ETロボコン2023
デベロッパー部門 プライマリークラス

地区：北海道　　地域：北海道札幌市

チームNo. 40　　チーム名：RWKYI

所属：株式会社スリーエス

チーム紹介、目標、意気込み

【チーム紹介】
メンバ：池田 和弘、倉谷 航、
　　村瀬 遼平、若林 祐弥
全員がロボコン初挑戦のチームです。

【目標・意気込み】
ダブルループ、ブロック de トレジャー をクリアし、走破することを目指します！

モデルの概要

安定してゴールに到達することを目指すことを目標とし、これを達成するために走行安定性の向上と処理の簡素化に重点を置いた。

●走行安定性を向上させる

・センサの位置と走行位置(車輪)が異なることを考慮した走行を行い、センサが常にライン上にあるような走行を行う。

・センサを検出しない期間を設定することで、同一ポイントで複数回処理が行われることによる誤作動を避ける。

●処理を簡素化する

- ブロックの各配置パターンにおける走行経路を固定化し、複雑な設計になることを回避する。
- 一関数に一機能を基本として疎結合を意識した設計を行い、処理の流れを分かりやすくする。

モデルの構成

1. 要求分析

安定してゴールに到達する目標のもと、
ブロック de トレジャーをクリアするための要求を抽出した。
・ラインレースや旋回でコースアウトしないための処理を実現する
・ブロックを押し出す距離やブロックの配置パターンにおける走行経路など、固定化可能なものは事前に定義することでモデルが複雑化することを防ぐ。

2. 機能モデル

・ブロック de トレジャーのクリアに必要な機能をユースケース図で抽出。
・各機能で行う処理を小さくすることで、可用性を高めた。

3. 構造モデル

・機能モデルで抽出した機能をもとにクラスを定義した。
・保守性を高めるため、単一責任の原則に基づいてクラス間が疎結合となるようにした。

4. 振舞いモデル

・走行体の現在の状態とセンサなどから得られたデータから、制御すべき状態をシーケンス図で明確化した。
・ラインの色彩検出の可否で状態を2つ用意し、複数回検出による誤作動を抑えた。

機能モデル

●走行戦略
事前にブロックのパターンに応じた走行ルートを設定する。
この方式の利点として、無駄なパターンを経由しないことやブロックを読み込まないことでマトリクス攻略の安定化が行える。
そのため、マトリクス間のノードのライントレースやブロックの押し出し、保持を高精度で行えるよう低速で実行することができる。

●機能定義
ブロック de トレジャーハンターの攻略に必要なシステムの機能をユースケース図、ユースケース記述で定義した。
ダブルループ、ブロック de トレジャーハンター で動作を切り替え、一連のシナリオとして動作させる。
汎用的なクラスは共通化し、それぞれのシナリオで使用できるようにする。

図1 ユースケース図

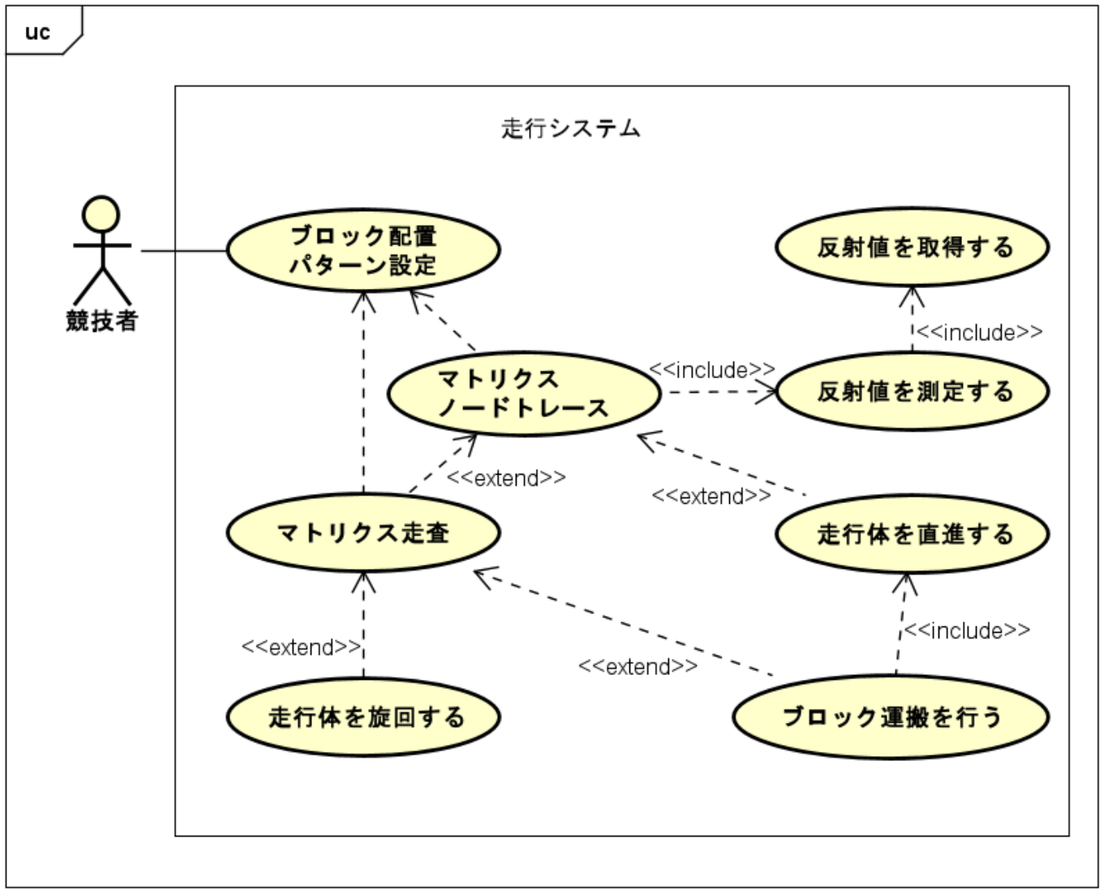


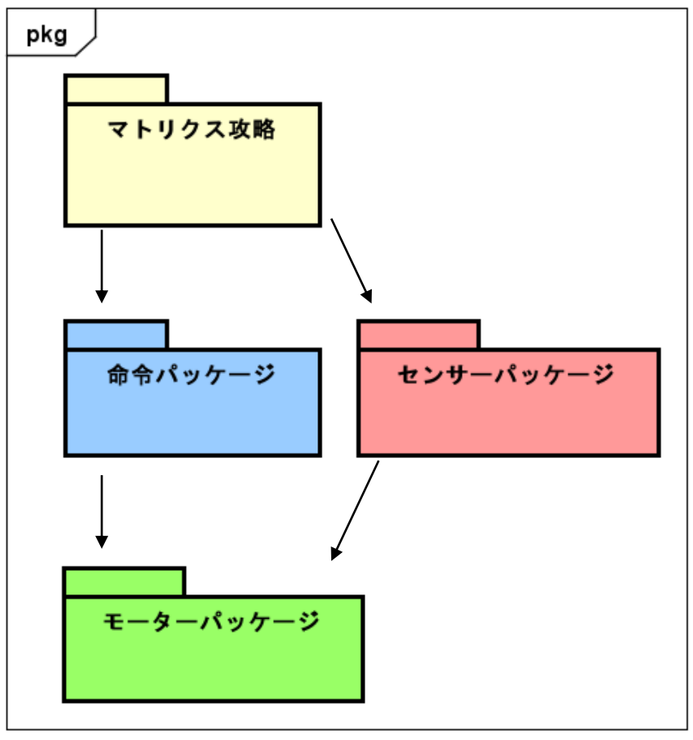
表1 ユースケース記述

ユースケース	ブロック de トレジャーハンター を攻略する
事前条件	・ブロックの配置パターンが設定されていること ・ブロック de トレジャーハンター のコースに走行体がいること
事後条件	トレジャーブロックを運搬し、ゴールエリアに到達する
処理フロー	1) 走行開始前に設定されたブロックの配置パターンをもとに、走行ルートを読み込む 2) 反射値からラインの色を判定する 3-a) 色が白/黒の場合、ライントレースを行う 3-b) 色が白/黒でない場合、走行ルートの次の命令を読み込む 4) 走行体がブロックの配置位置にある場合、ブロックの運搬を行う 5) 2~4 を繰り返し、ゴールまで移動する

構造モデル

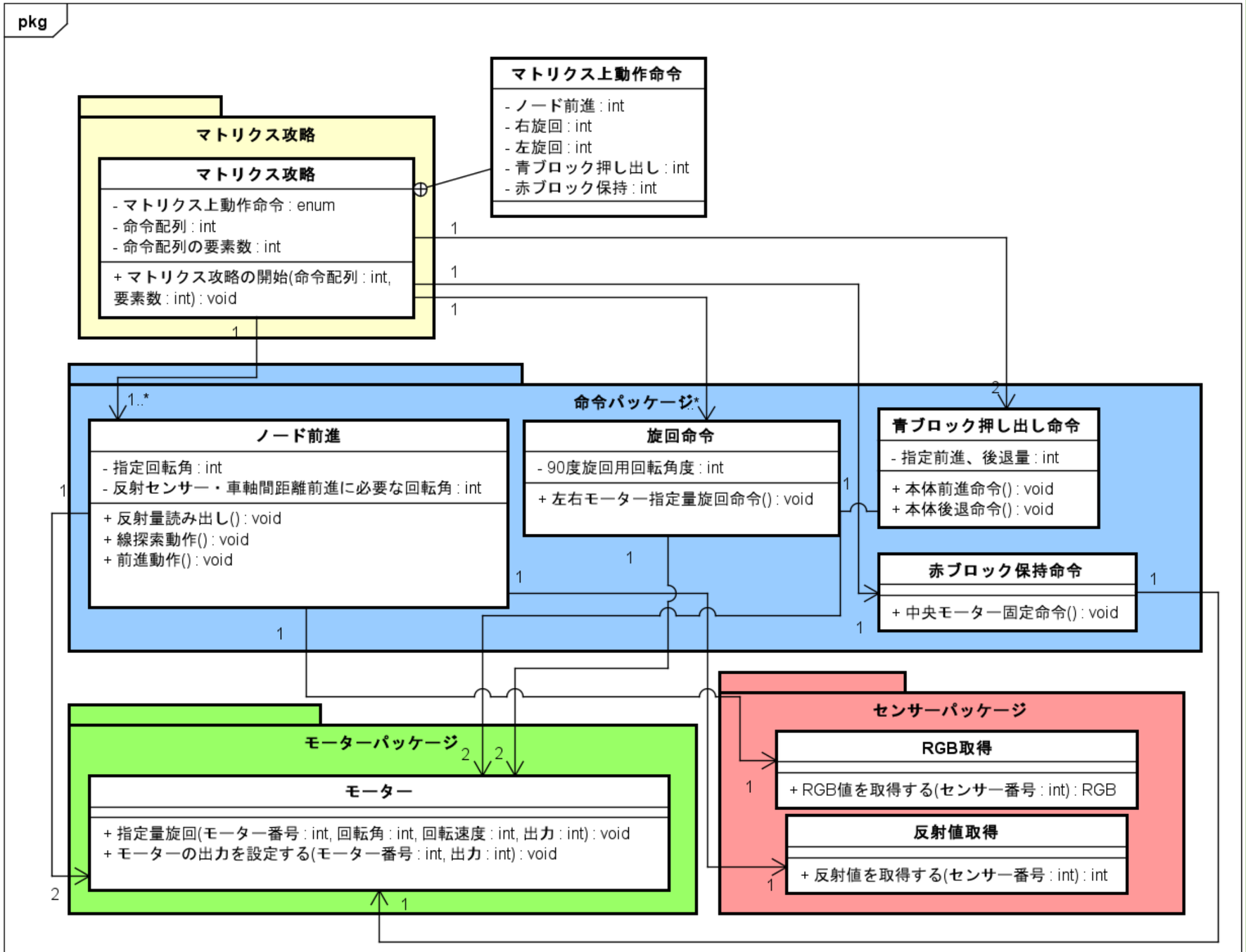
パッケージ図

パッケージ	説明
マトリクス攻略	命令を呼び出すパッケージ
命令パッケージ	走行の管理をするパッケージ
モーターパッケージ	モーターユニットの制御
センサーパッケージ	反射光センサーユニットの制御



クラス図

ブロックが配置される組み合わせに応じて命令配列にマトリクス上動作命令を設定し、命令パッケージで配列インデックス順に命令を呼び出す



ブロック　de　トレジャーハンター　要件分析

　ブロック de トレジャーハンターはライントレース終了後に走行する障害物コースである。この障害物のコースは格子状のマトリクスとマトリクスの交点にある赤、青、黄、緑のパターン、パターン間の黒線でできたノードで構成される。このマトリクス上に5つのパターンで、3つのブロックがパターン上に設置される。3つのブロックの2つは青で、1つは赤である。青ブロックを押し出し、赤ブロックをゴールまでもっていくことが求められる。ブロックの配置の組み合わせは上記の組み合わせの中から大会当日に決定される。

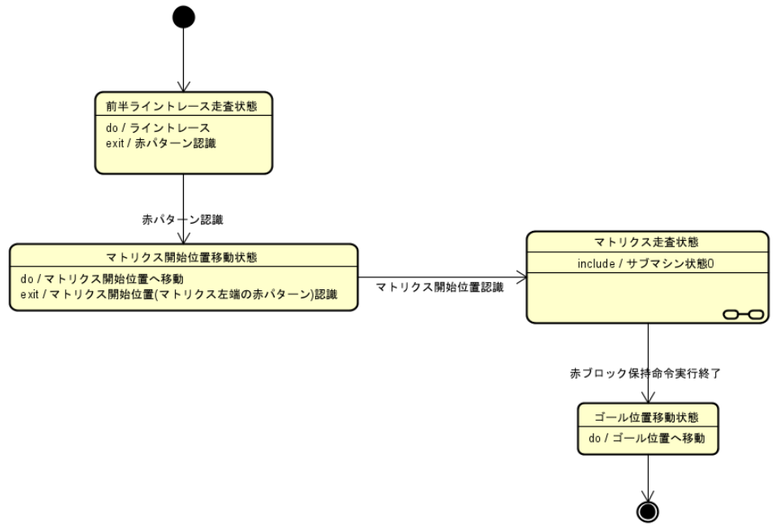
　本プログラムでは、あらかじめブロックが置かれる組み合わせに応じて、命令の組み合わせをあらかじめ記述し、それを読み込ませる方式とする。命令の組み合わせはブロックの置き方のパターンが５通り、赤ブロックの場所で３通り、左右のコースで２通りのため、5x3x2=30通りの命令の組み合わせをあらかじめ作成することができる。この命令の組み合わせはEV3マイコン本体に備えてある上下左右ボタンを用いて、走行前に読み込ませる。

　以下に命令の組み合わせの例を示す。

```
int matrix_order[]={move_forward,
                    move_forward,
                    push_blue_block,
                    turn_right,
                    move_forward,
                    turn_right,
***
```

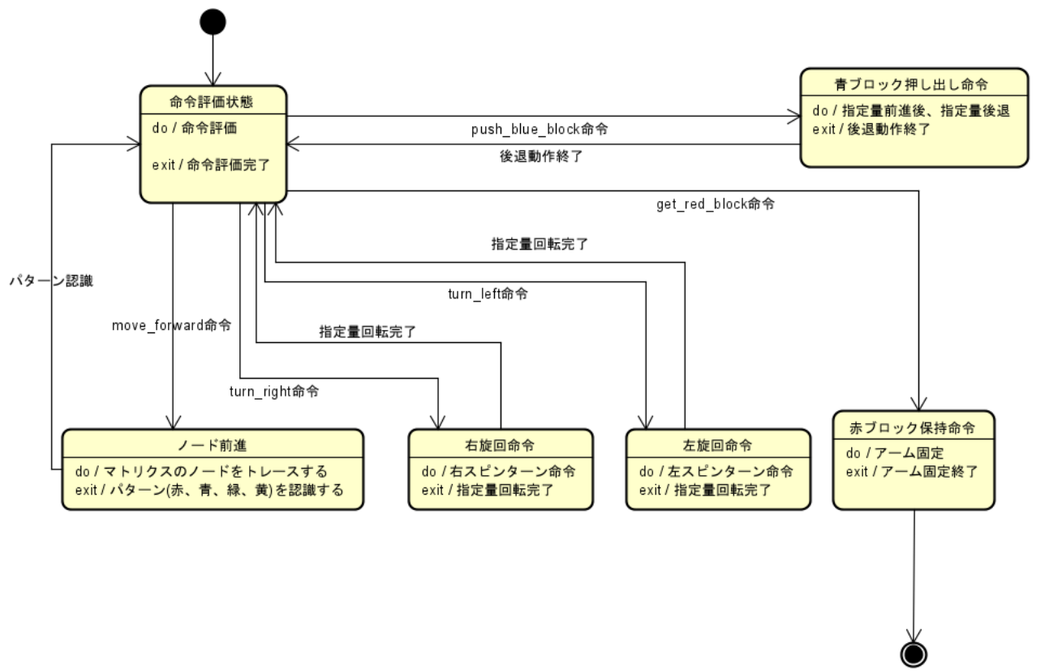
マトリクス進行の流れ　ステートマシン図

　マトリクス進行の流れをステートマシン図として以下に示す。前半のライントレースは赤いパターンで認識で判定される。判定後、マトリクス開始位置移動状態に遷移する。マトリクス開始位置とは、青に隣接する外側の赤パターンである。この位置の判定によってマトリクス走査状態に入る。マトリクス走査状態は右のステートマシン図に示す。マトリクス走査状態は赤ブロック保持命令実行終了で終了し、ゴール位置移動状態に入る。



マトリクス走査状態　ステートマシン図

　マトリクス走査状態の詳述したステートマシン図を以下に示す。本プログラムではブロックの配置ごとに用意された命令の配列を実行して、マトリクスを走査する。マトリクスの赤、青、黄、緑パターンの直上で命令を評価する。命令は前進命令、右旋回命令、左旋回命令、青ブロック押し出し命令、赤ブロック取得命令で構成される。マトリクス上の各パターンの上でそれぞれの命令が実行される。



マトリクス走査処理　シーケンス図

　マトリクス走査処理でのシーケンス図を右に示す。マトリクス走査処理ではマトリクス攻略、モーター、反射値取得、RGB取得のクラス間で相互にメッセージのやり取りが行われる。マトリクス攻略クラスはクラス図のマトリクス上動作命令クラスを読み込み、あらかじめ書き込まれた順序に応じて、命令パッケージの前進命令、旋回命令、青ブロック押し出し命令、赤ブロック取得命令、を実行してマトリクスを攻略する。