プログラム・ソースコードによる表現の可能性

日比野 光紘[1]　山田 晃嗣[1]

概要：プログラミング言語はコンピュータ上のシステムの動作を記述する形式言語である．しかし 筆者はプログラミング言語そのものを用いて，システムの動作ではなく現実世界の事象の表現を行ってきた．プログラミング言語を用いて表現を行うことはプログラマの思考を外在化することに繋がり，例えば現実世界の問題への理解を深めることなど様々な応用の可能性があると考えた．本論文ではこの手法を用いて様々な事象の表現を行い，制作物へのフィードバックを通じてこの表現の持ついくつかの可能性を示した．

1. 研究の背景と目的

プログラミング言語はコンピュータ上のシステムの動作を記述するための形式言語である．しかし，筆者はプログラミングを始めた頃から，プログラミング言語そのものを用いて現実世界の事象を表現することの面白さに魅力を感じ，楽曲の歌詞や料理の手順などをプログラミング言語を用いて記述してきた．

この表現手法は，筆者を含めプログラムを長年書いてきた人間にとって一度は考えたことのある手法だと考えられる．Twitter のハッシュタグ#songsincode[1] では，世界中のプログラマが楽曲の歌詞の一部を様々なプログラミング言語を用いてプログラムに落とし込んで投稿していることが確認できる．しかし，プログラマたちの中では一般的な手法であると考えられるため，この手法そのものについて着目した研究は少ない．

ウイング [2] は，問題を切り分け一つ一つの問題をシンプルに効率よく解決する思考法を「計算論的思考」と定義した．また日本では 2020 年から小学校でプログラミング教育 [3] が必修となる．文部科学省は計算論的思考を再解釈した「プログラミング的思考」をプログラミング教育の目的として掲げ，職業や時代を問わず普遍的な力として求められるものだと主張している．

特に構造化プログラミングの手法を用いて事象の表現を行う場合，条件によってプログラムの流れがどう変化するかを整理しながらプログラムに落とし込む必要がある．プログラミング言語によるこの表現は，現実世界の事象を計算論的思考をもって捉え，その結果としてプログラムに落とし込まれた表現と解釈することができる．小学校でのプログラミング教育が必修となる直前の現在において，この表現は価値のあるものだと考えている．

本論文では，プログラミング言語で扱うことのできるアルゴリズムと筆者の母語である日本語の識別子を用いた表現を試み，制作物へのフィードバックを通してこの表現の持つ可能性を示すことを目的とする．

2. 先行事例・関連研究

Code Poetry[4] はインターネット黎明期からプログラマたちの間で行われてきた，プログラミング言語処理系でコンパイルを行っても構文エラーが出ない範囲で詩的な表現を行うことを目指した活動である．Code Poetry では自然言語としての詩的な表現を目指しており，プログラムとして実行することは意識されていない．筆者の表現はむしろ現実世界の事象をプログラミング言語の世界でのルールに従って記述した表現である．

久保田 [5] は自身の作品『心をもつ言語』で「ソースコードの持つ実行的価値」について触れ，ソースコードを実行することによって実際に起こることが完全には予測できない既存の美術とは違った価値があると主張している．筆者の表現は人間がソースコードを読む，いわば人間の脳内でソースコードが実行されることに注目しており，ソースコードのみではない実行的価値を付与することができる表現であると位置づけられる．

Rockstar[6] は作詞を行うようにプログラミングをすることをコンセプトに設計されたプログラミング言語及び Rockstar から他言語へのトランスパイラ実装である．この言語自体が Code Poetry の文脈で作られた言語だと考えられるが，ロックンロールの詩として自然な英語の文章が実行可能なプログラムでもある，という点において参照すべき先行事例である．

[1] 情報科学芸術大学院大学 メディア表現研究科
Institute of Advanced Media Arts and Sciences

3. 手法と作品

3.1 手法

表現手法に関しては作者が日常的に使用する言語であり，また言語自体の使用人口も多い [7]JavaScript をベースにし，文法に関しては大まかなルールを定めた．
変数は変数名とその実体を紐付けることができる．

```
var 犬 = { 名前: 'ポチ' };
var 猫 = { 名前: 'タマ', 年齢: 5 };
```

関数は動作の手順など一連の流れを関数名という形で抽象化することができる．

```
function 歩く ( 動物 ){
  動物.現在位置 = 動物.現在位置 + 1;
  return 動物;
}
```

そして制御構文は受け取った条件式を元にプログラムの流れを制御する機能を持つ．

```
if( ご飯.存在するか () == true ){
  while( ご飯.量 > 0 ){
    ご飯 = 猫.食べる ( ご飯 );
  }
}
```

基本的な機能の他に，JavaScript をはじめ多くの言語に実装されている機能は必要に応じて使用する．例えば猫をオブジェクトとしてではなくクラスとして扱うことが適切だと判断した場合はクラスの文法を使用する．

```
class 猫 {
  constructor(名前){
    this.名前 = 名前;
    this.年齢 = 0;
  }
}
```

3.2 作品

この手法を用いて様々な事象について表現を行ってきた．その過程で誰もが一様に同じアルゴリズムとして捉えられる事象ではなく，作者の解釈をプログラムに落とし込んだ作品の方が鑑賞者から見て興味深い内容になると考えた．作品『＊ as Code』として，初期は作者の生活や世界的なニュースについて，最終的には作者の抱えている悩みや煩わしさをプログラムとして表現した．作品の内の一つを掲載する．

```
class 人間 {
  constructor(){
    this.誤差 = 90;
    this.フィードバック;
    this.理想の具合 = ( x ) =>
      x;
    this.伝わり具合 = ( x ) =>
      this.誤差 / 100 * x;
  }
  受け入れる (){
    this.誤差 =
      this.誤差 +
        (100 - this.フィードバック * 100) / 人数;
  }
  諦める (){
    this.誤差 =
      this.誤差 - this.フィードバック * 100 / 人数;
  }
}

var 人間たち = [];
var 人数 = 8000000000;

for( var i = 0; i < 人数; i++ ){
  人間たち [ i ] = new 人間 ();
}

var わたし = new 人間 ();

for( var i = 0; i < 人数; i++ ){
  for( var j = i + 1; j <= 人数; j++ ){
    var コミュニケーション;
    if( j == 人数 ){
      コミュニケーション = ( x ) =>
        わたし.伝わり具合 ( x )
          / 人間たち [ i ].理想の具合 ( x );
      わたし.フィードバック =
        コミュニケーション (100);
      わたし.諦める ();
    }else{
      コミュニケーション = ( x ) =>
        人間たち [ j ].伝わり具合 ( x )
          / 人間たち [ i ].理想の具合 ( x );
      人間たち [ j ].フィードバック =
        コミュニケーション (100);
      人間たち [ j ].受け入れる ();
    }
  }
}
```

このプログラムでは作者が抱えているコンプレックスの一つである「他者とのコミュニケーション」，その中でもコミュニケーションをどう捉えているかについて作者の考えを記述した作品である．作者と他者の間で起きているコミュニケーション，また他者と他者の間で起きるコミュニケーションをこのようなアルゴリズムで記述した．プログラムをディスプレイ上のエディタ等で表示するとその後の実行結果を期待してしまうアフォーダンスがあると感じ，またこの作品はあくまで人間が読むためのプログラムであるというコンセプトから紙に印刷し作品とした．初期作品は大判ロール紙と A3 コピー用紙に印刷し壁に掲示し，最

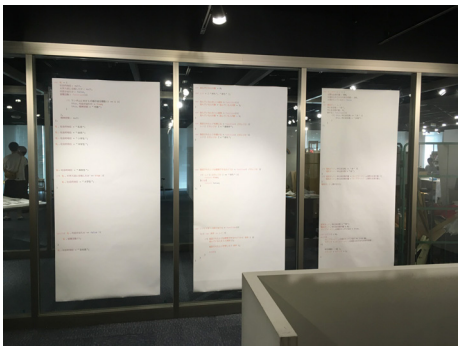

図 1　学内での作品展示 1 の様子

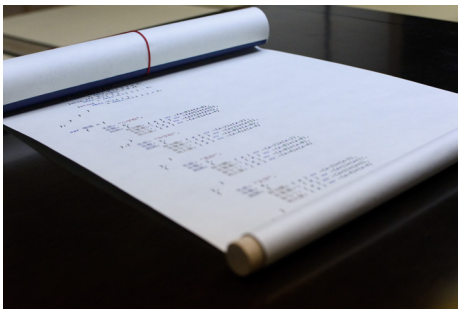

図 2　学内での作品展示 3 の様子

終的な作品では和紙に印刷したものを巻物の形に仕上げ，学内の和室に展示した．最終的な巻物という作品形式は，紙でありながらディスプレイと同じスクロールのような視線の動きを誘導できるのではないかと考えたからである．

4. 展示

これまでに 3 回の学内での作品展示，そして学内や学外の人間へのインタビューを行った．以下がそれらの詳細な日程である．

- 2018/09/11-12: 学内での作品展示 1
- 2018/10/16-17: 学内での作品展示 2
- 2018/11/17: 学内でのインタビュー
- 2018/11/18: 学外の方へのインタビュー
- 2018/11/19-20: 学内での作品展示 3

上記の機会で得たフィードバックから次章にてこの作品の考察を行う．

5. 考察

5.1 計算論的思考に基づいた思考を行うためのツールとしての可能性

学内での 3 回目の展示後のコメントとして担当の教員から，‘自分の周りの問題について書いているあなた自身が説明能力を持ってしまった。’というコメントをいただいた．作者自身の感覚としても，プログラムとして書くことによりその事象が何故起こるか，条件が変わるとどうなるのかを客観的な視点で捉え，作品展示の際にその内容について説明できていることを感じた．

5.2 プログラミング言語の教育ツールとしての可能性

学内での 1 回目の展示の時点では作者自身の考えを記述した作品の前段階として，プログラミングに触れたことの無い鑑賞者への配慮としてインスタントラーメンの作り方といったような誰もが共通認識として持っているアルゴリズムを読み易く記述したプログラムを展示していた．その際に，学内の事務職員の方がそのプログラムを読んでから作者の考えを記述したプログラムを読み，作者が三角関数と逆三角関数を誤用していた部分を指摘してきた．これはこの表現で記述された理解の易しいアルゴリズムを読んだ結果，プログラミング言語の作法を理解させることに繋がったと考えられる．またプログラム内の識別子を日本語で記述した結果，鑑賞者から，‘プログラムを読むのは苦手だが，日本語で書いてあるとスラスラと読める.’という感想をいただいた．識別子を日本語で記述することもプログラムを読み易くすることに貢献したと考えられる．

5.3 プログラマにのみ理解できるメッセージとしての可能性

```
function 認知症を患う ( 人 ){
  人.思い出 = 人.思い出 >> 1;
  return 人;
}
```

知人のプログラマからのインタビューの際に上記のコードスニペットに対して以下のような感想をいただいた．

‘これ好き．右シフトは，見た時，こんな印象でした.’

(´・ω・‘)　(´・ω・‘:;.:...　(´・ω・:;.:...
(´・ω:;.:...　(´・:;.:...　(´:;.:...　(:;.:...

2 進数の値が右にシフトされることで思い出の値がどんどん少なくなっていく，といった細かいニュアンスに関しても読み取ってもらえたことが確認できた．

また学内のインタビューではプログラミングに造詣が深い人間から，‘関数の再帰呼び出しが終わらず，いつかスタックオーバーフローすることが分かっているから悲しい.’といったような実行するとどうなるかが読み取れた上での感想をいただいた．単なるプログラミング言語の構文だけではなく，そこから実行の様子を読み取ることのできるプログラマにのみ理解が可能なメッセージとしての可能性があると考えられる．

5.4 自然言語の世界との乖離

一方でこの作品の作者自身として感じたことは，あらゆる事象はプログラムとして記述できるようなシンプルな条件の構成で記述することは不可能である，というコンピュータで扱うことのできる内容の限界である．この表現を用いて，簡単に捉えることは到底不可能な問題をシンプルなパラメータで記述することにより，この感覚を作者に

伝えることのできる作品を作ることができるのではないかと応用の可能性を考えている．

6. 結論

本論文ではプログラミング言語の考え方を用いて現実世界の事象の表現を行った作品『＊ as Code』を制作した．そして作品を読んだ鑑賞者から得たフィードバックからこの表現の持ついくつかの可能性について示すことができた．今後の展望として，この表現を用いた作品を更に書き，表現手法としても内容としても興味深くブラッシュアップしていくつもりである．また今後も展覧会やプログラマ，非プログラマを問わずインタビューを重ねていくつもりである．

参考文献

[1] Twitter: #songsincode にてツイート検索 https://twitter.com/search?q=%23songsincode （参照: 2018-12-25）
[2] Jeannette M. Wing: Computational Thinking，CACM Vol.49 No.3（2006）
[3] 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）（概要） http://www.mext.go.jp/b_menu/shingi/chukyo/chukyo3/074/siryo/__icsFiles/afieldfile/2016/07/07/1373891_5_1_1.pdf （2016）（参照: 2018-12-25）
[4] Source Code Poetry http://www.sourcecodepoetry.com/ （参照: 2018-12-25）
[5] 久保田晃弘: コードを記述し、実行し、保存する，情報科学芸術大学院大学 紀要第 9 巻 （2018）
[6] Dylan Beattie - Github: Rockstar https://github.com/dylanbeattie/rockstar （参照: 2018-12-25）
[7] 情報処理推進機構: 「IT 人材白書 2012」ＩＴ技術者動向調査結果（ＩＴベンダーのＩＴ技術者向け）【データ編】 https://www.ipa.go.jp/files/000010603.pdf ,（2012）（参照: 2018-12-25）