JSiSE Research Report
vol.33,no.5(2019-1)

ビジュアルプログラミングの評価支援システムの開発

福永理絵[*1], 佐々木整[*1], 岡本俊一[*1]
*1 拓殖大学工学部

Development of an Evaluation Support System for Visual Programming

Rie Fukunaga[*1], Hitoshi Sasaki[*1], Shunichi Okamoto[*1]
*1 Faculty of Engineering, Takushoku University, JAPAN.

小学生向けのプログラミングスクールの普及など、日本国内でもプログラミング教育が一般的に行われるようになってきている。2020 年からの小学校でのプログラミング教育の必修化もなされ、プログラミングやプログラミング的思考はこれまで以上に身近なものとなると考えられる。既に様々なプログラミング教育が児童・生徒を対象に行われ、そのノウハウなどの蓄積がなされつつある。その一方で、作成されたプログラムに対する評価の方法については明確になっていない。また、一人ひとりが作成したプログラムをどのように評価し、フィードバックを行うことは非常に多くの労力を必要とする。そこで、教員が児童の頑張りや工夫の度合い、不得意なところなどを見つけ、評価したり指導に繋げたりしていくことのできるアプリケーションを開発している。本稿では、このアプリケーションで使用している 5 つの評価観点やシステムの概要について述べる。

キーワード: ビジュアルプログラミング，評価，プログラミング教育

1. はじめに

将来，機械化が進むことが予想され，IT 業界の人材不足が懸念されている．現在でも IT 分野の人材が不足している状態であり，人材の育成が必要である(1)．経済産業省が 2016 年に発表した IT 人材の最新動向と将来推計に関する調査結果(2)では，2030 年の人材不足規模は約 59 万人になると予測されている．このような背景をもとに、2020 年から小学校でプログラミング教育の必修化される．また，海外でもプログラミング教育を初等教育から導入する動きがみられ，イギリスでは 2014 年から初等教育でのプログラミング教育の必修化が既に行われている (3)．

小学校教員が児童の作成したプログラムに評価をし，指導に役立てていく必要があるが，評価の明確な基準は定められていない．さらに，教員の多くがプログラミング未経験であり，適切な評価が行えるかという心配を多くの教員が持っている．地域によっては教員のためにプログラミング講習を開くなどの活動が行われているが(4)，教員全員に適切な指導をすることは現状困難であるとされている．教員は児童の作成したプログラムから，頑張りや工夫，不得意な箇所を見つけることで今後の指導に繋げなくてはならないため、教員が児童の作成したプログラムを評価する際の支援が必要とされている．

文部科学省が公開している資料(1)では，小学校でのプログラミング指導方法として，ビジュアルプログラミングが挙げられている．ビジュアルプログラミング言語学習環境には，MIT メディアラボが開発した Scratch，NTT が開発した VISCUIT，文部科学省が開発したプログラミンなどがある．本アプリケーションは，小学校教育でのプログラミング教育の例として書籍等で多く挙げられている Scratch のプログラムを対象とする(5)．この Scratch は 2006 年に MIT メディアラボが開発したビジュアルプログラミング言語学習環境で，150 以上の国で 2 千万人以上に利用されている．

現在のメジャーバージョンである Scratch2.0[6]では，12 種類 144 個のブロックを組み合わせることで図 1 のようにコードを用いず視覚的にプログラムを作成することができる．

本稿では、ビジュアルプログラミングで児童が作成したプログラムを教員が評価し、指導に反映させる時の難しさと、それを支援するために現在開発を行っているシステムについて報告する。

図 1 Scratch 2.0 でのプログラミング例

2. プログラミングの評価

プログラミング教育の必修化に向けて、カリキュラムの検討や作成，モデル校での実証授業がなされたり、茨城県つくば市や千葉県柏市のように市内の小学校でプログラミング教育を始めている自治体もある[7]．

しかし、つくば市が公開しているプログラミング学習に関するカリキュラムや指導案作成に関する資料[8]などでは，カリキュラムや授業内容の記載はあるものの，評価の基準は明確にされていない．また、教育者オンラインコミュニティーである ScratchEd[9]には，いくつかのプログラム評価用シートが投稿されているが，それを利用するには教員自身のプログラミング技術が必要である．文献(7)によると、柏市の教育委員会の委員は，忙しい教員が新たにプログラミングの技術を身に付けることは難しいため，プログラミングができなくても授業を実施できる仕組みが必要であると指摘しているが、これには当然実施した後の評価も含まれていなければならない．

文献（10）では，2 つのルーブリックが挙げられている．その 1 つでは，児童の作成したプログラムを以下の 4 つの観点に沿ってシート形式で評価を行なっている．

A　教科の知識：教科で学習した内容を結びつけ，理解できているか．

B　プロジェクトのデザイン：デザインの観点から見た独自性があるか．

C　プログラミングの知識：ブロックの働きを理解し，論的な組み合わせを行えているか．

D　学習の過程：基本的な過程を正確に活用し，要領よく作業が進められているか．

これらの観点は，ScratchEd に投稿されているスクラッチ・プロジェクトを評価する基準となるルーブリックを参考に日本の教育現場で活用できるようにデザインしたものであるが、これらのシートを利用して評価をする際，プログラムのどこを見れば良いかは明確にされていない．そのため，どのブロックがどれくらい，どのように使われているのかを教員が一つずつ確認し，理解したうえで評価する必要がある．少人数で実施する場合は大きな手間では無いかもしれないが、そうではない場合は、教員にとって大きな負担となってしまう。クラスやグループ単位の指導で，全員のプログラムを評価することになる．また，プログラミングの授業が作られるのではなく，既存の授業にプログラミング要素を取り入れるとされているため，授業内容を検討する必要がある．他にも，教員自身がプログラミングについての知識をつける必要があるなど，プログラミング必修化が教員に与える負担は大きいといえる．

3. プログラムの評価観点

3.1 評価観点の概要

そこで、評価のサポートを行うアプリケーションを開発して、自動の作成したプログラムのブロック数などのプログラムの特徴自動的に検出する。その検出結果を基に、いくつかの観点で自動が作成したプログラムを分析した資料を教員に提示することで，教員の負担を軽減することを考えた．

結果を数値化したり視覚化したりすることで，児童が授業内容を理解できているかの判断がしやすく，授業についていけない児童に対し早期で対応することができる．また，前回の評価結果や，クラスやグループ全体の平均と比較することも容易になる．

評価項目として、以下の５つを考えている.

① プログラムの長さ：縦に並んだブロック数
② 制御･構造:制御ブロックの中で使われている，条件などのブロック数
③ 変数等の活用 : 変数やリスト，定義のブロック数
④ メディア活用 : 動きブロック，音ブロックなどの数
⑤ ブロック化 : ブロック総数

以下、それぞれの評価観点について説明する。

3.2 長さ

ブロックをたくさん使うと，プログラムは縦に長くなる.そこで,多くのブロックを組み合わせることは，児童の取り組みの熱心さの一端を表していると考えた.しかし，制御構造やモジュール化を理解せず，冗長なプログラムだったり、適当にブロックを組み合わせているだけだったりすることも考えられるので，ブロックが長く繋がっていることが良いプログラムである事を意味しているとは限らない.

3.3 制御・構造

ネストや条件ブロックを適切に利用できていれば，制御構造を理解できていると判断できる．画面上のキャラクターを動かしたり，図形を描写したりするなどの比較的難易度の低いサンプルでもよく使用されているネストや条件ブロックを使用している数で評価する.しかし、これらのプログラムでは，ネストや条件ブロックを使用しなくても作ることができるので、3.2 で述べたプログラムの長さでの評価が高くなり、こちらの評価が低くなることになる.

3.4 変数等の活用

児童が自ら作成する変数やリストブロックを正しく使えていれば,それらの概念を理解しているとともに,問題を一般化して考えることができるかの判断に繋がると考え、変数やリストブロックを使用している数で評価する.

3.5 メディア活用

Scratch には、動きや音に関するブロックが数多く用意されている。これらのブロックを多く使っていることは、積極性や独自性が高いと考え、そのブロック数で評価する.

3.6 モジュール化

解決しなければならない問題を、小さな問題（モジュール）の集合と捉えることがどの程度できているかを、モジュールの数で評価する。意味のあるモジュールを作成することが重要で、とにかくブロック化すれば良いというものではないので、モジュールの数が多いから良い・悪いというものではない。

3.7 先行研究との比較と利用

次に，2.で述べた先行研究の観点，特にプログラムに直接関係のある B と C を, 本研究の評価項目と比較する.

B は，以下の３点で評価している.

① 作品の独自性
② 内容が体系化されているか
③ 他の児童が作品と触れ合う方法が与えられているか

本研究の評価項目である「4.メディア活用」で, イベントの発生, スプライトの動きや変更, BGM や効果音に関するブロックの数から評価を出すことで，最終的な B の評価に繋がる.

C では，以下の３点で評価している.

① ブロックの機能を理解して利用できているか
② 論理的に作れているか
③ エラーがあるか

条件や変数に関するブロックがプログラムの中で使用されており，かつ正しい結果を出力しているならば，ブロックの機能を理解し，論理的に作られていると言える．エラーがある場合，プログラムは正しく動作しないため，３点目は本研究では考慮しないものとする．本研究の評価項目である「2.制御･構造」と,「3.変数等の活用」で，条件や変数に関するブロックの数から評価を出すことで，最終的な C の評価に繋がる.

２つの長方形・正方形の面積を求めるプログラムを図 2 と図 3 に示す．これらは最終的に同じ結果を出力するが，表 1 に示すように使用しているブロックの数や種類が大きく異なっている.

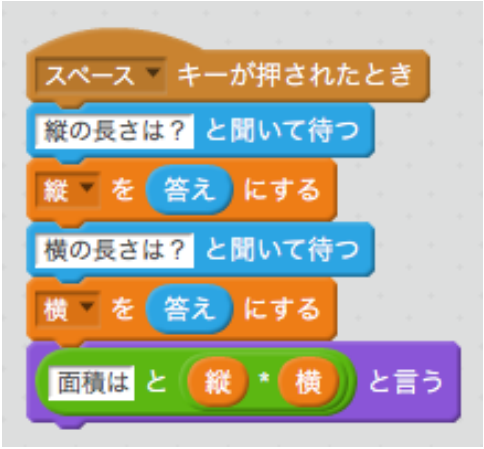

図 2 プログラム 1

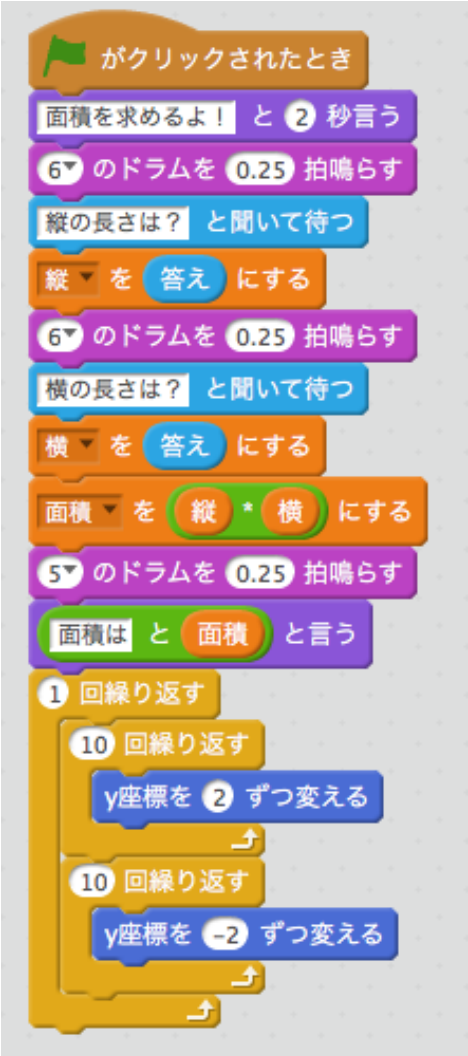

図 3 プログラム 2

表 1 種類別ブロック数

ブロック種類	プログラム 1	プログラム 2
動き	0	2
見た目	1	2
音	0	3
ペン	0	0
データ	4	6
イベント	1	1
制御	0	3
調べる	4	4
演算	2	2
その他	0	0
合計	12	23

表 1 の種類別ブロック数をもとに，本研究の観点に沿って評価したものが表 2 観点別評価である． この結果から，プログラム 1 は「長さ」「メディア活用」「モジュール化」の評価は低いが，「制御・構造」「変数等の活用」の評価は高いため，ブロックの働きを理解して使っていることがわかる．プログラム 2 はプログラム 1 と比べると，「制御・構造」と「変数等の活用」の評価の差はあまりないが，「長さ」と「メディア活用」の評価が高い．プログラム 1 と同じくブロックの働きを理解できており，さらにメディアも活用していることがわかる．

表 2 観点別評価

評価観点	プログラム 1	プログラム 2
長さ	2.5	11
制御・構造	11	13
変数等の活用	8	10
メディア活用	2	7
ブロック化	4	7.5

4. 評価アプリケーション

4.1 評価アプリケーションの概要

児童が Scratch で作成したプログラムを画像として読み込み，テンプレートマッチングを用いて評価項目に沿って解析する．各ブロックの画像を元画像とし，

児童の作成したプログラムから各ブロックの数や並びを検出する．

テンプレートマッチングの結果から図 4 の出力画面案のようにグラフを描写し，視覚的に評価結果を出力する．画面左側には読み込んだ Scratch のプログラム画像，中央には今回と前回のプログラム評価，右側には個人の評価結果と今までの評価結果をグラフで表示する．2 つのグラフを出力することで児童の学習進度や得意な分野を判断しやすくする．

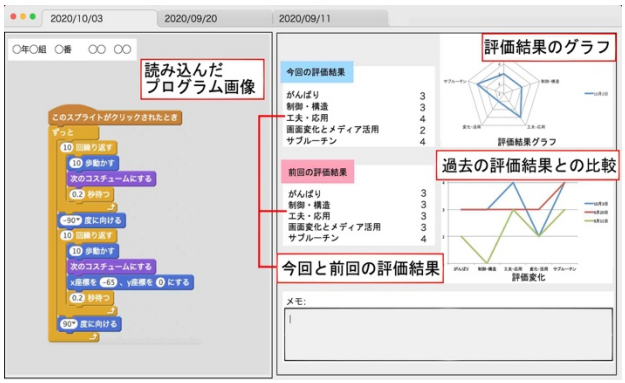

図 4 出力画面（開発中）

評価観点の概要で述べたように，教員はクラスやグループ単位で指導を行う．結果を図 5 のようなレーダーチャートで視覚的に出力することで，文字だけで出力するよりも容易に比較することができる．これによって，大人数での指導でも，児童の努力した点や得意な箇所，あるいは躓いてしまっている箇所の判断がしやすくなる．

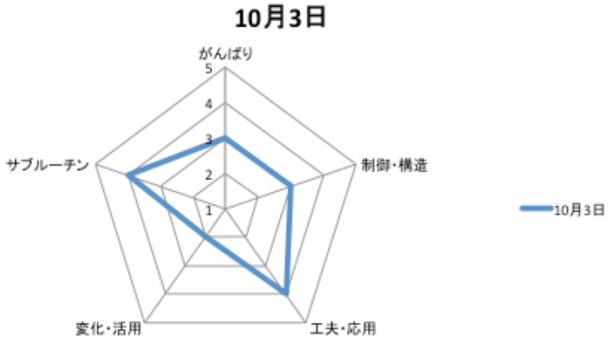

図 5 レーダーチャートによる結果表示の例

レーダーチャートの他に，図 6 のような折れ線グラフで今までの結果を出力することで，前回との比較が容易になる．回数を重ねるごとに，生徒がどのように成長してきたかを見ることができる．継続的に評価の高い場合には，より発展した指導をしたりなど，今後の指導方法の参考になる．

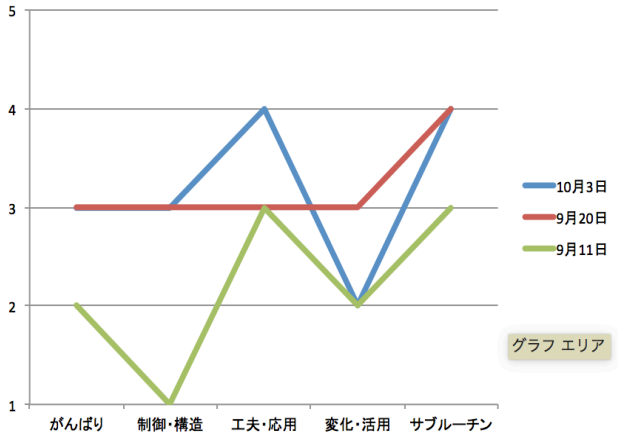

図 6 折れ線グラフ

4.2 プログラム分析

Scratch の各ブロックの画像をテンプレートとして，OpenCV のテンプレートマッチングで，作成されたプログラムから各ブロックの数と並び順を検出し,ブロックの種類ごとの合計結果をグラフとして出力した．同時に，Scratch のサンプルコードから評価項目を選定した．また，三重大学の Scratch プログラムの評価方法の研究の評価観点と選定した評価項目を比較した．

現在，ユーザー自身が名前をつける変数ブロックの検出ができない．ディープラーニングを用いて検出することが今後の課題となっている．また，図 7 のようなネストされたブロックの終端の検出率が低く，約 50%ほどだったが，現在は用意したサンプルプログラムの中で約 80%検出できるようになった．

図 7 ネストブロック

今後，ネスト構造と変数ブロックを判断できるようにし，評価項目をよりわかりやすくなるように見直す必要がある．

プログラムを完成させ，結果を保存し，出力案に沿って画面出力する．

5. おわりに

本稿では、児童が Scratch で作成したプログラムを自動的に分析し、教師に評価や指導の材料となる情報を提示する、評価サポートのシステムについて述べた。現時点では未完成で実践評価などが行えてはいないが、このシステムを利用することで、多くの児童のプログラムの中から、個別にケアが必要な児童を見つけ出したり、クラス全体の傾向などを比較的容易に見つけ出すことができると我々は考えている。

筆頭著者は，高校から情報系学科に所属していたかが，以前は理系分野の勉強かがとても苦手だった。そのため，プログラミングについて学ぶことの大変さは身を以て理解しているつもりである．プログラミングに苦手意識を持っている教員にも使いやすく，分かりやすいアプリケーションを目指すとともに、児童が適切な指導と評価を受けることで、IT 分野に進むきっかけとなるよう、未実装の部分について早急に開発を進め、本システムを完成させ実践を行いたいと考えている．

参 考 文 献

(1) 文部科学省: “プログラミング教育”, http://www.mext.go.jp/a_menu/shotou/zyouhou/detail/1375607.htm（2018 年 12 月 11 日確認）

(2) 経済産業省: “IT 人材の最新動向と将来推計に関する調査結果”, http://www.meti.go.jp/policy/it_policy/jinzai/27FY/ITjinzai_report_summary.pdf (2018 年 12 月 11 日確認)

(3) 文部科学省: “情報教育指導力向上支援事業（諸外国におけるプログラミング教育に関する調査研究）”, http://www.mext.go.jp/a_menu/shotou/zyouhou/detail/1408119.htm (2018 年 12 月 11 日確認)

(4) 毎日新聞: “教師も未経験　必須か控え指導準備”, https://mainichi.jp/articles/20170829/k00/00e/040/186000c（2018 年 12 月 11 日確認）

(5) 竹中章勝: “はじめてのプログラミング授業実践”, (2018)

(6) Scratch2.0, https://scratch.mit.edu/（2018 年 12 月 11 日確認）

(7) 日経 BP 社: “教育と ICT”, pp.7, pp.12, (2018)

(8) つくば市総合教育研究所: “つくば市プログラミング学習の手引き【第 2 版】”, https://www.tsukuba.ed.jp/souken/?p=7831 (2018 年 12 月 11 日確認)

(9) Scratch Ed, http://scratched.gse.harvard.edu/（2018 年 12 月 11 日確認）

(10) 大野恵理, 須曽野仁志: “スクラッチを用いたプログラミングプロジェクトの評価方法”, 日本科学教育研究会研究報告, Vol.31, No.8(2017)