マイクロマウスシミュレータを用いた プログラミング学習教材の提案

2016SE012 橋本佳門 2016SE044 小山哲明
指導教員：蜂巣吉成

1 はじめに

大学のプログラミング演習では，実用的なプログラミング言語を通してプログラムを開発する技術を身に付けることを目的とすることが多い．実用的なプログラミング言語とは，システムやアプリケーション開発に用いられる言語のことである．コマンドライン端末などの文字ベースでプログラミング演習を行うことがあるが，現在では GUI で PC を操作することが多く，学習者のプログラミングへの興味がわきにくい．

本研究ではマイクロマウスシミュレータを用いた学習教材を提案する．マイクロマウスシミュレータを用いた教材によって，実行結果が視覚的に分かりやすくなる．学習者が迷路探索アルゴリズムを用いたプログラムで迷路を解いていくことでプログラミング学習に興味を持つことができると考えた．マイクロマウスシミュレータとは，マイクロマウス競技 [1] で用いられる小型の移動であるロボットが迷路を駆け抜けるための迷路探索プログラムのシミュレーションを行うソフトウェアである．マイクロマウスシミュレータで迷路を解くことを通して，順次，分岐，反復といった手続き型言語の基礎的概念や，左手法などの単純な迷路探索アルゴリズムを学習することができる．さらに発展的な内容として迷路を記憶するようなアルゴリズムを学習できる．

本研究では，Web でマイクロマウスシミュレータを使用することを考慮して対象言語を JavaScript にする．また学習対象者は JavaScript によるプログラミングを初めて学習する者とする．

2 関連研究

学習者が興味を持ちやすいプログラミング言語やライブラリとして，Scratch[2] や TurtleGraphics[3]，Processing[4] などがある．それらの特徴として次の点があげられる．

- プログラム結果が視覚的にわかりやすい
- タートルの動きやグラフィカル出力のプログラムを簡単に実現できる

以上を踏まえてこれらの特徴を含みつつ，教材が作りやすいプログラミング学習環境が必要であると考えた．

3 教育用マイクロマウスシミュレータの提案

3.1 マイクロマウスシミュレータについて

マイクロマウスシミュレータとはマイクロマウス競技の迷路探索プログラムのシミュレーションを行うソフトウェアである．マイクロマウス競技 [1] とは，小型の移動ロボットが迷路を走り抜ける速さと知能を競う競技で，競技に出場するロボットのことをマイクロマウスと呼ぶ．使用する迷路は縦横それぞれ 16 区画，合計 256 区画の正方形でできており，スタートはその一区画，中心の 4 区画にゴールがある．

マイクロマウスシミュレータを教材として用いることで次のことがいえると考えた．

- 迷路探索アルゴリズムを学ぶことができる
- 実行結果が視覚的にわかりやすい

しかし，本研究の対象としている初学者が競技のための迷路探索アルゴリズムを学ぶことは難易度が高いと言えるので，教育用に設計する必要があると考えた．

3.2 教育用マイクロマウスシミュレータの設計

マイクロマウスシミュレータを構成する要素として次の 3 つがあげられる．

- マウスを模したキャラクタ
- 迷路
- ゴール

これらの要素を含んだ教育用マイクロマウスシミュレータを設計する．マイクロマウスシミュレータによって手続き型言語の基礎的概念から迷路探索アルゴリズムまで学習でき，教材が作成しやすくなるように設計を行う．そのための指針として次の 3 つをあげる．

- 学習者への負担が少ないようなシンプルな構造にする
- マウスの仕組みをシンプルでタートル [3] と似た操作にする
- 迷路の構造やゴールの位置を学習する目的に応じて変化できるようにする

指針に沿って，迷路とゴールを学習内容に応じて変更できるように設計した．迷路の構造を二次元配列を用いて実現する．壁の情報は表 1 のように 4bit で東西南北の壁の有無を表現し，0 ～ 15 までの整数値を配列に格納していくことで迷路を表現する．迷路は問題は合わせて教員が作成する．ソースコード 1

は図 1 の 5 × 5 の迷路の例だが，配列の大きさを変えることで任意の大きさの迷路を作成できる．

表 1 4bit で表される壁

西	南	東	北	整数値
0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
0	1	0	0	4
1	0	0	0	8

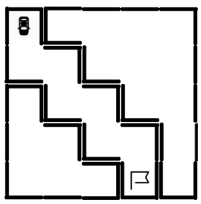

図 1 迷路 1

ソースコード 1 図 1 の迷路をつくるための配列

```
//迷路の作成
var maze = [
    [11, 13, 1, 1, 3],
    [12, 3, 12, 0, 2],
    [11, 12, 3, 12, 2],
    [8, 3, 12, 3, 10],
    [12, 4, 7, 14, 14]
]
```

ゴールは指針にあうように，競技用のように定位置に置くのではなく，迷路に合わせて自由に設置が出来るようにする．ゴールをクラスとし，迷路上の位置を示すフィールドと描画するメソッドを定義した．ゴールは教員が設置し，学習者が操作することはない．ソースコード 2 は図 1 の迷路の宣言例である．

ソースコード 2 ゴールのオブジェクト作成

```
var flag = new Goal(3,4);
```

マウスの動きは教育用マイクロマウスシミュレータの設計の指針にあうようにシンプルなものにする．直進，回転，壁を検知する，ゴールかどうか判断する次のメソッドを用意し，学習者はこれらを組み合わせてマウスを動かして迷路を解く．

- boolean go(length)
  マウスが今向いている方向に壁がなく length の数値分進めるときは進んで true を返し，壁があって進めないときはマウスは移動せずに false を返す
- void rotate(Direction)
  マウスの進行方向を変える
- boolean sence(Direction)
  壁があるかどうか判断する
- boolean goal()
  マウスの現在いる区画がゴールかどうか判断する

マイクロマウスシミュレータを扱う際にマイクロマウスの向きや壁の位置を扱うことがある．学習者が向きや位置を簡単に扱えるようにマウスの向きや壁の位置を LEFT や RIGHT，FRONT，BACK といった定数で表す Direction 型を用意した．また，マウスが向いている方向に対して右の位置と迷路中の壁の右の位置を同じ定数で扱うことで簡単に扱えることができると考えた．これにより意図しないエラーを回避し学習者への負担を減らすように設計を行った．マウスの目の前に壁がある時に go メソッドによって進もうとしたとき，マウスが壁にぶつかったということを学習者に知らせるように設計した．

4 Web による学習環境の設計・実現

本研究は初学者へのプログラミング学習を目的としている．初学者に学習環境の設定で負担をかけて，学習意欲を低下させるようなことは避けたい．そこで，学習環境を Web で実現することにした．

学習環境の設計にあたり，Web ページにテキスト編集エリア，描画エリア，実行ボタンを設置した．図 2 は学習環境の画面例である．図 3 はシミュレータの処理の流れである．JavaScript プログラムはテキスト編集エリアに記述する．マイクロマウスシミュレータは複数のプログラミング言語に対応することも考慮して，評価ステップと描画ステップの 2 段階で構成した．評価ステップは，テキスト編集エリアに入力された JavaScript プログラムをブラウザが取得し，eval 関数を呼び出して動的に評価する．eval 関数とは，文字列を JavaScript のコードとして解析評価する関数である．3.2 節で示した go() や rotate() メソッドは描画ステップの入力となる描画命令を出力する．評価ステップ終了後に，描画命令列を入力として描画ステップを実行する．描画ステップは requestAnimationFrame 関数を呼び出して，描画命令に従ってマウスをブラウザの画面に描画する．

学習の内容によって無限ループに陥ることがある．例えば，迷路がループを含む時に迷路探索アルゴリズムである左手法を用いると，マウスがゴールにたどり着けずに無限に回り続ける場合がある．無限ループ処理の対策として，go() の回数を記憶し，ある一定数を超えると途中でソースコードのプログラムを停止させ，“ループを抜けない可能性があります” と学習者に知らせるようにした．

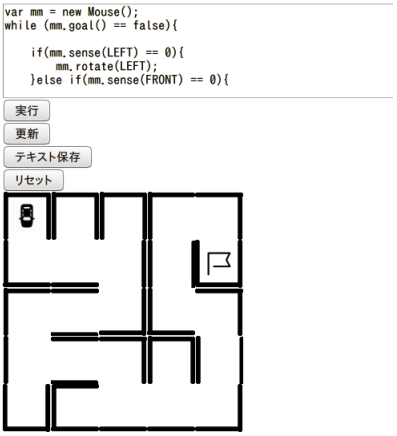

図 2 学習環境の画面例

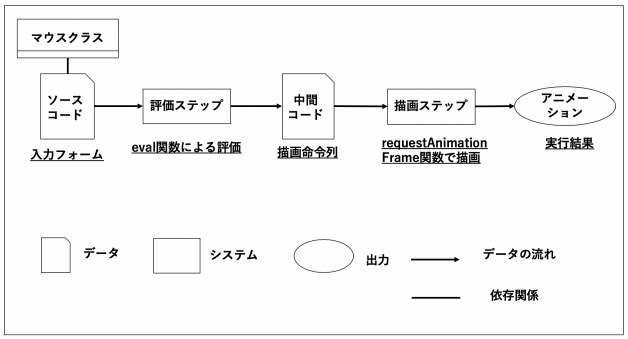

図 3 シミュレータの処理の流れ

5 教材例

最初に基礎的な繰り返し文や条件分岐を用いた問題を用意して，徐々に難易度を上げていくように問題例を作成した．教材を作成する上で次の事項を指針とした．

- 手続き型言語の基礎的概念を学ぶことが出来る
- 最終的に学習者自身で探索アルゴリズムを用いて迷路を解く事ができる

本研究のマイクロマウスシミュレータで扱うことが出来る迷路の学習教材を示す．迷路を記憶することを用いたアルゴリズムを学ぶことで配列の学習が出来るようにする．

1. 変数・演算
   - 図 4 の迷路 2 を解け

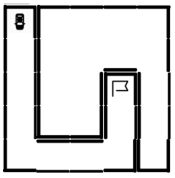

図 4 迷路 2

ソースコード 3 問題 1 の解答例

```
var mm = new Mouse();
var a = 4;
mm.go(a);
mm.rotate(LEFT);
a--;
mm.go(a);
mm.rotate(LEFT);
a--;
mm.go(a);
```

この問題では変数と演算について学ぶ事を目的としている．変数宣言を行い，その変数を引数として扱うことで変数と演算を理解させる．

2. 繰り返しと条件分岐
   - 図 1 の迷路 1 を解け

ソースコード 4 問題 2 の解答例

```
var mm = new Mouse();
for(var i=0; i<7; i++){
    mm.go(1);
    if(i%2==0){
        mm.rotate(LEFT);
    } else {
        mm.rotate(RIGHT);
    }
}
```

迷路 1 では繰り返しと条件分岐について学ばせることを目的としている．変数 i が奇数と偶数によってプログラムが分岐を用いて迷路を解かせることで繰り返しと条件分岐を理解させる．
左手法によって迷路 3 を解かせる．学習者に左手法についての説明を行った後に実装させるようにする．
左手法とは，図 6 のフローチャートに示すように常に左壁に沿って走る探索方法である．

3. 左手法を用いた迷路探索
   - 図 5 の迷路 3 を解け

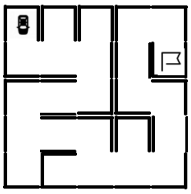

図 5 迷路 3

左手法を学ぶ中で繰り返しや条件分岐を学ぶことを目的としている．

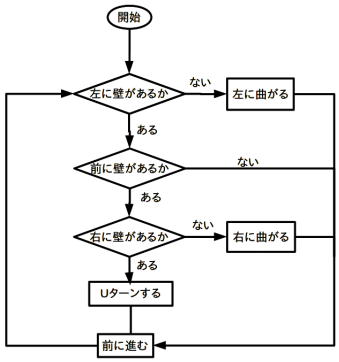

図 6 左手法のフローチャート

ソースコード 5 左手法のソースコード

```
var mm = new Mouse();
while (mm.goal() == false){

    if(mm.sense(LEFT) == 0){
        mm.rotate(LEFT);
    }else if(mm.sense(FRONT) == 0){

    }else if(mm.sense(RIGHT) == 0){
        mm.rotate(RIGHT);
    }else{
        mm.rotate(BACK);
    }
    mm.go(1);
}
```

4. 拡張左手法を用いた迷路探索

- 図 7 の迷路 4 を解け

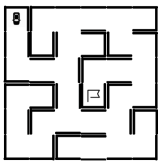

図 7 迷路 4

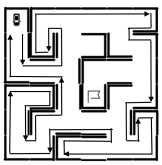

図 8 左手法による経路

図 7 の迷路 4 を左手法で解こうとすると，図 8 のようにゴールにたどり着けない．ゴールにたどり着くために拡張左手法のような迷路の内部構造を記憶する迷路探索アルゴリズムによって迷路 4 を解く必要がある．

拡張左手法とは，左手法を元に通過した場所に新たに架空の壁を作っていき，すでに通過した場所かどうかを判断し，すでに通過した場所であれば架空の壁の情報を含んだ迷路 (配列) で左手法を行う迷路探索アルゴリズムである．配列を区画の通過済みの記憶，架空の壁情報を格納する場所として扱うことで配列を学ぶことを目的としている．

6 考察

マイクロマウスシミュレータを用いたことでマイクロマウス競技を模した教材を作ることもできると考えた．学習者が迷路探索アルゴリズムを取り入れたプログラムで迷路を解いて，学習者同士でタイムを競うような競争性を含めば，学習が楽しくなると考えた．

本研究の学習環境ではコンパイルエラーを起こしても，どこが間違えているかはわからないので学習者自身でコンパイルエラーを発見する必要がある．コンパイルエラーの対策としてコンソールを開いて確認するという方法があるが，学習者に手間がかかる．実行された時点でコンパイルエラーを表示することができれば間違いに気づき，正しいプログラムを書くことができる．

マイクロマウスシミュレータの設計を考慮して，対象言語も JavaScript としたが，他のプログラミング言語でも扱う方法を考察する．他の言語に対応するには，評価ステップをその言語のプログラムで実行し，描画命令列を JSON 形式で記述できれば可能であると考えた．

7 おわりに

本研究ではマイクロマウスシミュレータを用いたプログラミング学習教材を提案した．既存のソフトウェアの学習教材のメリットを含み，既存のツールと併用して扱うことができるような教材用マイクロマウスシミュレータを設計した．迷路を解くことを通してプログラムの基礎的概念を学ぶことができ，左手法や迷路を記憶して探索を行うアルゴリズムを学ぶ際には，配列も学ぶことが出来るようにした．これらを Web で学習環境の設計・実現を行い，教材例も提案した．学習者が学習しやすく，教材の幅が広がるようなツールの設計にするための考察も行った．

今後の課題は実際のプログラミング演習に適用し，評価を行うことが挙げられる．

参考文献

[1] 公益財団法人ニューテクノロジー振興財団：マイクロマウス，入手先 <http://www.ntf.or.jp/mouse/>(参照 2020-01-17).

[2] ＭＩＴメディアラボ：Scratch, 入手先 <https://scratch.mit.edu>(参照 2020-01-17).

[3] Python Software Foundation：タートルグラフィックス，入手先 <https://docs.python.org/ja/3/library/turtle.html> (参照 2020-01-17).

[4] Processing Foundation: Processing, available from <https://processing.org/>(accessed 2020-01-17).