プログラミング学習のためのロボット制御システムの開発

益 崎 真 治*・猪 股 俊 光**

A Development of Robots Control Systems for Programming Learning

Shinji Masuzaki* and Toshimitsu Inomata**

1 はじめに

平成15年度より高等学校において「情報」という新教科がスタートする[1]。教科「情報」の中には「情報 A」,「情報 B」,「情報 C」の３科目があるが，その中の「情報 B」ではコンピュータの仕組みやコンピュータを活用した問題解決について学ぶことになっている。特定のプログラミング言語（以下，言語とかく）を用いてプログラム作りを行うところまでは想定されていないようであるが，科目の授業をつうじて，自らプログラムを作ってみたいと思いはじめる生徒は少なくないはずである。そのとき，初心者がくじけることなく続けられるようなプログラミング学習ツールや学習法が必要となる。本研究の目的は，そのための１つの学習ツールを開発することである。

筆者らは，主に大学・高専生を対象としたプログラミング学習のための処理系[10,11] の開発と，それを WWW（World Wide Web）環境で利用したプログラミングの授業[7,8] の研究を行ってきている。そのための言語としては，リスト処理言語の１つである Scheme[12,13] が適当であることを確認した。しかし，大学・高専生といった，自分の進む進路（専門分野）を決めた学生に対しては，このような Scheme を用いて学習を行うことも可能であろうが，将来の進路がまだ定まらない高校生に対しては，学習の持続性という観点からはこのやり方が最適とは思われない。そこで，キーボードから文字や数字を入力しながらのプログラミングではなくて，実際に物が動く姿を見たり，物の動きを直接的に指示したり，動くものを触ったりしながらプログラミングが学べられるならば，興味もわくであろうし，プログラミングの楽しさを実感でき，持続できるだろうと考え，タートルグラフィックスとロボットを取り入れることとした。

タートルグラフィックスとロボットを組み合わせるアイディアは，MIT の S. Papert 氏によるものである[2]。タートルグラフィックスは S. Papert が Logo というプログラミング言語の中で取り入れたグラフィックス描画法であり，コンピュータの画面上をタートル（亀）が命令にしたがって歩きまわるときの軌跡が図形となる。さらに，S. Papert は，画面上のみならず，床の上でも動きまわれるようなタートルロボットも実現している。本研究では，この S. Papert のアイデアをヒントとし，プログラミング学習のためのロボット制御システムを筆者らが作成している言語処理系上で実現した。

以下，２節ではタートルグラフィックスとロボットの関係について論じ，３節では対象とするロボットについて述べる。また，４節では開発してきた言語処理系について述べる。そして，５節で新しく開発したロボット制御システムについて，６節ではそれの適用例を述べる。最後に，７節でまとめを行う。

2 プログラミング学習とロボット

１節でふれたように，S. Papert はタートルグラフィックを用いながら子どもたちにプログラミングを教える試みを続けており，彼は，「プログラムすることは，コンピュータと人間であるユーザーとの両方が理解できる言葉で交流し合うことである」と述べている[2]。両方が理解できる言葉とはプログラミング言語のことであり，それには両者が正確に交流をはかるための考慮がされているべきである。そのような候補の１つが Logo という言語である。コンピュータと子どもはともに Logo を話しながら交流するが，子どもはコンピュータと直接話をするのではなく，画面上に現れているタートルと対話することになる。タートルに「前へ10歩進め」,「右を向け」などと伝えると，タートルはそのとおりに画面上を動き出す。そのとき歩いた軌跡が図形になり，それを利用して図形を描写することをタートルグラフィックスと呼んでいる。

タートルグラフィックスを利用した場合，プログラムをすることは，タートルにどう動いたらよいのかということを教えることになり，“ 限られた言葉を使って正確に

＊電子機械工学科
＊＊岩手県立大学ソフトウェア情報学部

平成13年９月28日受理

動き方を教える”にはどうしたらよいのかということを子どもは考えることになる。“限られた言葉を使って正確に動き方を教える”ということが，まさにプログラム作り（プログラミング）である。タートルグラフィックスであれば，間違った動きを教えるとそれが直ちに画面に反映されるため，自分の間違いにすぐ気がつき，そのことを通じて，正しく教えるにはどうしたらよいのかということを自分で考え，解決策をみつけることができるようになる。

さらに，画面上のタートルだけとの交流にとどまらず，床や机の上を実際に動きまわるタートルとの交流もはかれれば，より効果が増すと考えられる。つまり，画面上のタートルに教えた内容（プログラム）を，タートルと同じ言葉を理解するロボットにも教えることができれば，そのロボットもまた同じように動くことができる。そのとき，子どもはロボットが動く様子を観察したり，モーターの音を聞くことで，コンピュータの世界（理論的・理想的な世界）と現実の世界との違いを知るきっかけにもなるであろう。

このような考えにもとづきながら，図１に示すようなことが実現されるロボット制御システムを開発することにした。まず，画面上でタートルと会話しながら，動き方を教える。その内容をプログラムとして保存し，それをロボットが理解できる言葉に翻訳したのち，ロボットへ転送する。その後，ロボットは画面上のタートルと同じような動きをする。これは，見方をかえれば，ロボットの動きのシュミレーションをコンピュータ上で行っているともいえる。

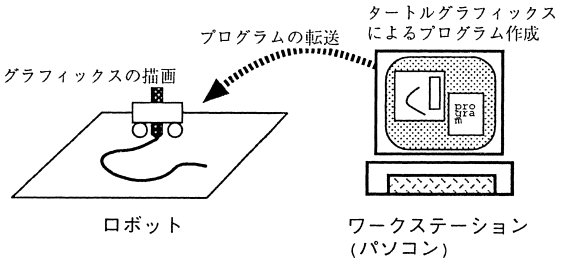


図1　タートルグラフィックスとロボット

3　対象とするロボット

3．1　ロボットの仕様

対象とするロボットは LEGO 社が製造したロボットシステム LEGO MINDSTORMS Robotics Invention System[1]であり，RCX（Robotics Command System）とよばれるロボット本体（CPU とメモリが内蔵されている）とロボットプログラム作成用のアプリケーションからなる[3,4]。RCX の形状はレゴブロックに対応してできており，モーターや光センサー，タッチセンサーといった各周辺装置もまたブロックとして本体に取付けることができる。ロボットプログラム作成用のアプリケーションを使って作成したプログラムを RCX へ転送（赤外線による）したのち，スタートスイッチを押すとロボットは単独で動作する。RCX のハードウェア仕様を表１に示す[5]。

表1　RCX のハードウェア仕様（［5］より）

項　　目	仕　　様
CPU	8bit マイクロプロセッサ H8／3292（16MHz）
メモリ	ROM 16KB，RAM 32KB
インタフェース	入力ポート×3 出力ポート×3 赤外線通信ポート×1 LCD ディスプレイ×1
電　源	単3形乾電池6本

入力ポートには，光センサー（赤色 LED の反射光をフォトトランジスタによって0～100の段階に分けて検出する）とタッチセンサー（導電ゴムのスイッチのオンオフの検出）を接続できる。一方，出力ポートには，モーター（PWM 制御で制御階調数は0～8）を接続することができる。これらの周辺装置を組み合わせることにより，自立走行が可能なロボットを作ることができる。図２にこれら周辺装置やレゴブロックを組み合わせてできたロボットの例を示す。

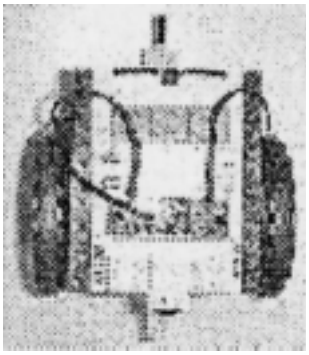


図2　ロボットの作成例

ロボットプログラムは，入出力ポートに接続された各周辺装置の動作を特定するコマンドの集まりであり，このプログラムにしたがってロボットは動作する。今回は，このロボットプログラムの作成を，Mindstorms 対応のオープンソースな OS の１つである。legOS[6,9] を用いて行うこととした。なぜならば，製品版アプリケーションでは限られた範囲のプログラムしか作成できない，特定

[1]LEGO MINDSTORMS は LEGO グループの登録商標である。その他，本文中の会社名，商品名，製品名は，一般に各社の商標もしくは登録商標である。

の OS（Windows）のもとでしか使用できない，他のソフトウェアとのデータの受け渡しが困難である，などの理由からであるとくに教育目的のためのソフトウェアが，特定の OS のもとでしか動かなかったり，ソースコードが公開されていないことは，利用に際して大きな障害となる。

3．2 legOS

legOS は，Markus L. Noga 氏によって開発された Mindstorms 用のオペレーティングシステム（ファームウェア）である。ソースプログラムが公開されているため[2]，OS の勉強になるとともに，独自の機能の追加も可能である。また，クロスコンパイラを用意することにより，Linux，Solaris，Windows といった OS 上で利用可能である。このように，教育目的として適しているため legOS を用いることとした。legOS の特徴は，次のとおりである[6]。

- プリエンプティブ可能なマルチタスク機能
- 省電力機能
- 動的メモリ管理機能
- POSIX（Portable Operating System Interface）準拠のセマフォ（プロセス間通信）対応
- 周辺装置（センサー，モーターなど）制御機能

legOS の構成は，図3の左側のように，カーネル（タスク管理，メモリ管理，周辺装置インターフェースなど）とロボットプログラムからなる。すべてC言語で記述されており，ユーザーは目的にあったロボットプログラムだけを作成する。カーネルとロボットプログラムは API（Application Program Interface）によって結ばれている。すなわち，ロボットプログラム中では，さまざまな API が呼び出されている。

このようにC言語で記述された legOS は，図3の中央のクロスコンパイラを用いて，H8／3292用のファームウェアに翻訳することにより，RCX 上で実行される。legOS の開発環境は，H8000シリーズ向のクロスコンパイラが利用できるものであればよい。そこで，今回は，Linux および Solaris の開発環境を用いた。

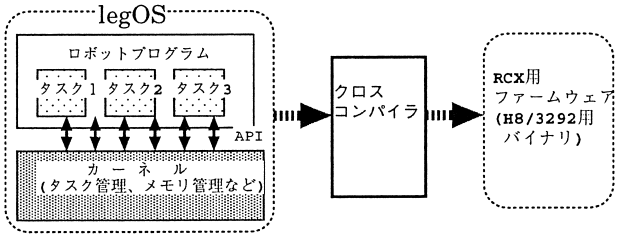


図3　legOS の構成

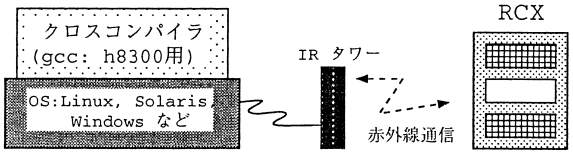


図4　legOS 開発環境と RCX

図4のように，作成した legOS（ロボットプログラム＋カーネル）をクロスコンパイラを用いてコンパイルし，そのデータを IR タワー（赤外線信号通信装置）によって，RCX にダウンロードする。

4　記号処理言語処理系

4．1　言語処理系の概要

1993年より開発をはじめた記号処理言語処理系 SIST-Scheme[10,11] は，記号処理向きの言語の1つである Scheme[12,13] のインタープリターである。当初は，記号処理教育のための言語処理系として，つぎの要求や条件を満たすよう開発を行った[10]。

a）無償であること。

b）ワークステーションやパソコンの区別なく稼動できること。

c）ソースコードが公開されていること。

d）OS に固有のライブラリやアセンブラは用いない。

e）必要最小限の言語仕様を満たすものとする。

f）日本語が扱えること。

開発した処理系（SIST-Scheme Vor.1.0）は，整数以外のデータ型，遅延評価といった Scheme の言語仕様[12] が実装されていないものの，Windows，MacOS，Solaris，Linux といった OS のもとで動作可能である。この処理系のもとで，数式処理，自然言語によるコンピュータとの対話，日本語文章の校正，演繹的推論などといった記号処理の分野のさまざまなテーマに関するプログラムが講義や卒業研究をつうじて作成された[10,13]。

さらに，1995年以降には，つぎの機能を追加した SIST-Scheme Vor.2.0を作成した。この追加機能を用いて弓削丸の計測制御システムを実現した[11]。

[2]配布条件は MPL（Mozilla Public License）による。

g）TCP/IP によるネットワーク通信機能

4．2　タートルグラフィックス機能

今回，タートルグラフィックスによるプログラミング学習を目指し，つぎの機能の追加を試みた。

h）タートルグラフィックスによる描画機能

i）RCX へのファームウェア（legOS）転送機能

タートルグラフィックス用の画面として，図5のようなウィンドウを設計した。

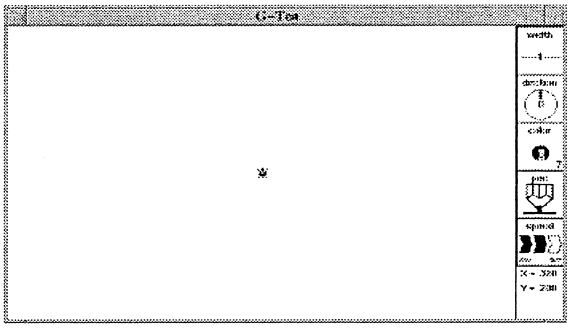
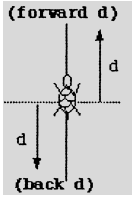


図5　ウィンドウの初期画面

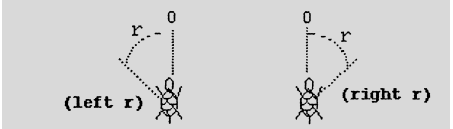
画面中央に位置するのがタートルであり，大きさ640×400ドットの範囲を移動する。右に並んでいるマス目は上から順に，ペンの太さ，頭の向き，ペン色・背景色，ペンの状態，移動速度，現在の座標位置を表す。

座標は，図6（a）のように左上を原点とし，x 軸は右方向へ，y 軸は下方向へ伸ている。中央の座標（320, 200）は，タートルの初期状態での位置である。タートルは，図6（b），(c）のように前進（forward），後進（back）左右方向回転（left，right）という動きができる。

(a)　タートルの座標



(b)　前進・後退



(c)　方向回転

図6　タートルの座標と移動

たとえば，一辺が100ドットの正方形を描くには，タートルに対して「100ドット前進し（forward 100），90度向を変える（right 90）」という指示を4回繰り返せばよい。この手順に square という名前を付けるためには，Scheme の場合はつぎのようにする。この手続きを用いた描画例を図7に示す。

```
(define (square)
    (forward 100) (right 90)
    (forward 100) (right 90)
    (forward 100) (right 90)
    (forward 100) (right 90))
```

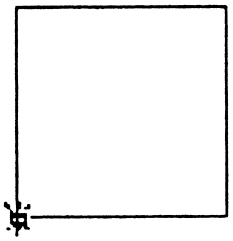


図7　一辺が100ドットの四角形の描画

タートルグラフィックスによる描画例を図8に示す。

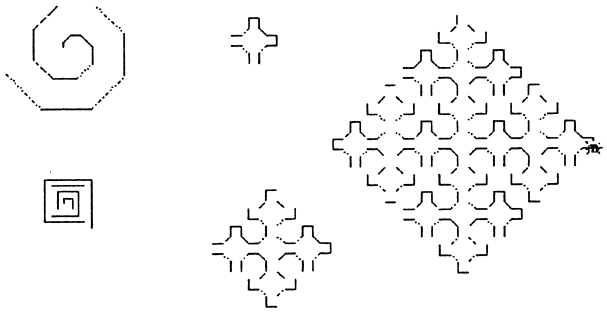


図8　タートルグラフィックスによる描画例

表2にタートルグラフィックス用の主な手続きの一覧を示す。

表2　タートルグラフィックス用の手続き

手 続 き	動　　作
(jump x y)	座標（x, y）へ移動
(home)	座標（320, 200）へ移動
(init)	画面を初期状態に戻す
(penup)	描画可能状態にする
(pendown)	描画不可能状態にする
(fast)	移動速度を速くする
(slow)	移動速度を遅くする
(left r)	左回りに r 度向きをかえる
(right r)	右回りに r 度向きをかえる
(north)	0度（真上）を向く
(forward d)	d ドット分、前方へ移動
(back d)	d ドット分、後方へ移動

5 ロボット制御システム

5．1 構成要素

図9にロボット制御システムの構成を示す。同図中（a）と（b）が，今回新たに SIST-Scheme に追加した部分であり，（c）と（d）は既存のプログラム（（d）は legOS に含まれているコマンド）である。

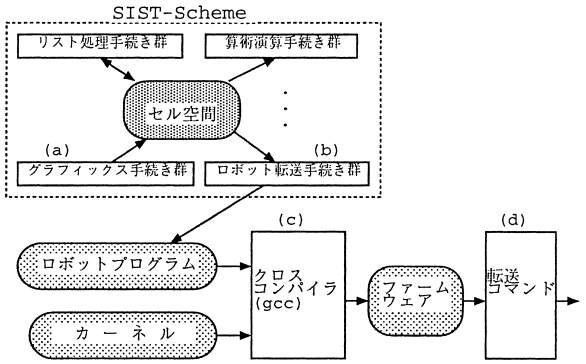


図9 ロボット制御システムの構成

ユーザーは（a）グラフィックス用の手続き群を用いてロボットの動作を特定する Scheme のプログラムを作成する。つぎに，（b）のロボット転送用手続きを実行すると，まず，その Scheme プログラムをもとに，C 言語によるロボットプログラムが生成される。そして，（c）のクロスコンパイラによってファームウェアが生成され，最後に（d）によって RCX へファームウェアが転送される。

5．2 実 現 法

SIST-Scheme のもとで作成された手続き *proc* からロボットプログラム（ファイル名 *file*）を生成するための手続きはつぎの lego である。

（lego *proc file*）

たとえば，図7のような動きをロボットにさせたい場合を考える。生成するロボットプログラムの名前を test.c にする場合には，つぎのようにする。

（lego square "test.c"）

このとき，Scheme の手続きからロボットプログラムへの変換は次のようにして行われる。Scheme の1つの手続きを1つの C 言語の関数に対応させる。手続き中で呼び出されている各タートルグラフィックス用の手続きは，該当する API に対応させる。また，逐次処理や繰り返し処理は，それぞれ該当する C 言語の制御構文に対応させる。たとえば，つぎのように対応させる。

Scheme の手続き	対応する C 言語の文
(forward 100)	forward(100)；
(right 90)	right(90)；
(penup)	penup( )；
(repeat 8 square)	{ int loop； for(loop＝1； loop＜＝8 ＋＋loop){ square( )； } }

なお，ロボットプログラムに変換できるタートルグラフィックス用の手続きは，表2中の forward，back，right，left，penup，pendown，north に限られており，センサー（光，タッチセンサー）に対応した手続きはまだ実現されていない。

6 開発したシステムの適用例

開発したシステムを，高校生を対象とした1999年度～2001年度の岩手県立大学ソフトウェア情報学部のオープンスクールにおいて教材として使用した。そこでは，ワークステーションの基本的な操作から始めて，タートルグラフィックスによる描画の仕方，ロボットへのプログラムの転送といった一連の作業を練習課題を盛り込みながら行った。そこでの練習課題の内容は，次のとおりである。

- 四角形（一辺が20ドット）の描画
- 自分のイニシャル1文字の描画
- 四角形描写手続きのロボットへの転送
- イニシャル描画手続きのロボットへの転送

四角形の描画手続き square をロボットへ転送する際に，生成されたロボットプログラムは次の通りである。ここで，mindstorms-02x.h は200行ほどのヘッダファイルであり，そこでは各種の宣言・定義がなされている。

```
#include "mindstorms-02x.h"
static int
square(void)
{
    forward(20)；
    right(90)；
    forward(20)；
    right(90)；
```

```
    forward(20);
    right(90);
    forward(20);
    right(90);
    return 1;
}
int main (void)
{
    square( );
    return 0;
}
```

このロボットプログラムを含む legOS から生成されたファームウェアをペンを付けた図2のロボットへ転送し，紙の上に四角を描かせた。

オープンスクールに参加した高校生の感想は，たのしかったというものが多く，タートルグラフィックスとロボットの組み合せは，プログラミング学習に対して効果があると思われる。

7 おわりに

タートルグラフィックスとロボット制御を組み合わせたプログラミング学習を実現するために，SIST-Scheme へ，タートルグラフィックスによる描画機能と RCX へのファームウェア（legOS）転送機能の追加を試みた。それと既存のクロスコンパイラなどからなるロボット制御システムを作り，高校生を対象としたオープンスクールの教材として使用し，所期の動作が得られることを確認した。

これからの課題としてつぎのものがあげられる。

- 光センサーやタッチセンサーに対応した手続きを SIST-Scheme に組み込むこと。
- タートルグラフィックスとロボット制御を組み合わせたプログラミング学習の効果を定量的に評価すること。

謝　辞

岩手県立大学ソフトウェア情報学部のオープンスクールでの開発したシステムの適用に際しては，同学部曽我研究室の曽我教授をはじめ，新井助手，研究室所属の学生の方々に協力していただいた。ここに記して謝意を表す。

参考文献

［1］文部省，“高等学校学習指導要領解説　情報編”，開隆堂出版，2000.

［2］Seymour Papert，“新装版　マインドストーム”，未来社，1995，（“MINDSTORMS”，Basic Books, Inv，1980）.

［3］The LEGO Groupe, “LEGO MINDSTORMS Robotics Invention System 1.5 USER GUIDE”, 1999.

［4］http : //mindstorms. lego. com/

［5］古川剛（編），“LEGO MINDSTORMS パーフェクトガイド”翔泳社，1999.

［6］Markus L. Noga, http : //www. noga. de/ legOS/

［7］猪股俊光，“学内 LAN の教育利用に関する考察”静岡理工科大学紀要，Vol－6，pp. 19－33，1997.

［8］益崎真治，猪股俊光，“WWW 環境を用いたプログラミング教育に関する考察”弓削商船高等専門学校紀要，Vol－21，pp. 25－30，1999.

［9］衛藤仁郎，http : //www. ylw. mmtr.or.jp/ arcadia/ legos/ index. html，1999

[10] 猪股俊光，“記号処理教育用言語処理系の開発と応用”静岡理工科大学紀要，Vol－4，pp. 25－42, 1995.

[11] 益崎真治，猪股俊光.“記号処理教育用言語 Scheme の開発と弓削丸への応用”弓削商船高等専門学校紀要，Vol－20，pp. 83－88，1998.

[12] Richard Kelsey, William Clinger, and Jonathan Rees (Editors), “Revised[5] Report on the Algorithmic Language Scheme”,http : //www. cs. indiana. edu/ schemerepository/ doc. standards. html

[13] 猪股俊光，益崎真治，“Scheme による記号処理入門”森北出版，1994.