タブレット端末を活用したプログラミング教育（３）
― プログラミング・オン・モバイル ―

Programming Education Using the Tablet Device（III）
- Programming on Mobile -

田近 一郎，本多 一彦，杉江 晶子[1)]，森 博

Ichiro TAJIKA, Kazuhiko HONDA, Akiko SUGIE, Hiroshi MORI

前回本紀要で報告した第 2 報では,「CodeToGo」を利用するプログラミングでの能力向上と「TouchLua」を利用するアルゴリズムの理解を促進する試みを紹介し，これらが学生のスキルアップに有効であることを示した．またマルチタッチ入力や重力センサーが利用可能な 2 つの iPad アプリ「Codea」,「Pythonista」の紹介と将来への展望を示した．

本年度は，実際に「Codea」を使って本格的に教育を行った結果の報告と，タブレット上でのビジュアルプログラミング環境「Hopscotch」利用によるプログラミング教育の実践例を報告する．また，新しい動きとして，PC 上での有力なビジュアルプログラミング環境「Scratch」とほぼ同等な機能を有する iPad アプリ「Pyonkee」が登場した．オブジェクト指向を強く意識させるビジュアルプログラミング環境がタブレットに実装されたことによる新しい可能性（「プログラミング・オン・モバイル」と呼ぶことにする）にも言及した．

We have shown previously the progress of programming capability for programming exercises and the enhancement of understanding of the algorithmic concepts for lectures are accelerated using applications on iPad, “CodeToGo”and“TouchLua”, respectively. We also have briefly mentioned a review about software environments on iPad, “Codea” and “Pythonista”, using multi-touch input and gravity sensor for further development in the programming education.

In the present study, we report a programming exercise using “Codea” in the seminar during a semester and the practical lecture using visual programming environment, “Hopscotch”. New wave for visual programming arises on iPad by “Pyonkee” which is compatible with the famous environment, Scratch on PC. The appearance of “Pyonkee” breaks new field for object-oriented programming and the effectiveness for programming on mobile is discussed.

キーワード : プログラミング教育，タブレット端末，iPad，プログラミング言語，ビジュアルプログラミング，iPad アプリ
programming education, tablet device, iPad, programming language, visual programming, iPad app

１．はじめに

本学では全国に先駆けて学生への iPad の無償貸与を開始した平成 23 年度以来，iPad を教育に活用するさまざまな試みに取り組んできた[1)−10)]．平成 25 年度にはタブレット端末の機動性を活かして，アルゴリズムの理解を促すプログラミング教育について報告するとともに PC にはないタブレット端末特有のインターフェースを利用したプログラミング教育の可能性について紹介した[11)]．平成 26 年度はタッチセンサや重力センサなどタブレット端末の機能を充分生かしたプログラムをタブレット端末上で作成する「プログラミング・オン・モバイル」の実践を行ったので報告する．本論文の構成は以下の通りである．2 章ではソフトウェアキーボードを用いる伝統的なプログラミングアプリケーションを利用したプログラム演習の実践例について述べる．ここではゼミナールの学生を対象にやや高度なシミュレーションプログラムの制作を行った．3 章ではプログラムの構成要素である各種命令が記述されたブロックを組み上げてプログラムを作成する「ビジュアルプログラミング」アプリケーションを利用した初心向けプログラミング教育の実践例について述べる．4 章ではビジュアルプログラミングの新たな動きを紹介し今後の展望を考察する．

２．「Codea」によるプログラミング教育

２.１ ゼミナールにおける活用例

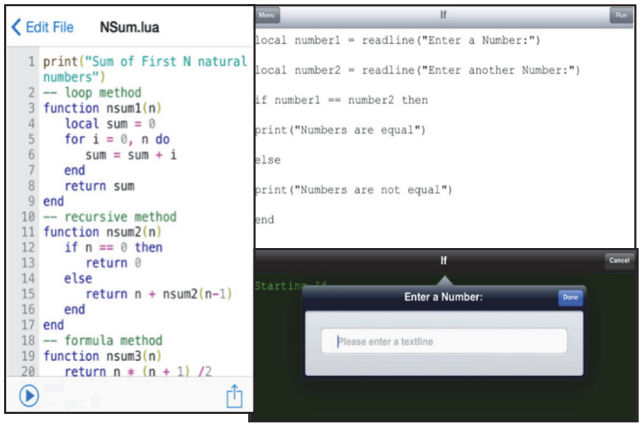
平成 25 年度の報告で iPad 上で利用可能な 2 つのプログラミングアプリケーションである Codea[12)] と Pythonista[13)] を紹介し，プログラミング・オン・モバイル

[1)] 名古屋文理大学短期大学部

の可能性について言及した．本章では，2 年次後期のゼミナールにおいて 2 年生を対象にした Codea を用いたプログラミング演習の実践について報告する．

Codea と Pythonista は共に iPad 上で利用できる高機能プログラミングアプリケーションであるが，Pythonista ではその名前が示すように言語としては Python[14] を用いる．Python は軽量スクリプト言語として人気があり，情報科学の分野はもちろん、自然科学およびデータサイエンス分野でも積極的に利用されているオブジェクト指向言語である．一方の Codea は Lua[15] という言語を利用している．Lua は Python と同様，軽量スクリプト言語であるが，ゲームの AI パートを記述するために用いられるなど，柔軟な言語仕様に特徴がある．Codea では，Lua に独自にクラスが拡張され導入されているが，オブジェクト指向の概念は必須ではない．以上の 2 種類の言語の特性を踏まえ，2 年次 1 年間のゼミナールでは Codea，3～4 年次 2 年間のゼミナールでは Pythonista の選択がそれぞれ適切であると考えた．

2 年次ゼミナールの受講者は 7 名で，大半が Visual Basic や C 言語などのプログラミング言語の演習を履修していた．しかし iPad のように画面のタッチ操作に対応したイベント駆動型のプログラムをどのように作成するか経験を積む必要がある．加えてなじみのない Lua の文法を最低限は理解しなければならない．そこで Codea とは別に Lua を理解するため iPad で利用可能な Luna[16] と TouchLua[17] の 2 種類の Lua アプリケーションを用いることにした．TouchLua は“アルゴリズムとデータ構造”の講義で利用されており，一部の学生にはなじみがあった．また Luna も TouchLua 同様，Lua の処理を記述し，その結果を出力するだけの単純なソフトウェアである．



(a) TouchLua　　(b) Luna

図1　(a)TouchLua と(b)Luna．TouchLua は iPhone 版を拡大表示．Luna は条件分岐によるダイアログを表示した例．

これら Lua アプリケーションでは，コードを書いて即実行するだけで，複雑な操作を覚える必要はなく，C 言語のような既学習のプログラミング言語から無理なく移行できることを期待した．Lua アプリケーションを利用した学習内容は以下の通りである．

- 入出力文の使い方
- 判断文の使い方
- 繰返し文の使い方
- 関数の呼び出しとローカル変数について

判断文，繰返し文については Lua でも C 言語と同等な処理ができることを理解することが目的であった．また関数の呼び出しについては，主プログラムから関数を呼び出す典型的な非イベント駆動型プログラムについて充分理解した上で Codea でのプログラミングと対比できるように計画した．演習としては 3 回を割り当て，それぞれのプログラムの説明とそれを応用したプログラミングを課題とした．少人数のゼミナールであるため，学生の理解度や興味を確かめながら進行したが，Luna や TouchLua に比べて言語体系がはるかに大きく複雑な Codea によるプログラミングを最終目標に設定したため，そこに至る過程が長くなり持続的に学生の興味を喚起することが難しいことを実感した．この点は後述する．Lua アプリケーションによる Lua プログラミング演習後に，Codea による演習を開始した．演習内容は Codea から参照可能なチュートリアルに沿って以下のような演習をおこなった．

- 円を描く
- 円を動かす
- 画面にタッチして円を表示し，指を動かすことで円に速度を与える→壁で跳ね返る

円の描画と移動では，フレームごとに呼び出される draw 関数と，draw 関数の間で呼び出され，画面のタッチを検出する touched 関数が用いられる．演習ではこれらの関数の扱い方を理解し，毎回呼ばれる関数や画面をタッチする状況に応じたプログラムの作成に慣れていく．チュートリアルを順に追っていくと，画面にタッチした指を動かすことで円に初速度を与え動作させるプログラムに行き着く．初速度を与えられた円は四方の壁で跳ね返るようになっている．この跳ね返りの処理は単純なアルゴリズムであるが，アルゴリズムを完全に理解するのは容易ではない．そこで Codea のサウンド付加機能を用い，円が壁で跳ね返る瞬間に音を発するプログラムに改良した後で，跳ね返りアルゴリズムの学習を進めるようにした．Codea でのサウンドの付加は関数を一行加える

だけであり，音の種類を表す関数の引数設定もプログラムを実行する前に確認でき容易である．このように漠然とプログラミングコードを眺めるのではなく，注目する処理に目印をつけることで，アルゴリズムそのものの理解に役立ったと考えている．以上の演習で，“タッチして円を動かし壁で跳ね返るプログラム”の概要を把握できるようになった．

Codea のチュートリアルはその後「パラメータ」と「重力」の項目が並んでいるが，パラメータの利用は上記の円の運動と直接関係のないサンプルプログラムであり，また重力の利用については，ゼミナール実施当時は項目のみで内容が存在していなかった．そこで，“タッチして円を動かし壁で跳ね返るプログラム”に重力による影響を含めるように改良することにした．プログラミングの初歩は，まず既存のプログラムを真似ることから始まるといわれるが，チュートリアルに従った手順はまさにプログラムを動かして理解する段階であった．重力による影響を加える課題になって，自発的にプログラミングを行う段階に入ったことになる．Codea には豊富なサンプルプログラムが含まれており，その中に重力に従って針の画像が伸縮するプログラムがあった．まずこのサンプルプログラムを示し，円の跳ね返りプログラムで，iPad を傾けることで円がその傾きに沿って動くプログラムへの改良をおこなわせる課題を与えた．重力センサを活用したプログラミングは，タブレット端末利用プログラミングの利点の 1 つである．PC は重力センサを組み込まれていないため，重力センサを利用したモバイル機器のアプリを作成する場合，まずプログラムを PC で作成し，重力センサが搭載された機器にプログラミングを転送した上で動作を確認し，不具合があれば再び PC でデバッグを行うという煩雑な作業が必要であった．iPad では，プログラミングと動作確認が同じ機器上で行える．プログラミング初学者の学生にとって画面をタッチし指を動かした方向に円が移動するプログラムでもインパクトのあるものであるが，重力センサの利用はそのプログラムの自然な拡張になっている．プログラムを拡張することで円が指の動かした方向に追随して自由に運動し，iPad を傾けることで運動を変化できるようになった．しかしここまで完成させると，我々が現実の世界で体験する運動に比べ何か違和感があることに気がつく．それは iPad 上を運動する円は決して減衰して止まることがないことである．そこで次の課題は，系に粘性抵抗を導入し，減衰運動をするようにプログラミングすることである．粘性抵抗は描出するフレーム毎の運動に追加することで可能である．実際修正すべきプログラム箇所を示し，プログラムを修正・実行の作業を行ってもらった．試行錯誤でも運動する円の速度が減速する“それらしい”プログラミングが可能である．しかし粘性抵抗を考慮した円の運動については，運動方程式が既知であり，微分形式で記述されたこの運動方程式をコンピュータでのグラフィック表示のようなフレーム毎の差分形式に変換する場合，差分形式のアルゴリズムによる正確な表現が存在することも付け加え解説した．

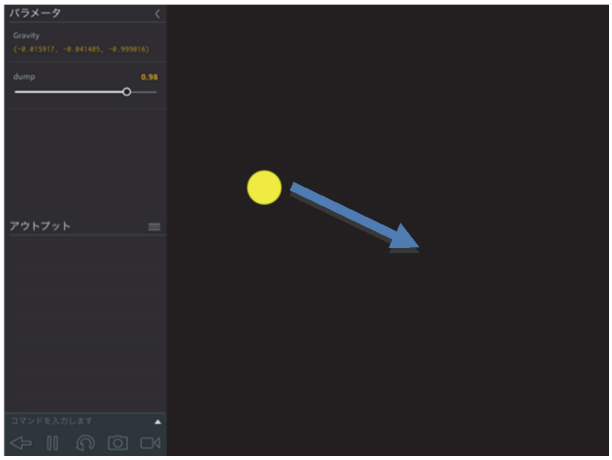


図2　Codea の実行画面．円をタッチして任意の方向に速度を与えることができる．左端上は (x,y,z) 方向の加速度を表示している．その下のスライドバーを操作することで，円の粘性抵抗を変えることができる．

円の運動について重力の影響と粘性による減衰の効果を付け加えることができたが，しばらく円の運動を辿ってみると動きとしてさらに不自然な箇所があることがわかった．それは元のサンプルプログラムでは四方で円が跳ね返る運動であったが，重力と粘性抵抗のため iPad を傾けた状態で保持すれば境界で円が静止するのが自然である．しかし iPad の画面上，フレーム毎に送られてくる重力の情報と粘性抵抗の処理のバランスから円は境界で静止せず微小振動が止まらない現象が現れた．この現象を抑えることが次の課題となった．具体的には，四方の境界域で閾値以下では動きを抑制するようにプログラムすることになる．四方の境界域で円の運動を抑制するプログラミングに学生たちはかなり苦労していたが，この課題を通して得られものは大きかった．ある学生のプログラムでは iPad を傾けて円を隅に追いやると，円は振動運動を始めその振幅が小さくなっていった後，円が突如消失してしまい全く別の隅から円が出現するプログラミングコードになってしまった．これは境界域での単純な処理の誤りではあるが，実行時のアルゴリズムの誤りを

示す重要な体験となった．プログラミング初学者にとって，プログラム中の文法の誤りと実行時のアルゴリズム自体の誤りの違いはあまり区別がつかない．プログラム中の文法の誤りに意識が行きがちであるが，この円の運動では明らかに学生が想定した動きとは異なっており，視覚的に実行時の誤りを体験できたことになる．この例では思った通りでない誤ったプログラムを作成したことになるが，iPad を傾けて円を隅に追いやると別の隅から円が出現するアルゴリズムは視覚的に面白く，ゲーム作成の場合などでは活用の可能性があるように思われた．重力の影響と粘性による減衰の効果をプログラミングした後，最後にチュートリアルにあったパラメータの設定を組み込むことにした．パラメータとして粘度を設定することで，ソースコードを変更することなく円の運動の減衰の度合いを制御できるようになった．

以下では非イベント駆動型プログラムからイベント駆動型プログラムへステップアップを図る際の困難性についてまとめる．今回はゼミナールにおける iPad を利用したプログラミング・オン・モバイルの最初の適用例となるため，プログラミングの手順を踏んで Luna や TouchLua などの Lua アプリケーションを用いて Codea での学習を行う前の基礎固めにも努めた．しかし，非イベント駆動型からイベント駆動型プログラムといった学習上の流れは，教える側の理論体系であってもプログラミング初心者にとって理解しやすい手順であるとは限らない．特にスマートフォンやゲーム機と日常的に接している世代にとっては直接 Codea を利用してプログラミングの経験を増やしていく方が効果的であるようであった．時間の関係で今回出現させた円は１つだけであり，今回のゼミナールの内容はタッチセンサや重力センサを利用したプログラミング入門であった．しかし，粘度が異なった複数の円を生成させそれらの円の当たり判定を行えば重力を利用したゲームに簡単に拡張できる．そのためより高いプログラミングの到達度を設定しても工夫次第で充分克服可能であり達成感の向上も期待できると感じた．

２.２ ゼミナールを終えて

ゼミナールを終え，学生には演習内容をまとめたレポートの提出を求めた．その中で Codea を利用したプログラミングは他のプログラミングの演習や他のパソコン演習と比べてどうだったか，利点と欠点をまとめるように指示した．いつでもどこでもプログラミングができることを全ての学生が利点として挙げていた．一方でキーボードレスの環境での入力の大変さについても全ての学生が欠点として挙げていた．キーボードに関しては無線接続のハードウェアキーボードも安価で入手できるようになってきているので，重量は増加するがキーボードを携帯することで欠点は解消できると考えている．一方“どんな姿勢でも利用できる”というコメントが１件あった．この“どんな姿勢でも”というのは，“どこでもプログラミング”というコメントと若干ニュアンスが異なる．“どんな姿勢でも”では，極端ではあるが寝ながらプログラミングも含まれることになる．プログラミングでは既存のプログラムを真似ることが始まるので，入力・実行の作業が円滑に行えることは重要である．したがってハードウェアキーボードの利用は充分価値がある．しかし自らコードを書き始めると，入力のための障壁よりもアルゴリズムをどのように組み立てるかが問題となることもあり，コードを見て考えている方が長い場合がある．iPad では，就寝前に急によいアルゴリズムを思いつけば寝ながらでもすぐに新たなコードを試してみることができる．寝ながらの姿勢では画面タッチとハードウェアキーボードの併用よりソフトウェアキーボードの単独利用が便利である．ユーザとの近さがモバイル機器の最大の利点であり，その利点を活かすことがプログラミング・オン・モバイル成功の鍵となる．

Codea の機能に関連して複数の学生が問題点として挙げていたことは，プログラミングをする際の関数パラメータ指定の誤動作の問題である．例えばグラフィクスの色を指定する際，RGB を数値だけで直感的に理解するには経験が必要である．Codea ではグラフィクス描出関数に与える RGB パラメータは，希望する色と RGB それぞれの値を併せて確認できるような編集機能を備えている．便利なように思われるが，コードを書いていて誤って関数のパラメータ部分をタッチしてしまうと意図しないパラメータ設定のモードとなり作業が中断されることになった．ハードウェキーボードではカーソルキーが利用できるので，こうした問題は回避できるが，上で述べたようにハードウェアキーボードなしの環境でも効果的に iPad が使えることが望ましい．演習を行った当時に比べ，現在の Codea ではパラメータ設定の利便性は残しつつタッチでも誤動作しにくいように改善されている．

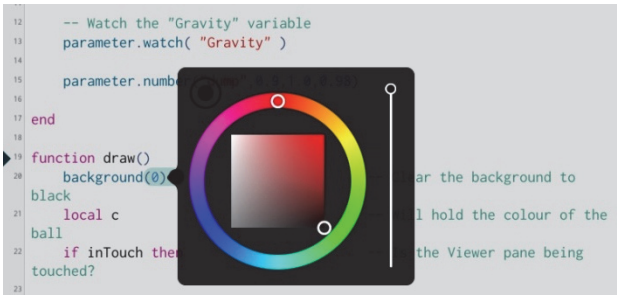


図3「カラーピッカー」による色の指定

なお，Codea ではオンラインマニュアルだけでなく，開発者と利用者が意見を交換するフォーラムが存在し，Codea から簡単にアクセスできるようになっている．フォーラムでの活発な議論が Codea の改善に大いに貢献しているようである．

PC で既にプログラミングの経験がある学生のレポートでは iPad を利用したプログラミングの利点として環境整備の容易さを挙げている．PC では様々なプログラミング言語を利用でき，またエディタを含め開発環境を自分に合うようにカスタマイズできる．しかしその自由度のためどのような設定をすればよいか，また様々なプログラミングツール間のバージョンの整合性の問題など，プログラミング初心者にとって難しい問題が存在する．iPad ではソフトウェア間の連携機能は乏しい，いわゆるオールインワン型のアプリケーションとなる．しかしよいプログラミング開発環境ソフトウェアであれば，オールインワン型アプリケーションの簡便さが活かされることになり，今回利用した Codea はその数少ない例であるといえる．

ゼミナールでのプログラミング演習は，多人数で利用するパソコン室と異なった“どこでも演習室”の実践でもあった．学生のレポートで，実習室で行うプログラミング演習と比べゼミナールを行っている小教室での欠点として，“提示装置がなくプログラミングの進度がわかりにくかった”という意見があった．本学の実習室ではパソコン 2 台の中間に提示装置を配置しており，教員のディスプレイや書画装置の映像を参考に演習を進めていけるようになっている．ゼミナールではこうした提示装置を用いずにプログラミングの解説を行った．主にホワイトボードを用い，熟考する必要があった円の跳ね返りのアルゴリズムについては紙の資料を用意した．しかし提示装置の利便性は大きかったようである．現在のゼミナールでは短焦点距離のプロジェクタを活用しており，演習室の提示装置の代用になっている．加えて演習室と違い少人数であるので，個々人の iPad をプロジェクタに接続してゼミナールでの発表やディスカッションにも活用している．個々人の iPad の画像を無線で動的に切り替えてプロジェクタに投影できる機器を活用すれば，ディスカッションのさらなる活性化も期待できると考えている．タブレット端末は機能的な制限が大きいため，他の機器との連携を行うことが重要であるが失敗も多い．しかし失敗の積み重ねが教育システムそのものを変えていく可能性を秘めていると考えており，今後も新たな試みとその結果を逐次報告していく予定である．

3．「Hopscotch」によるビジュアルプログラミング教育

3．1 iPad を活用するビジュアルプログラミングの試み

本学では高校教育から大学教育への速やかな移行を目指す高大連携事業の一環として高校での出前授業をおこなっている．情報メディア学科の出前授業では特に，他大学に先駆けて iPad を教育に導入した本学が実践しているさまざまな iPad 活用授業の実際を高校生徒にも体験してもらっている．著者の一人も愛知県立起工業高校にて定時制課程（昼間）普通科クラスの 20 名の生徒を対象に iPad によるビジュアルプログラミングの出前授業をおこなった（2013 年 12 月 12 日 10:45～12:15）．授業は，ビジュアルプログラミングアプリケーション（以下，ビジュアルプログラミングアプリと略記）をインストールした iPad を生徒二人に一台ずつ配布し，著者の一人が用意したプリントに沿ってゲームのプログラムを作成するものとした．従来のプログラムをキーボードからテキスト入力するというプログラム初心者にとって煩雑な作業を伴わないビジュアルプログラミングの採用により，プログラム作成の本質的なところを押さえつつ気軽にプログラミングを体験してもらい，同時にプログラミングに対する興味，関心や楽しさを感じてもらうことを目指した．

まず一般的なビジュアルプログラミングについて解説する．従来のプログラミングでは，プログラムは変数・制御構造を表すキーワードや処理の対象である数や文字列などから構成され，それらをキーボードからのテキスト入力により編集することでプログラムを作成する．それに対してビジュアルプログラミングでは，変数・制御構造は特定の形をした「ブロック」として，また数や文字列も「ブロック」内の入力用テキストボックス内にデフォルト値がそれぞれあらかじめ用意されている．PC 上で数や文字列を変更するときは「ブロック」内のテキストボックスをクリックし，キーボードから適切な数や文字列を入力する．iPad 上では「ブロック」内のテキストボックスをタップしてソフトウェアキーボードを表示さ

せて入力する．プログラムの作成ではそれらの「ブロック」をジグソーパズルのように画面上で適切に結合してゆく．特に，iPad 上で動作するビジュアルプログラミングアプリの多くでは，通常の「ブロック」の他にタブレット端末特有のタップ操作や傾きセンサ等のユーザインターフェースを介した入力に対して処理を実行する「ブロック」も用意されている．ビジュアルプログラミングが可能な iPad アプリとして現時点（2014 年 9 月 30 日）で KineScript[18]，Hopscotch[19]，ScratchJr[20]，Pyonkee[21]等がある．ただし，出前授業では，その実施時点で安定動作をしており操作性が高い Hopscotch を採用した．なお，Hopscotch のトップ画面とビジュアルプログラミング時の編集画面はそれぞれ以下の通りである．

図4 「Hopscotch」のトップ画面

図5 「Hopscotch」のプログラム編集画面

生徒がおこなった Hopscotch によるゲームプログラムの作成過程について説明する．ゲームの内容は画面の最上部のスタートラインに立つキャラクタ「サル」を，iPad を適切な向きに随時傾けることで，進行を妨害する敵キャラクタ「スペースポッド」をうまく避けながら，画面最下部のゴールラインに向かわせるというものである．ゲームプログラムは以下の 3 要素から構成される．オブジェクト指向プログラミングの考え方に基づき，ゲームに出現するキャラクター一体一体に必要なプログラムを追加する形式でプログラムを構成する．

（1）一つ目は iPad の傾きセンサの入力によりキャラクタ「サル」を iPad が傾いた向きに移動させるプログラムである．このプログラムはキャラクタ「サル」に実装する．図 6 のように iPad の傾きセンサからの入力に反応する「制御ブロック」を利用してプログラムを作成する．

図6 「サル」の動作用プログラム

（2）2 つ目は左右に往復運動しキャラクタ「サル」の進行を妨害する敵キャラクタ「スペースポッド」の動作プログラム（図 7）およびキャラクタ「サル」が敵キャラクタ「スペースポッド」に当たりゲームオーバーとなるときの当たり判定用のプログラム（図 8）である．共に敵キャラクタ「スペースポッド」に実装する．

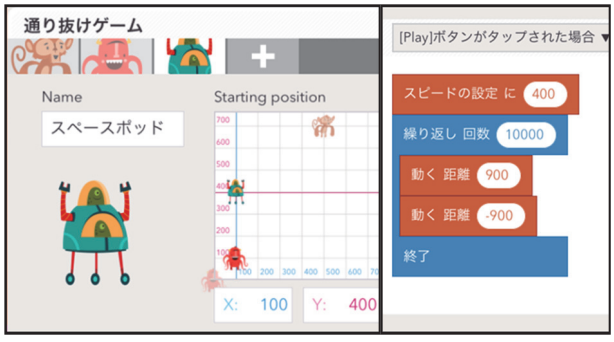


図7 「スペースポッド」の動作プログラム

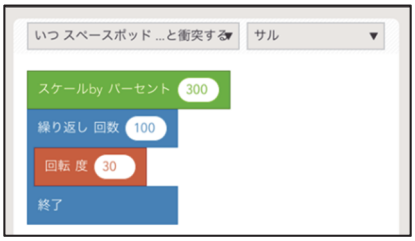


図8 「スペースポッド」のゲームオーバー時のプログラム

（3）3つ目はキャラクタ「サル」がゴールラインに到達したらアニメーションでゲームクリアを表示するプログラムである（図9）．当たり判定用のプログラムをベースに作成したものでキャラクタ「ゴール」に実装する．

図9 「ゴール」のゴール判定用プログラム

以上のプログラムをプリントに沿って順次作成した．

3.2 ビジュアルプログラミングに関する調査結果

ビジュアルプログラミングの出前授業（2013年12月12日）を受講した高校生徒20名に対し，授業直後にアンケートを実施し全員から回答を得た．以下に集計結果と考察をまとめる．なお，Q1，Q2，Q3は著者の一人が用意したアンケートの集計結果であり，Q4，Q5は高校側が実施したアンケートの集計結果を山田隆司教諭より提供していただいたものである．

Q1. プログラミング体験の有無

4分の1の生徒がプログラミングを体験している．ただし，自由記述によると体験の多くはHTMLファイルの編集やエクセルの関数設定などで，C言語やJavaなどはごく少数である．

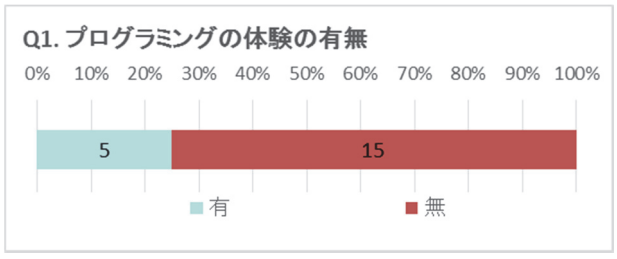


図10 プログラミング体験の有無

Q2. ビジュアルプログラミングの理解度

8割以上の生徒がビジュアルプログラミングの内容を理解しながら進めている．

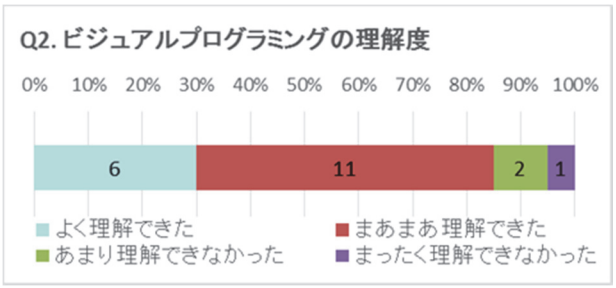


図11 ビジュアルプログラミングの理解度

Q3. ビジュアルプログラミング作業の大変さ

試行錯誤を繰り返しながら望み通りに動作するプログラムへと近づけていくというプログラミング初心者にとっての大変さはビジュアルプログラミングの場合も変わらないことが見て取れる．このようなプログラミングの大変さを3分の2の生徒が感じている．ただし，Q2で見たように最終的には多くの生徒が一定の理解に達していることからビジュアルプログラミングによりテキスト入力に煩わされずにプログラムの仕組みそのものに集中できていることがうかがえる．

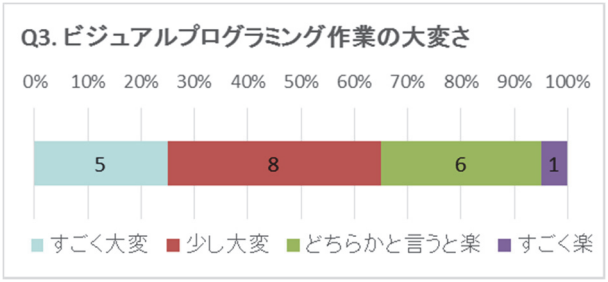


図12 ビジュアルプログラミング作業の大変さ

Q4. ビジュアルプログラミングへの興味・関心

半数弱の生徒がビジュアルプログラミングに一定以上の興味・関心をいだいている．

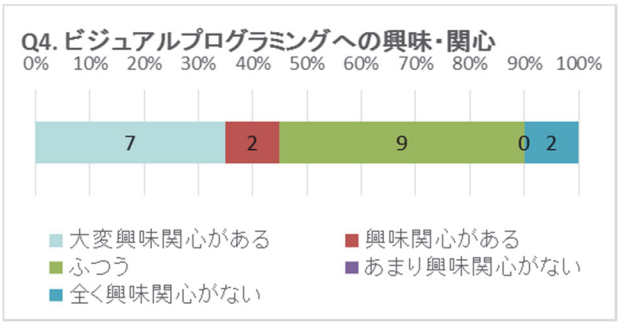


図13 ビジュアルプログラミングへの興味・関心

Q5. ビジュアルプログラミングの楽しさ

3分の2以上の生徒がビジュアルプログラミングに楽しさを感じている．従来のプログラミングとは違って，

プログラミング作業を大変に思いつつも楽しく進めていけるというビジュアルプログラミングの特徴が表れていると思われる．

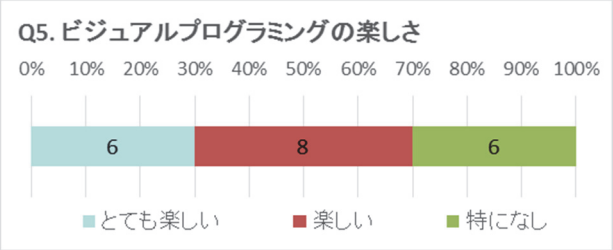


図 14 ビジュアルプログラミングの楽しさ

4．新しい動きについて

4.1 iPad アプリ「Pyonkee」などの登場

前章までで，タブレット型 PC を活用したプログラミング教育において，「ビジュアルプログラミング」が有効であることはかなり明確になってきた．非タブレット型 PC においては，MIT が提供している「Scratch」[22]がビジュアルプログラミング環境の定番として定着してきている．2014 年 8 月の時点では，iPad で動作し「Scratch1.4」とほぼ同等の機能と外観を備えた iOS 向けアプリケーションである「Pyonkee」[18]が登場している．

MIT メディアラボのレズニックが作った「Scratch1.4」は，子供向けプログラミング環境として有名であるが，そのオブジェクト指向を強く意識したプログラミングスタイル[23]は，高等学校や大学におけるプログラミング教育ツールとしても興味深いものがある．「Scratch1.4」とほぼ同等の環境を iPad 上でも実現した「Pyonkee」は，その意味で，高等教育機関においても従来のプログラミング教育とは一味違ったものが展開できる可能性を感じさせるものである．

図 15 「Pyonkee」の編集画面

「Pyonkee」は，「Scratch1.4」同様，スプライトと呼ばれるステージ上のキャラクタ（オブジェクト）に対し，8 つのカテゴリーに分類されたブロックを組み込んでいく，または，ブロックの中のパラメータ部分の数値を変化させていくことでコードを書くことなくプログラミングしていくことができる．この点では他のビジュアルプログラミングと同じである．しかし，「ペン」カテゴリーの中の「ペンを下ろす」ブロックを活用することにより，「Pyonkee」上で「タートルグラフィックス」を実現することができる．別途ライブラリを追加することなく「タートルグラフィックス」をプログラミングすることにより，オブジェクト指向プログラミングを直感的に体験していくことが可能となる．また，「見た目」カテゴリーの「隠す」ブロックを使えば，スプライト自体も見えないようにすることができ，純粋なタートルグラフィックスの軌跡だけを表示することも可能となっている．

図 16 に示すスクリプトはステージ上の中央座標 (0, 0) を起点として，正 8 角形の各辺を 1 辺とした正 5 角形を順に 8 個タートルグラフィックスとして描かせるものである．

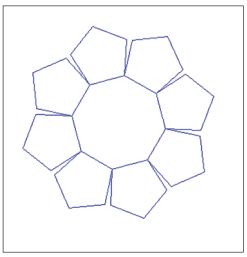


図 16 スクリプト例　　図 17 スクリプトの実行結果

具体的には 1 辺 50 ドットの正 5 角形を左回りに描いた後，50 ドット進んで右に 45 度回転させることを 8 回繰り返す 2 重ループで実現している．スクリプト先頭の「消す」ブロックはステージ上のすべての描画を消去させるもので，最後の「隠す」ブロックはキャラクタを隠して描画のみを残すための命令である．実行結果を図 17 に示す．

タートルグラフィックスが「Pyonkee」で使えることのメリットは何であろう．ビジュアルプログラミングであるため，プログラム作成の容易さと実行結果がすぐに目に見えることがそのメリットの 1 つである．しかし最も重要な点は，一般的なコンピュータグラッフィクスプログラミングでは，固定されたステージ上の x，y 座標を指

定することにより描画するのに対し，タートルグラフィックスでは，常に現在のタートル（亀）の位置と向きからの進む距離と角度の指定で描画するという違いがあることである．言い換えると，座標が書かれたキャンバスを上方から俯瞰して図形等を描画するのと，そのキャンバス上の描画点に自分を置き，そこからの相対的な座標をもとに描画する違いである．タブレット上に構築されている「Pyonkee」ではキャンバス上で亀の進む向きを自由に変えたとき，それに合わせてタブレットを回転させ，亀の進む向きを常に一定にして描画方法を考えることができる．亀中心の相対座標で考えられるうえに，ディスプレイ上で亀の進む向きと人間の向きを同一にすることによりさらに亀の動作の理解が容易になる．これは人間とタブレットを一体化させ，プログラミングに没頭させる可能性を示唆しており非常に興味深いテーマである．

タートルグラフィックスによってオブジェクト概念の中心にスプライトを据える場合，オブジェクトの機能として他にどのようなものが考えられるだろうか．プログラムからディスプレイへ出力させる際，従来型のプログラミング言語の教科書では，例えばコンソールディスプレイ上に「Hello, world!」を出力させるため，出力用の関数を用いるコードを記述させることが多い．これに対し「Pyonkee」ではスプライトの発言として出力させることになる．このようにオブジェクトに動きの機能だけでなくオブジェクトから（メッセージを）出力するという機能が加わることで，オブジェクトの多機能な特質が理解でき，オブジェクト指向らしさを自然に習得することが可能となっている．さらにスプライトの種類を工夫することで「出力」のバリエーションが広がっておもしろい作品を作ることが可能となる．

図 18 「Hello,world!」と表示させる

また，オブジェクト指向プログラミングのひとつの特徴である「オブジェクトにメッセージを送る」という部分にスポットをあて，「あいさつ」というメッセージをキャラクタに送ると，それを受け取ったキャラクタが「Hello,world!」と出力するようなスクリプトも作成できる．

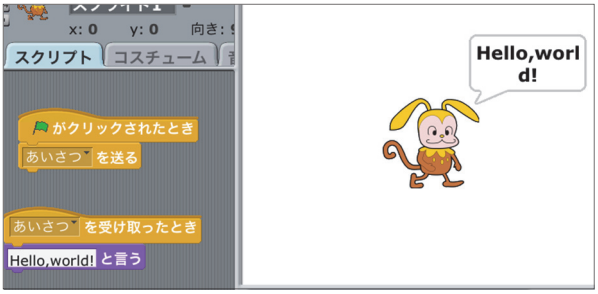


図 19 メッセージを送ることにより「Hello,world!」と出力

4.2 ビジュアルプログラミングに不向きな例題

さて，このように「Pyonkee」は，従来型プログラミング教育にも対応し，オブジェクト指向プログラミング教育にも適応可能で万能のように思われるが，意外にも比較的単純な計算処理には不向きな点もみられる．

従来型プログラミング教育の初期段階で課題となることが多い摂氏・華氏変換などの単位変換プログラムでは，ビジュアルプログラミングの弱点も見えてくる．次のスクリプトは，「摂氏」温度を入力させ，「華氏」温度に変換して出力する一例である．

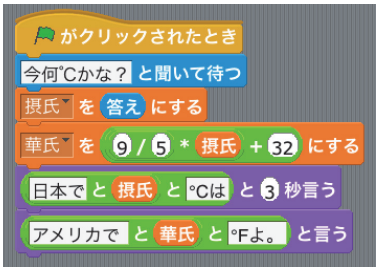


図 20 摂氏温度から華氏温度へ変換するスクリプト例

ここで「答え」は，「調べる」カテゴリーにデフォルトで存在する変数で，「摂氏」と「華氏」は，「変数」カテゴリーで新しい変数として作成した．

F = 9 / 5 * C + 32 という変換式を「Pyonkee」上で作成するにはやや複雑で，計算の優先順位を誤ると正しい結果を得られない．コードによるプログラミングでは，式が自由に記述できるうえに，() (括弧) を使うことにより計算の優先順位を正確に記述できる．一方でビジュアルプログラミングでは，この優先順位の設定がコードに比べて煩雑であり，「Pyonkee」の欠点の一つといえる．

プログラムを実行すると，下図のようにデータの入力待ちになる．

図 21　摂氏温度の入力待ち画面

ソフトキーボードから数値を入力しチェックボタンをタッチすると下図のように計算結果が表示される．

図 22　計算結果として華氏温度が表示された画面

計算式が複雑になってくるアルゴリズムを実装する場合，ブロックのパラメータの値を変更したり，加減乗除の単純なブロックを組み合わせて計算式を構成していく作業は想像以上に苦痛な作業で，ブロックを合成すると計算順序がほぼ判別不能となるため，計算結果の正当性を常にチェックしていないと正しいプログラムが作成できない．括弧を示すブロックがぜひ欲しいところである．複雑な式は自由にコード入力できる汎用計算式ブロックがあるとさらに活用の幅が広がるであろう．

また， iPad のルールにより，「Pyonkee」に限らず多くのプログラミングアプリケーションではアプリ外部からのソースコードの読み込みを禁止している点もプログラミング教育ではマイナスであろう．ただしこの問題については，教育機関に対してその問題をクリアしたバージョンを別途配付する試みもあるらしい．教育機関で iPad アプリを使用する場合の新しいスタイルとなる可能性もある．

５．プログラミング教育の展望

タブレット端末を教育に使用する利点として，いつでもどこでもすぐに利用できるといった機動性については本紀要のシリーズで述べてきたことである．本論文では，プログラミング・オン・モバイルの教育事例，ビジュアルプログラミングの新しい流れと教育事例を紹介し，プログラミング・オン・モバイルがプログラミングの楽しさと同時にプログラミングの理解の向上に役立つであろうことを示唆した．しかし，iPad だけでも複数のビジュアルプログラミングアプリケーションが存在し，個々のアプリの特徴について充分な知識が得られていない．この点はアプリケーション自体が発展の段階であり，今後もプログラミング教育での実践を基に検証をしていく必要がある．

ビジュアルプログラミングは新しい流れであるものの，実社会においては伝統的な文字ベースのコーディングが主流であることには変わりがない．ビジュアルプログラミングでアルゴリズム等の基礎的概念を把握した後，いかに文字ベースの中～大規模なプログラミングに接続していくかが課題となる．現行カリキュラムでは 1 年次後期に Visual Basic，2 年次前期・後期に C 言語の演習を行っている．そこでプログラミング教育の効果を高める方策として，上記のプログラム演習科目の履修で理解できなかった部分を克服してもらうためのビジュアルプログラミング演習による支援，それに加えて再度文字ベースのコーディングについても知識と技能の向上を支援していく必要がある．これらの問題については今後の課題としたい．

参考文献

1) 本多一彦:「モバイル機器の変遷から情報教育機器としての iPad を考察する」，名古屋文理大学紀要，11，97-104 (2011)
2) 森博，田近一郎，杉江晶子 :「タブレット PC を活用したマルチメディア教育の試み」，名古屋文理大学紀要，12, 97-104 (2012)
3) 佐原理，大橋平和，長谷川旭，長谷川聡，KAISER Meagan :「タブレット端末による学校教育現場向け多言語情報配信システム」，名古屋文理大学紀要，12, 105-112 (2012)
4) 長谷川旭，佐原理，尾崎志津子，本多一彦，山住冨也，長谷川聡 :「名古屋文理大学における iPad 導入とアクティブラーニング」，モバイル学会研究報告集，7(2), 45-48 (2011)
5) 長谷川旭，長谷川聡，本多一彦，山住富也，佐原理:「大学教育でのタブレット端末の利用とその効果—iPad を無償配布した名古屋文理大学における学生意識」，コンピュータ＆エデュケーション，31，70-73, (2011)
6) 尾崎志津子 :「iPad を活用したオンライン英語多読の

導入―名古屋文理大学情報メディア学科における事例―」, コンピュータ&エデュケーション. ; 32, 49-52 (2012)
7) 長谷川聡:「ソーシャルリーディングとソーシャルラーニング」, 現代の図書館, 50(2), 114-120 (2012)
8) 長谷川旭, 小橋一秀, 山住富也, 長谷川聡 :「タブレット端末の教育利用と情報インフラ:名古屋文理大のiPad 無償配布と大学図書館」, 医学図書館, 59(3), 186-191, (2012)
9) 斉藤徹, 河原潤, 高下義弘:「教える! 名古屋文理大学」, 『iPad で現場を変える!』, 日本経済新聞出版, 132-147 (2011)
10) 本多一彦, 田近一郎, 杉江晶子, 森博 :「タブレット端末を活用したプログラミング教育」, 名古屋文理大学紀要, 13, 85-92 (2013)
11) 田近一郎, 本多一彦, 杉江晶子, 森博 :「タブレット端末を活用したプログラミング教育 (2)」, 名古屋文理大学紀要, 14, 75-86 (2014)
12) Codea,
https://itunes.apple.com/jp/app/codea/id439571171 より 2014 年 10 月 21 日検索
13) Pythonista,
https://itunes.apple.com/jp/app/pythonista/id528579881 より 2014 年 10 月 21 日検索
14) Python,
公式 web サイト https://www.python.org/ より 2014 年 10 月 21 日検索
15) Lua,
公式 web サイト http://www.lua.org/ より 2014 年 10 月 21 日検索
16) Luna,
https://itunes.apple.com/jp/app/luna/id396308761 より 2014 年 10 月 21 日検索
17) TouchLua,
https://itunes.apple.com/jp/app/touch-lua/id525273327 より 2014 年 10 月 21 日検索
18) KineScript,
https://itunes.apple.com/jp/app/kinescript-visual-programming/id674887500 より 2014 年 9 月 30 日検索
19) Hopscotch,
https://itunes.apple.com/us/app/hopscotch-programming-designed/id617098629 より 2014 年 9 月 30 日検索
20) ScratchJr,
https://itunes.apple.com/jp/app/scratchjr/id895485086 より 2014 年 9 月 30 日検索
21) Pyonkee,
https://itunes.apple.com/jp/app/pyonki/id905012686 より 2014 年 9 月 30 日検索
22) Scratch,
http://scratch.mit.edu/ より 2014 年 10 月 21 日検索
23) 阿部和広 :「簡単だけど奥深い ! Scratch プログラミング」, http://itpro.nikkeibp.co.jp/article/COLUMN/20111019/371081/ より 2014 年 10 月 21 日検索