゲームやシミュレーション環境などを自由にデザインすることができる．

3. コンテンツデザイン支援環境

3.1 設 計 方 針

本稿ではプログラミング初心者を対象としたコンテンツ構築支援環境を提案する．我々の提案する環境では，シミュレーションやゲームのデザイン支援に加え，デザインされた環境内でオブジェクトを認識して振る舞いを変える物理的なロボットのプログラミングも簡単に行うことができる．対象ユーザがプログラミング初心者であることから，以下の２点を設計方針としてコンテンツデザイン支援環境の構築を行った．

- 情報の入力を明確なものにする
- ユーザが混乱しうる要素を排除する

我々はこれらの設計方針を満たす環境として，先行研究で開発されたテーブルトップ環境である RoboTable を採用した．RoboTable[6] はマルチタッチ入力だけでなくテーブル上のロボットの位置や角度情報なども高い精度で取得することができる．この環境を利用することでユーザは以下の直接的な入力を行うことができる．

- 物理的なロボットを直接掴んで行う入力
- ボタンやアイコンなどに指で触れる入力
- マルチタッチによるハンドジェスチャ入力

これらの入力手法では対象に直接触れるため，情報を入力する対象がより明確なものになる．さらに，入力を行う過程でユーザは身体感覚を伴うため，自身の入力をより明確に把握することが可能となる．

我々の提案するコンテンツデザイン環境を実現するために最低限必要となる操作は，オブジェクト生成，パラメータ操作およびロボットプログラミングである．我々は RoboTable の特性を考慮して以下の操作方法を割り当てた．

- オブジェクト生成：メニューのアイコン一覧からデザイン環境へのドラッグ
- パラメータ操作：マルチタッチジェスチャによる操作
- ロボットプログラミング：物理的なロボットを用いたプログラミング

3.2 デザイン環境

ユーザは，前述した直接的な入力によりコンテンツのデザインを行うことができる．オブジェクト生成はメニューパネルからアイコンを選択し，メニューパネル外に指で触れてドラッグすることで行う．生成したオブジェクトに対し，一般的に用いられるマルチタッチジェスチャを用いることで，ユーザは移動や回転，拡大・縮小などの操作を簡単に行うことができる．生成したオブジェクトは，メニューパネル上に表示されているゴミ箱のアイコン上にドラッグすることで削除される．

3.3 提案するロボットプログラミング手法

3.3.1 プログラムモデル

本稿ではロボットプログラミングのための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る舞いの組み合わせで構成される．本稿ではこの組み合わせをプログラムブロックと呼ぶことにする．ユーザはプログラムブロックを生成し，登録するだけで簡単に条件分岐を含むプログラムを記述することのできる環境を実現することができる．プログラムブロックはRoboTable に登録され，RoboTable 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．例えば，「障害物の接近の認識（イベント）」と「右に 90 ° 回転して回避（振る舞い）」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．

3.3.2 プログラミング手法

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 RoboTable 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．認識イベントの保存には，あらかじめロボットが認識可能な領域を RoboTable 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でユーザが半径と角度を自由に調整できる．RoboTable 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．このとき，発生したイベントを保存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保存できる (**図 1** 左)．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．

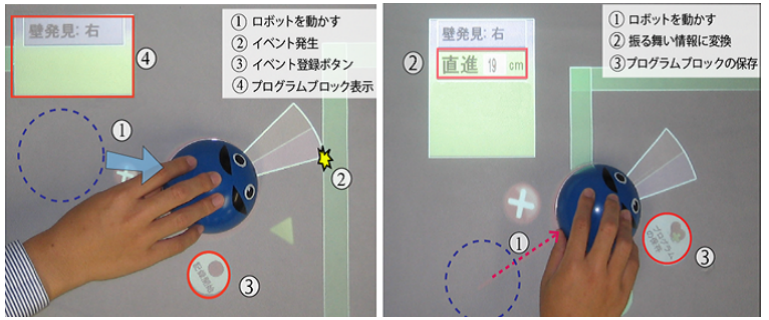

図 1 提案するロボットプログラミング手法
Fig. 1 Proposed robot programming technique

プログラムブロックのテンプレートに認識イベントが保存されると，ロボットの周囲にはそのイベントを保存した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ことにより，そのイベントに対応するロボットの振る舞い情報を作成できる．振る舞い情報は，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 RoboTable により生成される．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る舞い情報を確認する．意図した通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「振る舞い情報保存ボタン」に触れることで，作成された振る舞い情報を保存する．振る舞い情報が保存されると，RoboTable 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プログラムブロックを登録するためのボタンが表示される（図 1 右）．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 RoboTable に登録する．その後，同様にロボットを動かして登録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．

4. パイロットスタディ

4.1 目的と設定

提案したロボットプログラミング手法を用いることで，プログラミング初心者でも簡単にプログラミングできることが先行研究で確認されている[7]．しかし，コンテンツのデザイ

ン支援に関しては，ユーザの要求に十分応えることができるかを明らかにする必要がある．そこで，必要となる機能や問題点の確認を行うため，パイロットスタディにおいて，ロボットプログラミングを含むコンテンツのデザインを被験者に行ってもらった．実験で扱ったデザイン環境は以下の２種類である．

- 実験１：交通シミュレータ
- 実験２：ゲーム環境

実験中は被験者の質問に適宜答え，サポートを行うとともに，被験者の様子をビデオ撮影した．実験後は以下の設問を含むアンケートに回答してもらった．それぞれの回答は記述方式である．

- 設計時に不足していると感じた機能
- 設計時に負担を感じた点
- 他に気づいたこと

4.2 実　験　1

パイロットスタディの題材として，コンテンツを構築するためのツールがある程度限定されている交通シミュレータを採用した．実験で扱う交通シミュレータは，交通事故や渋滞などの事象発生プロセスを観察する用途を想定して設計している．

交通環境を表現するための一般的な要素として，道路や交通標識，信号機などのオブジェクトが挙げられる．実験ではこれらの基本的なオブジェクトに加え，プログラミングによって個々のロボットが異なる経路を移動可能にするための方向指示オブジェクトを実装した．さらに，個々のドライバ（ロボット）ごとに，注意深さ（認識領域）と移動速度を調整可能にした．注意深さは認識領域のサイズをジェスチャで調整することにより設定可能である．また，移動速度はロボットの周囲に速度調整アイコンを配置し，3 段階（早い，普通，遅い）の定性的な速度設定を行えるようにした．

4.3 実験１結果

研究室のメンバー 3 名（男性 3 名，平均 22.6 歳）を被験者とし，自由に交通シミュレータを構築してもらった．実験を通して，被験者が構築した様々な交通モデル上でロボットが移動するのを観察することができた．現状では単純に交通規則に従って移動するロボットを観察することしかできないが，騒音や交通量などの数理モデルを提供し可視化することで，本格的な交通シミュレータとしての展開も期待できる．さらに，実験で得られたアンケートから，以下の拡張機能が要求されることが明らかとなった．

- 移動経路の指定

  被験者のコメントから，交通シミュレータのデザインにはロボットの移動経路を指定する機能が要求されることが分かった．我々の提案したロボットプログラミングでは，道路の白線に対する認識イベントを設定することで，道路の中を走行させることができる．しかし，道路の白線を確認しながら走行するロボットは，交通シミュレータとして扱うには動作が不自然であると指摘されている．実際にプログラム実行中の環境を確認すると，認識領域に白線が衝突するたびロボットが回転するため，スムーズに道路を走行できていないことが確認できた．このことから，移動経路の指定機能が必要になると考えた．厳密に移動経路を指定できるようにすることで，道路上をスムーズに移動するロボットを表現できる．また，道路内を走行するためのプログラミングの手間も省くことができ，ユーザへの負担を抑えることが可能となる．これを実現するための方法として，指で描いたライン上をロボットに走行させるためにライントレース機能の実装などが考えられる．

- ロボットの振る舞いの拡張

  我々が交通シミュレータのデザイン環境を設計する過程において，ロボットの振る舞いの拡張が必要となることも確認されている．交通シミュレータでは，交通標識を発見したロボットが交通標識の指示に従って一時的にスピードを落としたり停止するなど，受動的に振る舞うことが要求される．我々の開発したプログラミング環境では，自由にこれらの挙動をロボットにプログラミングすることができない．そのため，新たに受動的な振る舞いを登録できるよう，受動動作を引き起こすオブジェクトを設計する，登録できる振る舞いの拡張を行うなどのアプローチによりプログラミング環境を拡張する必要がある．

4.4 実　験　2

交通シミュレータ同様，ゲーム構築支援環境の設計を行った．ここで扱ったゲームでは，ロボットがオブジェクトに衝突することで，その展開に動的な影響を与えることができる．この点が交通シミュレータとは異なる点であり，ロボットプログラミングをロボットの動きを表現する以外の用途で扱うことも可能となる．また，プレーヤーが直接テーブルに触れることによってもゲームの展開に影響を与えることができるという点も交通シミュレータとは異なる．我々はピンポンやサッカーなどの球技ゲームを参考に以下の要素を含む簡単なプレーヤー参加型ゲーム構築支援環境を実装し，被験者に提供した．

- ボール
- ゴール（プレイヤ１，プレイヤ２）

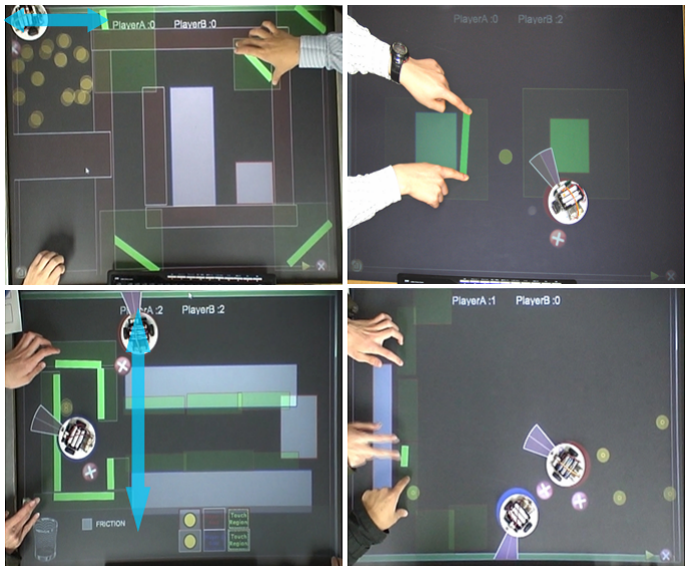

図 2 被験者のデザインしたゲーム環境
Fig. 2 Game environments designed by subjects

- パッド作成領域
- 障害物

提案する支援環境では，一方のゴールにボールが一定回数進入するとゲーム終了の画面が表示される．ゴールは複数用意され，複数のユーザまたはチームの対戦型ゲームを作ることができるようにした．

4.5 実験２結果

研究室のメンバー 5 名（男性 5 名，平均 23.8 歳）を被験者とし，自由にゲーム環境を構築してもらった．被験者が製作したゲームの例を**図 2** に示す．製作されたゲームからロボットプログラミングを用いることで，以下に示す様々なゲームオブジェクトの表現を確認することができた．

- ゲーム開始のトリガを行うロボット（図 2 左上）
  静的に配置されたボールに対し，動的に振る舞うロボットを意図的に衝突させることで，ゲームを開始するためのトリガとしてロボットを活用した例．
- 障害物としてのロボット（図 2 右上）
  単純な前後移動するようプログラミングしたロボットにより，ダイナミックな障害物を実現した例．さらに，ロボットに認識イベントを登録することで，適切なタイミングで障害物として振舞う例．
- ゲームのチームメイトとしてのロボット（図 2 左下）
  「ゴールを認識しながら移動」，あるいは「ボールを発見した時ははじき返す」ようにプログラミングされたロボットをゴール付近に配置することで，一方のプレイヤのチームメイトとして活用した例．
- ゲームを複雑化するためのロボット（図 2 右下）
  周囲のオブジェクトを回避しながら前後左右に動き回る複数のロボットをゲームに投入することによって，ユーザの推測できない複雑な衝突現象を生み出しゲームを複雑化するためにロボットを活用した例．

さらに，被験者から得られたアンケートから以下の機能に対する要求が明らかとなった．

- ロボットの移動区域を制限する機能
  一部の被験者はロボットを特定の区域に留まらせたいという要求を述べていた．これはロボットを一方のプレーヤー側に留まらせることで，プレーヤーのサポートを行うロボットをデザインしたいという要求によるものだと考えられる．移動区域の制限は，現状のロボットプログラミングでも表現することが可能であるが，オブジェクト配置や調整，プログラミングなど非常に手間がかかってしまう．そこで，新たに移動区域を指定するためのオブジェクトを定義し，これを提供することでデザインのサポートを行う必要がある．
- 特定のオブジェクトを追尾する機能
  被験者から得られたコメントのひとつに，ロボットにボールを追跡させて欲しいというものがあった．これは，ロボットに接近したボールを弾き飛ばすようにプログラミングしても，ロボットがうまくボールにぶつからなかったことが被験者にストレスを与えたものだと推測できる．このことから，ボールの追跡を含めた，より能動的な振る舞いを表現できるようにする必要があると考えられる．

5. 議　　論

実験では交通シミュレータとゲームのデザイン環境を提供し，それぞれの環境でどのような機能が要求されるか確認した．その結果，どちらの環境においても，ロボットの行動制限

と振る舞いの拡張が要求されていることが確認できた．交通シミュレータではロボットの移動経路の厳密な制限に加え，交通標識に応じてスピード低下や一時停止を行うなどの受動的な振る舞いを自由にプログラミングできる環境が要求されている．一方のゲーム環境では，ロボットの行動範囲の制限に加え，ボールを追跡して衝突するようなより能動的な振る舞いが要求されている．これらの機能をデザイン支援環境に実装すればシミュレータやゲームなどのデザインに使用できる，より汎用性の高いデザイン環境を構築できると考える．しかし，多様な機能を実装したデザイン環境は，ユーザの混乱を引き起こす可能性も考えられる．そこで我々は実験の題材として扱った二つの環境から，ロボットの役割と，それに対する要求の関連性について考察する．

交通シミュレータを含むシミュレーション環境全般において，ロボットはその挙動がユーザの注目の対象となる．そのため，個々のロボットはユーザの観察対象として理想的に振る舞う必要がある．その結果，周囲のオブジェクトに対する多様な反応や，移動経路を指定してスムーズに移動させたいという要求が生じたのだと考えられる．一方で，ゲームという題材においてユーザが注目する対象はロボットではなく，ゲーム環境である．ロボットは障害物やキャラクタなどの様々なゲームの要素として振る舞うことで，ゲームへの没入感を高めることが要求される．これにより，環境に適切に影響を与えるため能動的に振る舞わせたい，影響を与える範囲を指定したいなどの要求が生じたと考えられる．以上から，注目する対象によってロボットプログラミングに要求される機能も変化すると推察できる．これらの要求を満たしつつ，かつ直感的なロボットプログラミング手法によってコンテンツ構築を支援できる環境を実現することが今後の課題となる．

6. おわりに

本稿では物理的なロボットを導入したコンテンツデザイン支援環境を提案した．直感的なロボットプログラミング手法を導入し，プログラミング初心者でも扱うことのできるデザイン環境の設計を行った．設計した環境のパイロットスタディを行った結果，ロボットプログラミングに要求される機能と，ロボットの用途には関連性があることが示唆された．また，被験者がデザインしたコンテンツから，提案したロボットプログラミングによる表現の多様性も確認することができた．今後はロボットプログラミングに要求される機能の拡張を行うとともに，ユーザの要求を満たすコンテンツ構築支援環境の実現に向けて取り組みたい．

参考文献

1) Kojima, M., Sugimoto, M., Nakamura, A., Tomita, M., Nii, H. and Inami, M. (2006). Augmented Game Environment with Small Vehicles. In Proc. of IEEE TABLETOP 2006. 3-8. Washington, DC.
2) Leitner, J., Haller, M., Yun, K., Woo, W., Sugimoto, M. and Inami, M. (2008). IncreTable: A Mixed Reality Tabletop Game Experience. In Proc. of ACM ACE 2008. 9-16. Yokohama, Japan.
3) Calife, D., Bernardes, J. and Tori, R. (2009). Robot Arena: An augmented reality platform for game development. In Proc. of ACM Computers in Entertainmnet (CIE). 7(1).
4) Sugimoto, M. (2011). A Mobile Mixed Reality Environment for Children's Storytelling using a Handheld Projector and a Robot. In Proc. of IEEE Transactions on Learning Technologies (to appear).
5) Fischer, G., Sugimoto, M. (2006). Supporting Self-Directed Learners and Learning Communitieswith Sociotechnical Environments. Journal on Research and Practice in Technology Enhanced Learning. 1(1). 31-64.
6) Krzywinski, A., Mi, H., Chen, W. and Sugimoto, M. (2009). RoboTable: A Tabletop Framework for Tangible Interaction with Robots in a Mixed Reality. In Proc. of ACE 2009. 107-114.
7) 藤田, 米, 杉本. (2011). 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. 東京. (to appear)
8) Resnick, M., et al. (2009). Scratch: Programming for All. In Proc. of ACM Communications of the ACM. 52(11). 60-67.
9) レゴマインドストーム公式サイト. Retrieved November 7, 2010, from http://www.legoeducation.jp/mindstorms/index.html.
10) Horn, M. and Jacob, R. (2006). Tangible Programming in the Classroom: A Practical Approach. In Proc. of CHI 2006. 869-874.
11) Horn, M., Solovey , E.T., Crouser, J., Jacob, R. (2009). Comparing the Use of Tangible and Graphical Programming Languages for Informal Science Education. In Proc. of CHI 2009. 975-984.
12) Frei, P., Su, V., Mikhak, B. and Ishii, H. (2000). curlybot: Designing a New Class of Computational Toys. In Proc. of CHI 2000. 129-136.
13) Raffle, H., Parkes, A., Ishii, H. (2004). Topobo: A Constructive Assembly System with Kinetic Memory. In Proc. of CHI 2004 647-654.