

チームNo. 169 チーム名：ISC走る.exe

所属：学校法人 岩崎学園 情報科学専門学校

## チーム紹介、目標、意気込み

ISC走る.exeは学校法人 岩崎学園 情報科学専門学校の3年生と4年生の学生で構成されたチームです。私たちは学校の授業の一環でETロボコンに取り組んでいますが、対面での活動が難しかったり、授業だけだと時間が足りないので、オンラインで集まって作業をしています。ほとんどのチームメンバーが初めてのETロボコンなので手探りで頑張っています。

ISC走る.exeの目標は  
「総合タイム0秒」  
目標達成目指してチーム一丸頑張ります！

## モデルの概要

### 1. 開発工数の削減

ETロボコン経験者及びプログラミング言語を巧みに扱える生徒は少ないためできるだけ定義や設計を簡素化し開発工数を抑える必要があるため設計モデルにシングルトンクラスを設ける等を行いまとめた。

### 2. 平行処理

PC側と走行体側のリソースを両方使い、通信で情報をやり取りする。  
そうすることで、走行の安定化と経路探索の高速化を図った。

## モデルの構成

### 1. 要求モデル

ETロボコン2022の審査規約と競技規約を踏まえて、このチームの全員で話し合い実現可能な目標を設定した。目標達成のために必要な機能をユースケース図に示した。目標とユースケース分析ででた機能ごとに必要な要件を検討し要件定義を行った。

### 2. 分析モデル

ゲームエリア攻略のための走行体の動作定義を図と文章でまとめた。チームでゲームエリア攻略指針を決め、それを攻略していく解法を定めた。  
解法を実現するために必要な要素を定義し、クラス図に示した。

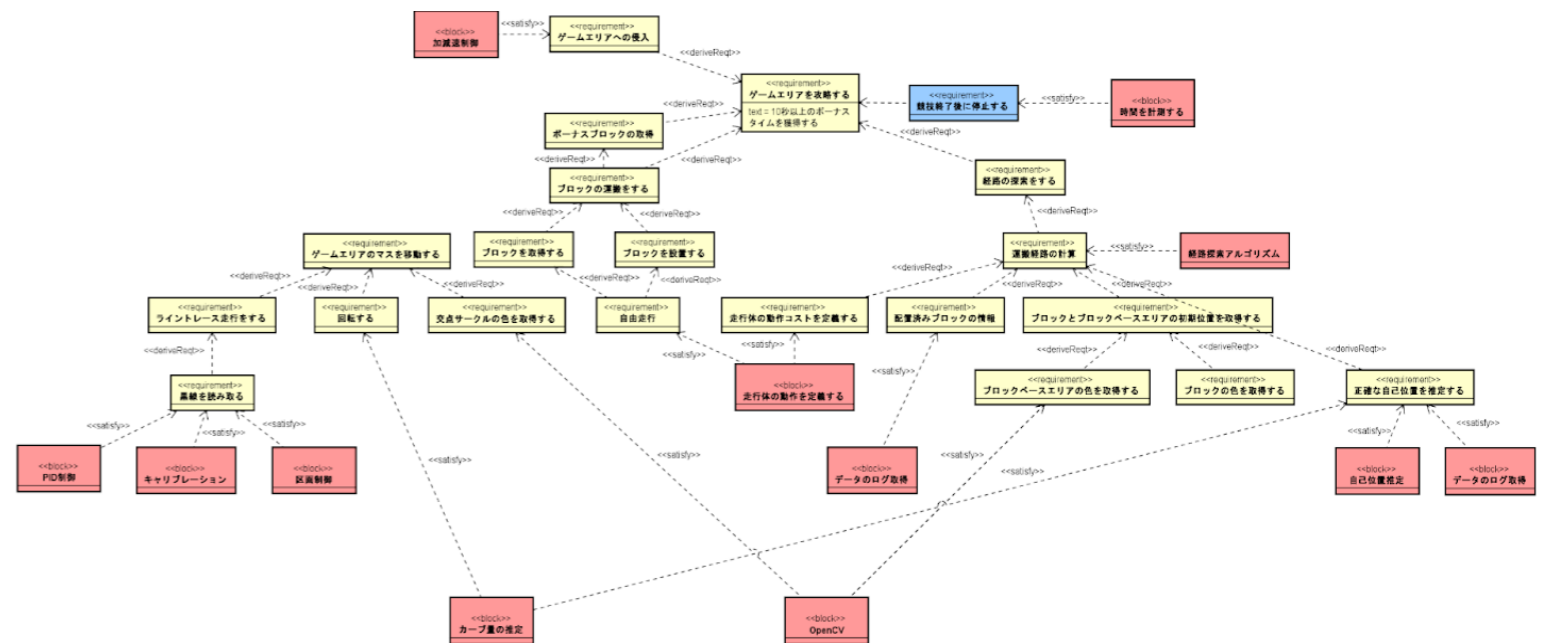
### 3. 設計モデル

各種要求モデル、分析モデルからパッケージの構造を模索し、パッケージ内の構造について示した。  
走行体による区画制御の詳細な状態遷移を示した。  
PCと走行体との通信を重点に置いてシーケンス図を示した。

### 4. 制御モデル

様々な観点から出た課題に対し対策を考案し、実施した。  
人口知能を用いたブロックの座標及び色取得の課題を重点に記載した。

# ISC走る.exe



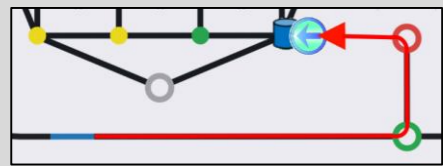
# 2. 分析モデル

## 2-1. 走行体の動作定義

ゲームエリアの攻略を安定して行うためにゲームエリア内及び侵入方法を定義した。動作定義には各種の動作を含め、制約を持たせることがある。

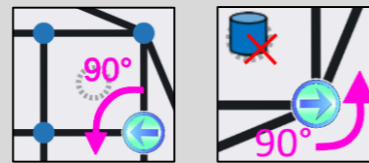
↑ : 走行体

### ゴールからブロックエリアへの侵入経路



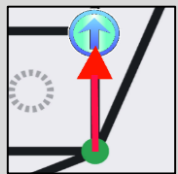
[動作] ゴールした後、ライトレースで走行し続ける。  
端点サークル到着後90°回転し侵入サークルへ移動する。  
その後90°回転してブロックエリアの右下の交点マーカ  
へ侵入する

### <1>交点マーカ上での回転



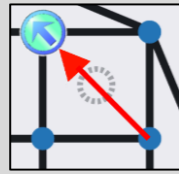
[動作] 交点サークル上で走行体が90°回転する  
[制約] 回転時に尻尾がブロックに当たってしまう  
場合はできない

### <2>交点マーカ間直線移動



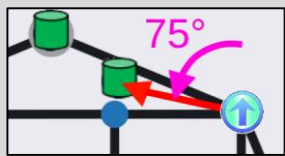
[動作] 走行体が向いている方向の  
交点サークルに直線移動をする。

### <3>交点マーカ対角線移動



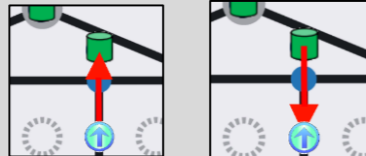
[動作] 対角線の交点サークルに直線移動をする。  
[制約] 進行方向のブロック置き場にブロックが置いてある  
場合はできない。

### <4>ブロックの配置(方法1)



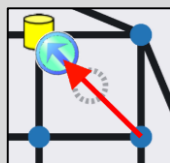
[動作] 配置するブロックサークルから  
直角の方向に向いている走行体  
から75°回転し配置処理をする。  
[制約] 既にその交点マーカで設置処理  
が行われている場合はできない

### <5>ブロックの配置(方法2)



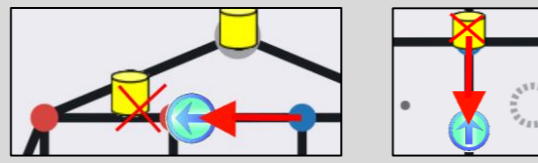
[動作] ブロックサークルに隣接する交点マーカの内側のみの配置  
方法。一つ後ろの交点マーカから開始して設置処理を行う  
[制約] 既にその交点マーカで設置処理が行われている  
場合はできない

### <6>ブロックの取得



[動作] 交点マーカ対角線移動と同じ要領  
でブロックを取得する  
[制約] 既にブロックを所持している場合は  
できない

### その他の制約



[制約] 左の図のように既にベースサークルにブロックが置かれている  
場合はベースサークル付近を走行できない  
[制約] 右の図のようにブロックを所持したままのバック走行はできない  
(ブロックの配置処理を除く)

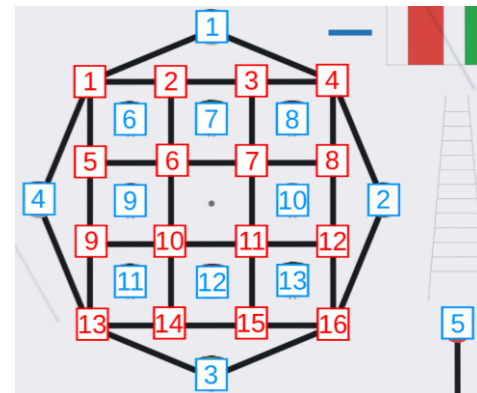
## 2-2. 指針

- 1、ボーナスブロックを先に攻略する
- 2、ブロックを確実に取得、設置しボーナスタイム10秒を目指す
- 3、ゲームエリア攻略のため経路探索が必要。経路探索の手法はダイクストラ法を用いる。

## 2-3. 解法

経路探索を行うために、ベースサークル、侵入サークル  
及びブロック置き場にブロックが配置される場所に青色で番号を振った。  
また、走行体を通る交点マーカには赤色で番号を振り、図2-1に示す

図2-1

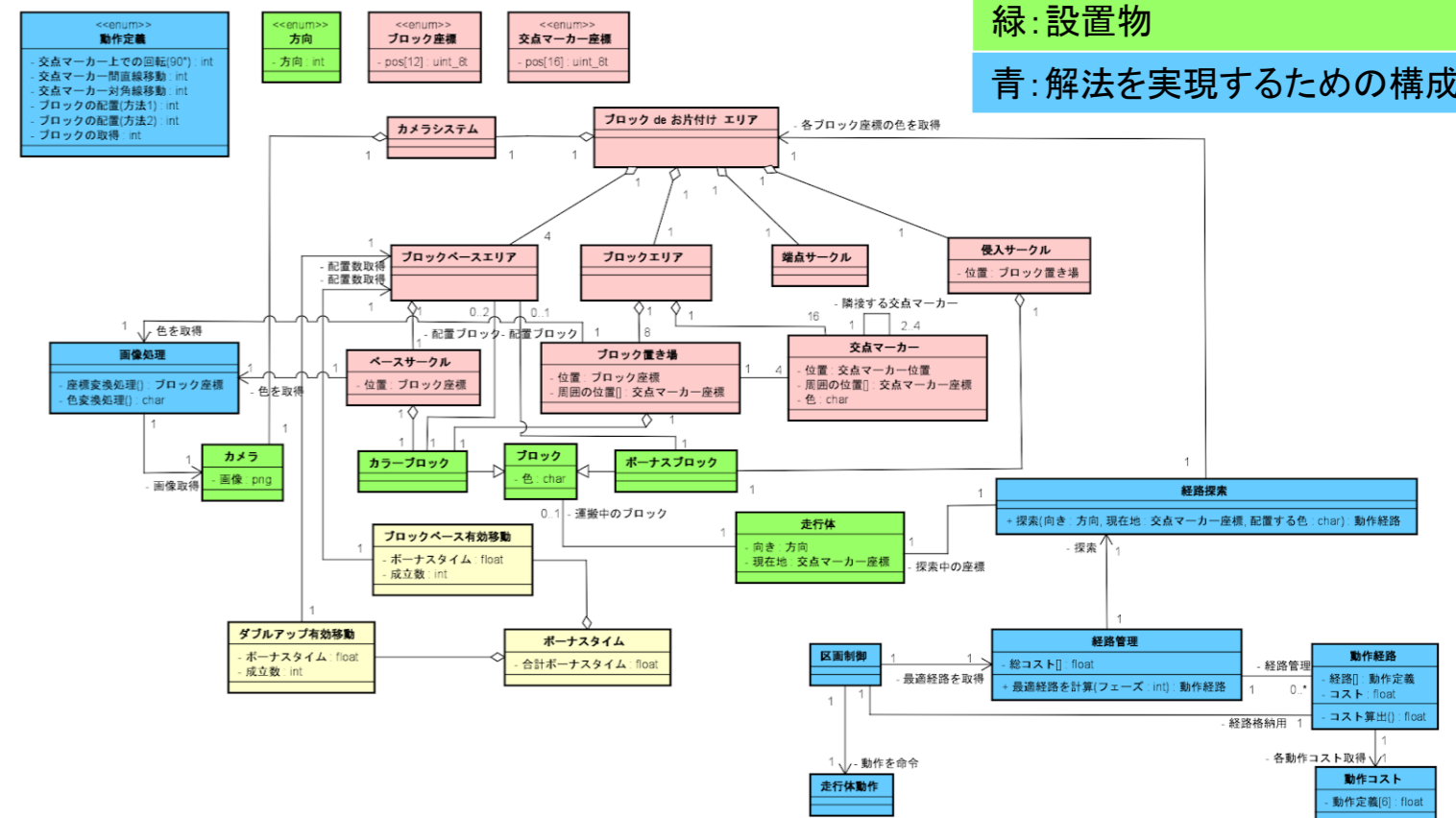


複数の経路を探索し格納する機能が必要になるため、  
走行体がブロックの周囲の交点マーカに移動、取得をし、  
ベースサークルに配置するまでの流れを1フェーズとして、探索をする。

ゲームエリア攻略の途中でも経路探索を実行し続け、  
フェーズが終わる度にそのフェーズからの探索は全て行わずに  
探索を続行する。

## 2-4. 解法を実現するための構成

※setやgetメソッドは省略している



赤: ゲームエリアの構成

緑: 設置物

青: 解法を実現するための構成

3. 設計モデル

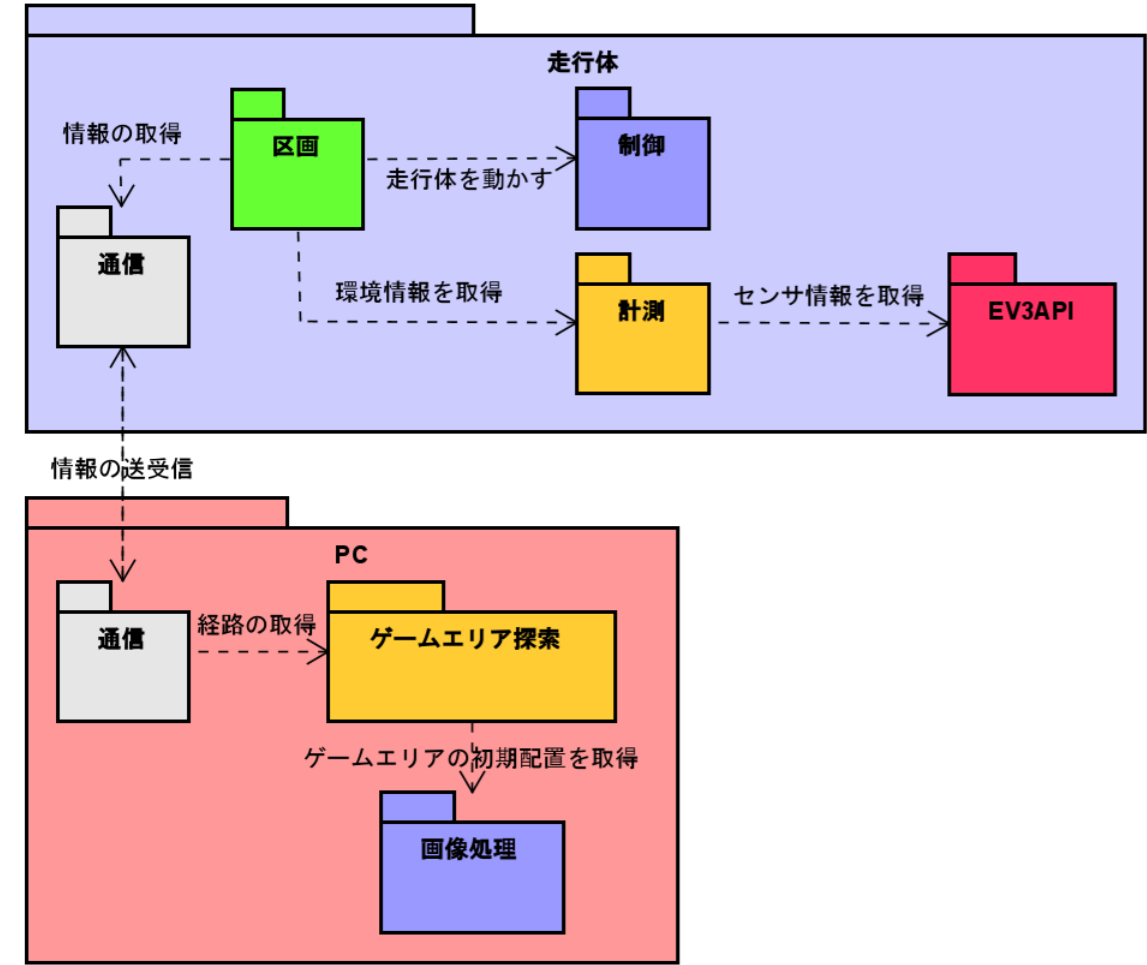


3-1.構造設計(全体)

要求モデルの分析結果からソフトウェア全体の構造設計を行い、それぞれ走行体側とPC側に分けた。  
下記の表と図3-1 パッケージの構造に示す

デバイス	パッケージ	役割
走行体	通信	Bluetooth通信によりPC側との情報の送受信を行う
	区画	区画制御を行い状況に従って制御をする
	制御	EV3機能や走行体の動作を管理する
	計測	EV3APIの情報を管理し、データの加工を行う
	EV3API	各種モータ・センサ
PC	通信	Bluetooth通信により走行体側との情報の送受信を行う
	ゲームエリア探索	早く攻略するため、経路探索を行う
	画像処理	各種ブロックの座標及び色を推定する

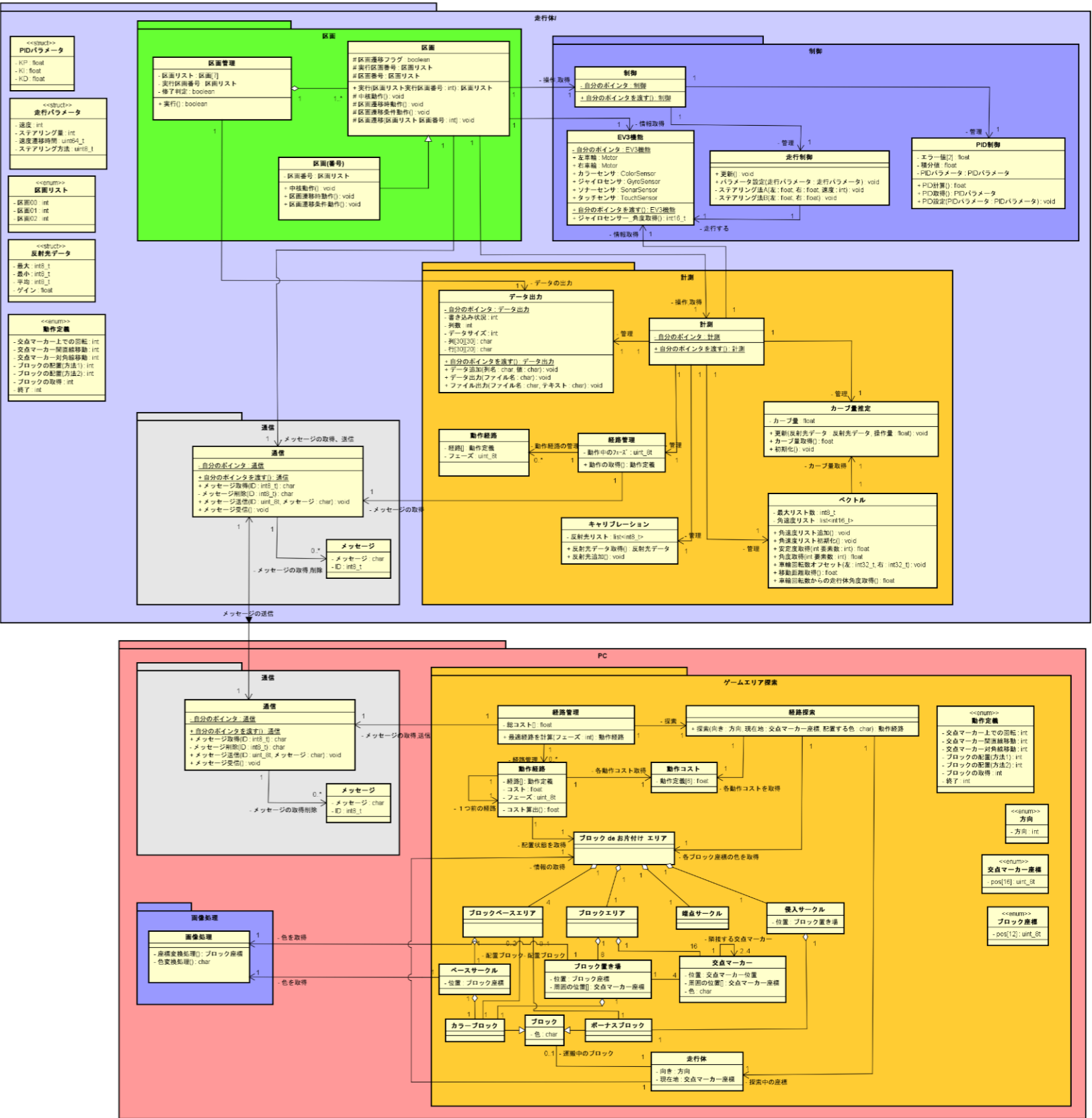
図3-1 パッケージの構造



3-2.構造設計(詳細)

3-1で定めたパッケージの構造を元にパッケージ内の詳細な設計を図3-2 パッケージの詳細設計に示す。  
また、区画制御の流れやパッケージ間の振舞いに関しては次のページの図3-10から図3-13を参照

図3-2 パッケージの詳細設計



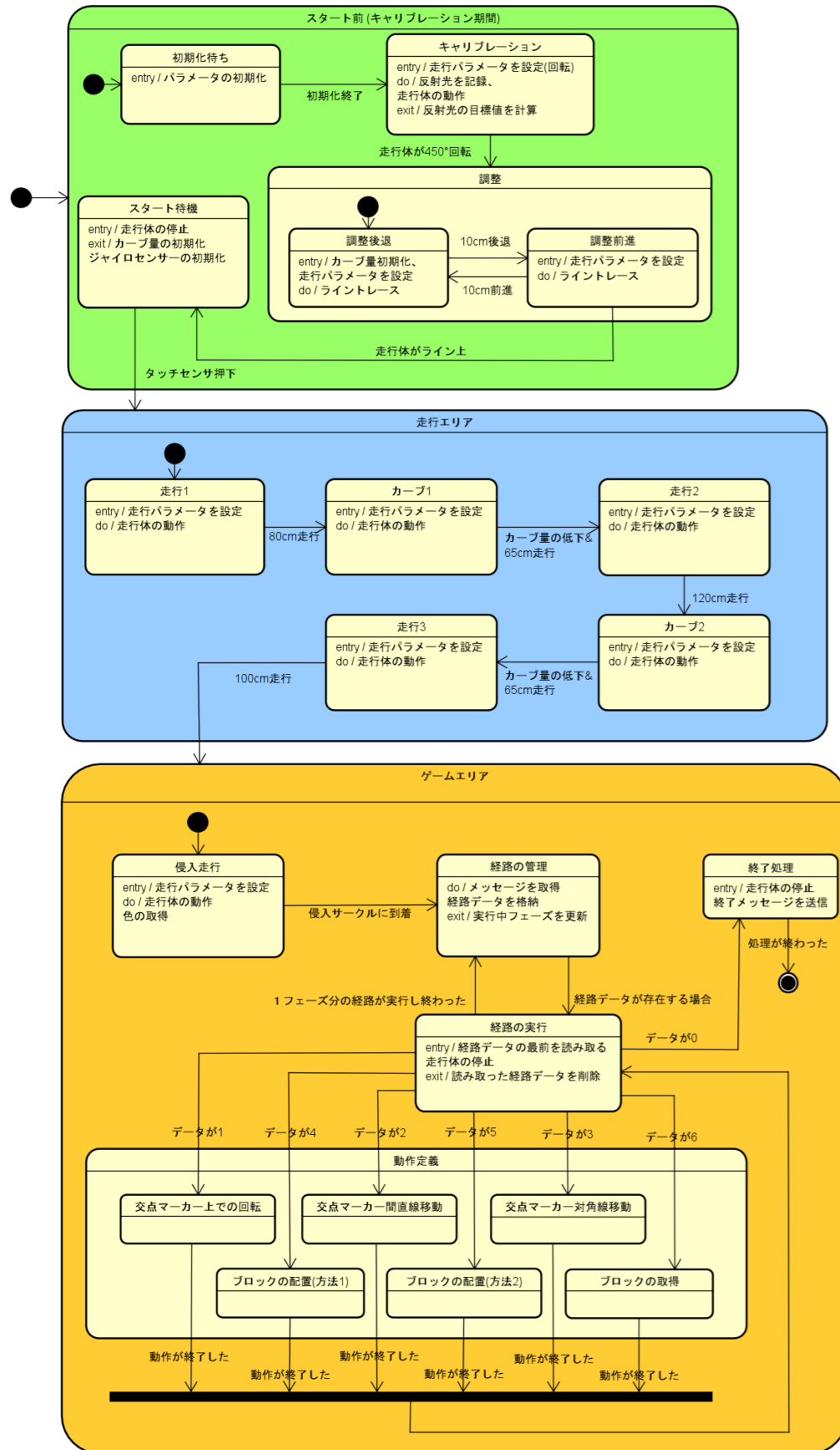
# 3. 設計モデル



### 3-3.振舞設計(詳細)

走行体の区画制御による遷移状態を下記の図3-10のステートマシン図で表す。

図3-10



競技の開始~終了まで各パッケージごとの全体的な振舞を図3-11に示す。

ゲームエリア攻略中のPC(ゲームエリア)と走行体(区画)との通信を図3-12 ゲームエリアに示す。

ゲームエリア内のブロックまでの経路~ベースサークルまでの経路(1フェーズ)を探索する  
ゲームエリア経路探索 図3-13に示す。

図3-11

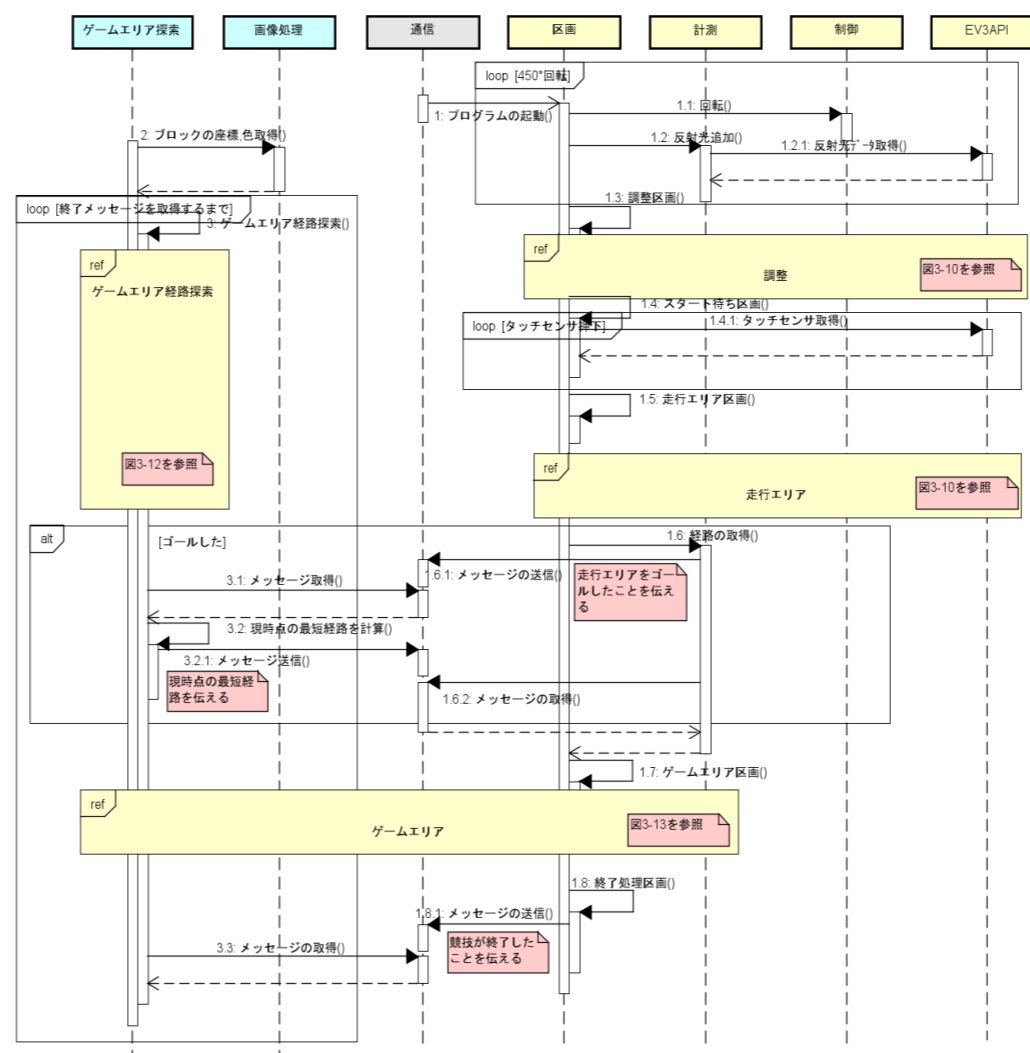


図3-12 ゲームエリア

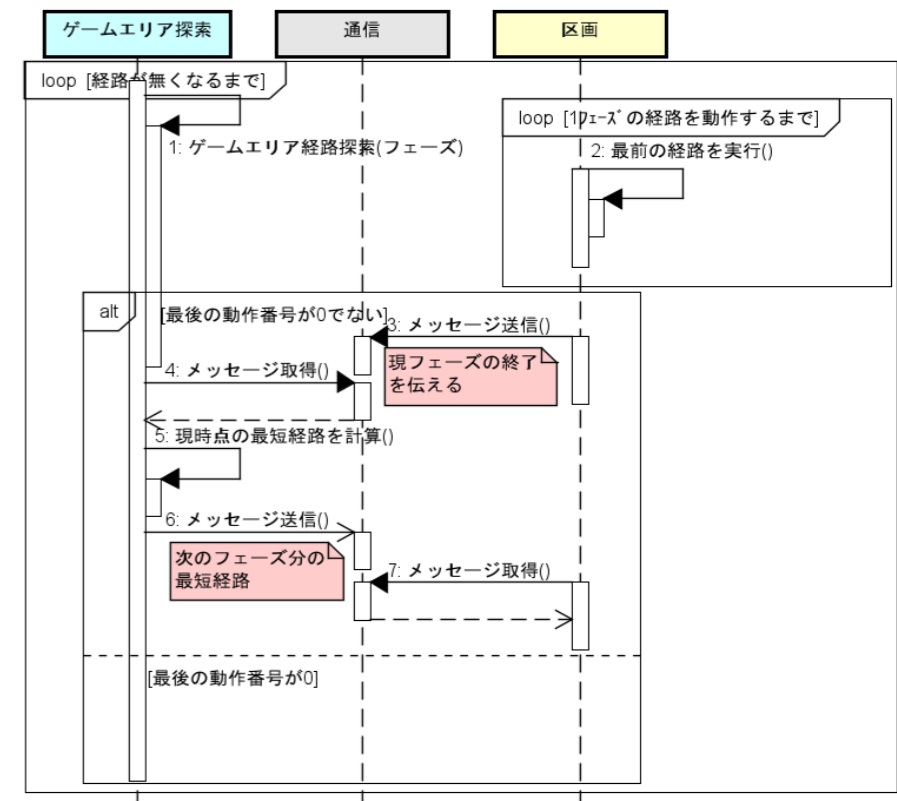
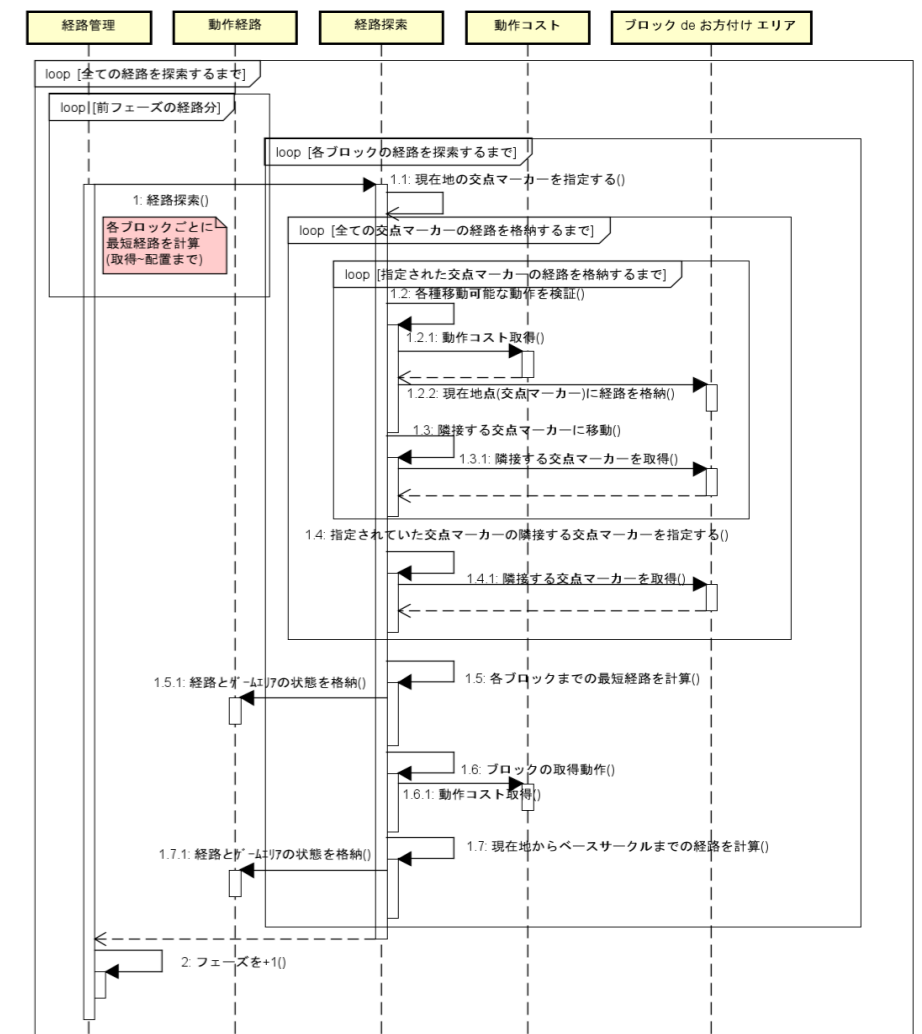


図3-13 ゲームエリア経路探索



# 4. 制御モデル

## 4-1.走行エリア

**課題:** 走行体がカーブにさしかかっているかどうかを判断することが難しい

**対策①:** カーブ量を推定する機能を実装

走行体の各種センサーから独自の計算方法を用いてカーブ量を推定した。  
計算方法は下記に示す。

$$a = (\text{反射光} - \text{目標値})^2$$

$$b = a * \text{操作量} * \text{期間角速度(gyro)}$$

$$C(x) = b + C(x-1) * 0.9$$



$$C(x) = (\text{反射光} - \text{目標値})^2 * \text{操作量} * \text{期間角速度} + (C(x-1) * 0.9)$$

※操作量 = PID制御により生じた補正量

※期間角速度 = ある一定期間のジャイロセンサーの変化の平均値

計測したグラフからカーブにさしかかっている場合他の方法に比べ変化が大きかったが、過去の影響も含めて計算しているので反映に時間がかかってしまいカーブの判定が遅れてしまった。

**対策②:** 一定量走行した後カーブ走行に遷移する

カーブの判断を諦め、一定時間走行したときカーブ走行に遷移することにした。  
直線走行に遷移する条件にカーブ推定を用いることによりうまくゴールまで走行することができた。

## 4-2.OpenCVによる色の分析

**課題:** ゲームエリアを攻略するためには各種設置ブロック、交点マーカの色取得をする必要がある

**対策:** その手法としてOpenCVによる色の分析を利用  
四色のブロックを撮影した画像を集め、機械学習を行った

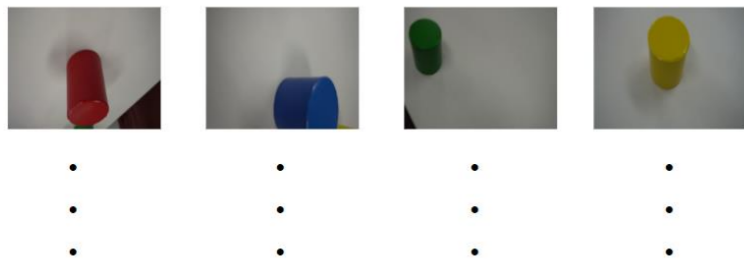


図4-1学習データの例

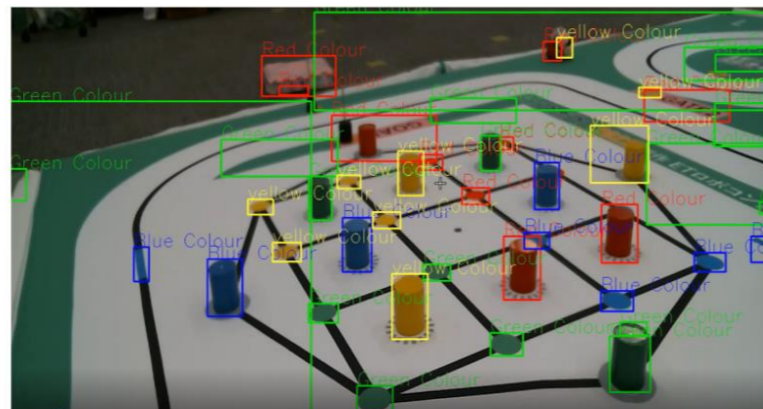


図4-2Opencvによる色の分類

集めたデータでk-meansとSVMを利用した  
**教師なし学習**を行った

**課題:** 緑色のブロック、交点マーカの認識時、ゲームエリアの地面の緑色を認識してしまう。  
また、本番ではスポンサーの表示があるため、それを誤認識してしまう危険がある。

**対策:** キャリブレーション中に色を認識する範囲の選択をできるようにした。

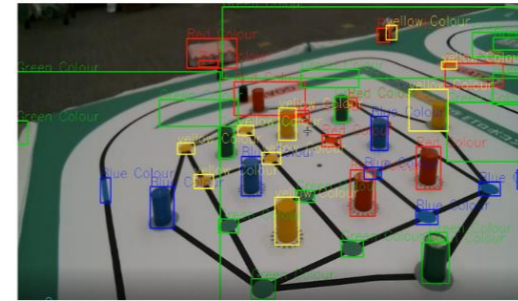


図4-3Opencvによる色の分類

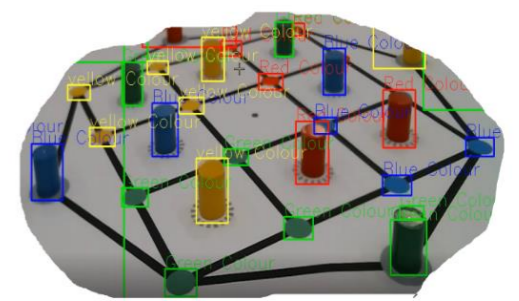


図4-4範囲選択後の画像

**課題:** 認識した色の位置を初期位置に当てはめるのが難しい

**対策:** 射影変換をしてブロックと交点マーカの配置を単純化した。

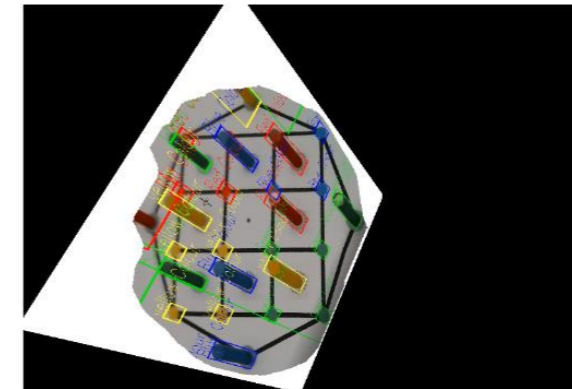


図4-5射影変換後の画像

## 4-3.経路探索

**課題:** ブロックベースエリアを走行時、ブロックベースエリアに設置済みのブロックに車体がぶつかってしまう。

**対策:** ブロック配置後、配置したブロックに隣接する直線を通行不可にする

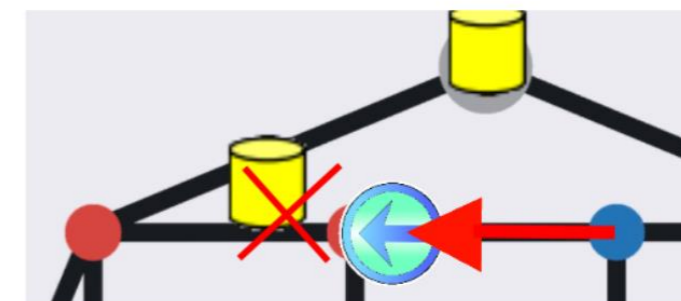


図4-6直線使用不可の例