VR を用いた没入型プログラミング環境

Immersive Programming Environment using VR

大西 敦生，西口 敏司，橋本 渉，水谷 泰治
Atsuki ONISHI, Satoshi NISHIGUCHI, Wataru HASHIMOTO, Yasuharu MIZUTANI

大阪工業大学 情報科学部
Faculty of Information Science and Technology, Osaka Institute of Technology
Email: e1n15016@st.oit.ac.jp

あらまし：プログラミングの入門者を対象として，Scratch のようなビジュアルプログラミング環境が提案されており，2 次元上に表現された機能ブロックを並べることでプログラミングが可能である．しかしながら，画面上のブロックが多くなると一覧性に欠けるという問題がある．そこで本研究では，プログラムの一覧性の向上を目指し，VR を用いた没入型の 3 次元空間上に機能ブロックを並べることでプログラムを作成する環境を提案する．
キーワード：**プログラミング学習**，Virtual Reality

1. はじめに

C 言語や Java 言語などのテキストベースのプログラミング言語では，一般に，キーボードを用いてプログラムを開発する．一方，ビジュアルプログラミング言語では，GUI を利用し，主にマウス操作やタッチ操作で様々な機能を持つブロックを配置していくことでプログラムを開発するため，テキストベースのプログラミング言語で頻発する文法エラーが発生しにくく，プログラミングの初心者が学習を始めるのに適している．実際にプログラミングの入門段階で成功を収めている例[1)2)]も存在する．

一方，従来のビジュアルプログラミング言語では，2 次元空間上にブロックを配置していくため，プログラムの構造が複雑になると画面に収まらなくなり，全体の把握が困難になるという問題があった．

そこで本研究では，視野の広い没入型 HMD を利用した没入型 Virtual Reality(VR)技術を利用し，プログラム全体を把握しやすく，機能ブロックの配置がしやすい没入型プログラミング環境を提案する．

2. ビジュアルプログラミング言語

一般的なエディタを使用してソースコードを記述し，コンパイルすることで実行可能バイナリを生成するプログラミング方法は，コンピュータの性能を最大限に引き出すことが可能である一方，当該プログラミング言語固有の文法を知っておく必要があり，プログラミングの初心者には敷居が高い．

一方，Scratch などのビジュアルプログラミング言語では，キャラクタやロボットなどの対象を制御するために，各機能を持つ GUI 部品をマウス操作やタッチ操作によってモニタ画面上で組み合わせることで実現することが多い．この方式では，文法に関する正確な知識が不要であるため，プログラミングの入門用として活用されている．しかしながら，ブロックの数が増えるとモニタ画面に収まりきらないため，一覧性が低いという問題や，各機能ブロックを自由に配置しづらいという問題がある．

3. 没入型プログラミング環境

没入型 HMD で実現される仮想空間では，奥行き方向の有効活用が可能であり，また視野も広いため，ブロック配置の一覧性および操作性が高まることが期待される．そこで本研究では，VR 技術を用いた没入型プログラミング環境を提案する．

3.1 制御対象

一般的なビジュアルプログラミング言語に倣い，本研究でも，碁盤の目状に区切られたステージ上を 1 ブロック直進，その場で右 90 度回転，その場で左 90 度回転の 3 種類の制御ができる，人型の 3DCG モデルを制御対象とする．

3.2 没入型 VR 空間内でのプログラミング手順

本研究では，没入型ヘッドマウントディスプレイ(HMD)を装着した利用者が，VR 空間内で機能ブロックを配置し，各機能ブロックをエッジで接続することで処理の流れを構築する．機能ブロックとしては，制御対象に 3 種類の動作を指示する命令ブロックと，プログラムの制御構造を表現する制御ブロックを提供する．制御ブロックの一覧を表 1 に示す．

表 1 制御ブロックの一覧

開始	終了	変数	リテラル
四則演算	比較	分岐	リセット

各機能ブロックを接続するには，モーションコントローラで獲得した操作者の指の姿勢情報を利用して VR 空間上に出現させた利用者の指でブロックをタッチするとそのブロックから指にエッジが伸び，その状態で，次に実行したいブロックをタッチすることで接続する．提案環境でプログラミングをしている被験者の様子と，被験者が見ている画面の様子をそれぞれ図 1 と図 2 に示す．

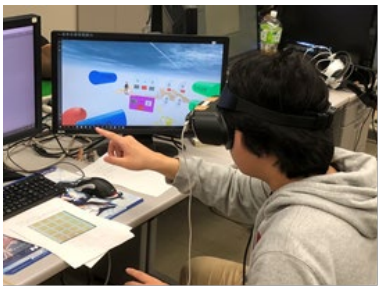

図 1 提案システムによるプログラミングの様子

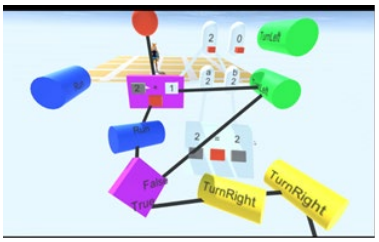

図 2 提案システムの画面提示の例

4. 実験

4.1 実験方法

表 2 に挙げる内容の課題プログラムを，情報系学部に所属する 10 名の被験者に，Scratch と提案した没入型 VR プログラミング環境で作成してもらい，完成にかかる時間を計測した．また，プログラム全体の把握のしやすさ，操作のしやすさを 5 件法(5 が最大)でアンケート評価した．

表 2 課題プログラムと標準使用ブロック数

課題	主な使用ブロックの種類	Scratch	VR
練習	命令＋分岐/反復	6	10
A	命令のみ	13	8
B	命令＋分岐	15	12
C	命令＋反復	6	10
D	命令＋分岐/反復	16	12

4.2 実験結果

各課題プログラムを作成するのにかかった時間の平均及び作成完了した人数を表 3 に示す. 没入型 VR システムでは VR 酔いのため被験者 1 名がプログラムを作成できず，課題 D は作成完了できない被験者がいた．結果から，使用するブロックの種類が増えるにつれて，課題プログラムの作成に必要な時間が増加する傾向があり，全体的に VR 環境のほうが，時間がかかることが分かった．一方，課題 D では，プログラム作成時間にかかる時間の差が小さくなった．この理由として，Scratch のほうが VR 環境よりも使用するブロック数が多く，プログラムの把握がしづらくなった一方，VR 環境では使用するブロックが多くてもエッジを用いて接続していくスタイルであるため，プログラム全体を把握しやすかったことなどが挙げられる．なお，VR を用いた没入型プログラミング環境での実験後に感想を聞いたところ，それほど時間がかかったとは思わなかったという感想を述べた被験者が多かった．このことから，VR 環境下では多少時間がかかっても，飽きることなく作業を進めることができる可能性がある．

表 3 実験結果

	作成時間の平均（分） / 人数	
課題	Scratch	VR
練習	4:27 / 10	18:58 / 9
A	1:26 / 10	2:44 / 9
B	3:39 / 10	7:20 / 9
C	1:56 / 10	9:24 / 9
D	11:28 / 7	12:03 / 5

また，把握のしやすさ，および，操作のしやすさに関するアンケート結果を表 4 に示す．今回の実験では把握のしやすさについては同等な結果となった．

操作のしやすさについては，Scratch によるプログラミングの評価が高かった．この理由としては，被験者が既に Scratch に触れた経験があったため直感的に操作できたことや，マウス操作によるブロックの配置の精度が高かったことなどが挙げられる．一方，VR 環境下の評価が低かった理由としては，初めて触れた環境であったことや，指の姿勢を獲得するためのモーションコントローラである Leap Motion による指の動きの検出精度が低く，思うような操作がし難かったことなどが挙げられる．

表 4 アンケート結果（最大 5）

評価（平均）	Scratch	VR
把握のしやすさ	4.0	4.0
操作のしやすさ	3.8	2.8

5. おわりに

本研究では，プログラム全体の把握や操作がしやすいビジュアルプログラミング環境の構築を目指して，VR を用いた没入型プログラミング環境を提案した．従来の 2D 環境でのプログラミング言語である Scratch と比較した結果, プログラム全体の把握のしやすさでは同等であったものの，操作のしやすさの点では不利であることが分かった．今後の課題としては，操作性の向上と，より没入型 VR の特性を生かしたビジュアルプログラミング環境への拡張が挙げられる．

参考文献

(1) 服部，井上，志賀野，平井，“AR 技術を用いた物理/ビジュアルプログラミングのハイブリッド型開発環境の提案,” エンタテインメントコンピューティングシンポジウム(EC2018), p.248 – p.254, 2018.

(2) 大澤，浅井，鈴木，杉本，斎藤，“マルチモーダルインタフェースを利用した 没入型プログラミングシステム: おうぎ,” 日本バーチャルリアリティ学会第 7 回大会, pp.545-546, 2002.