

マイクロロボット制御アルゴリズムの見える化に向けて — 組込みシステムの教材として —

大熊一正*, 恐神正博*, 籠谷隆弘†, 四折直紀‡, 杉原一臣*, 山西輝也*

Toward Clarification of Control Algorithm for Micro-Robot — as a teaching material for embedded systems —

Kazumasa OHKUMA, Masahiro OSOGAMI, Takahiro KAGOYA,
Naoki SHIORI, Kazutomi SUGIHARA, Teruya YAMANISHI

We composed teaching material for education about information and communication technology using micro-robot which is the smallest soccer robot in the RoboCup soccer international competition. The feature of this material is that control programming for the micro-robot is clarified using Scratch which is produced by MIT Media Lab. However, since micro-robot is currently controlled by program in C++ language, we made translator system from Scratch code to C++ code using PHP language. In this paper, we report the reason why we come to think clarification of our teaching material is important and how to realize such clarification.

Keywords: プログラミング教材, 組込みシステム教材,

1. はじめに

今日の急速な情報通信技術（Information and Communication Technology：ICT）の発展は、社会の情報化を加速させて続けている。その結果、今や情報通信ネットワークは我々が生活する上で必要不可欠なものとなった。その一方で、厚生労働省による統計データでは、平成 24 年度 1 月時点での ICT 技術者の新規求人倍率が、2.07 倍（全体平均 1.10 倍）¹⁾ と平均値よりも高い。さらに、この平均を上回る倍率は、数年間継続されているため、我が国では ICT 技術者、特に、組込みシステムに関連する技術者不足は深刻となっている。このような情勢に配慮したためか、我が国の情報教育の中核を担う学習指導要領が刷新され、小学校では平成 23 年度から、中学校では平成 24 年度から全面施行され、高等学校では平成 25 年度より年次進行で施行される。この刷新された学習指導要領では、小・中・高校の全ての課程において、「ICT の使い方」中心の学習から「ICT を活用した問題解決」の学習へと内容が拡充された。特に、中学校では「プログラムによる計測・制御」がすべての生徒が受講すべき実習項目となり、プログラミング学習の早期化が実現された。さらに、高等学校においても、「情報に関する科学的な見方や考え方を養う」という単元の充実が図られた。このため、これら新しく導入された単元において、生徒の興味・関心を惹き付けつつ学習効果を保証できる教材が求められている。

一方、我々は、既にマイクロロボットと呼ばれる約 3cm 立方の小型ロボットを制御するプログラミングの体験授業を提案し、実施してきた²⁾。そこで本稿では、実施した体験授業のアンケー

* 経営情報学科 † 仁愛大学 人間生活学部 ‡ 経営情報学科 学生

ト結果から，使用した教材の改善点の洗い出し及びそれらの改善策とその実施状況を報告する．

本稿は，第 2 節において先行研究で行ったアンケート結果から浮かびあがった問題点を整理し，それらの問題解決に利用した **Scratch** と呼ばれるプログラミング環境を紹介する．続く第 3 節においては，今回考案した **Scratch** によって作成したプログラムから，マイクロロボットを制御するプログラムへの変換手順を説明し，その実装結果を示す．そして，第 4 節において，本稿のまとめを行い，今後の課題について述べる．また，補足では，今回作成した変換プログラムの主要部に関する説明を行う．

2. マイクロロボットを用いたプログラミング授業

本節では，まず学習教材として利用するマイクロロボットを紹介する．さらに，実際に行ったマイクロロボットを用いたプログラミング体験授業から得られた問題点及びその解決手段として利用する **Scratch** と呼ばれるプログラミング環境について述べる．

2-1 教材ロボットとしてのマイクロロボット

マイクロロボットは，シチズン時計株式会社[†]が，RoboCup[†]において使用するサッカーロボットとして開発した一辺約 3cm 立方の小型ロボットである（図 1）．このロボットには，それぞれのバッテリーに接続されたモータが，左右の車輪に 1 台ずつ付いている（図 2）．これら車輪の回転を制御す



図1: マイクロロボットの外観



図2: マイクロロボットの構造

ることによって，前進及び後進・回転だけでなく，曲線的な動きを可能にする．そして，そのような動きは，本体上部に据え付けられるマイコンに組み込まれたり，パソコンから赤外線を利用して伝送されたりする制御情報によって実現される．また，制御情報は，左右の車輪それぞれの回転方法を指示するプログラム群が C++言語用に用意されているため，一般には，それらを用いて構築する．しかしながら，C++言語に不慣れな場合，既存の関数が用意されているとはいえ，実際にそれらを理解して利用することは容易ではない．このことは，実際に行った授業の評価アンケートからも見て取れたため²⁾，続く 2-2 節において少し触れることにする．

2-2 授業アンケートからの課題

マイクロロボットを授業に利用しようとする試みは既に幾つか行われており，名城大学の高橋氏らのグループによる試みが先駆的である³⁾．それらの試みに触発され，我々のグループもマイクロロボットを利用した高校生対象の体験授業を提案し，実施してきた．この際に行った授業アンケートの結果から，体験授業自体は楽しかったようであるが，プログラミングや組込みシステ

[†] RoboCup は，“ロボット工学と人工知能の融合，発展のために自律移動ロボットによるサッカーを題材として日本の研究者らによって提唱され，西暦 2050 年「サッカーの世界チャンピオンチームに勝てる，自律型ロボットのチームを作る」という夢に向かって人工知能やロボット工学などの研究を推進し，様々な分野の基礎技術として波及させることを目的としたランドマーク・プロジェクトである．詳しくは，RoboCup 日本委員会の Web サイト（<http://www.robocup.or.jp/>）を参照のこと．

ム自体への興味関心が増加したかという点に関しては、(必ずしも否定的な結果であったわけでもないが,) 肯定的な結果を得ることはできなかった。この理由は、授業では、マイクロロボットの動作プログラム作成としつつも、実際には(変数の)数値入力(もしくは、変更)だけに限定してプログラミングを体験させたことに起因すると考えられる。つまり、受講者達は、プログラミング作業として、既存プログラムを一部変更するだけに留まったため、プログラミングを行ったという実感及び達成感を得られなかったのではないだろうか。しかしながら、授業実施当時のマイクロロボットシステムは、C++言語をある程度理解していなければ、変数の値変更以外のプログラミングを行うことは容易ではなかった。それでも無理にC++言語による本格的なプログラミング教育を行うことも可能であるが、それは逆に、受講者に苦痛を与え、興味関心を低下させる恐れがある。このため、マイクロロボット制御プログラミング環境として、制御アルゴリズムを視覚的に理解しながら、主にマウス操作によってプログラミングできる「Scratch」を注目するに至った⁴⁾。Scratch 以外にも、プログラミングの視覚化を意識し、比較的新しく、日本語対応がなされているプログラミング環境としては、文部科学省が提供している「プログラミン⁵⁾」もある。しかし、ここでは、作成したプログラムの流用性及びインターネット接続を必要としない等の理由から Scratch を採用した。

2-3 Scratch を用いたプログラミングの見える化

2-2 節で、問題点としてあげた“プログラミングを行った実感を与えづらい”を解決するために、マサチューセッツ工科大学メディアラボ(MIT Media Lab)の Lifelong Kindergarten グループによって開発されたプログラミング環境である Scratch を導入し、マイクロロボットの制御プログラムの見える化を試みる。

Scratch は、主にスプライトと呼ばれる画像オブジェクトを操作するプログラムの作成を通し、プログラミングの楽しさをより多くの人に伝えることを目的に開発された。このため、プログラムの流れや命令の意味が視覚的に理解しやすいように設計されている。この設計思想のもと、Scratch は、主に以下の4つの領域(図3の①～④)；

- ① コマンドパレット：プログラミングに必要な命令ブロックの一覧を表示するエリアであり、上部で、命令の種類を選択すれば、下部に上部で選択された命令に関連する命令ブロックが表示される。
- ② スクリプティングエリア：このエリアで①のコマンドパレットからドラッグした命令ブロックを組み合わせ、プログラム(スプライト)を作成する。
- ③ スプライト/ステージ ドック：作成しているプログラムで使用する(使用できる)スプライトとステージ画像を一覧表示する。



図3: Scratch開発環境

- ④ **アクションエリア**：②のスク립ティングエリアで組み上げたプログラムがどの様に動作するかを確認できる。

から構成されるプログラミング環境となっている。

その特徴としては、

- 命令ブロックが、例えば“○歩動かす”，“△が押されたとき”や“○回繰り返す”と表記されており（ここで、○は数値、△は対応するキーをそれぞれ表す.），アルゴリズムを言葉として理解しやすい（図 3 の①参照）。
- プログラミングは、命令ブロックの組み合わせによって行うが、構文として成り立っていない場合はブロックが組み合わないと、構文を覚える必要がない。
- Scratch 本体が Web ページ⁴⁾からダウンロードでき、無料で使用可能であるだけでなく、環境の日本語化も行われている。
- 作成されたソフトウェア（主にゲーム）及びそのプログラムが Scratch の Web ページ⁴⁾から参照でき、それらを利用して自主的な学習が可能である。

という点があげられる。さらに、Scratch は、スプライトを操作するプログラム作成に長けているため、このスプライトを実機であるマイクロロボットに置き換えることが比較的容易であると考えた。これらが、今回、マイクロロボット制御用プログラムの開発の環境として Scratch を採用した理由である。

3. Scratch によるマイクロロボットの制御

2 節でも記したが、マイクロロボット は C++言語によるプログラミングによって制御されることが一般的である。そこで、今回、Scratch を用いてマイクロロボットを制御するために、Scratch によるプログラムを C++言語によるマイクロロボット制御プログラムへと翻訳するプログラムを作成することにした。

3-1 C++言語によるマイクロロボット制御プログラム

図 4 にマイクロロボットを 5 秒間前進させるための C++言語による制御プログラムを示す。ここで、3 行目において読み込む“ConnectRC.cpp”によって、10 行目から 12 行目にあるマイクロロボット制御関数が利用可能となる。これら制御関数は、以下の通りである。

`connectToRobotcontrol()`：マイクロロボットを制御するサーバとマイクロロボット間の通信を確立する際に利用する関数

`sendData(id, left, right)`：マイクロロボットに制御データを送る関数

ここで、

`id`：マイクロロボットに割り振られた番号

1	#include <stdio.h>
2	#include <unistd.h>
3	#include "ConnectRC.cpp"
4	int left=100, right=100;
6	int time=50000;
7	main()
8	{
9	int id = 1;
10	connectToRobotcontrol();
11	sendData(id,left,right);
12	usleep(time);
13	}

図4:マイクロロボットの制御プログラム

left : 左車輪の回転速度

right : 右車輪の回転速度

である.

usleep(time) : sendData 関数によって指定された id のマイクロロボットの車輪を left / right で指定された回転速度で time ミリ秒だけ回転させる関数

つまり, マイクロロボットは, 左右の各車輪の回転速度と回転時間が設定されることによって制御される.

3-2 Scratch によるマイクロロボット制御プログラム

Scratch は, 命令ブロックを組み合わせることにより, 視覚的に確認しながら, プログラミングを行える開発環境であるが, 作成したプログラムをテキスト形式で保存することも可能である[‡]. 例えば, Scratch を用いてスプライトが正方形を描くように動くプログラムを命令ブロックで組むと, 図 5 のようになる. そして, そのプログラムをテキスト形式で保存すると図 6 のようになる. この図 6 において, 図 5 の命令ブロック部が記述されている部分は, 27 行目から 30 行目である. それ以外の 1 行目から 26 行目, 31 行目と 32 行目は Scratch によるプログラムの変更履歴や画像ファイル等の情報が書かれている部分となっているが, 制御には関係がないため, 続く 3-3 節以降では, 主に 27 行目から 30 行目までの部分を取り扱うことにする.

3-3 Scratch によるマイクロロボット制御

Scratch で作成したプログラムのテキスト形式を図 4 に示したような C++ 言語のプログラムに変換するために, 図 6 の 27 行目から 30 行目までを考察し, (以下, ○は整数値として,)



図5: 命令ブロックによるプログラム

1	Project: square
2	Author:
3	Scratch: 1.4 of 30-Jun-09
4	History:
5	20yy-mm-dd hh:mm:ss save square
6	
7	Totals:
8	Sprites: 1
9	Stacks: 1
10	Unique costumes: 3
11	Unique sounds: 2
12	-----
13	Sprite: ステージ
14	Costumes (1):
15	背景 1 (480x360)
16	Sounds (1):
17	ポップ (0:00:00)
18	No stacks.
19	-----
20	Sprite: スプライト 1
21	Costumes (2):
22	コスチューム 1 (95x111)
23	コスチューム 2 (95x111)
24	Sounds (1):
25	ニャー (0:00:01)
26	Stacks (1):
27	4 回繰り返す
28	10 回繰り返す
29	5 歩動かす
30	90 度回す
31	
32	-----

図6: Scratchのプログラムをテキスト形式で保存したもの

[‡] 命令ブロックで作成したプログラムをテキスト形式で保存する場合は, シフトキーを押しながらファイルメニューの【プロジェクトのまとめを書き出す】を選択する. 保存されたファイルは, 文字コードが UTF-8N, 改行コードが CR+LF となっている.

- “〇回繰り返す”は、For 文
- “〇歩動かす”と“〇度回す”は、sendData 関数と usleep 関数の組み合わせ

に置き換える方法を考案した。

そして、将来的には、インターネット経由での利用を想定し、PHP 言語を用いて、Scratch のテキストプログラムをマイクロロボット制御用の C++言語プログラムに変換するスクリプト（以後、変換スクリプト）を有する Web サイトを構築した。

作成した変換スクリプトの処理の流れは、

- ① **変換データの整形**: Scratch によって組み立てられたプログラムのテキストデータ（以後、Scratch プログラム）は、改行コードが統一されていない場合があるので、全ての行の改行コードを CR+LF に統一する。さらに、マイクロロボット制御に不要な部分（例えば、図 6 の 27 行目から 30 行目以外）を削除する。
- ② **“〇回繰り返す” 処理の置き換え**: Scratch プログラムにおける“〇回繰り返す”の文字列を“for (int i=0; i< 〇; i++)”に置き換える。（ただし、この処理は、より詳しい説明が必要であると思われるため、補足において詳説する。）
- ③ **“〇歩動かす” 処理の置き換え**: Scratch プログラムにおける“〇歩動かす”の文字列を“sendData(id, 41, 41); usleep(〇*125*1000);”に置き換える。（ここで、sendData 関数の id は、任意のマイクロロボットに対応する id 番号であり、車輪の回転速度「41」は、試行実験の結果から決めた値である。同様に、usleep 関数の「125*1000」も試行実験によって決めた値である。）
- ④ **“〇度回す” 処理の置き換え**: Scratch プログラムにおける“〇度回す”の文字列を“sendData(id, 31, 10); usleep(〇*14*1000);”に置き換える。（③と同様に、2 つの関数内の数値は、試行実験によって求めた値である。）
- ⑤ **マイクロロボット制御に必要な語句の挿入**: ここまでの処理結果を C++言語の構文体系に則し、さらに、マイクロロボット制御関数を利用できるようにするため、図 4 の 1 行目から 8 行目までの文字列を文頭に付加する。
- ⑥ **終了括弧の付加**: 置き換え処理の最終行に、main 関数の終了を示す括弧“}”を付加する。
- ⑦ **C++プログラムの作成**: 上記の処理を行った結果を、入力ファイルの拡張子部を cpp に変更した C++のプログラムファイルとして作成する。（つまり、Scratch プログラムのファイル名から拡張子のみが cpp に変更されたファイル名となる。）

の順で行われる。

4. 動作確認

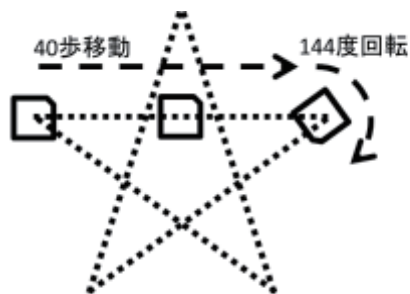
本節では、マイクロロボットが一筆書きの星（五芒星）型を描くように動くプログラムを Scratch で作成し、それを C++言語のプログラムに書き換える過程を紹介する。そして、変換さ

れた C++言語のプログラムを用いてマイクロロボットを制御した結果を示す.

4-1 Scratch による制御プログラムから C++言語プログラムへ

マイクロロボットに, 図 7 に示す星型を描かせるには,

- ① 40 歩動かす[§]
- ② 144 度回す
- ③ ①と②の処理を 5 回繰り返す



の処理を行う必要がある.

図7: マイクロロボットが描く星形の軌道

図8: 星形を描かせる命令ブロック

そして, その処理を実行させるプログラムを Scratch の命令ブロックを用いて表現すると図 8 のようになる. (ここで, 直線移動は, 1 歩動かすを 40 回繰り返すことによって実現している.)

図 8 の命令ブロックをテキスト形式で保存すると, 図 9 のようになり, それを今回作成した PHP スクリプトを介して C++言語のプログラムに変換すると図 10 のようになる.

1	Project:
2	Totals:
3	Sprites: 1
4	Stacks: 1
5	Unique costumes: 3
6	Unique sounds: 2
7	-----
8	Sprite: ステージ
9	Costumes (1):
10	背景 1 (480x360)
11	Sounds (1):
12	ポップ (0:00:00)
13	No stacks.
14	-----
15	Sprite: スプライト 1
16	Costumes (2):
17	コスチューム 1 (95x111)
18	コスチューム 2 (95x111)
19	Sounds (1):
20	ニャー (0:00:01)
21	Stacks (1):
22	5 回繰り返す
23	40 回繰り返す
24	1 歩動かす
25	144 度回す
26	
27	-----

図9: Scratchによる星形軌道プログラムのテキスト形式

1	#include <stdio.h>
2	#include <unistd.h>
3	#include "ConnectRC.cpp"
4	int left=100, right=100;
5	int time=50000;
6	main()
7	{
8	int id = 1;
9	connectToRobotcontrol();
10	for(int i=0; i<5; i++){
11	for(int i=0; i<40; i++){
12	sendData(id, 41, 41);
13	usleep(125*1000);
14	}
15	sendData(id, 31, 10);
16	usleep(2016*1000);
17	}
18	}

図10: PHPスクリプトによって変換されたC++のプログラム

注) 図 10 の 13 行目と 16 行目の `usleep` 関数の引数の値は (3-3 節のスクリプト処理の流れ③及び④で記したように), それぞれ, 1 歩進むに対し, $1 \times 125 \times 1000$ マイクロ秒, 144 度回転に対し, $144 \times 14 \times 1000$ マイクロ秒として算出されている.

[§] Scratch によるプログラムでは, 直線移動による距離を○歩と表記するため, ここでもその記述方法を踏襲した.

4-2 マイクロロボットの動作結果

4-1 節で記した手順に基づいて作成した C++言語によるプログラムをマイクロロボット制御サーバ**に投入し、マイクロロボットを実際に動かしてみた。その結果は、図 11 に示すように、マイクロロボットに星形を描かせることに成功した。

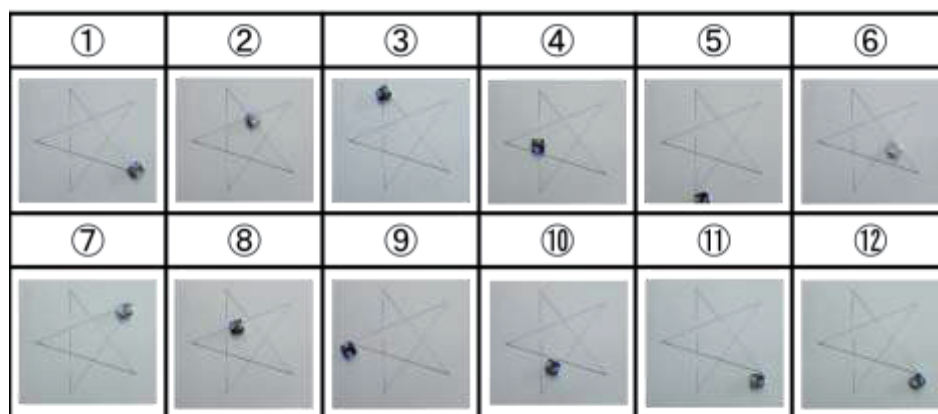


図11: マイクロロボットが描いた星形移動のスナップショット

5. まとめ

日本における ICT 技術者（特に組み込みシステムの技術者）不足が深刻化している中、平成 24 年度より施行される学習指導要領では、中学生が受講すべき項目として「プログラムによる計測・制御」が組み込まれた。これらの現状を受け、マイクロロボットと呼ばれる 3cm 立方の小型ロボット実機を MIT Media Lab が無料公開している Scratch を用いて制御する教材システムを構築した。本構築教材システムの特徴は、Scratch を用いることによってマイクロロボットの制御アルゴリズムを視覚化した点である。この視覚化、つまりアルゴリズムの見える化によって、プログラミング初学者にとって 1 つの難関となるプログラミング言語の文法修得を排除する（もしくはその労力を低減する）ことに成功し、プログラミング学習において必要かつ重要なアルゴリズムを直感的に学習することが可能になると期待できる。

本教材を用いた授業内容としては、先行研究²⁾で行った授業に習い、マイクロロボットに直線運動や円運動などを行わせる簡単な制御から始める予定である。そして、最終的には各自の好きな図形をマイクロロボットに描かせるアルゴリズムを考案させ、実際にそのアルゴリズムを実装し、その動作確認を行う授業への展開を試みる。それらの授業後には、アンケートを実施し、本システム及び教授内容の改善に努め、中学校の教育現場でも簡単に利用でき、生徒たちがプログラミングに興味を持てるような教材へと改善したい。

一方、授業アンケートに基づくシステム及び教授内容の改善に並行し、3 節で紹介した PHP スクリプトによる Scratch プログラムを C++言語のプログラムに置き換える Web システムの拡張

** マイクロロボットを制御する環境については、文献 3 に詳しいので、ここでは割愛する。

を模索している．そして，将来的には，インターネットを経由して Scratch プログラムを Web サイトにアップロードさえすれば，自動的に C++言語に変更されたプログラムがマイクロロボット制御サーバに転送され，転送された制御データに基づいて動作するマイクロロボットを Web カメラ経由でも確認できるようにしたい．

謝辞

本研究は，日本学術振興会 科学研究費補助金(基盤 C 21500962, 23560543)，福井県大学連携リーグ研究推進事業及び福井工業大学特別研究費クラスタ研究 D からの補助を受けて実施された．

補足

本補足では，3-3 節において記したスクリプト処理の流れ② “○回繰り返す”処理の置き換え方法を詳細に記す．

本文中で記したように，単に“○回繰り返す”を“for (int i=0; i<○; i++) {”と置き換える^{††}だけでは，その処理範囲の指定ができない．何故なら，Scratch プログラムのテキストデータでは，“○回繰り返す”の処理範囲が空白文字によるインデントで表記されており，繰り返し処理の終了括弧と置き換えるべき文字列が存在しないからである．この結果，図 5 のような“○回繰り返す”が，入れ子構造である場合には，各繰り返し処理の終了判定が重要になる．

そこで，今回作成したスクリプトでは，Scratch プログラムのテキストデータのインデント構造を調べ，インデントの空白文字数から“○回繰り返す”の有効範囲を決める方法を考案し，実装した．

以下に，Scratch プログラムのテキストデータ形式における繰り返し処理のインデント構造の特徴と今回考案した処理手順の概略を記す．

例えば，Scratch で図 12 のような繰り返し処理をプログラミングした場合，そのテキストデータの該当処理部は，図 13 のように記述される．図 13 より，繰り返し制御のインデント空白文字数は，4 個であり，繰り返し処理される命令文（ここでは，“1 歩動かす”，“90 度回す”）のインデント空白文字数は 8 個であることが分かる．次に，繰り返し処理が，入れ子構造になった場合の Scratch プログラム（図 14）とそのテキストデータ（図 15）を見てみれば，1 つ目の繰り返しの制御文は，インデントの空白文字



図12: Scratchによる“繰り返し”処理

1	<input type="text" value="4"/> 4 回繰り返す
2	<input type="text" value="8"/> 1 歩動かす
3	<input type="text" value="8"/> 90 度回す

図13: 図12のテキスト表現(抜粋)

ここで， 内の数値は空白文字数を示す．

^{††} 実際には，まず，“繰り返す”という文字列を有する行頭に“for (int i=0; i<”を挿入し，その後，“回繰り返す”の文字列を“; i++) {”で置き換え，“○回繰り返す”を“for (int i=0; i< ○; i++) {”と置き換えている．

数が 4 個であり，2 つ目の繰り返しの制御文のインデント空白数は，8 個である．そして，2 つ目の繰り返し文で繰り返し処理される命令文（ここでは，“1 歩動かす”）のインデント空白文字は，12 個である．そして，2 つ目の繰り返し制御領域を抜け，1 つ目の繰り返し処理だけの制御対象となる“90 度回す”の処理命令のインデント空白数は，8 個となっている．さらに，2 つの繰り返し処理の制御領域を抜けた後の命令文（ここでは，“-10 歩動かす”）のインデント空白数は 4 個である．

以上より，1 つ目の繰り返し処理は，4 個のインデント空白文字の後から始まり，その有効範囲は，8 個以上のインデント空白がある（別の繰り返し処理も含む）処理命令部であることが分かる．

つまり，Scratch プログラムのテキストデータにおける“○回繰り返す”処理の入れ子構造では， N 番目の繰り返し処理命令文前のインデント空白数は， $[4(N-1)+4]$ 個であり，その処理の有効範囲は，インデント空白数が， $[4N+4]$ 個以上ある命令文である．このことを利用して，

1. 各行のインデント空白文字数を数え，その行中に“繰り返す”の文字列があれば， N に 1 を加える．
2. インデント空白文字数が $[4(N-1)+4]$ 個となる命令文の前には， N 番目の繰り返し処理の終了括弧を挿入する．次の命令文が無い場合にも，終了括弧を挿入する[‡]．

という置き換えを実行することによって，“繰り返す”の有効範囲を決定することができる．



図14: Scratchによる“繰り返し”処理の入れ子処理

1	4	4 繰り返す
2	8	10 回繰り返す
3	12	1 歩動かす
4	8	90 度回す
5	4	-10 歩 動かす

図15: 図14のテキスト表現(抜粋)

ここで， 内の数値は空白文字数を示す．

参考文献

- 1) 厚生労働省 職業安定局雇用政策課，“一般職業紹介状況（平成 24 年 1 月分）について”，（2012）．
- 2) 山西輝也，杉原一臣，大熊一正，“マイクロロボットを用いたプログラミング導入教育の試み”，福井工業大学研究紀要 第 41 号（2011）486－496．
- 3) 市橋浩典，内山雅文，岡谷賢，高橋友一，“EcoBe!とウェブカメラによる教材用ロボットシステム”，人工知能学会研究会資料，SIG-Challenge A901-7，（2003）34－38．
- 4) MIT Media Lab, Scratch, <http://scratch.mit.edu/>（2012 年 3 月 31 日現在）．
- 5) 文部科学省，プログラミン，<http://www.mext.go.jp/programin/>（2012 年 3 月 31 日現在）．

（平成 24 年 3 月 31 日受理）

[‡] 次の命令文が無い場合は，その行が最終行となるため，繰り返し文の終了括弧及び，main 関数の終了括弧も挿入することになる．