Vol.2023-GN-118 No.52
Vol.2023-CDS-36 No.52
Vol.2023-DCC-33 No.52
2023/1/24

複数人でのプログラミング体験を向上させる Scratch上での共同実行環境とその評価

古家 一樹[1]　市村 真希[2]　高田 秀志[2]

概要：近年情報化社会の発展により，2020 年度から，小中学校においてプログラミング学習が必修化された．それに伴い，全国で児童向けプログラミングワークショップが多く開催され，プログラミング体験を向上させる要求が高まっている．本研究では，プログラミング体験の向上を支援するために，Scratch の共同実行環境を実現する．共同実行としては，Scratch 上で実現された対戦ゲームなどを複数の PC で同時に実行できることを目的とする．共同実行環境を実現するには，実行画面の情報を共有する機能が必要である．本稿では，Scratch 上で描画されているオブジェクトの座標情報などを端末間で共有することにより実現したシステムについて述べる．また，小中学生を対象に行ったワークショップで本システムを評価した結果を紹介する．

キーワード：プログラミング学習，プログラミング体験向上，共同実行

Co-execution Environment on Scratch Enhancing Programming Experience with Peers and its Evaluation

Abstract: In recent years, programming has become compulsory in elementary and junior high schools from the 2020 school year due to the development of the information society. In line with this change, many programming workshops for children have been held throughout Japan. As opportunities to learn programming are increasing, there is a growing demand to enhance the programming experience. In this study, we realize a collaborative execution environment for Scratch to support the enhancement of programming experience. The goal of the collaborative execution is to enable multiple PCs to simultaneously run games such as competitive games implemented in Scratch. In order to realize a collaborative execution environment, it is necessary to have a function to share information on the execution screen. This paper describes a system that shares information such as the coordinates of objects drawn in Scratch between terminals. We also present the results of a workshop for elementary and junior high school students in which this system was evaluated.

Keywords: Programming learning, Enhancing programming experience, Co-execution

1. はじめに

近年，情報化社会の発展に伴い，プログラミングの重要性が一般の人々にも広まりつつある．例えば，日本においては 2020 年度から，小中学校においてプログラミング学習が必修化された [1]．それに伴い，全国で児童向けプログラミングワークショップが多く開催されている．我々の研究グループは 15 年以上に渡って，NPO 法人スーパーサイエンスキッズと協力し，児童向けプログラミングワークショップを開催している．

ワークショップでは，一人でプログラムを書いたり実行したりすることが多い．これに対して，市販のゲームが複数人でプレイできるように，自分で作ったプログラムも複数人でプレイすることで，より楽しくプログラミングを進めていくことができるようになり，プログラミング体験が向上すると考えられる．

2019 年度の中学校のプログラミング教育に使われた教

[1] 立命館大学大学院情報理工学研究科
Graduate School of Information Science and Engineering, Ritsumeikan University, Kusatsu, Shiga 525-8577, Japan
[2] 立命館大学情報理工学部
College of Information Science and Engineering, Ritsumeikan University, Kusatsu, Shiga 525-8577, Japan

Vol.2023-GN-118 No.52
Vol.2023-CDS-36 No.52
Vol.2023-DCC-33 No.52
2023/1/24

材は，Scratch[2] が 38.3 %であり，最も高い [3]．Scratch では，Imagine，Create，Share，Reflect のプロセスで構成されている Creative Learning Spiral[4] が重要視されている．この中で，Create の部分に対しては，複数人でプログラムを書くことができる Progummy[5] というシステムが構築されている．一方で，Play に対しては，作成したプログラムを共同複数人で同時に実行するような環境はない．

そこで本稿では，プログラミング体験の向上を目的として，Scratch の共同実行環境を実現する手法を提案する．共同実行としては，Scratch 上で実現された対戦ゲームなど，一つのプログラムを複数の PC で同時に実行できることを目標とする．また，本手法を実際のプログラミングワークショップの PC で利用し，プログラミングの体験向上にどの程度効果があるかを評価した結果について述べる．

2. 研究背景

2.1 Scratch

2.1.1 Scratch 上でのプログラミングと実行

図 1 を用いて，Scratch でプログラミングを行う方法を説明する．Scratch では，赤枠で囲んだ部分のように，「10 歩動かす」「どこかの場所へ行く」などのブロックを組み合わせていくことによってプログラムを作成していく．作成したプログラムの実行結果は，紫で囲んだ部分（ステージと呼ばれる）に表示される．

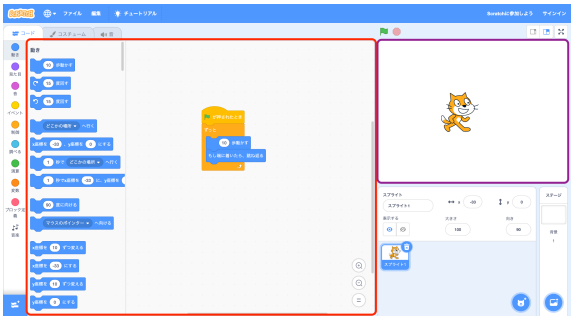

図 1 Scratch

2.1.2 プログラムの共同編集

Scratch には，複数人でもプログラミングできるような取り組みが行われているものがある．例えば，Progummy というサービスでは，複数人が共同でスクリプトの共同編集を行うことができる．

図 2 は，Progummy でスクリプトの共同編集をおこなっている様子を表している．図の 2 つのウィンドウは，別の PC で表示されている Scratch の画面である．図中の丸で囲まれた部分に示されているように，右側の PC の利用者がブロックを選択して移動させると，左側の PC に表示されている同じブロックが半透明で表示され，同じように移動する．また，右側の PC の利用者が移動後にブロックを離すと，左側の PC では半透明の表示が解除され，移動操作が完了する．このような機能により，複数人が共同で同時にプログラムを作成できるようになっている．

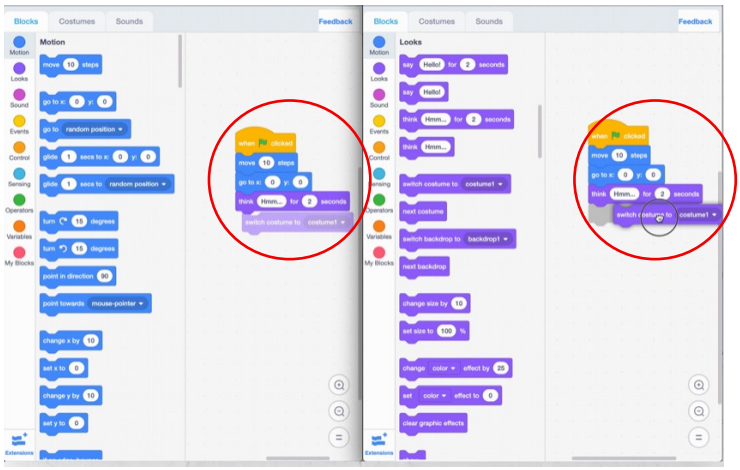

図 2 Progummy （[5] から引用）

2.1.3 プログラムの共同実行

Progummy では，プログラムの共同編集が行えるようになっているのに対して，本研究では，作成したプログラムを共同で実行できるようにする．共同実行とは，図 1 のステージ部分について，「一つのプログラムを複数の PC 上で同時に実行できる」ということ表す．

図 3 は，共同実行で想定するプログラムの実行画面である．このプログラムは，図中に示すボールを蹴るキッカーと，ボールを守るキーパーのそれぞれを，異なる PC 上で操作して対戦を行うサッカーの PK のゲームである．キッカーを操作する人は，ゴールのどこを狙うのかを指定してボールを蹴る．キーパーを操作する人は，左右に動かしてゴールを守る．このように，複数人で一つのプログラムを同時に実行し，それぞれの PC 上で操作を行えるようにすることを共同実行としている．

図 3 共同実行で想定する画面例

2.2 関連研究

双方向性のあるコンテンツの考え方と，そこに含まれる情報伝達の仕組みを Scratch 上で学ぶためのシステムとして， Scratch の Mesh 機能を用いた双方向性のプログラミング教材の開発が行われている [6]．この研究では，Scratch 1.4 が持つ Mesh 機能を用いて複数の PC を接続し，双方向性のプログラミングを行なっている．Scratch でのネットワーク通信は，Host がサーバ機能を提供し，クライアントが Host に接続することで実現している．こうして接続された全ての Scratch は変数を共有することができる．こうすることで，互いに接続された Scratch プログラム間で双方向性のある情報伝達ができるようになっている．

このように，変数を共有することで双方向性の通信が可能になるが，対戦ゲームなどを実現するには，共有する変数を事前に決めてプログラミングを行う必要があるなど，難易度が高い．本稿で述べる手法により，事前に共有する変数を決めることなく双方向性のある通信を可能にすることを目指す．

また，小学校のプログラミング教育において学習状況を共有することの効果について調べられている?．この研究では，黒板の前のスクリーンにクラス内でプログラミングの状況と作品名を投影して共有する．プログラミングの方法は，IoT ブロックを組み合わせて操作を行うプログラミングツール MESH [7] を利用している．このシステムを利用することで，他のグループの工夫に気づきやすくなることや，使用されたブロックの数と種類の平均値が高くなることが示されている．

学習状況を共有することで，他の人の工夫に気づき，参考にすることで自分もより工夫することが可能になる．しかし，学習状況をスクリーンに投影して共有するだけでは，どのような理由でプログラムを書いているか分かりにくい．そこで本研究では，複数人で共同実行することを通して直接相談することで，プログラミングの体験向上を目指す．

3. 共同実行の実現方法

3.1 手法の全体像

本節では，Scratch の共同実行環境を実現する手法を述べる．共同実行の際には，前節で述べたサッカーゲームのように，一つのスプライトは一台のマシンのみで動かす場合がほとんどで，同じスプライトを複数台で同時に動かす状況は少ないため，一つのスプライトに対して一つのマシンでしか動かさないことを想定する．

図 4 を用いて実現方法の概要を説明する．左の白いキーボードとマウス，右の黒いキーボードとマウスはそれぞれ別の利用者が操作する．白と黒のキーボードとマウスで操作が行われると，その結果が自身の実行画面に反映され，スプライトの座標や向きが変化する．その変化を相手の実行画面に反映するために，変化したスプライトの変化後の座標や向きを抽出し，通信サーバに送信する．通信サーバは受信したデータを相手のマシンに送信し，抽出したデータを共有する．これにより，共同実行を実現する．

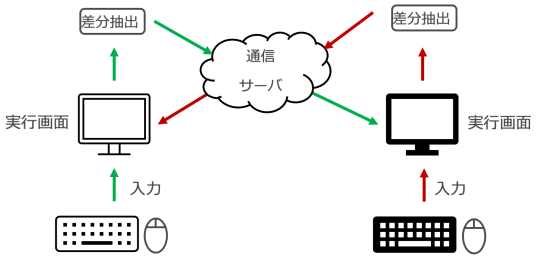

図 4 概要

本稿で提案する共同実行環境を実現する手法は，以下の 3 つのステップで構成される．

- Scratch には，実行画面上でスプライトの座標や向き，大きさなどを描画している関数がある．その関数を用いて**スプライトの属性取得**を行う．
- 取得されたスプライトの属性を，通信サーバに送信する**スプライトの属性送信**を行う．送信は，属性が変化したスプライトのみとする．
- 通信サーバに送られたスプライトの属性データをサーバを介して受け取り，自分の PC に反映させる**スプライトの属性受信**を行う．

3.2 実装

実装は，オープンソースとして公開されている Scratch 3.0 のソースコード [8] を改変して行う．Scratch 上でのプログラムは，共同実行を行う PC 上で同じプロジェクトファイルを開いておき，実行するものとする．

PC 間の通信には，ブラウザとウェブサーバとの間で双方向通信を行う WebSocket を用いる．一つの PC からデータの送信が行われると，全ての PC にそのデータが送信されるようなサーバを用意している．

3.2.1 スプライトの属性取得

第一ステップでは，実行画面上で描画されているスプライトの属性を取得する．キーボードやマウスが操作されると，Scratch 上でその操作が反映され，スプライトの座標や向き，大きさが変化する．この変化を，Scratch のソースコード上において，WebGL によるスプライトの描画を行っている関数上で取得する．Scratch ではこの関数が一定周期で呼び出されている．

3.2.2 スプライトの属性送信

第二ステップでは，スプライトの座標などの属性に変化があった際、変化後のデータを WebSocket サーバに送信する．スプライトの属性を取得している描画関数はミリ秒単位の短い周期で呼び出されているため，全てのスプライ

Vol.2023-GN-118 No.52
Vol.2023-CDS-36 No.52
Vol.2023-DCC-33 No.52
2023/1/24

トの属性データを送信することは，動作の遅延の原因になる．そこで，属性の変化があったスプライトのデータの変化後のデータのみを送信することにする．送信するデータは，スプライトの ID，座標，向き，大きさ，スプライトが表示されているかどうかのフラグである．

3.2.3 スプライトの属性受信

第三ステップでは，WebSocekt サーバから送られてきたデータを受け取りスプライトに反映させる．他の PC から送信された情報を受け取った PC 上では，受け取った情報の中に含まれているスプライトの座標や向きに従って，スプライトの属性値を変更する．変更結果は，スプライトの描画を行っている関数により実行画面に反映される．このようにしてスプライトの属性を受信して自身に反映させることでスプライトの動きを共有している．

ここで，端末間でデータの送り合いが発生しないような工夫が必要である．データの送り合いについて図 5 を用いて説明する．例えば，左の白い PC で座標が変化した場合，その変化した情報は右の黒の PC に送信される．黒の PC では，受け取った属性データを自身に反映させる．この時，スプライトの属性送信によって属性が変更されたと判定され，さらに左の白い PC に送り返される．本手法では，このような送り合いが発生しないように，属性の変化が受け取ったデータによるものである場合には，属性送信が起こらないようにしている．

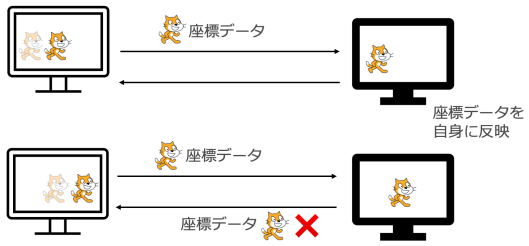

図 5 データの送り合い阻止

4. 評価実験

本節では，本システムを児童向けのプログラミングワークショップに適用した実験の結果と考察について述べる．

4.1 実験概要

NPO 法人スーパーサイエンスキッズが主催するプログラミングワークショップで本システムの適用実験を実施した．本実験では，ワークショップに参加した児童は，本システムを使用し，Scratch のプロジェクトを共同実行した．評価では，本システムを利用して作成したプロジェクトファイルの内容，共同実行中の動画，実験終了後のアンケートを検証する．

4.2 実験内容

実施日は，2022 年 11 月 20 日であり，参加者は小学 1 年生から中学 1 年生の 12 人である．12 人を二人ペアとし 6 グループに分けた．この時，年齢が近い 3 グループと，年齢が離れている 3 グループに分け，年齢による影響が出ないようにした．プロジェクト制作を行う PC は，全ての児童に対して一台ずつ用意した．共同実行は，共同実行用の PC を 3 セット用意した．自分の PC で制作したプロジェクトをペアの相手と共同実行するために，ファイルサーバにファイルをアップロードし，共同実行用の PC でペア同士で共同実行できるようにした．

ペア同士で共同実行するにあたり，対戦型のサッカーゲーム（図 3）と協力型のシューティングゲーム（図 6）といった異なるテーマを用意した．実験では，前半にサッカーゲームを制作し，後半にシューティングゲームを制作することとした．

前半のサッカーゲームでは，児童は最初の 85 分間，プログラムの制作方法の説明を聞きながら，それぞれのプログラミングを行った．その後，ペアごとに共同実行を行った後，25 分間で改良と共同実行を行った．後半のシューティングゲームでは，最初の 40 分間はプログラムの制作方法の説明を聞きながらプログラムを作成し，その後，30 分間で改良と共同実行を行った．シューティングゲームの制作は，ワークショップの時間の都合で敵キャラに攻撃が当たるかどうかの判定の部分のみを児童が制作した．これら 2 つのテーマに取り組んだ後，アンケートに回答した．

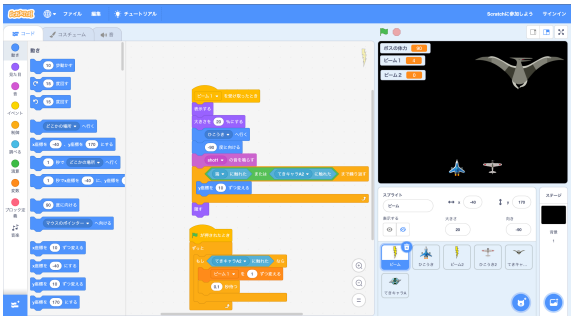

図 6 シューティングゲームの画面例

4.3 実験結果

4.4 プロジェクトファイルの内容と映像の結果

最初に，児童 12 名がプログラミングワークショップ中に何回共同実行をしたかを**表 1** に示す．サッカーゲームでは共同実行の利用方法の説明も含めて，一度全員で一斉に共同実行をした．シューティングゲームでの共同実行

Vol.2023-GN-118 No.52
Vol.2023-CDS-36 No.52
Vol.2023-DCC-33 No.52
2023/1/24

表 1 共同実行の回数

ペア	サッカーゲーム	シューティングゲーム
A	5	13
B	1	1
C	1	0
D	3	3
E	1	3
F	1	1

は，全て自主的に行なったものである．表 1 より，ほとんどの児童が共同実行の回数がサッカーゲームに比べて，シューティングゲームの方が同じか増えていることがわかる．さらに，サッカーゲームでの共同実行は全員が一斉に行なった回数も含まれており，自主的に共同実行した回数はシューティングゲームの方が増加していると考えられる．また，サッカーゲームとシューティングゲームでの改良時間の差はほとんどない．以上より，本システムの利用に慣れると，共同実行の回数が増加しプログラミングの体験向上につながると考えられる．

表 1 のペア C について考えていく．ペア C は本実験で唯一共同実行の回数が減少している．ペア C の一人の参加者は Scratch の経験が約 3 年あり，一人でプログラムの改良を行なっていた．そのため，ある程度 Scratch の経験があり一人でプログラムを進めていくことに慣れている人は，あまり共同実行をしない可能性がある．また，この参加者は本実験のなかで唯一の中学生だったので，ペアの児童から共同実行を誘いにくかった可能性があると考えられる．

次に，プログラミングワークショップ内で，児童たちがサッカーゲームとシューティングゲームのファイルを保存した数を**表 2** に示す．表 2 の 0 の部分は，プロジェクトファイルが保存されていなかった人を表している．今回，共同実行のためにファイルを保存する際には，保存するプロジェクトファイルの名前に順番に番号をつけ保存するように指示したが，順番に番号を変えることなく上書き保存をしたり，ワークショップ終了時にファイルを保存することなく削除したという可能性がある．

表 2 より，ほとんどの児童がサッカーゲームの作成数に比べて，シューティングゲームの作成数が増加していることがわかる．共同実行の回数も増加傾向にあることから，プログラムを何度も改良しながら共同実行を行なっていた様子が伺える．

4.5 アンケート結果

アンケートの質問内容を**表 3** に示す．アンケートは自由記述による回答とし，回答結果の多くは「楽しかった」や「面白かった」などであった．以下に，アンケートの回答の中からポジティブな意見とネガティブな意見を抜粋し示す．

表 2 最終作成数

ペア	サッカーゲーム	シューティングゲーム
A	4	22
	2	0
B	6	4
	2	3
C	1	5
	3	4
D	3	1
	2	0
E	2	2
	2	2
F	4	0
	3	4

ポジティブな意見としては，「相手と楽しく遊べた」「共同作業がとても面白かった」という回答があった．プログラミングをして一人で楽しむだけでなく，共同実行を通してペアの相手と一緒に楽しむことができたことがわかる．また，「一緒に遊んだ人と友達になれた」「友達関係ができた」といった回答も得られた．プログラミングをペア同士で楽しむことができ，さらにその相手と友達関係を築くことができたり，プログラミングを勉強するというより，楽しく遊びながら学ぶことができたと考えられる．これらより，Scratch で共同実行をすることはプログラミングの体験向上につながったと考えられる．さらに，「作っている間にだんだん Scratch のコツを掴んできた」という回答があった．この回答から，共同実行することで，プログラミングの体験向上に繋がっただけでなく，プログラミングの中身をより理解できるようになったと考えられる．

ネガティブな意見としては，「改造のアイデアが少なかった」という回答があった．この回答をした児童は，共同実行する回数が少なくペアとの相談回数が少なかった．そのため，一人でプログラムの改良を進めていくことになり，改造のアイデアが生まれにくくなったと考えられる．共同実行をする回数を増やすためには，共同実行をするハードルを下げる必要がある．共同実行をより簡単にするには，プロジェクトファイルの読み込みボタンを作り簡単に読み込めるようにしたり，プロジェクトを制作するマシンと共同実行用のマシンをまとめて一台で行えるようにする方法が考えられる．

また，「作るのはそれほど楽しくなかった」とう回答がった．この児童が保存したファイルの内容を分析すると，前半のサッカーゲームのプログラムは発展した内容になっていたが，後半のシューティングゲームのプログラムの内容は発展した内容はなく基本のままであった．また，この児童は表 2 ペア D の上段であり，保存したファイルの数が 3 から 1 に減少していた．ワークショップの映像より，対象の児童の集中力が切れてしまった可能性が考えられる．しかし，「遊ぶのは楽しかった」という回答もあり，プログ

Vol.2023-GN-118 No.52
Vol.2023-CDS-36 No.52
Vol.2023-DCC-33 No.52
2023/1/24

表 3 アンケートの質問

	内容
Q1	今日のワークショップの感想をしえてください
Q2	対戦ゲームを作ったり遊んだ感想を教えてください

ラミングによる体験向上には繋がったと考えられる．

5. おわりに

本稿では，Scratch を用いたプログラミング活動において，プログラミング体験の向上を支援するために Scratch 上での共同実行環境を実現する手法について述べた．児童たちが共同実行した結果について評価したところ，共同実行をすることはプログラミングの体験向上に繋がり，プログラミングの中身をより理解できるようになっていることがわかる．一方で，プログラミングしている最中に共同実行をするハードルが高いことがわかった．

今後の課題としては共同実行をするハードルを下げる必要があると考えられる．本研究では，プロジェクトを作成する PC と共同実行用の PC を分けて行ったが，二つの PC を一台のみで行えるような改良が必要である．また，作成したプロジェクトファイルの保存方法についても検討を行う予定である．

謝辞 ワークショップ中の共同実行やアンケートへの回答に協力してくださった参加者及び保護者の皆様に感謝いたします．

参考文献

[1] 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). https://www.mext.go.jp/b_menu/shingi/chousa/shotou/122/attach/1372525.htm.
[2] Mitchel Resnick, John Maloney, Andrés Monroy-Hernández, Natalie Rusk, Evelyn Eastmond, Karen Brennan, Amon Millner, Eric Rosenbaum, Jay Silver, Brian Silverman, et al. Scratch: programming for all. *Communications of the ACM*, Vol. 52, No. 11, pp. 60–67, 2009.
[3] 中学校プログラミング教育の実態調査 -R 元年度技術・家庭科技術分野「D 情報の技術」の現状-:. https://www.jste.jp/main/teigen/200201_jr_chosa_repo.pdf.
[4] Mitchel Resnick. All i really need to know (about creative thinking) i learned (by studying how children learn) in kindergarten. In *Proceedings of the 6th ACM SIGCHI conference on Creativity & cognition*, pp. 1–6, 2007.
[5] 株式会社プログミー Progummy Inc. プログミー — いっしょに作るともっと楽しい - Progummy. https://progummy.com/ja.
[6] 木下崇, 鎌田敏之, 本多満正. Scratch の mesh 機能を用いた双方向性のプログラミング教材の開発˜ 中学校技術科のネットワークを用いたコンテンツ制作の導入として˜. 愛知教育大学技術教育研究, No. 6, pp. 7–12, 2018.
[7] ソニー株式会社,「MESH」. http://meshprj.com/jp/.
[8] scratch-gui. https://github.com/LLK/scratch-gui.