2007－CE－88 (13)
2007／2／17

ViPPER -ロボットを使用した初等教育向けビジュアルプログラミング-

瀬古俊一[†] 山岸真弓[††] 中西健太[††]
伊与田康弘[††] 永井敏裕[††]
服部隆志[††] 萩野達也[††]

ViPPER(Visual Programming for Primary Education with Robot) とは，小学生でもプログラミングの流れや繰り返しや条件分岐などの動きが簡単に理解できる情報教育用ビジュアルプログラミング環境である．ViPPER はロボットの動きをプログラミングする環境を提供し，その制御データを送信することでロボットを動かす．さらに記述したプログラムの流れをアニメーションで表現する．このことにより，実際に動いているロボットの動きとプログラムの流れを見比べられるので，プログラムの動きや流れを理解しやすくなる．結果，小学生でも簡単に理解できるビジュアルプログラミング環境が実現される．

ViPPER - Visual Programming for Primary Education with Robot -

Shunichi Seko,[†] Mayumi Yamagishi,[††] Kenta Nakanishi,[††]
Yasuhiro Iyoda,[††] Tomohiro Nagai,[††] Takashi Hattori[††]
and Tatsuya Hagino[††]

ViPPER(Visual Programming for Primary Education with Robot) is visual programming environment for information technology education that primary schoolchild can easily understand motion of programming such as flows, loops and IF Statements. ViPPER offers the environment that dose programming of movement of the robot. When a user sends data that made in ViPPER to a robot, the robot moves as the data was described. Furthermore, ViPPER expresses flows of program with animations. Because it is compared movement of a robot and a flow of a program at the same time by this, It is easy to come to understand movement and a flow of a program. Therefore, the visual programming environment that even a primary schoolchild can easily understand is realized.

1. はじめに

1.1 プログラミング教育の必要性

文部科学省による学習指導要領における情報教育の改善内容[1] によると，情報教育による目標の一つに「社会生活の中で情報や情報技術が果たしている役割や及ぼしている影響を理解する」とある．情報技術の役割や影響の理解を深めるには実際に技術を体験することが一番である．情報技術の基盤はプログラミングであるため，プログラミングを学べば情報技術の役割や影響を知ることができる．

プログラミングを学ぶ利点はそれだけではない．目的を達成するためのプロセスを考える力や試行錯誤しながら完成を目指す力を身につけることもできる．小学生のうちにこれらの力が身につけられれば，初等教育で習う他の教科と同様に，社会生活の中で必ず役に立つ．これらの理由により，プログラミング教育を初等教育から行う必要がある．

1.2 プログラミング教育の現状

初等教育におけるプログラミング教育の現状を小学生に実際に聞いたところ，インターネットやパソコンでの文書作成などは教育されているが，プログラミングは教育されてはいなかった．また，プログラミングという言葉自体知らない小学生が非常に多い．これは小学生がプログラミングを学びやすい環境がないため，教えるのが困難なのが原因であると考えられる．したがって，初等教育におけるプログラミング教育のために，学びやすい環境を新たに構築することが必要である．

† 慶應義塾大学大学院政策・メディア研究科
Graduate School of Media and Governance, Keio University
†† 慶應義塾大学環境情報学部
Faculty of Environmental Information, Keio Unibersity

2. 初等教育に適したプログラミング環境

初等教育におけるプログラミング教育に適したプログラミング環境は，以下の3つがポイントとなる．

- ビジュアルプログラミング
- ロボット
- モバイル性

2.1 ビジュアルプログラミング

ビジュアルプログラミングとは，プログラムのコード自体を視覚的に表し，実行結果や実行経過も視覚的に表すことができるものである．そのため，文字だけのソースコードよりも，図形や矢印を用いたソースコードの方が直感的に理解しやすい．また，決められた命令を選択して組み合わせていくだけでプログラムを作成できるので，感覚的にプログラミングしていくことができる．

プログラミングの初心者は，プログラム実行時の動きとソースコードが対比できていない場合が多い．動きとソースコードとの対応付けができないことが，初心者がプログラミングを難しいと感じてしまう原因となっている．ビジュアルプログラミングはプログラムの流れや動きをアニメーションなどで視覚的に表しやすく，動きとソースコードとの対応付けの理解を深めることができる．

以上のことにより，初等教育向けのプログラミング環境にはビジュアルプログラミングが適している．

2.2 ロボット

プログラミングした結果が画面内で動く環境よりも，プログラミングした結果，実世界上のものが動く環境のほうが小学生に親しみを持たれやすい．そこで，実世界で動くものとしてロボットを使用する．ビジュアルプログラミング環境でロボットの動きをプログラミングし，その制御データをロボットに送信して動かすことができれば，画面内で完結せず実世界のものを動かす環境になる．また，プログラムの実行結果がロボットの動きとなるため，画面上のプログラムの進行にしたがってロボットを動かすようにすれば，プログラムの動きとソースコードとの対比がしやすくなる．

2.3 モバイル性

ロボットを使うとなると，ロボットをいろんな場所に持ち運んで動かしてみたいという心理が働きやすくなる．しかし，プログラミングをする環境そのものが持ち運べなければロボットを動かすことができない．そこでプログラミング環境を携帯ゲーム機上に構築すれば，ロボットとプログラミング環境を持ち運ぶことが可能になる．持ち運べるということは，それだけプログラミングに触れる機会を増やせることができるので，プログラミングを学ぶ時間を増やせる．携帯ゲーム機はパソコンに比べ入力機器が少ないため，C やJava などの言語よりも，ビジュアル言語のほうが相性が良い．

2.4 目標とする環境

目標とする環境はビジュアルプログラミングとロボットという特性を活かし，実行過程とソースコードとの関係性をいかにわかりやすく表していくかである．どのようなコードを書くとどういう動きをするかを明確にわかるようにすれば，プログラミングを繰り返していくうちに自然と力が身についていくようになる．したがって，実行過程とソースコードとの対比・係わり合いが分かりやすいビジュアルプログラミング環境を携帯ゲーム機上で構築することを目指していく．

3. 関連研究

2章で挙げたキーワードを元に選出した以下の3つの関連研究を紹介する.

- LEGO MindStorms
- WonderBorg
- スペースジョイナーズ

3.1 LEGO MindStorms

MindStorms[3] はデータフローモデルのビジュアルプログラミング環境である. あらかじめ用意されたコマンド郡から命令を選び, それらを組み合わせていくことでロボットの動きを制御するプログラムをプログラミングしていく. 上から下へ組み合わせていき, その順番通りに実行されるので, プログラムの動きや流れを直感的に理解しやすい. また, 好きな形 (機能を持った) ロボットが作れるため, 色々な動きを試すことができる.

しかし, 繰り返しや条件分岐を入れ子にできなかったり, プログラムの規模が大きくなるとプログラム全体を見渡しにくくなってしまうという欠点がある. また, マウス操作を基本としているため, 携帯ゲーム機という限られた入力機器の上では制約を受けてしまうという問題もある.

MindStorms はプログラミングの教育に向いているかもしれないが, 携帯ゲーム機上で動作させることを考えると理想とする環境には届かない.

3.2 WonderBorg

WonderBorg[4] は, PC からでも携帯ゲーム機からでもプログラミングが可能なビジュアルプログラミング環境である. ロボットに搭載されたセンサに対して優先順位を決め, センサが反応したときの行動を用意された命令の中から選んでいくとこでプログラミングしていく. これを各センサごとに設定することで, 状況に応じて動きを変えるプログラムを作成できる. パネル (Excel のシートのようなもの) を切り替えることでより複雑な行動を作り上げることもできる.

しかし, すべて IF 文の制御として記述していくため, 条件分岐の考え方しか教えることができない. また, プログラムを本体に送信してからしかロボットを動かせないため, プログラミングをしながらそのロボットの動きを確認していくということができない. したがって, プログラムの動きとソースコードとの対応付けを意識させにくい.

WonderBorg は携帯ゲーム機上で動くプログラミング環境という面では良いが, プログラミング教育に使うには不十分である.

3.3 スペースジョイナーズ

スペースジョイナーズ[5] は, インターネットを使ってプログラミングを学べるロボットシミュレーションである. 格子状に配置されたパネルの上にロボットを制御するための命令チップを貼り付けることでプログラミングしていく. プログラムは, スタートパネルから始まり, 矢印の向き通りに隣接するパネルの命令を順番に実行していく. 実行時には画面の中のロボットが動き, 現在どこの命令を実行しているかが分かる.

しかし, 実行している命令が分かる機能は, アニメーションの動きが早すぎるため, プログラムの流れを捉えづらい. また, プログラミング環境もロボットも PC 上でしか動作せず, インターネットに接続されていることが必須であるため, モバイル性が低い.

スペースジョイナーズは情報処理室等の PC とインターネット環境が完備されている場所での教育には向いているかもしれないが, 自宅や外でも学べることを考えると, 理想とする環境には適していない.

3.4 考　　察

以上の3つの関連研究の特徴をまとめると 表1 のようになる. これらの結果から既存のものは, プログラムの動きとソースコードとの対比がしやすいものではないといえる. また, その機能とモバイル性を兼ね備えたものはない. したがって, 2.4 節で述べた通り, 実行過程とソースコードとの関係がわかりやすい環境とモバイル性との両立がポイントとなる.

	Mind Storms	Wonder Borg	スペース ジョイナーズ
実世界性	○	○	×
ロボットの自由度	○	△	○
ビジュアル言語	○	○	○
手続き型言語の機能	△	×	△
実行過程とコードとの対比	×	×	△
モバイル性	×	○	×
携帯機器向けのインターフェイス	×	○	×

表 1　関連研究の特徴

4. システムの概要

携帯ゲーム機である PSP 上にビジュアルプログラミング環境を構築し，それを用いてロボットを制御するプログラムを書く．ロボットには無線 LAN モジュールが搭載されており，無線 LAN を用いて制御データを PSP からロボットに送ることでロボットを動かす．

4.1 ロボットの仕様

ロボットの形状は左右に 1 つずつ無限軌道を搭載した車型である (図 1)．それぞれの無限軌道には独立したモータがついており，無線 LAN を使ってデータを送ることでモータを制御する．無線 LAN を用いているため，電波が届く位置であればどこからでもロボットを動かすことが可能である．また，複数のセンサを搭載しており，センシングの結果によって分岐するプログラムを書くことが可能である．

4.2 言語の仕様

PSP 上に実装したビジュアルプログラミング環境は，BASIC 風のビジュアル言語である．プログラムは Start から順番に下方向へ矢印をたどっていき，End に到達すると終了する (図 2)．Start と End の間にあらかじめ用意されている命令をパネルとして挿入していくことでプログラミングしていく．各命令は引数を 1 つ持つので，ロボットが前進する距離や旋回する角度などを細かく指定できる．これらに加え，以下の特徴がある．

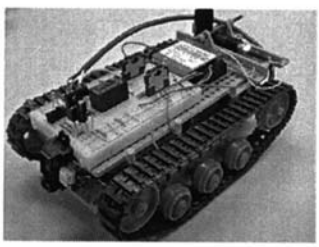

図 1　ロボット

図 2　プログラミング画面

- 自動インデント
- 繰り返しや条件分岐の入れ子が可能
- 繰り返しの残り回数を引数として利用可

繰り返しや条件分岐は入れ子可能であり，書いたプログラムは自動インデントされる．自動インデントとは，入れ子の中のベースラインが自動的に一段右にずれることによって繰り返しや条件分岐などの範囲を分かりやすくすることである (図 2)．このように，手続き型言語の表記に似せることによって，将来手続き型言語を習うときに知識が繋がりやすくなる．また，条件分岐は通常の条件式による分岐のほかに，ロボットに搭載されているセンサの状態による分岐も可能である．例えば赤外線センサを用いて，前に障害物があるかないかで異なる振る舞いを行わせることができる．

このビジュアルプログラミング環境は，MindStorms で不可能だった繰り返しや条件分岐の入れ子が可能である．また繰り返しでは，繰り返しの残り回数を命令の引数として利用することができる．このように，より手続き型言語に近づけている．

4.3 プログラミングの理解を深める工夫

プログラミングの理解を深める工夫として以下の機能を取り入れた．

- 実行過程のアニメーション
- 繰り返しの残り回数

実行過程のアニメーションを行うことによってプログラムが実行されていく様子を可視化できる．命令が実行されていく順番を明示的に見ることができるので，プログラムがどのように実行されていくかの理解を深めるられる．また，繰り返しや条件分岐の動きなども

見ることができ，特に繰り返しに関しては残り回数がカウントダウンされていくため，制御構文が理解しやすい．

このアニメーション時に，実行過程と同期してロボットも動作する．そのため，画面を見ながらロボットの動きを追うことでプログラムの動きとソースコードとの対比ができる．これによりMindStormsやWonderBorg，スペースジョイナーズといった従来のロボットの動きを制御するビジュアルプログラミング環境では難しかった実行過程とソースコードとの対応付けの手助けが可能となる．したがって，従来のものよりもプログラミングの理解を深めることができる．

以上の工夫をすることで，プログラミングへの導入の敷居をさげ，小学生にも分かりやすいプログラミング教育を行うことができる．

4.4 携帯ゲーム機上の工夫

PCに比べ限られた入出力機器しかもたない携帯ゲーム機上でも問題なくプログラミングできるように以下の工夫をした．

- 裏面の利用
- 画面の拡大・縮小
- コードの全体表示

裏面の利用とは，各命令パネルの裏面に引数を表示することで，1画面上の情報量を2倍にすることである．これにより，狭い画面でもソースコードが見づらくならず，情報量も減ることもない．

画面の拡大・縮小は，パネル全体をズームイン・ズームアウトできる機能である．ユーザが自由にパネルの大きさを設定できるので，プログラムの規模に合わせて見やすい大きさにできる．どこまでもズームアウトできるため，プログラムがどれだけ膨大になっても全体を見渡すことができる．しかし，いくら全体を見渡せても小さすぎて何が書いてあるかわからなくなってしまっては意味がない．そこで，コードの全体表示機能を用いることでこの問題を解決する．

コードの全体表示とは，コードの全体と詳細を同時に見ることができる機能である．画面が左右の二つに分かれ，左側にソースコードの全体像と選択範囲の枠が，右側に選択されている範囲の拡大図が表示される(図3)．この選択範囲の枠を動かすことで右の拡大画面もその場所に移るので，常にソースコードの全体像と詳細を同時に見ることができる．これにより，プログラムが肥大化した際に全体が見渡しにくくなることを解消できる．

図3 コードの全体表示

5. まとめ

5.1 関連研究との比較

初等教育向けのビジュアルプログラミング環境である ViPPER と関連研究との比較をすると 表 2 のようになる．ViPPER は今回のポイントである実行過程とソースコードとの対比をさせやすい環境を実現できた．そのため，他の研究に比べプログラムの動きとソースコードとの関係の理解を深めやすい環境となった．ロボットを利用し実世界との連携を実現することで，自分が作ったプログラムで目の前のロボットを動かす楽しさを得ることもできる．また，モバイル性にも優れているので，いつでもどこでもプログラミングをすることも可能である．

以上のことにより，小学生でも簡単にプログラミングを学ぶことができる環境を構築することに成功した．この環境を小学校でのプログラミング教育に利用することにより，初等教育における情報教育の充実化が実現できる．

5.2 改善点・追加点

- 関数の定義
- モータ制御の自由化

現在の環境では関数を定義できないので，手続き型言語の機能としては不十分である．関数を定義できるようにし，関数を使った際の実行順序もアニメーションで表すことができれば，さらにプログラミングの理解を深めることができる．

前進・旋回・後退といった用意されている命令では，あらかじめモータの制御が決められている．このモータの制御を自由に設定でいる命令を実装すると，斜めに前進・その場で回転などといった新しい動きをロボットにさせることができる．モータ制御の自由化を実装すれば，自分で新しい動きを作り出すといった考える力を身につけさせることができる．

5.3 今　後

今後の方向としては，まだ小学生に利用してもらったことがないため，実際に使ってもらいフィードバックを受けて初等教育で扱ってもらえるよう改良していく．また，ゲーム性を用いるなどして，より楽しく継続してプログラミングの勉強をしてもらえるような仕組みもとりいれていきたい．

	ViPPER	Mind Storms	Wonder Borg	スペースジョイナーズ
実世界性	○	○	○	×
ロボットの自由度	×	○	△	○
ビジュアル言語	○	○	○	○
手続き型言語の機能	○	△	×	△
実行過程とコードとの対比	○	×	×	△
モバイル性	○	×	○	×
携帯機器向けのインターフェイス	○	×	○	×

表 2　ViPPER と関連研究の比較

参考文献

1) 文部科学省: “学習指導要領における情報教育の改善内容”, 文部科学省資料, 2002.
2) 兼宗 進, 中谷 多哉子, 御手洗 理英, 福井 眞吾, 久野 靖: “初中等教育におけるオブジェクト指向プログラミングの実践と評価”, 情報処理学会誌:プログラミング vol.44 No. SIG13(PRO18), 2003.
3) F. K. Scott: “LEGO MindStorms: Not Just for K-12 Anymore”, available at http://citeseer.ist.psu.edu/600374.html
4) BANDAI: “WonderBorg”, available at http://www.roboken.channel.or.jp/borg/
5) 科学技術振興機構: “インターネットロボット競技会　スペースジョイナーズ”, available at http://www.netrobo.net/index.html