クリック操作とフローチャートを用いた教育用プログラミングツールの開発

川上 由乃[1,a)]　鈴木 優[1,b)]

概要：2020 年より改訂される学習指導要領によって小学校でプログラミング教育が必修化される．本研究では，プログラミング教育の目的の 1 つであるプログラミング的思考を養うためのツールとして，選択式プログラミングツール「ふろみん」を開発した．選択式プログラミングとは，本研究で提案するビジュアルプログラミング言語で，クリック操作のみで選択肢から命令を選び，フローチャートを作成するものである．操作方法をクリックのみにすることでパソコン操作の習熟度の違いによらず学習に取り組みやすくなる．また，プログラミングをするためには問題解決のために物事を順序立てて考えることが必要で，プログラミング教育はその力を身に付けるためのものである．そこで，問題解決のための手順を図式にするフローチャートを用いてプログラミングができるようにすることで，手順をプログラミング言語でプログラムに変換することを考える必要がなく，考えた手順をそのままフローチャートにするだけで良くなる．また，プログラムの流れが読み取りやすく，複雑に見える動作も「順次」「分岐」「反復」という要素の組み合わせであることの理解を促す．これにより簡単で取り組みやすい学習でプログラミング的思考を養うことを支援する．

1. はじめに

2020 年に改訂される学習指導要領によって小学校でプログラミング教育が必修化される．学習指導要領とは，文部科学省が定める教育課程（カリキュラム）の基準である．プログラミング教育の目的は，子供たちにコンピュータに意図した処理を行うよう指示することができるということを体験させながら次のような資質及び能力を育成することである [1]．

知識及び技能

身近な生活でコンピュータが活用されていることや問題の解決には必要な手順があることに気付くこと

思考力、判断力、表現力等

「プログラミング的思考」を育成すること

学びに向かう力、人間性等

コンピュータの働きを，よりよい人生や社会づくりに生かそうとする態度を涵養すること

プログラミング的思考とは，自分が意図する一連の活動を実現するために，どのような動きの組合せが必要であり，一つ一つの動きに対応した記号を，どのように組み合わせたらいいのか，記号の組合せをどのように改善していけば，より意図した活動に近づくのか，といったことを論理的に考えていく力のことである [2]．その他にも，ルールに従って過不足なく指示を出す力や，意図した結果にならなかったときに必ずどこかに要因があると考え改善するといった力を身に付けることができる．プログラミング教育とはプログラミング単体の授業を行うわけではなく，算数や理科といった既存の教科の中でプログラミング的思考を養うような学習を実施するもので，学年や，教科，時間といった具体的な内容は各学校が判断することとなっている．子供たちにとって難しいプログラミング教育を行うことでプログラミングに対する苦手意識を子供たちが強める可能性がある．そのため苦手意識を感じない教育方法が必要であると考える．そこで，本研究では簡単で取り組みやすくプログラミング的思考を養うことを支援する．

2. プログラミング教育

2.1 プログラミング教育の方法

プログラミング教育の方法はアンプラグドプログラミングとソフトウェアを使うプログラミング，ハードウェアを使うプログラミングの 3 つに分けられる．

2.1.1 アンプラグドプログラミング

アンプラグドプログラミングとは，コンピュータを使わ

[1] 宮城大学
Miyagi University, 1-1 Gakuen, Taiwa-cho, Kurokawa-gun, Miyagi 981-3298, Japan
[a)] p1622034@myu.ac.jp
[b)] suzu@myu.ac.jp

ずにパズルやカードなどを用いて，順序立てて物事を考えることによって問題解決の手順を学ぶ方法である．ICT 環境の整っていない場所でも学ぶことができ，パソコンなどのコンピュータを使ったことのない子供でも取り組みやすい．例えばカードを使った授業では，命令の書かれたカードを意図した動作になるように並べ，相手はその並べられた命令の通りに行動をする．様々な動作を組み上げて，実際にその通り動くか，動かなかった場合はどこに要因があるのかを考え，試行錯誤をする力を身に付ける．

2.1.2 ソフトウェアを使うプログラミング

ソフトウェアを使うプログラミングとは，コンピュータでビジュアルプログラミング言語などの専用のアプリケーションを使う方法である．この方法は視覚的にわかりやすく，簡単な操作だけで行うことができる．また，結果がすぐに出るため，結果をもとに試行錯誤することを後押しする．コンピュータを使って算数の授業で多角形の作図を行ったり，音楽の授業で音を鳴らしてリズムを作ったりといった授業が行われている．多角形の作図や，音楽は同じ行動の繰り返しでできていることに気付き，プログラミングにおける反復処理についても学ぶことができる．

2.1.3 ハードウェアを使うプログラミング

ハードウェアを使うプログラミングとは，教育用のロボットやマイコンなどのハードウェアを制御し，動かす方法である．センサやモータなどと組み合わせられるものは身の回りの様々なものにコンピュータが活用されていることを考えるきっかけになる．パソコンやタブレット上でプログラミングをして自分で組み立てたロボットやマイコンを制御するものや，ロボットを組み立てること自体がプログラミングとなるものがある．自らロボットを組み立てることで愛着が湧き，組み立てる過程の中でどう動かしたいか想像を膨らませながら楽しく学ぶことができる．

2.2 3 つの方法の比較

アンプラグドプログラミングは，パソコンがない環境でも実践ができるため，気軽に挑戦できるというメリットがある．その反面，コンピュータを使わないということは 1 章で述べたプログラミング教育の目的の「知識及び技能」と「学びに向かう力、人間性等」を満たすことが難しい．ソフトウェアを使う方法は，パソコンやタブレットを用意する必要があるが，コンピュータが様々なものに活用されていることに気付くきっかけにはなりやすい．ハードウェアを使う方法は，子供たちの興味を惹きやすく，また，ロボットを組み立てることはものづくりの楽しさに気付くきっかけになるというメリットがある一方で，専用のロボットを用意する必要がある．

そこで，本研究では，パソコンやタブレット以外の特殊な装置を使わずに行うことができるソフトウェアを使う方法を用いることにする．

3. ビジュアルプログラミング言語

3.1 ビジュアルプログラミング言語の概要

パソコンやスマートフォンで学べるプログラミング教材は数多くあり，未就学児でも使える簡単なものから，本格的なプログラミング言語を学べるものまで，年齢やスキルにあわせて選ぶことが可能である．その中でも未就学児から小学生にも扱いやすいのが「ビジュアルプログラミング言語」である．ビジュアルプログラミング言語とは，プログラムを英語や数字などのテキストで記述するのではなく，視覚的なオブジェクトでプログラミングするプログラミング言語である．マウス操作だけで簡単にプログラミングできるため，子ども向けのプログラミング的思考を養う教育ツールにも用いられている．

3.2 既存のビジュアルプログラミング言語

ビジュアルプログラミング言語は主に図 1 に示すブロックタイプとフロータイプに分けられる．

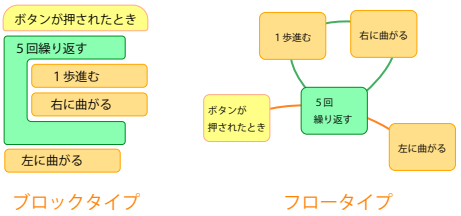

図 1 ビジュアルプログラミングの例

ブロックタイプとは，指示の書かれたブロックをマウス操作で繋げて配置することでプログラミングする方法である．米国マサチューセッツ工科大学に設置された研究所である MIT メディアラボが開発したプログラミング言語学習環境の Scratch[3] などがある．ブロックを並べる順序が一般的なテキスト言語の考え方と似ており，順次・選択・反復というプログラムの基本構造をしっかりと学ぶことができるという利点がある．一方で，繋がったブロックのかたまりから 1 つだけブロックを抜き出そうとしたときに，ドラッグしたブロックより下に繋がっているものがすべてくっついて移動してしまうため，プログラムの修正に手間がかかるという欠点がある．また，ブロックはジャンル分けして用意されているものが多いが，種類が豊富なため使いたいブロックを探すのが困難である．

フロータイプとは，指示の書かれたオブジェクトを自由に配置して矢印や線で繋ぐことでプログラミングする方法である．ソニー株式会社が提供する MESH アプリなどがある．MESH という，ボタンやスピーカーなどの様々な機能を持つ MESH ブロックを MESH アプリ上で繋げてプログラミングをすることで制御するツールである．フロータイプの利点は，オブジェクト自体を動かさなくても線の繋

がりを消したり，繋げたりすることで簡単に命令を削除，追加できることである．しかしながら，オブジェクト同士の間にスペースをあらかじめ空けておかないと途中にオブジェクトを足したいときにオブジェクトをすべて動かす必要などがある．また，オブジェクトの配置も自分でするため配置によっては流れを読み取りにくいという欠点がある．

その他にもブロックタイプやフロータイプに分類されない独自の手法を用いたタイプもある．例えば Viscuit[4] は，自由に絵を書き，メガネという左右２つの輪がつながったツールの，それぞれの輪に違う絵を置くことでプログラミングをする．

ビジュアルプログラミング言語を用いた教育ツールは，言語の知識がなくてもプログラミングすることができる．また，タイピングミスによるエラーがないこと，ゲーム感覚で勉強ができることなどの利点がある．基本的なマウス操作だけでプログラミングを行えることが利点だが，実際にプログラミングをする際は，まず制作や問題解決のために必要となる手順を順序立てて考え，そのあとに考えた手順をどのようにプログラムにするかを考える必要がある．つまり，教員は順序立てて考えることと，ビジュアルプログラミング言語の操作方法やプログラムにする方法を教えなければならない．さらに，操作を教える際，パソコンを使ったことのない子供がいる場合は，ビジュアルプログラミング言語の使い方の前にマウス操作も教える必要がある．パソコンを使ったことのある子供と使ったことのない子供で進行度や積極さに差が生まれてしまう可能性があると考える．

4. 選択式プログラミングの提案

4.1 選択式プログラミングの概要

本研究で提案する選択式プログラミングとは，選択肢から「命令」を選ぶだけでフローチャートを作成してプログラミングができるものである．必要な操作がクリックのみなことが特徴で，ブロックタイプやフロータイプと違って命令が書かれたブロックやオブジェクトを一覧から探し出してドラッグで並べる必要がない．

授業の中でパソコン操作を教えながら既存のビジュアルプログラミング言語を使う場合，普段からパソコンをよく使う子供は経験から別の操作を行うことでできることを自ら考えることができるのに対して，はじめてパソコンを使う子供は習った操作しか使えないということが起きる．また，反対にパソコン操作を教えることに時間を使うことで，パソコンを使ったことのある子供が授業に飽きてしまう可能性もある．本ツールでは，必要な操作をクリックのみにすることで，パソコンの習熟度によって学習に差ができないようにする．また，パソコン操作の指導によって授業時間が足りなくなることを防ぐ．また，命令の書かれたオブジェクトを並べるためには，一覧から必要なオブジェクトを探し出してドラッグで持ってくる必要がある．既存のビジュアルプログラミング言語は，画面が実行する部分と，ブロックを並べる部分，オブジェクトの一覧部分に分かれているものが多いが，本ツールはフローチャートを作成する部分に選択肢が表示されるため視線の移動が少ないことも特徴である．

フローチャートとは問題解決の手順を図式に表したものである．プログラミングをするためには必ず問題解決のために物事を順序立てて考える必要がある．そこで，その考えた手順をそのままフローチャートとして作成するだけでプログラミングを行うことができるツールを使用すれば，教員は手順をプログラムにすることを教える必要がなく，物事を順序立てて考えることを教えるのに重点を置いた授業を行うことができる．また，自分の考えている動きを実現するためには「順次」「分岐」「反復」のどれを使うのかを考え，それをフローチャートの形にして，実際に動作させることで，複雑に感じる動作も実際の動く仕組みは，「順次」「分岐」「反復」という単純な要素の組み合わせでできているということに気付くきっかけとなる．

作成中のプログラムの途中に命令の追加をする場合や，命令を削除するとき，他の命令に書き換えるときにもクリックだけで全て行うことができ，修正したいときにその都度ブロックやオブジェクトをドラッグで移動させるといった余計な操作を必要としない．そのため，ブロックやオブジェクトが画面内に乱雑に置かれることがなく，プログラミングする本人も他の人にもプログラムが読みやすくなる．フローチャートを作成する場所をブロックタイプやフロータイプのように自由に決めるのではなく，固定の位置にすることで，どんな内容のプログラムでも命令の流れがわかりやすくなる．

子供が「操作方法が難しくて覚えるのが大変」や「ブロックやオブジェクトが多すぎて何を使ったらいいのかわからない」という印象を抱かなくすることで，プログラミングに対する難しそうというイメージを払拭し，楽しく学べるようにする．本ツールの対象はこれからプログラミング教育を受ける小学生を想定する．本ツールによって以下の力や習慣を身に付けることができる．

- 目的を達成するためにはどのような処理をどのような順番で実行すればいいのか，ということを論理的に考える力
- 意図した結果にならなかったときにどこを変えたらいいのか，どう変えたらいいのか，ということを考える習慣
- もっと簡単に目的を達成するにはどうしたらいいのか，他にも達成する方法はあるのか，ということを考える習慣

4.2 タスクのデザイン

選択式プログラミングでは，障害物を避けながらキャラクタをスタートからゴールまで連れていくというプログラムを作成させる．目的を提示せずに自由に作成させることも検討したが，子供の性格によっては「作りたかった動きにはなっていないけど面白いからこれで完成でいいや！」と，自分で設定した目的を達成できないまま途中でやめてしまったり，目的を達成するために命令を選ぶのでなく，「こうしたら面白くなるのでは？」という気持ちから別の選択をしてしまったりという可能性もあるのではないかと考えた．興味を持って自らほかの命令を選んだらどのような動きになるのかを考え，実際に動かすことは学習するうえで重要なことではあるが，それによって達成しようとしている目的を見失うことはないようにしたい．そこで，意図した結果にならなかった際にどこに要因があるかを考え，改善する力を身に付けるために明確な目的を提示することにした．

プログラミング教育で用いられる教材には図形を書くものや計算を解くことを目指すものもあるが，キャラクタのイラストを使うことで興味を惹き，愛着がわきやすいと考えた．そこで，図 2 のような，キャラクタを障害物を避けながらスタートからゴールまで連れていくプログラムを作成させることにした．これは迷路がベースになっているので子供たちにも取り組みやすいと考えたためである．動かすキャラクタは，動物によって動きや使うことのできる命令に制限を設けることで同じステージでも難易度が変わるようにする．

ツールの使用者は，キャラクタがどこにいて，どこに連れて行かなければいけないのか，そのためにはどのような道を通ればいいのかをまずは考える．そして「ここを通ってゴールまで連れていく」という達成すべき目的に対して，何歩進んで左右のどちらかに曲がるという処理をどのような順番で実行すればいいのか，ということを論理的に考えることができる．また，スタートからゴールに行くまでのルートが複数あったり，同じルートを通る際にも様々な命令の組み合わせ方で通ることができたりするように障害物を配置する．これによって一度目的を達成できた後にも，他にもゴールにたどり着く方法はあるのではないかと考えることができる．

5. 「ふろみん」の開発

5.1 ツールの概要

ふろみんは，クリック操作だけでフローチャートを作成する選択式プログラミングでキャラクタをスタートからゴールまで連れていく迷路を解くことでプログラミング的思考を身に付けさせる学習を行うものである．使用者には，チュートリアルを順番に解いてもらうことでツールの使い方と，繰り返しや分岐の活用方法を学んでもらう．

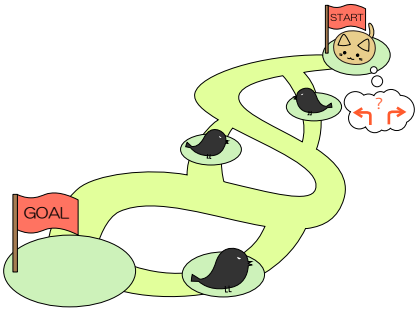

図 2 タスクデザイン

図 3 は実装したツールの画面である．画面は左側の命令を選択してフローチャートを作るフローチャート入力部と，右側の制作したプログラムを実行するフローチャート実行部で構成される．

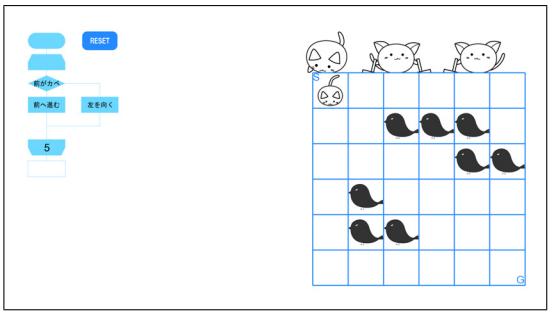

図 3 画面イメージ

フローチャート実行部に表示される迷路を解くためのプログラムをフローチャート入力部で作成する．本ツールは openFrameworks で実装した．

5.2 フローチャート入力部

フローチャート入力部ではクリックで選択をしてフローチャートを作成することでプログラミングを行う．フローチャートは，予め開始を表す楕円のみ書かれた状態になっており，フローチャート内の四角い枠の部分に命令を追加することができる．命令を追加したい枠内をクリックすると図 4 のような記号の選択肢が出てくる．記号には次の 6 つがある．

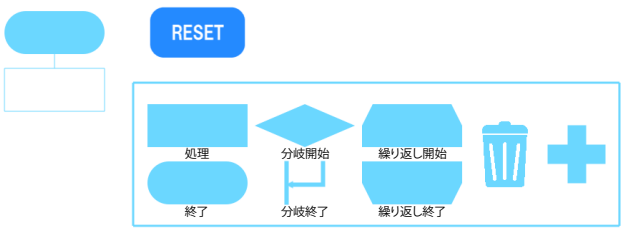

図 4 記号の選択肢

処理

前へ進んだり，横を向いたりという基本的な動きを選

択する

分岐開始

分岐の条件を選択する

分岐終了

フローチャート内から分岐の合流する地点を選択する

繰り返し開始

繰り返しの開始を指す

繰り返し終了

繰り返しの回数を選択する

終了

フローチャートの終了を指す

記号を選択すると，さらに詳しい内容や条件などの選択肢が出てくる．図 5 は処理を選んだときと，分岐開始を選んだときの選択肢の例である．これらを選択することでフローチャートに選択した命令が表示される．

図 5 記号の選択肢

この選択を繰り返すことでプログラミングを行うことができる．さらに，すでに命令が入っている場所をクリックしても同じ選択肢が出てくる．そこで記号や詳細を選択することで，「前へ進む」という命令が入っていた場所を「右に曲がる」に変更するなどのプログラムの修正が容易にできる．また，選択肢にあるゴミ箱を選択することでそこに入っていた命令を削除することができる．隣のプラス記号をクリックすることでそこに新たに命令を追加することができる．フローチャートが完成して終了の記号を最後に選択することで，フローチャート作成モードと実行モードの切り替えボタンを押すことができるようになる．

5.3 フローチャート実行部

画面上部にはモードの切り替えボタンと一時停止・再実行ボタン，実行停止ボタンがある．モードの切り替えボタンを押すとキャラクタがフローチャート通りにアニメーションで動く．実行中に一時停止・再実行ボタンを押すと一時停止し，もう一度押すことで途中から実行を再開する．再生中，または一時停止中に実行停止ボタンを押すとキャラクタがスタート位置に戻り，その後に一時停止・再実行ボタンを押すことで最初から実行し直すことができる．

また，実行と同時にフローチャート入力部にフローチャート内のどこの命令を実行しているのかが表示されるため，思いどおりの動きをしなかったときにフローチャートのどこを修正すればいいのかがわかりやすくなっている．さらに実行中にキャラクタが壁や障害物にぶつかったときは表情などでこれ以上進むことができないことを知らせる．

5.4 実行例

このステージではスタートから階段状にジグザグに進むルートや，一番下まで直進したら曲がってさらにゴールまで直進するルートなどがある．図 6 は階段状にジグザグに進むルートを通った場合の例である．

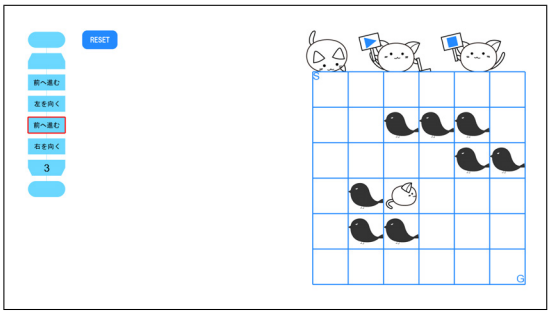

図 6 ジグザグに進むルートの例

同じルートを通る場合でも繰り返しや分岐を使うことで簡略化をすることもできる．

6. まとめ

本研究では，プログラミング教育の目的の 1 つであるプログラミング的思考を養うためのツールとして，選択式プログラミングツール「ふろみん」の開発を行った．本ツールは，問題解決の手順を表す図式であるフローチャートに着目し，フローチャートを作成するだけでプログラミングができる．これにより，教員は手順をプログラムにすることを教える必要がなく，順序立てて考える方法について教えることに重点を置くことができる．ツールを使用する子供は問題解決のために考えた手順をそのままフローチャートにすることでプログラミングをすることができる．今回はキャラクタをスタートからゴールへ連れていくプログラムを作成させることとしたが，選択肢や実際に作成させるプログラムを変更することで，さらに学年や授業の科目にあった教育に活用することができると考える．

参考文献

[1] 文部科学省．小学校段階におけるプログラミング教育の在り方について（議論の取りまとめ）．
http://www.mext.go.jp/b_menu/shingi/chousa/shotou/122/attach/1372525.htm （2019-12-01 参照）

[2] 文部科学省．小学校プログラミング教育の手引（第二版）．
http://www.mext.go.jp/component/a_menu/education/micro_detail/__icsFiles/afieldfile/2018/11/06/1403162_02_1.pdf （2019-12-01 参照）

[3] Mitchel Resnick, John Maloney, Andrés Monroy-Hernández, Natalie Rusk, Evelyn Eastmond, Karen Brennan, Amon Millner, Eric Rosenbaum, Jay Silver, Brian Silverman and Yasmin Kafai . Scratch: Programming for All. Communications of the ACM, Volume 52, Issue 11, pp. 60-67, 2009

[4] 原田康徳．子供向けビジュアル言語 Viscuit とそのインタフェース．情報処理学会研究報告，ヒューマンインタフェース研究会報告 pp. 41-48, 2005