

CS 知的システム演習

# 強化学習による ライントレーサー(光-色-光) のプログラム

---

松吉 俊

# 強化学習を利用する手順

---

- (1) 状態を定義する
- (2) 報酬関数を定義する
- (3)  $Q$  学習のアルゴリズムを適用する
- (4) 十分な学習を行ったのちに最適政策が得られる(はずな)ので、それに従って行動する

# 強化学習を利用する手順

---

- (1) 状態を定義する
- (2) 報酬関数を定義する
- (3)  $Q$  学習のアルゴリズムを適用する
- (4) 十分な学習を行ったのちに最適政策が得られる(はずな)ので、それに従って行動する

# 状態と行動を定義する

---

- 状態:

- 光センサーの値: WHITE or BLACK
- 色センサーの値: WHITE or BLACK or **BLUE**
- まずは、センサー3つの値の組み合わせを状態とすると良い
  - $2 \times 3 \times 2 = 12$ の状態

- 行動: 光-光-光の時と同様

# 強化学習を利用する手順

---

- (1) 状態を定義する
- (2) 報酬関数を定義する
- (3)  $Q$  学習のアルゴリズムを適用する
- (4) 十分な学習を行ったのちに最適政策が得られる(はずな)ので、それに従って行動する

# 報酬関数を定義する

---

- 報酬の例:

- ゴール: プラスの値
- ライン上: プラスの値
- マークの上: プラスの値
- ライン外: マイナスの値

# 強化学習を利用する手順

---

- (1) 状態を定義する
- (2) 報酬関数を定義する
- (3)  $Q$  学習のアルゴリズムを適用する
  - 光-光-光の時と同様
- (4) 十分な学習を行ったのちに最適政策が得られる(はずな)ので、それに従って行動する
  - 光-光-光の時と同様

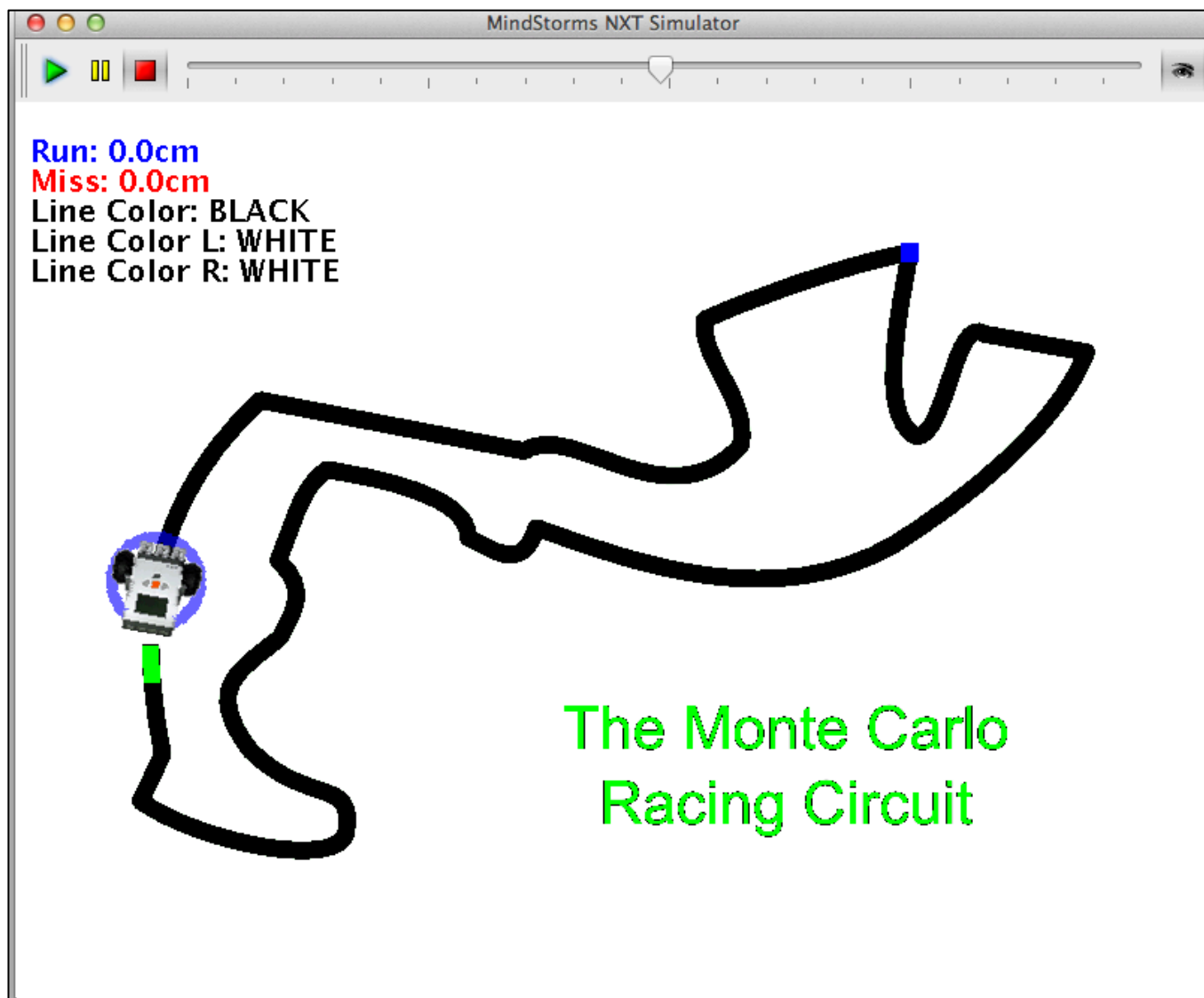
# 光-色-光に対応したシミュレーター

---

- 修正版のシミュレータープログラムを配布
  - 色センサーに対応
    - getColor(LIGHT\_B) が3値:  
WHITE, BLACK, **BLUE**
  - 後退すると、Runが増加する
    - backward()時にRun増加
  - センサーの値を常に画面の左上に表示
    - LはLIGHT\_Aに、RはLIGHT\_Cに対応



# 修正版シミュレーターの画面




# 雛形プログラム

## ソースファイル

- |                       |                     |
|-----------------------|---------------------|
| ● Simulator.java      | 起動用のクラス             |
| ● Model.java          | シミュレーションデータを管理するクラス |
| ● View.java           | 描画を担当するクラス          |
| ● ControlToolBar.java | 実行制御用のツールバー         |
| ● Robot.java          | 抽象ロボットクラス           |
| ● <b>MyRobot.java</b> | <b>サンプルロボットクラス</b>  |

このクラスと  
QLearning.javaを  
利用する



## マップデータ (適当な箇所にマーク付き)

- sample/map1-rect.png ~ map9-amida.pngまでの9種類

## その他

- ロボットやアイコンなどの画像データ6つ

# 雛形プログラムのダウンロード

---

1. Moodleから、linetracerLCL-template.tar.gz  
をダウンロードする  
(LCL = Light-Color-Light)
2. 適当なディレクトリに保存する
3. 圧縮ファイルを解凍する  

```
% tar xvzf linetracerLCL-template.tar.gz
```

# 雛形プログラムのコンパイルと実行

## ● コンパイル

```
% javac *.java
```



すべての.javaファイルから、  
.classファイルが生成されます

## ● 実行

```
% java Simulator MyRobot sample/map1-rect.png
```



起動用のメインclass



ユーザが作成した  
ロボットclassを指定する



マップの名前。  
独自にマークを付けたマップも  
元のファイルと同じファイル名を  
付けること

シミュレーターは、マップ名からロボットの初期位置と初期の向きを決定している  
(そのファイルがどのディレクトリにあるかは見ていない)

# クラスRobotの概要

---

- **init()** ロボットを開始位置に戻す
- **delay()** 速度調整 & 描画用のメソッド
- **forward(double cm)** ロボットを指定距離だけ前進させる
- **backward(double cm)** ロボットを指定距離だけ後進させる
- **rotate(double angle)** ロボットを指定角度だけ回転させる  
(正の角度は右回転、負は左回転)
- **rotateRight(double angle)** ロボットを指定角度だけ右回転させる
- **rotateLeft(double angle)** ロボットを指定角度だけ左回転させる
- **getColor(int lightNo)** 指定センサーから色を読み取る  
センサー: LIGHT\_A、LIGHT\_B、LIGHT\_C  
**Bの値: WHITEとBLACKとBLUEの3色**  
AとCの値: WHITEとBLACKの2色
- **isOnGoal()** ゴールに到達すると true を返す

# 第5回の出席確認

---

- 実装できたところまでで良いので、  
Q学習によるライトレーサー(光-色-光)の  
プログラムをMoodle上で提出する
  - tar.gz形式のファイルを提出する
  - マークを付けたマップもmyMark/ディレクトリに  
含める
  - 最終評価には直接関係しません

# 課題

---

- Q学習によるライトレーサー(光-色-光)のプログラムを完成させ、関連ファイル一式をMoodle上で提出する
  - tar.gz形式のファイルを提出する
  - マークを付けたマップも、マークを付けずに利用したマップもmyMark/ディレクトリに含める
  - 必ずたくさんのコメントを書く
- 上記に加え、以下を記述したレポートもMoodle上で提出する (PDF形式)
  - 「状態」と「報酬」の説明
  - 工夫した点と苦勞した点
  - レポートの先頭に、学籍番号と名前を書く

---

● Special thanks:

● 山本 泰生先生

● 鍋島 英知先生