ロボティクスを題材にした実習型授業の総括

松本成司 (信州大学), 鈴木治郎 (信州大学)
matsu@johnen.shinshu-u.ac.jp, szkjiro@shinshu-u.ac.jp

概要

LEGO 社のマインドストームという組立・プログラミング可能なロボットキットを活用して，大学初年次の教養課程の学生を対象に 5 年間，実習型授業を実施してきた。この授業の総括を通じて，情報教育からものづくり教育まで幅広く現在の学生が抱えている課題を総括し，そこにおける課題の多くは高校までで体験するほうが学習効果が多く期待できるものであることを指摘する。

1 はじめに

コンピュータはその万能性ゆえなのか，それが人間の設計したものであり，その意図にしたがって動作する装置であるという本質的部分が忘れられかけているのではないだろうか。そうした懸念を最近のコンピュータシステムに関わる事件報道，たとえば東証の取引システムの不手際に関する論評を見ても強く感じる。だからこそ，人間がコンピュータ上にモノを設計する際の責任を取り戻すには，モノの仕組みを理解した上でそれを使うという，かつて当たり前だったことを教育の場にも取り戻す問題として提起したい。

現代の社会像をこのように捉えてみると，従来の情報教育では不可欠な要素であったプログラミングやアルゴリズムの学習は，コンピュータを正しく理解するための近道にならないだろうか。ただし近道とみなすためには，こうした学習行為と実際に世の中にある複雑なシステムとの距離を埋めなくてはならない。

この目的において本件で扱うロボット制御キットを扱うプログラミング実習は，コンピュータの仕組みの理解と同時に，身近にあふれかえるコンピュータ制御製品を理解する上でも適切な教育的課題であると考える。

本報告では，一般教養科目および情報教育の一環として大学初年度の学生を対象に 5 年間にわたって実施したロボティクスの授業事例を紹介する。この実習はプログラミングの進度に対応して種々のロボットを製作していくことで，大学 1 年次の半期 (90 分× 15 回) の授業時間で自然とプログラミングの基礎学習ができるように構成されているが，実習で実際に採用した課題の紹介を通じて，本来なら高校や中学においてこそ同等の学習が経験されるべきであることを訴えたい。

2 プログラミング実習としての特徴

■ロボットは興味をもちやすい対象である　コンピュータだけを使ってプログラミングの学習を行うことはもちろん可能であるが，プログラミングのことをコンピュータの専門家が行う抽象的な作業と感じている学生も少なくなく敬遠されやすい。しかし近年の話題性もあり，ロボットに興味を持っている学生は比較的多い。だから教材としてロボットを採用し，実際に自作のロボットを自作のプログラムで動かすことはプログラミングに対する興味にもつながりやすい。

■プログラミングに具体性を与える　手を動かしてロボットを製作することは，コンピュータ上だけの抽象的とも捉えられがちなプログラミングをより具体的なものとして実感できることに結びつく。同じロボットであってもプログラムを変更すればさまざまな動作をさせることが可能であり，機械的な工夫とプログラムの連携も必然的に重要になってくる。もちろんプログラムに間違いがあった場合にロボットは思い通りに動いてくれない。このようにロボットという『実体』を動かすことによればプログラミングをより身近なものとして学習できるだけでなく，広い意味での『バグ取り』という試行錯誤の経験を積むことができて総合的な問題解決能力の養成に役立つと期待できる。さらにチームでのロボット製作は，学校で習う科目以外のさまざまな『遊び』を多く体験してこなかった (世代の) 学生にとって，貴重な創作的共同作業の体験にもなる。

3 ロボット教材

3.1 RIS について

近年になって二足歩行ロボットを含む多種多様なロボット製作キットが市販されるようになってきたが，LEGO 社から 1998 年に発売された MindStorms Robotics Invention System (RIS)[1] とよばれるキットは，組み立て・プログラミング可能で比較的安価なことからロボティクスの入門的な学習に適している。RIS には LEGO 社の従来部品であるブロックやモータ，ギヤ，タイヤ，プーリなど 700 個以上の部品に加え 8bit CPU を積んだ RCX と呼ばれるコントローラと光センサ，タッチセンサなどがキットに含まれる。モータやセンサの数はそれほど多くないものの[*1]，これらの部品を組み合わせるパターンは事実上無限であり，オリジナ

[*1] モータは 3 個，センサは合計 3 個まで使用できる。

ル・ロボットを容易に製作することができる。

3.2 LEGO マインドストーム用の開発環境

Mindstorm 用のプログラミング環境としては，MS Windows 用の開発ソフトがキットに標準で添付されている。しかしそれ以外でも C, C++, Java, Perl, Tcl, Forth などオープンソースで配付されているさまざまな言語を使用することができ，これらは Linux や Mac などの他の OS 上でも動作する[*2]

本事例では Dave Baum 氏によって開発され Mozilla Public Licence[4] で配付されている NQC (Not Quite C)[5, 6] という言語[*3]を採用した。NQC は C 言語風の文法の記述であるが，以下の例のようにモータ制御やセンサ入力を非常に簡単に扱える API が充実しており，プログラミング経験が全くなくても容易にプログラミングを始めることができる。

NQC プログラムの例

```
task main ()
{                    // C++ 風のコメント文も使える
  OnFwd(OUT_A);      // 端子 A のモータを前転
  OnRev(OUT_C);      // 端子 C のモータを後転
  Wait(300);         // 3 秒間待機 (単位 1/100 秒)
  Off(OUT_A+OUT_C); // 端子 A と端子 C のモータを停止
}
```

ただし NQC の場合，LEGO 社のファームウェアを使用するため変数やサブルーチンの数などに制限があり，また従来の手続き型プログラミングしかできないので，それらのことを不満に思う先進的な学習者が増えてきた場合には，JAVA[7, 8] などのより汎用的なオブジェクト指向言語を用いるのがよいであろう。

本事例では，次節の各段階に合わせた NQC のチュートリアル的な教材をウェブ上に用意した [9]。

4 実習の事例紹介

4.1 目標と概要

本節ではロボット製作を通じてなるべく形式ばらない入門的プログラミング学習や総合的な問題解決能力の養成を目標にした実習型の授業事例を紹介する [10]。対象は文科系・理工系・医学系・教育系などさまざまな学部学科の学生からなる信州大学の 1 年次生であり，数人でチームを構成して作業を行った。プログラミングやホームページ作成の経験は特に前提としていない。

受講生に自然とプログラミングに接してもらうためには，次節で述べるようにプログラムの作成からコンパイル，変数，関数，サブルーチンの使用，制御構造などのプログラミング学習における一連の基本的な要素を順次採り入れた課題を与える必要がある。すなわちプログラミング学習が自然と進展するよう新たなロボットの製作課題を順次出していく必要がある。

また，各課題についてチームごとにウェブ上で成果を発表しあうことで情報発信能力やプレゼンテーション能力を養うことも付加的な目標として設定している。そのような意味では，かなり総合学習的な内容と言える。

実際には，二段階に分けてロボットの製作やプログラミングを行った。

まず第一段階では 2〜3 人のチームに分かれて比較的簡単なロボットを製作しプログラミングやロボット製作に慣れていく。この段階では，可能な限りチームの各人がプログラムを作成し，プログラミングの基本的事項について全員が理解できるよう配慮した。

第二段階では，第一段階の 2 チームを合わせて 6 人程度の新たなチームを形成し，簡易ロボコンを目標にしたロボットを製作する。キットを 2 セット使用できるので，使える部品数も増え，また RCX 間の通信機能を利用することで 2 台のロボットを連携させたり，合体させてより複雑な動作を行うロボットを製作することができる。この段階では，チーム内で作業をある程度分担しながら，各メンバーがプログラミングやロボット製作などそれぞれの得意なところで力を発揮してチーム力を高めるようにした。

4.2 各段階ごとの課題例

プログラミングの各段階ごとにロボットの課題例を紹介する。授業では各段階においてごく簡単なサンプル・プログラムやサンプル・ロボットを紹介しながらプログラミングの基本的な要素を説明していくが，単にそのポイントを理解するだけでなく，授業では説明しない付加的な工夫が必要になるよう課題設定にも注意している。

■図形や文字を描くロボット　最初のこの課題ではセンサを全く使わず，基本的な動作である前進・後進・左折・右折に加え，ペンを上下するロボットを製作することで，エディタを使ったプログラムの作成，プログラムのコンパイルとバイナリコードのロボットへの転送，モータを回転させるための基本的な API[*4]の利用，といった基本的事項を習得する。また一連の作業を通じてチーム内の交流を深める。この課題では，決まった角度に何回も曲がったり，ペンの上げ下げという動作を繰り返したりする必要があるので，マクロや関数，サブルーチンを使用した手続きの抽象化についてもその必要性を理解して実践することが求められる。その他，ペンの上げ下げをどうやって行うか？ 適切な筆圧をどのようにして維持するか？ 角や曲線をどのように描くか？ などもそれぞれのチームで独自に工夫すべき点である。

■机の縁で方向を変えて動きまわるロボット　これは論理式や制御構文についての基礎を学習する課題である。具体的には『押されている (1)』『押されていない (0)』という 2 値を取るタッチセンサをロボットに取り付け，if 文や while 文などを使ったプログラムを作成する。タッチセンサを 2 個使用することで論理積や論理和といっ

[*2] 現在使うことのできる，さまざまな開発 (プログラミング) 環境については，例えば文献 [2, 3] を参照のこと。
[*3] 現在は John Hansen 氏によってメンテナンスされている。
[*4] Application Programming Interface

た論理演算についても学習する。机から落ちないように動かすためにはロボットのあらゆる位置や状態を想定し，すなわち完全な場合分けを行ってプログラムを作成しなければならない。ただし，机からロボットが本当に落ちて破損する可能性を心配するならば，例えば，床の上を机の外周に沿って動くロボットというような課題も考えられる。

■黒いラインに沿って走るロボット　この課題では多段階の値を与える光センサを使用する[*5]。紙の質やペンの濃さ，部屋の明るさ，センサの個体差などによって明るさの閾値 (例えば白と黒を見分けるための閾値) を調整する必要がある。また光センサを 1 個使った場合，2 個使った場合，モータを 1 個使った場合，2 個使った場合など種々のロボットを製作してスピードや確実性をアップさせるよう工夫する。

■光を追いかけるロボット　ここでは懐中電灯などで指示した最も明るい方向を探し，さらに光源を動かした時にそれに追従するロボットを製作する。変数を使用して最大値を求めたり，近傍での明るさの差分を取って比較して進むなどこれまでよりも多少複雑なアルゴリズムを考えなければならない。またタイマ (時間センサと考えてもよい) を積極的に活用することでより複雑な動作にも対応できるプログラムを作成する。

■曲を演奏しながら動くロボット　この課題では曲を演奏しながら光センサとタッチセンサの両方を監視して行動する，例えば，ライントレースを行いながら障害物に当たったときは違う曲を演奏しながら動作を変更する，といった行動をするロボットを作成する。複数のタスクを並列で実行させたり，他のタスクをスタート/ストップさせる必要が生じることで，割り込み処理やリソース (モータ) へのアクセス権についても理解する。

■簡易ロボコン　これまでの学習の集大成として，簡単なロボット・コンテストを開催する。例えば，空き缶を集めてきて積み上げるロボットや，紙パックを運搬して所定のカゴに投げ込むロボットなどを製作して点数や時間を競う。キットを 2 セット使うことで，より複雑な作業を必要とする課題を設定できる。そしてこれまでの課題と違い，通信機能を用いて 2 台の RCX を協調させなければならないため原始的なプロトコルを設計することも必要とされる[*6]。チームの人数も増えるため，作業分担を上手に行うことも重要である。

4.3 ウェブ上での実習報告

各課題で作成したロボットについては，実習時間中にミニ発表会を開いて他チームのロボットについても意見交換を行った。そしてチームごとにホームページを開設して製作したロボットの詳細な説明・報告をウェブ上で行うという形式で各課題のまとめを行った。

ウェブ上での発表の場合，文章での説明だけでなく写真や図，動画も簡単に張り付けることができ，他チームのページやインターネット上のページにリンクを張って参照することも容易である。また大勢の人に見てもらうということを前提にレポートを作成するので表現方法だけでなく著作権やプライバシーに関する意識も高まる。

さてこの実習を 5 年間に渡って継続的に行ってきたが，古い年度のページを残しておくことで新受講生が旧受講生のページを参考にできるようにした。そうすることでロボットを完成させる前に解答例を見てしまい独自の努力や工夫が足りなくなるという心配も皆無ではないが，むしろ過去の受講生の豊富なページを教材として積極的に参考にすることで自主的な学習が促進されるという利点の方がはるかに大きいように思われる。実際に，まったく同じロボットを作成しない限り他人の作成したプログラムをそのまま適用してもうまくロボットが動かないこと，また，大勢の人が閲覧することを想定してページを作成しなければならないこと，などの理由で技術的にも心理的にも他人のページやプログラムを丸写しすることのほうが困難であると考えられる。

ところで毎年度いくつもの新しい課題を考えることは教員にとってかなりの重労働であるが，過去の課題をわずかに変更するだけでもその効果は大きい。例えば文字や図形を描くロボットについて課題の文字や図形を変更するだけでも違ったプログラムが必要になる。API の使用方法やサブルーチンの活用の仕方など習得すべき基本的事項については以前の年度の報告を参考にできるのだが，その一方で独自に調整したり工夫しなければならないところも多く残っているので，他人のプログラムをただコピーするだけでなく内容をある程度理解してから改良する必要がでてくる。このような学習方法はプログラミング学習において極めて自然かつ一般的で学習効率の高い方法ではないだろうか。同様にライントレース・ロボットについてもチーム独自のコースを作成させることでそのコースに適した，ロボットやプログラムの作成が必要になる。そしてそのようなオリジナルコースを上手にトレースするロボットを製作できたとき，学習者の感動や達成感も大きい。

4.4 Wiki の活用

さて 2001 年度にこの実習を開始した当初は，ロボット用のプログラムを作成するのと同様，エディタを使用して HTML ファイルを編集することで各チームのホームページ作成を行った。しかし時間的な制約や静的なページという特性上，チームのすべてのメンバーがページの作成や更新に関わるということは困難であった。

そこで 2004 年度より，Wiki クローンである PukiWiki[11] を導入した [10]。Wiki サイトの開設によって，ホームページの作成経験のない受講生でも短期間で容易にホームページを作成することが可能になった。また Wiki サイトを単なる一方的な成果発表の場としてだけでなく掲示板機能やコメント機能などのさまざまなプラグインを利用することで共同作業の補助ツール

[*5] 通常モードでは 0〜100 までの値，raw モードでは 0〜1023 までの値を取る。
[*6] RCX は赤外線を用いて 0〜255 までの数を送受信できる。

としても活用できる。もちろん教員や授業補助員にとっても各チームの進行状況や理解度を把握しやすくなり，ロボットやプログラムに対するアドバイスもしやすくなった。結果的に更新頻度も高くなり，平均的なページの質が大幅に向上したと考えられる。もちろんHTMLの学習時間を節約できるメリットはとても大きい。

5 実習上の問題点

前節で述べたような課題を順次遂行して行く中で，感じられた問題点の中で主なものを述べる。

■教員側で行う準備について　総合的な問題解決能力を養う，というこの実習の主旨を考えた場合，ロボット教材以外の教材をどの程度親切に準備するべきか？ という問題に直面する。なるべく金銭的な負担が少ないよう，ロボコンを含め各課題で使用する小道具類は紙パックや懐中電灯など身近にあるものを選ぶようにしているが，例えば『ライントレースのコースを作成するためのA0版くらいの画用紙と太いペンを用意しておく』ということが今時の学生にとっては難しいようだ。どこで画用紙を購入すればよいのかわからない，小さな画用紙をたくさん用意したものの張り合わせるためのテープなどは用意しなかった，といった事例は毎年ある。また『ロボコンで空き缶を使用するが，練習時には各チームで用意した空き缶を使うこと』と指示したところ，教室のそばに空き缶入れがあるにもかかわらず，教員に対して『空き缶をください』と申し出る学生もいる。ここまでくるとプログラミング学習どころではない感じがするが，やはり時間がかかってもどうしたら空き缶を入手できるのかさまざまな方法を一緒にその場で検討するような教育が必要なのかもしれない。

■1チームの構成人数について　受講希望者が定員より多いにもかかわらず受講制限をしない場合，1チーム4人のチームができてしまうことがある。しかし3人のチームに比べ4人では何も作業しないメンバーが出てくる確率は飛躍的に高くなる。また受講生各人の学習が授業の目的であるにもかかわらず，チームに対する貢献度が少ないと感じて他のメンバーへの遠慮から途中で放棄してしまう学生も少ないながら存在する。そのような場合，ロボットは一体しかなくても少しずつ違う動きをさせるプログラムを必ず各人に作成させることで多少は状況を改善することができるであろう[*7]。

■時間的制約について　90分×15回という時間的な制約のため前節で紹介した練習課題にそって授業を進めると，課題一つあたり2回程度の実習で完了しなければならない。とりあえず動くロボットを製作することは十分可能であるものの，製作したロボットやプログラムを洗練させるには残念ながら十分な時間とは言えない。ちなみにLEGO社の他の製品同様，RISの場合，汎用的な部品を組んでアイデアを形にして行くのは比較的容易である[*8]。しかし使用した一つひとつの部品が本当に必要かどうか吟味して無駄な部品を削って行く作業が無ければ，部品点数を減らして強度を高めたり，他の機能を追加するための部品を確保することが難しくなり保守性も悪くなる。これはもちろんプログラムの作成にも当てはまる。作品を洗練させるこの過程をもっと重要視するならば，思い切って内容を減らすか，時間数を多少増やす努力が必要であろう。

■日常の機械音痴　変速機付きの自転車に乗ったり，キャスター付きの椅子に座っていても，それらの仕組みや原理を知らない，あるいは考えたことがない，という学生は少なくない。そのため，たとえRISが12才以上対象の製品であってもギアを使った減速の方法など，初歩的な機械の説明に意外と時間を使う。しかしながらボタンだけで操作できる玩具や生活製品が溢れてしまっている以上，それらの仕組みに興味を持つということ自体現在では難しいのかもしれない。

6 まとめ

本論文では，LEGO社のマインドストームを活用して過去5年間にわたって実施してきたロボティクス実習の授業を紹介した。

本事例は大学初年度における一般教養科目および情報教育の一環として大学1年次の学生を対象に行ってきたものであるが，履修生がキーボード入力に多少慣れていれば，プログラミングの入門的学習を含め高校生や中学生でも30時間程度の時間で実現可能なコースであると考えられる。授業時間外にチームで集まっての作業が可能ならば対象が大学生でなければならない理由は特に見当たらない。また，実習上のいくつかの問題点を指摘したが，いずれも大学特有のものではない。年少時から創作的な共同作業に慣れるという面ではむしろ高校や中学校で採り入れたい総合学習的な実習であると言える。

参考文献

[1] http://www.mindstorms.com/
[2] Luis Villa, Lego Mindstorm with Linux Mini-HOWTO, http://tldp.org/HOWTO/Lego/index.html, http://www.linux.or.jp/JF/JFdocs/Lego/(日本語訳).
[3] LEGO User Group Network (LUGNET) のニュースグループ, http://news.lugnet.com/robotics/
[4] Mozilla Public License, http://www.mozilla.org/MPL/
[5] NQC - Not Quite C, http://bricxcc.sourceforge.net/nqc/
[6] Mark Overmars, Programming Lego Robots using NQC, http://www.cs.uu.nl/~markov/lego/
[7] LeJOS, Java for the RCX, http://lejos.sourceforge.net/
[8] Brian Bagnall,『マインドストーム・プログラミング入門』, CQ出版社, ISBN4-7898-3711-4
[9] http://yakushi.shinshu-u.ac.jp/robotics/NQC 入門
[10] http://yakushi.shinshu-u.ac.jp/robotics/
[11] http://pukiwiki.org/

*7 RCXは同時に5つのプログラムを内蔵することができる。
*8 LEGO社によるとRISは12才以上対象製品となっている。