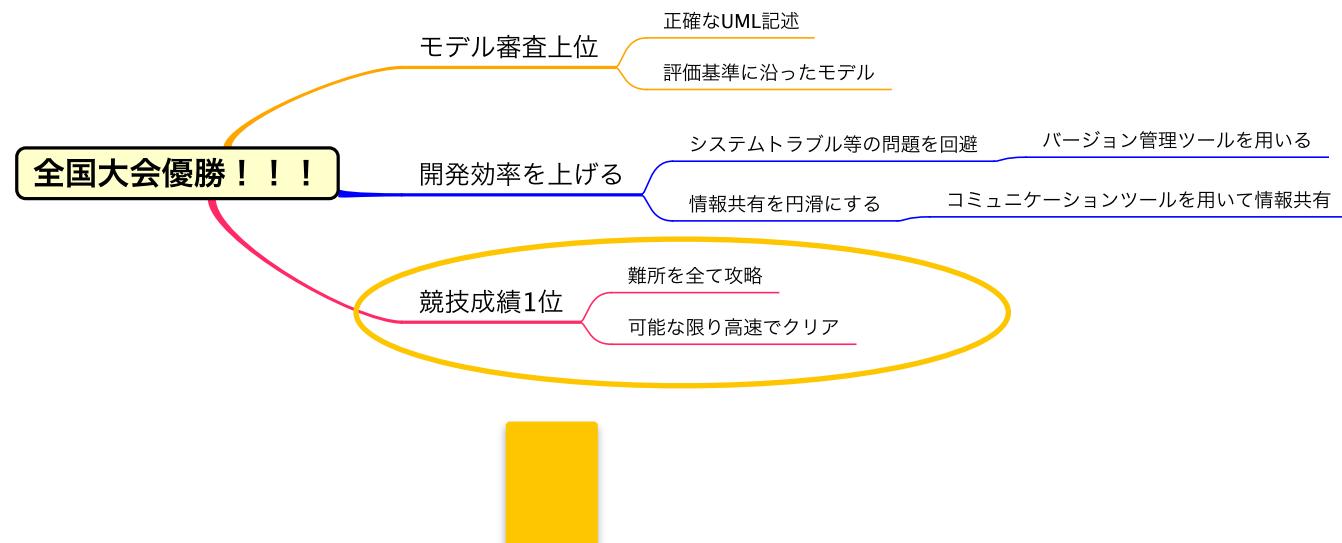


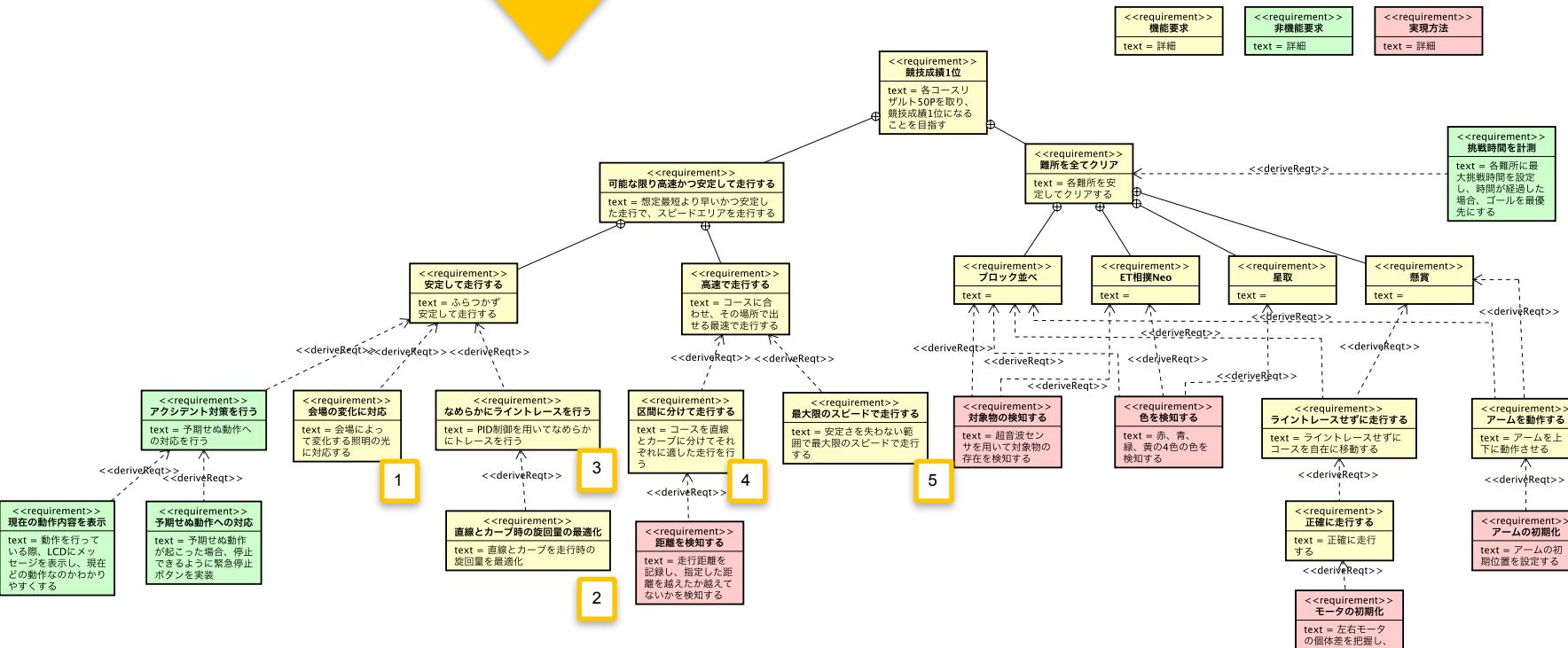
P.1 要求モデル

P.1-1 要求分析

授業で参加することを踏まえ、授業外で行う時間を少なくするため、徹夜せずに地区大会優勝を目指に掲げ、それを実現するために必要な要求をマインドマップで抽出、さらに競技ノードを要求図で細分化した。

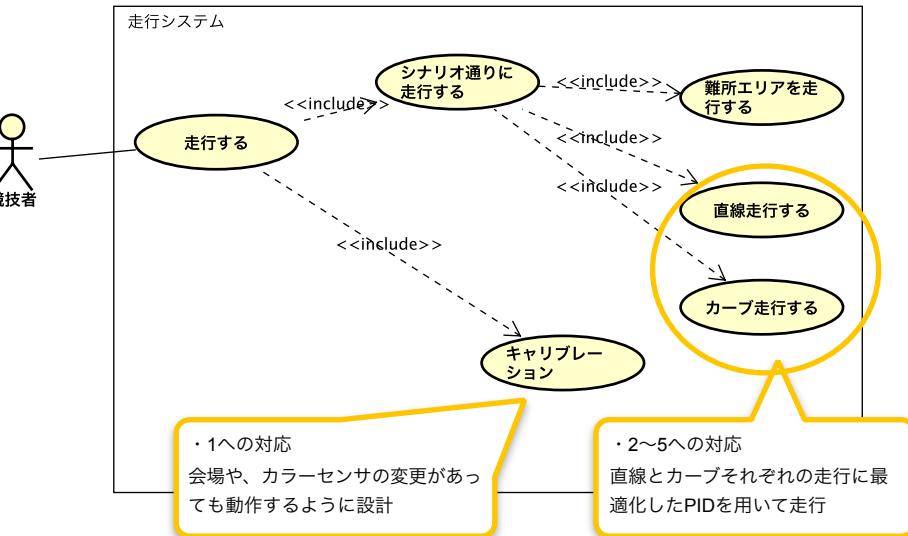


機能要件と非機能要件を明確化するためさらに細分化



P.1-2 機能要件

要求図から機能要件を抽出し、ユースケースとして明確化



ユースケース名	走行する
概要	初期動作を行い、走行を開始する
アクター	競技者
事前条件	ロボットの電源がONになっていること
事後条件	走行できている
基本系列	1. 競技者が走行システムを起動する 2. 走行システムは動作を開始するとデバイス、ライブラリを初期化する 3. 初期化後、競技者がキャリブレーションを行う 4. キャリブレーション終了後、スタート命令があるまで待機する 5. 競技者からスタート命令があるとシナリオ通りに走行を開始する。

※一例として"走行する"の記述を上げ、それ以外のユースケース記述を省略する。

P.1-3 非機能要件

1. アクシデント対策

- ・予期せぬ動作が起こった際に停止できるようなシステム設計

2. 時間管理

- ・難所エリアに時間を費やしてしまい、ゴールできない場合を考え、各難所に挑戦時間を設定し、経過するとゴールを優先した動作を行う

3. 競技者にわかりやすく

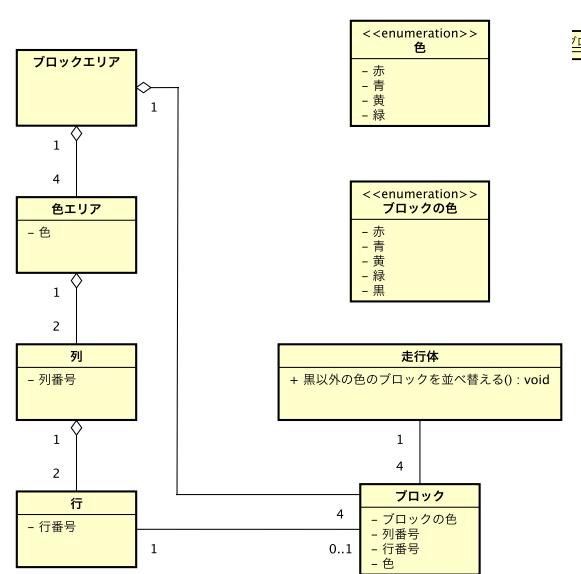
- ・バッテリーの残量、動作中のシナリオ、その他センサの値等が見てわかるようLCDに表示し、デバッグ、

P.2 分析モデル

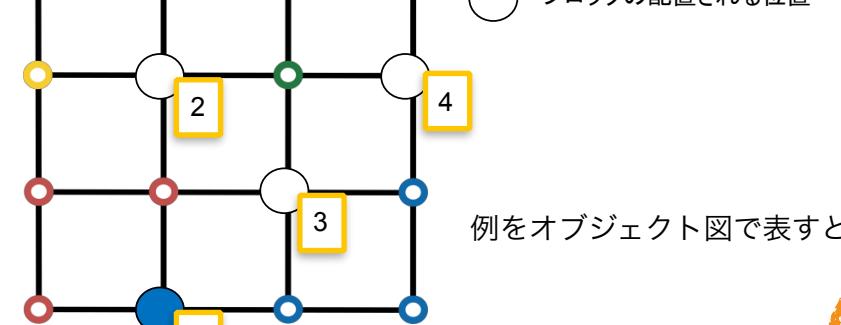
P.2-1 問題分析

モデリング対象としてRコースを選択、
Rコースのゲーム課題である「ブロック並べ」を分析した。

用語	詳細
色エリア	ブロックエリア上の各色のついたエリアのこと
ブロックエリア	ブロック並べを行うコース全体のこと



ブロックの配置される位置

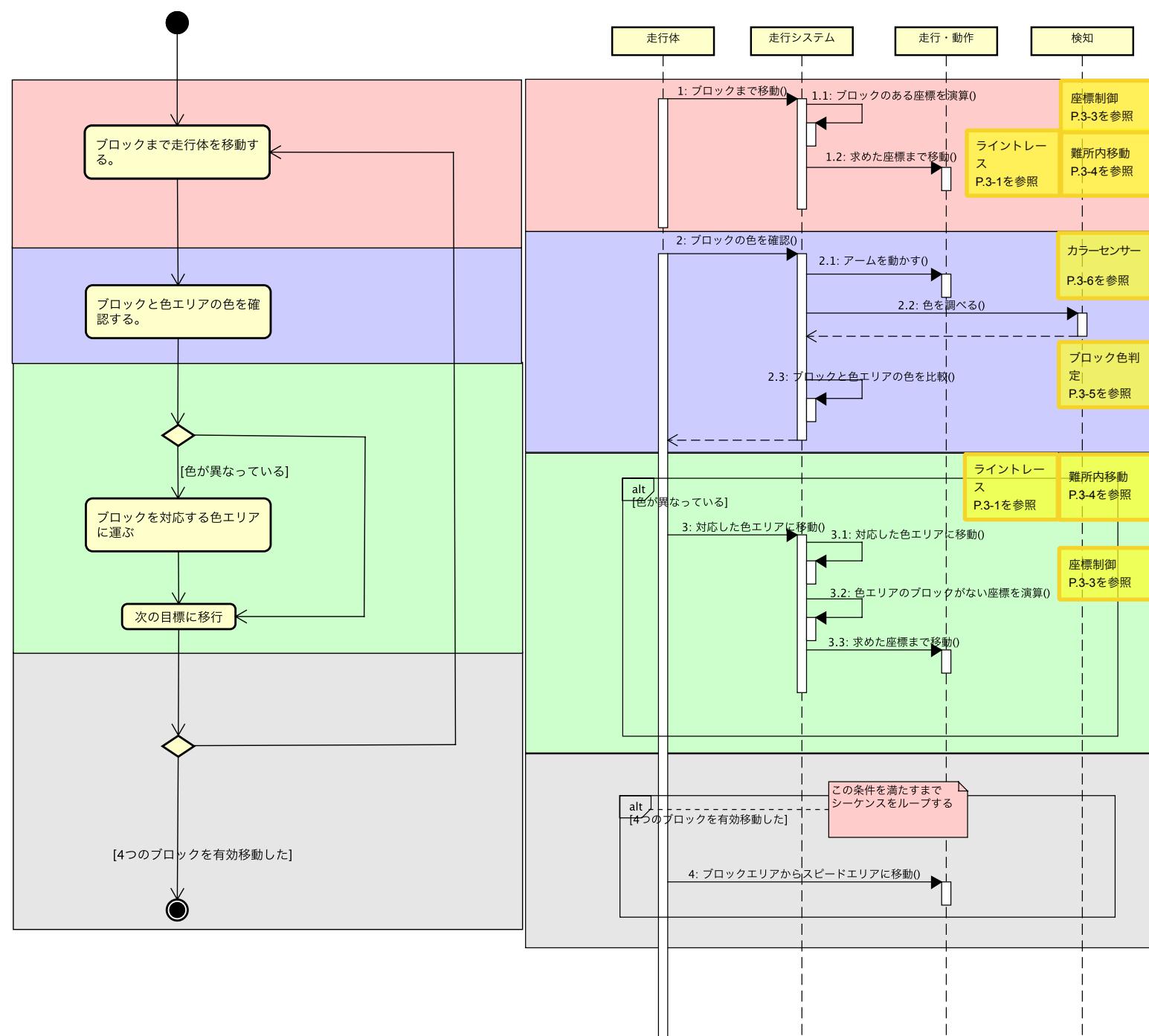


例をオブジェクト図で表すと

ブロック配置の例(競技規約より抜粋)

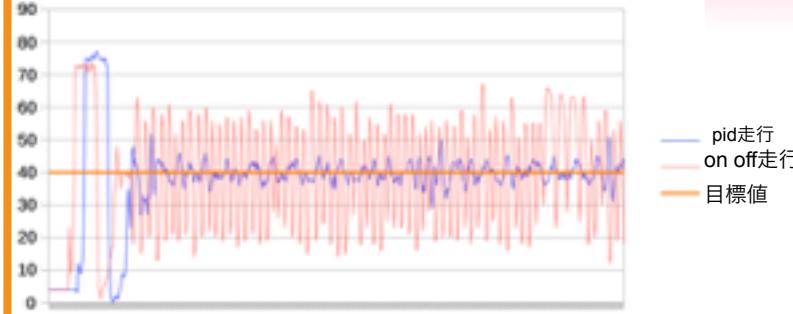
P.2-2 攻略

問題分析にしたがって指針として以下のアクティビティ図を作成した。さらに細分化してシーケンス図を作成した。



P.3 制御モデル

P.3-1 ライントレース



上グラフはPID走行時とon off走行時のカラーセンサの値をとったものである。オンオフ走行は目標値の周りを大きく行ったり来たりするのに対し、PID走行では目標値付近で収束しているのがわかる。よって、今回はPID走行を採用した。

PIDの値は、直線とカーブで別のパラメータを使用しており、大きいカーブほど、I値とD値に加算補正を掛けている。

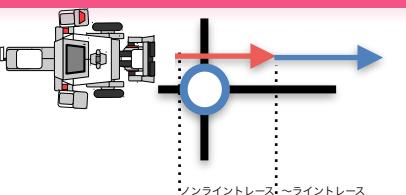
P.3-2 モータ速度補正

	5s	10s	15s
右車輪	4333	8644	13043
左車輪	4261	8514	12855

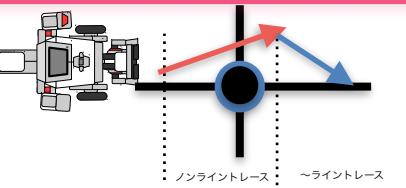
上表は、15秒間モータを回転させた場合の左右のモータの値の差を表したものである。この左右の差は、主にノンライントレース時の走行に影響を与えることが予想されるため、左モータの速度に補正(約1.015倍)をかけることで左右の速度が概ね同じになるように調整している。

P.3-3 難所内移動

P.3-4-1 ブロックが置いてない場合



P.3-4-2 ブロックが置いてある場合



難所内を移動するとき通常はライントレースをつかう。しかし、色の付いた床や他の軸をトレースすると正確な移動ができない。そのためノンライントレースを使用し、一定距離進んで通常のライントレースに戻ることで正確な走行ができるようしている。またこの移動で障害となるブロックをよけることもできる。

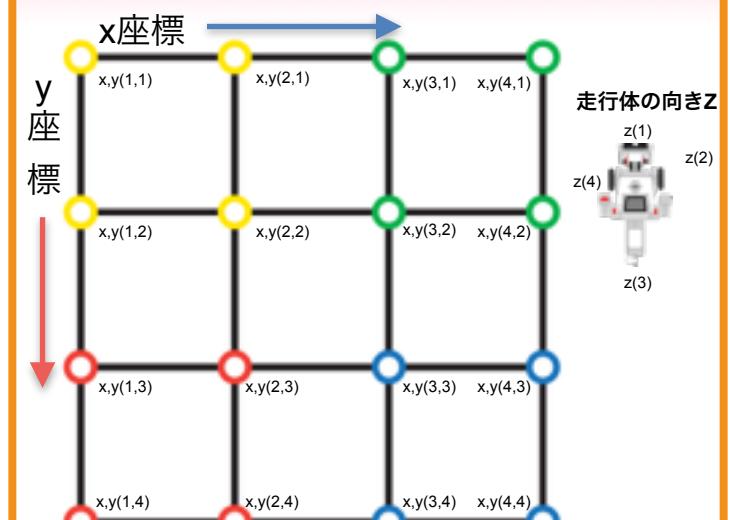
P.3-4 ブロック色判定

移動させる必要のあるブロックの元に移動したあと、そのブロックが何色かを判断するために、走行体を一定距離後退させたあと、カラーセンサ部を約50度上昇させることで、ブロックの色を検知できる。

P.3-5 LCD出力

走行制御などで動作を判断するために、`ev3_lcd_draw_string()`を用いて、現在の動作とバッテリー情報等をLCDに出力している。

P.3-8 座標制御



ブロック並べの攻略には、座標制御を用いている。エリアの左上の黄色の円を座標x,y(1,1)とし、さらに走行体の向き4パターンをzと置き、移動した先の円の数と色を判定することで自己位置を推定している。初期位置としてはx,y(1,3)のz(2)が想定されるが、x,y(3,1)に移動したい場合、赤の円1つ、青の円1つを検知したところで走行体の向きをz(1)へ変更し、緑を2つ検知するまで移動する。この際の移動には(3-4: 難所内移動)を用いる。

P.3-6 カラーセンサ

明度を20%ずつ変化させたRGB値	R	G	B	明度を20%ずつ変化させたRGB値	R	G	B
	100	245	51	17	127	193	29
80	164	52	17	97	173	25	
	60	106	47	16	80	123	23
40	55	38	13	63	82	20	
	100	60	66	59	278	242	29
80	56	61	54	225	189	29	
	60	55	49	43	140	116	22
40	48	44	29	100	81	20	
	100	230	50	17	114	177	26
80	226	53	20	116	183	26	
	60	236	63	31	127	191	32
40	240	102	53	158	205	58	
	100	58	59	54	249	224	27
80	56	59	50	261	233	30	
	60	66	70	59	276	245	42
40	158	205	58	273	242	67	

上表は明度と彩度を20%ずつ変化させたものを、カラーセンサを使用しRGBの値を取ったものである。この値から「赤」、「緑」は明度が変わると値の変動が大きく、「黄」は彩度の変化で値が変動することが分かった。また青は半分以下にならないとあまり変動が無いので黒との誤検知を防ぐため値を区切って使っている。色を判断する部分では、これらの値を使用して判断する。

P.3-7 ベーシック走行制御

・凡例

LT_①
(②)

- ①: ライントレースの種類
S:直線用 C:カーブ用
②: スピードの大きさ
大, 中, 小

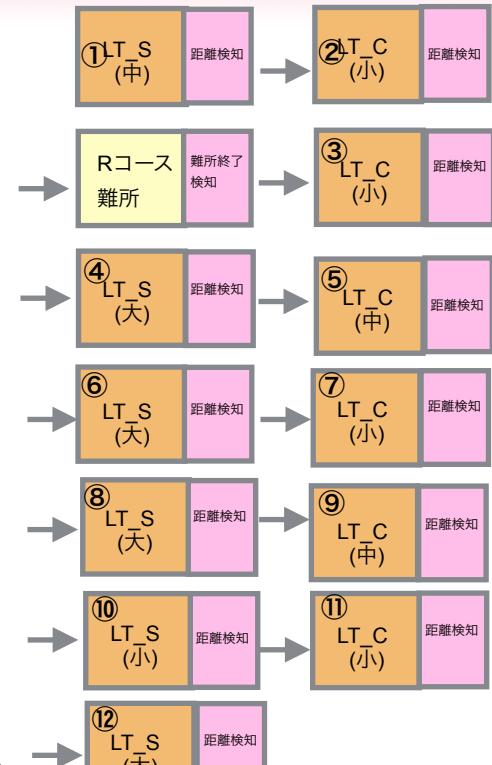
色分け

直進用

カーブ用

ここでライン
を変更する

p i d
ストレート 0.30 0.05 0.08
カーブ 0.70 0.16 0.03



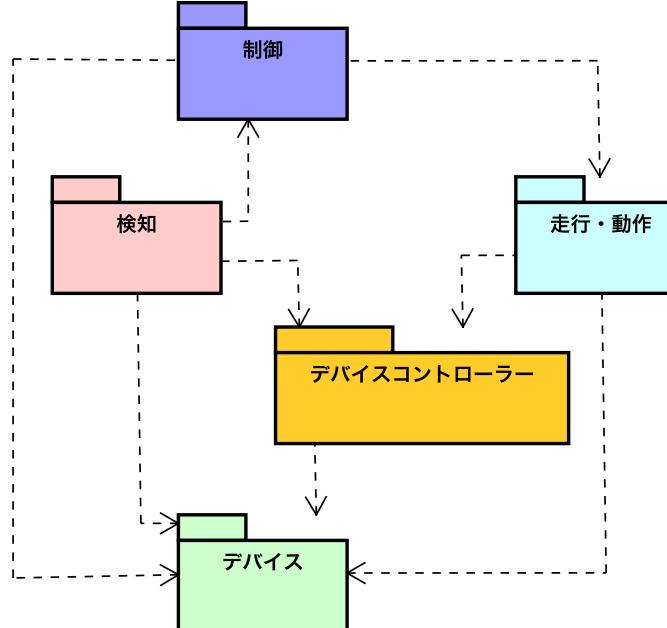
走行タイムはベーシックと難所の攻略を合わせて2分しかないので、ベーシックの攻略速度を出来るだけ上げる必要がある。そのため、区間毎に速度とPID(3-1)の値を変更し、滑らかで高速な走行を実現する。

P.4 設計モデル(1)

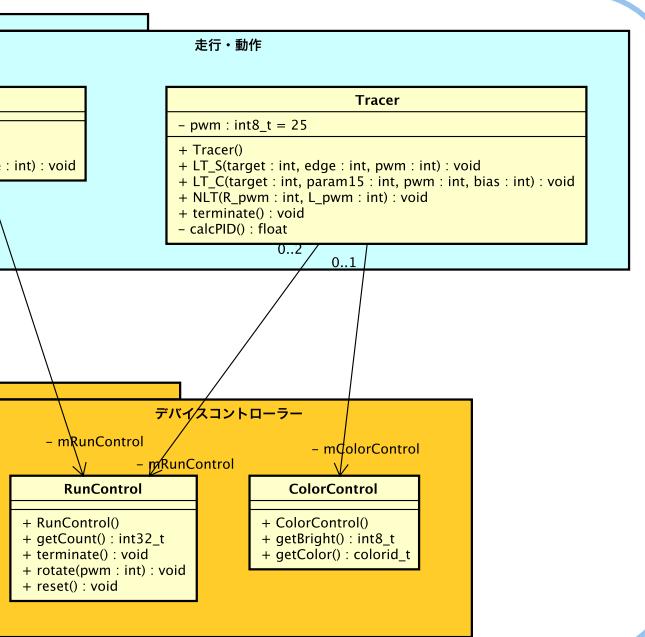
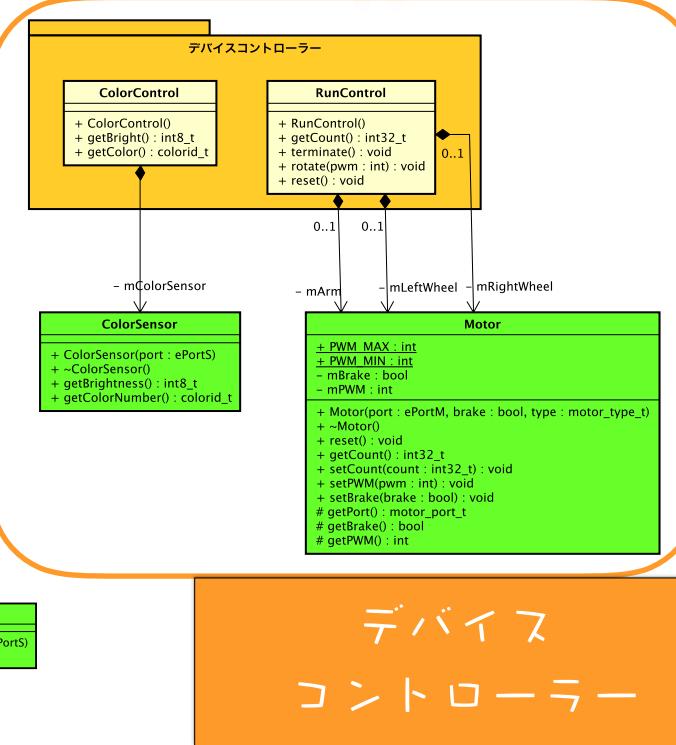
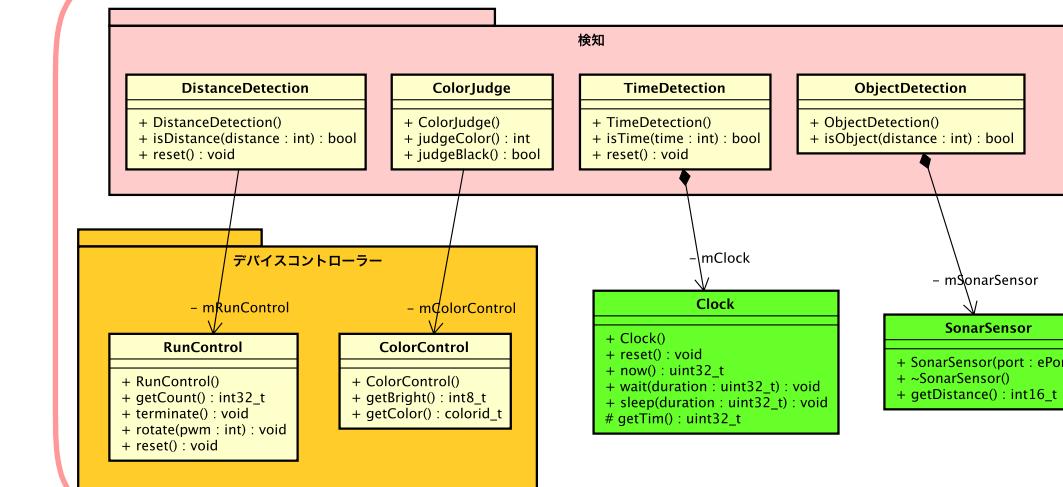
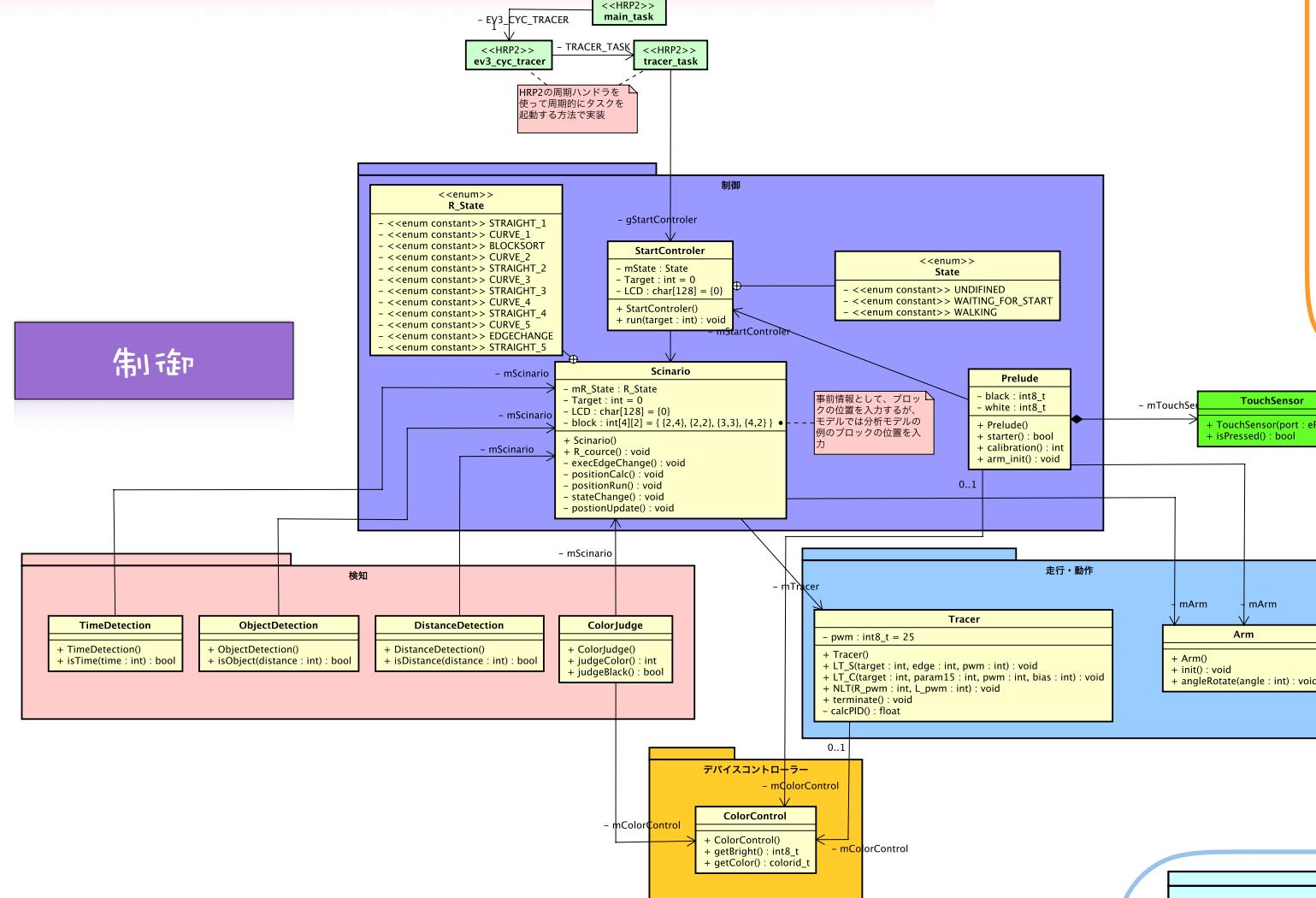
P.4-1 ドメイン分析

制御モデルを考えた制御を行うために、システムを6つのドメインに分割し、構造と振る舞いを明確化し、それぞれが行う処理をまとめた。

ドメイン	役割
制御	コースをクリアするためのシナリオを読み込み、走行指示/検知指示/初期動作指示を行う
走行・動作	制御から指示された走行や動作を行う
検知	指示された検知を行い、検知したかしていないかを制御に知らせる
デバイスコントロール	複数のクラスが操作するデバイスをコントロールする
デバイス	各センサやモータなどのデバイス



P.4-2 クラス分析



檢知

P.5 設計モデル(2)

P.5-1 振る舞い分析

P4で求めた構造を用いて、P2の攻略方法の一部の動作を分析した。以下のステートマシンとシーケンスは、動作の全体動作のステートマシン図とステートマシン図内のステートマシン”Rコース”内で行っている動作のシーケンスで例として、区間制御のシナリオ”STRAIGHT_1”と”CURVE_1”的シーケンスを記述している

