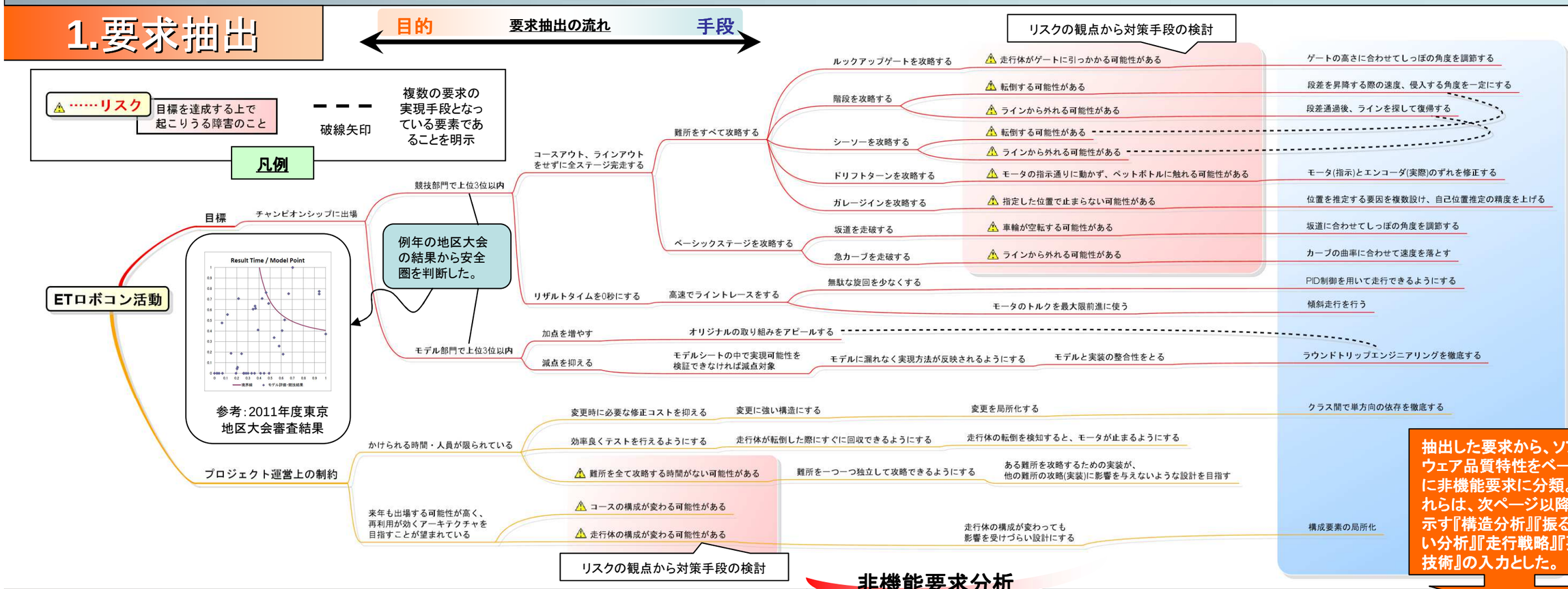
1.要求分析

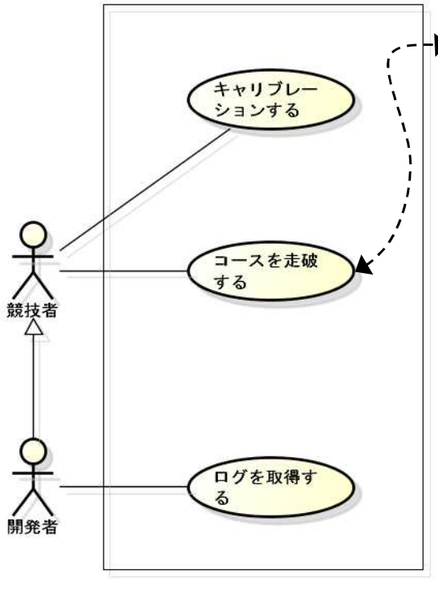
例年の地区大会突破チームの成績の傾向を踏まえ、チャンピオンシップ出場を果たすために必要な要求を整理した。
『目的』から達成するための要素を洗い出して分析し、それぞれの要素に対して必要な『手段』を決定した。

オージス総研 芝浦雑伎団

1.要求抽出



2.ユースケース分析



項番	B01
ユースケース名	コースを走破する
概要	走行体をライン上に沿って走らせる
アクター	競技者
事前条件	ユースケース「キャリブレーションをする」が終了している 走行体がスタート地点に置かれ、完全停止状態になっている
事後条件	完全停止状態になっている
基本フロー	1. アクタは、走行開始指示を行う(※1) 2. システムは、設定された区間に従い走行体を走行させる 3. システムは、区間の終了を認識する （最後の区間終了を認識するまで2～3を繰り返す） 4. システムは、ガレージの中で走行体を完全停止状態にする
例外フロー	e1. 走行体の転倒を検出した場合 e1-1. システムは走行体を完全停止状態にする e2. 走行体の停止指示を受信した場合(※2) e2-1. システムは走行体を完全停止状態にする
備考	(※1)走行開始指示は、タッチセンサによる指示と、Bluetoothによる遠隔指示の両方に対応している。 (※2)他の走行体との衝突、コースからの転落を回避するため、Bluetoothによる遠隔指示で走行体を止めることができる。

※「キャリブレーションする」「ログを取得する」は紙面の都合上、省略。

非機能要求分析

品質特性	要求	実現手段	
機能性	リザルトタイムを0秒にするため、高速でライントレースをする	PID制御を用いて走行し、無駄な旋回を少なくする	P5.『輝度補正』
		傾斜走行を行い、モータのトルクを最大限前進に使う	P4.『ベーシックステージ』
信頼性	『ルックアップゲート』で走行体をゲートに引っかけない	ゲートの高さに合わせてしっぽの角度を調節する	P4..『ルックアップゲート』
	『階段』『シーソー』で転倒しない／ラインから外れない	段差を昇降する際の速度、侵入する角度を一定にする	P4..『階段』『シーソー』 P5.『方位補正』 P5.『振動検知』
		段差通過後、ラインを探して復帰する	
	『ドリフトターン』でペットボトルに触れずに通過する	モータ(指示)とエンコーダ(実際)のずれを修正する	P4..『ドリフトターン』 P5.『曲率補正』
	『ガレージイン』で指定したエリア内に完全停止する	位置を推定する要因を複数設け、自己位置推定の精度を上げる	P4..『ガレージイン』 P5.『複合切替条件』
	『ベーシックステージ』の坂道で車輪を空転させずに登りきる	坂道に合わせてしっぽの角度を調節する	P4..『ベーシックステージ』
	『ベーシックステージ』の急カーブでラインから外れず走行する	カーブの曲率に合わせて速度を落とす	P4..『ベーシックステージ』
保守性	効率良くテストを行える構造にする	走行体の転倒を検知すると、モータが止まるようにする	P5.『振動検知』
	変更に強い構造にする	クラス間で単方向の依存を徹底する	(P2.構造分析全般)
移植性	走行体の構成が変わっても影響を受けづらい設計にする	構成要素の局所化	(P2.構造分析全般)

2.構造分析

『コース』の構成要素である『エリア』を走破するための戦略を、基本的な動作である『プロセス』の組み合わせで表現。
『プロセス』は『走法』と『切替条件』の組み合わせで構成しており、特定の難所に依存しない設計を実現している。

オージス総研 芝浦雑伎団

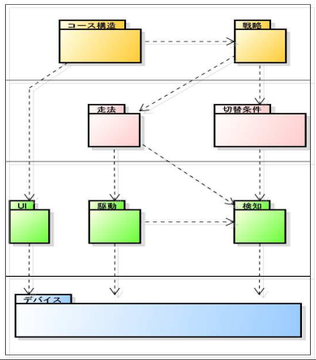
パッケージ図

部品化の推進

レイヤ構造をとり、
レイヤ及びパッケージ
別の責務を明確化
することで実現。

変更の影響を局所化

パッケージ間において
双方向の依存を
排除することで実現。



レイヤ説明
コースを走破するという目的を達成するためのレイヤ
目的を達成するための情報や走り方を提供するレイヤ
データ収集や動作するために、デバイスからの情報取得や、具体的な動作を提供するレイヤ
デバイスAPIが存在するレイヤ

パッケージ名	説明
コース構造	コースがどのようなエリアで構成されているか、コースの構造を保持する
戦略	どうすればコースのエリアを走破できるかプロセスを保持する
走法	どのように走行体を走らせるか姿勢と走法を保持する
切替条件	戦略の切替タイミングを保持する
UI	ユーザとシステム間のやりとりをサポートする
駆動	走行体を動かすための手段を保持する
検知	走行体が各種センサから得た値を提供する
デバイス	NXTのデバイスAPIを保持する

全体構成

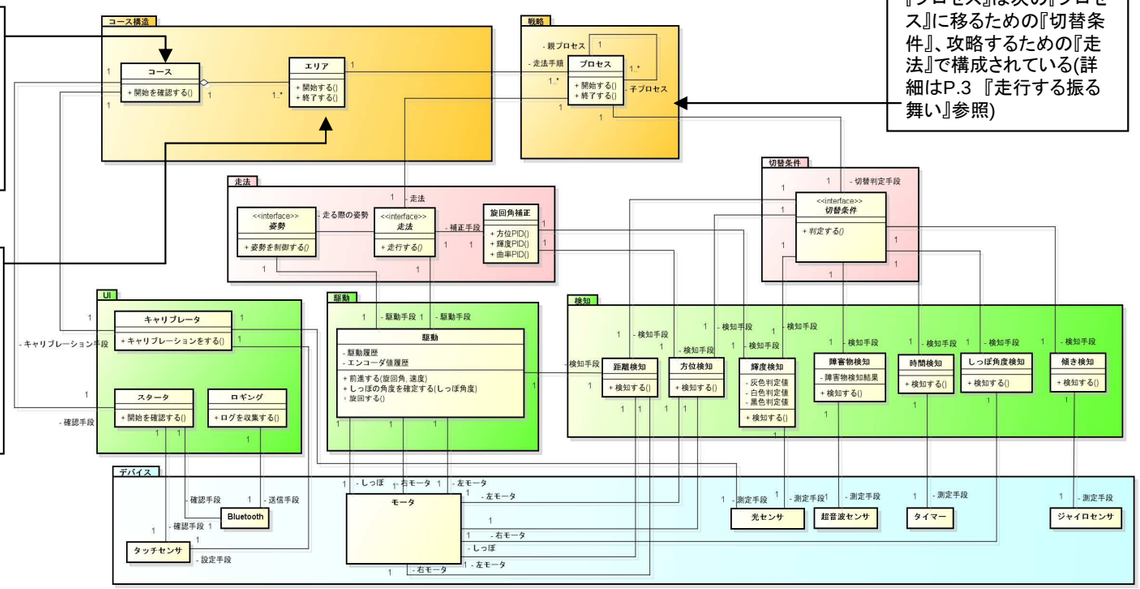
設計意図

戦略や走行方法を司るレイヤを、デバイスと直接関連を持たない設計にすることで、走行体の構成が変わっても影響が少ないようにしている。

<コース>
インコースまたはアウトコースにおけるそれぞれの『エリア』と順番の情報を保持している。

<エリア>
『コース』を構成している要素。具体的には「階段」「シーソー」などが挙げられる(詳細はP.4『走行戦略』参照)

<プロセス>
『エリア』を攻略するための手順。
『プロセス』は次の『プロセス』に移るための『切替条件』、攻略するための『走法』で構成されている(詳細はP.3 『走行する振る舞い』参照)



走法パッケージ詳細

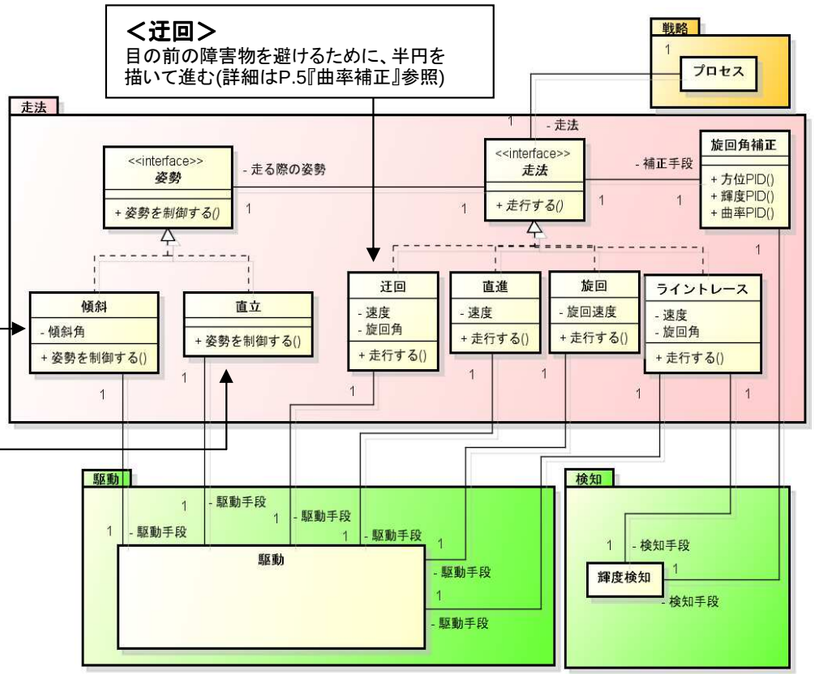
設計意図

『プロセス』は具体的な走法を意識せずに走行できる。
『走法』と『姿勢』を分けることで、柔軟に走行時の走法と姿勢を切り替えられる。

<迂回>
目の前の障害物を避けるために、半円を描いて進む(詳細はP.5『曲率補正』参照)

<傾斜>
尻尾を用いて走行体が傾斜している状態

<直立>
尻尾に頼らず、倒立振子制御を行ない直立している状態



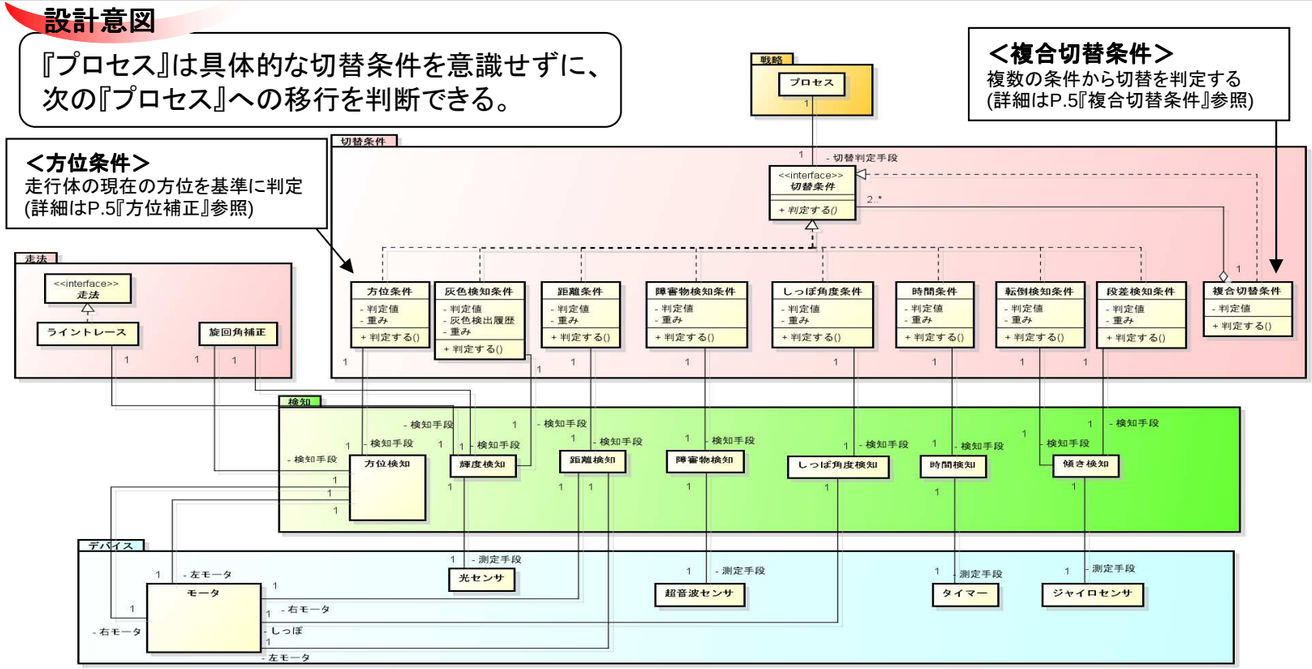
切替条件パッケージ詳細

設計意図

『プロセス』は具体的な切替条件を意識せずに、次の『プロセス』への移行を判断できる。

<方位条件>
走行体の現在の方位を基準に判定
(詳細はP.5『方位補正』参照)

<複合切替条件>
複数の条件から切替を判定する
(詳細はP.5『複合切替条件』参照)

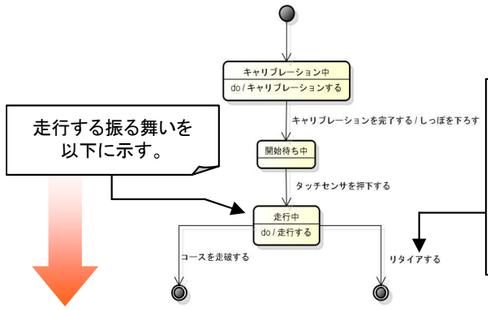


3.振る舞い分析

構造分析で示した『エリア』を走破するための一つ一つの手順である『プロセス』を用いて、
『コースを走破する』ユースケースが実現できることを検証した。また、ドリフトターンで用いる『障害物検知タスク』が、コースを走破する目的の『走行タスク』に影響を及ぼさないよう、並行性設計を実施した。

オージス総研 芝浦雑伎団

システム全体の状態遷移



走行する振る舞いを以下に示す。

リタイアするとは、コースアウトや転倒など、競技継続が困難であると判断されたことを意味する。

難所:ドリフトターンの振る舞い

『プロセス』ごとに走法を変化させ、『切替条件』で切替えることでドリフトターンを走破できることを示す。

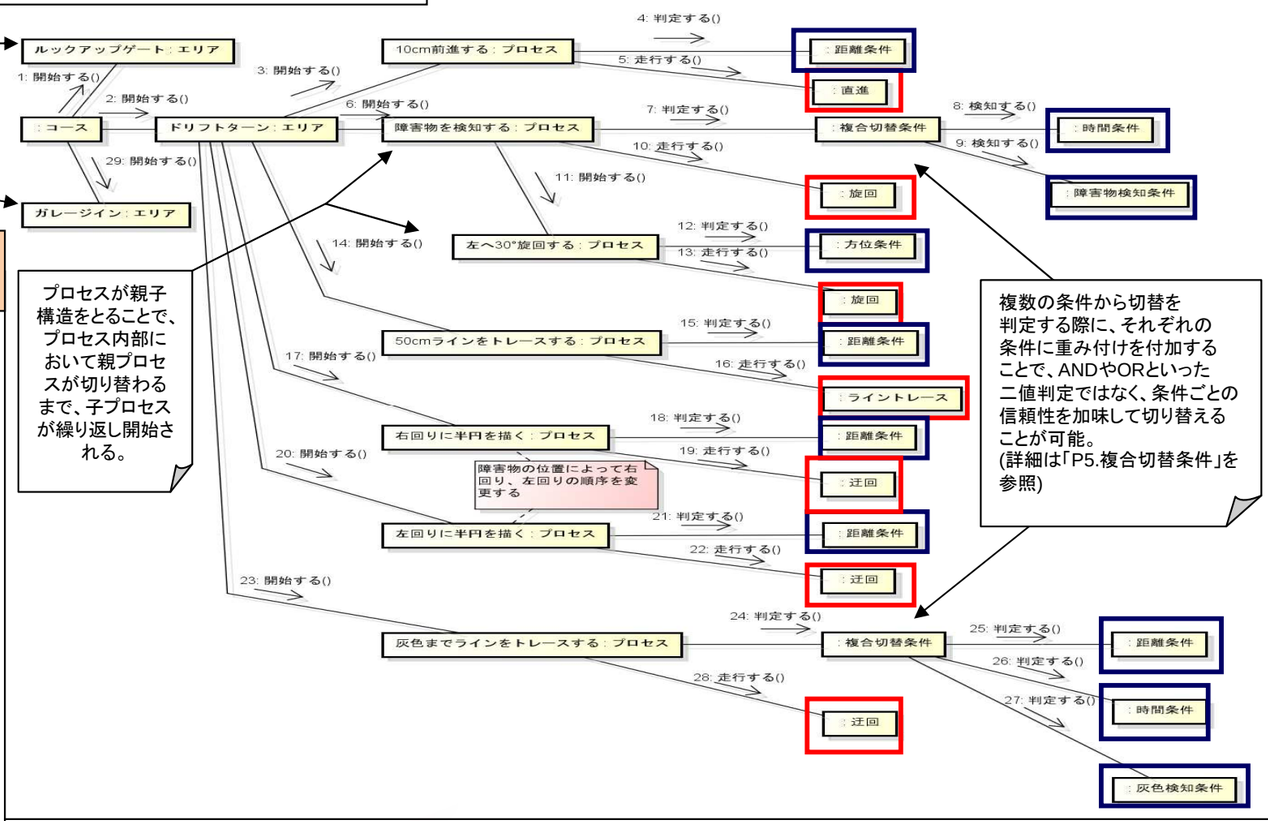
青枠で示された『切替条件』は下図の「切替条件の振る舞い詳細」を参照。

赤枠で示された『走法』は下図の「走法の振る舞い詳細」を参照。

ドリフトターンの前のエリア

エリアの最後のプロセスが終了したら次のエリアを開始する。

ドリフトターンの後のエリア



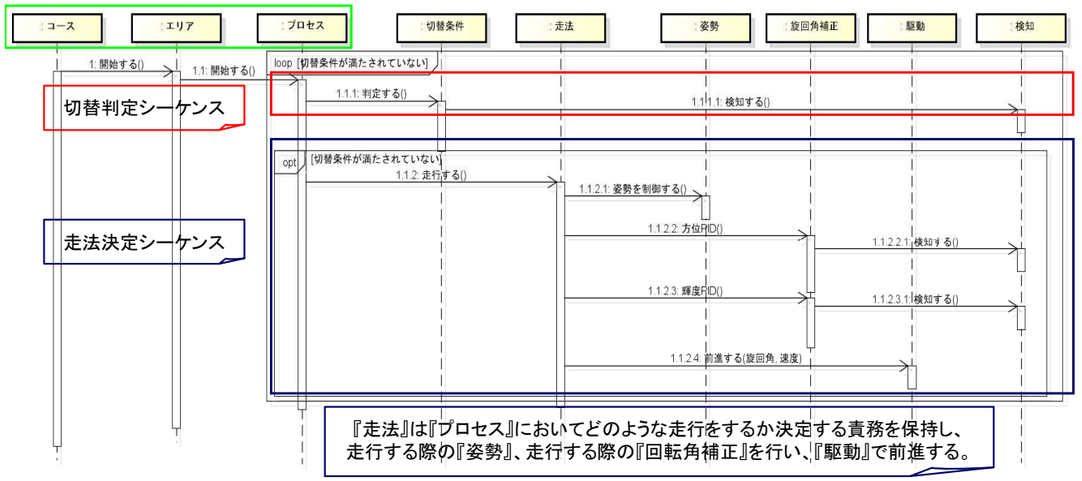
プロセスが親子構造をとることで、プロセス内部において親プロセスが切り替わるまで、子プロセスが繰り返し開始される。

複数の条件から切替を判定する際に、それぞれの条件に重み付けを付加することで、ANDやORといった二値判定ではなく、条件ごとの信頼性を加味して切り替えることが可能。
(詳細は「P5.複合切替条件」を参照)

走行する振る舞い

『コース』は階段やシーソーなど複数の『エリア』を順番に保持している。
『エリア』は10cm前進するなどの複数の『プロセス』を順番に保持している。

『切替条件』は『検知』から必要な情報を取得し、次の『プロセス』に切り替わるか否か判定する。



『走法』は『プロセス』においてどのような走行をするか決定する責務を保持し、走行する際の『姿勢』、走行する際の『回転角補正』を行い、『駆動』で前進する。

並行性設計

タスク分割の理由

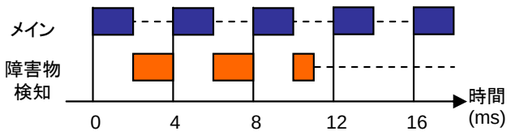
走行体の倒立振子制御には4msの周期での実行が、超音波センサが正確に障害物を検知するには40msの周期での実行が必要である。これらの制御が互いに与える影響を排除するため、右表のようなタスク設計を行った。

タスク名	周期	優先度	実行時間
走行タスク	4ms	高	1～2ms
障害物検知タスク	40ms	低	約5ms

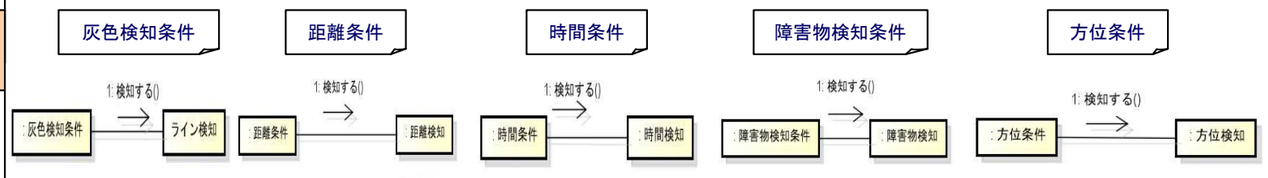
走行タスクと障害物検知タスクの優先度を逆にしてしまうと、倒立振子制御のデッドライン(4ms)を守ることができない。

タスク分割の妥当性検証

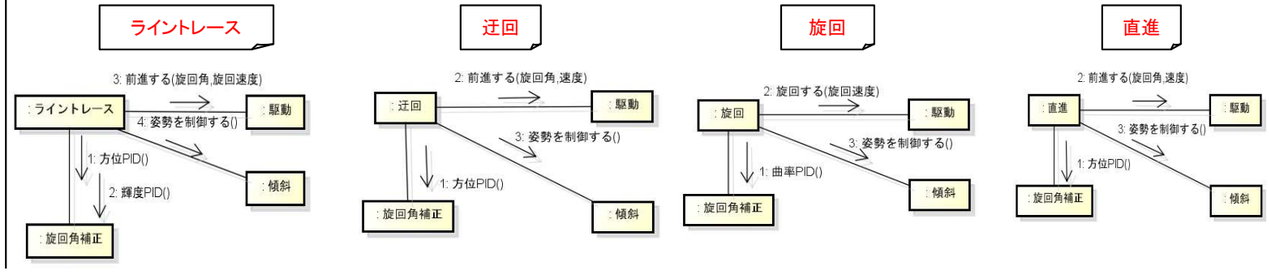
メインタスクの倒立振子制御はジャイロセンサの性能制約上、4ms毎に必ず実行される必要がある。一方、障害物検知タスクは40ms以内に「障害物検知結果」を更新できていれば十分である。右図に、各タスクが実行周期内で終わることを示す。



切替条件の振る舞い詳細



走法の振る舞い詳細

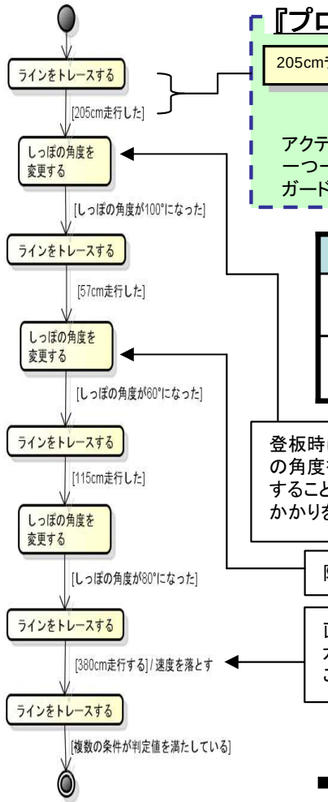


4.走行戦略

コース上に存在する『エリア』に対してどのような振る舞いを用いて走破するのか記述した。
また、要求抽出で明らかになった各エリアを走破する際のリスクと、その具体的な回避策を決定した。

オージス総研
芝浦雑伎団

ベーシックステージ



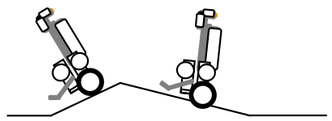
『プロセス』の組み合わせで全てを走破

205cmラインをトレースする：プロセス
205cm：距離条件
速度90：直進
80°：傾斜

アクティビティの
一つ一つが、『走法』『姿勢』クラスの組み合わせを、
ガード条件が『切替条件』クラスを示している。

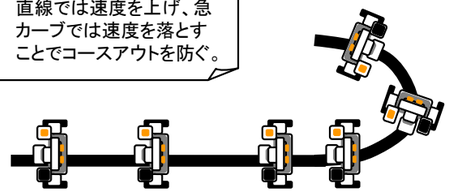
リスク	回避策
坂で速度が落ちる。	坂の傾斜に合わせてしっぽの角度を変更する。
急カーブでラインを見失う。	急カーブ走行中は速度を落とす。

登板時はしっぽの角度を急にすることで引っかかりを無くす。

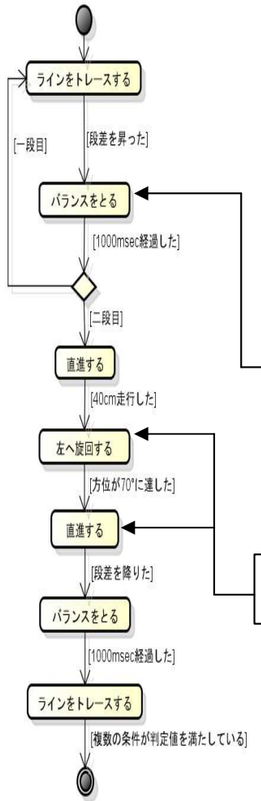


降板時はしっぽの角度を緩くすることで転倒を無くす。

直線では速度を上げ、急カーブでは速度を落とすことでコースアウトを防ぐ。

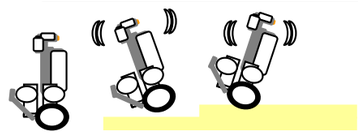


階段

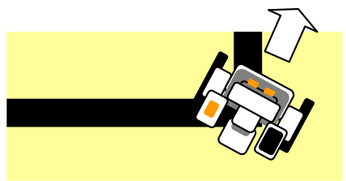


リスク	回避策
段差の昇降時に転倒する。	段差を登るごとにバランスをとる。【要素技術：振動検知】
昇段・降段時の衝撃で走行ラインを見失う。	昇段・降段後に方位を指定して直進し、ラインに復帰させる(必ずラインの右側に走行体を移動させる)【要素技術：振動検知、方位補正】
しっぽが段差に引っかかり階段に登れない。	しっぽを用いた走行から倒立振子制御を用いた走行に切り替える。【関連：走法と姿勢を分離する設計】

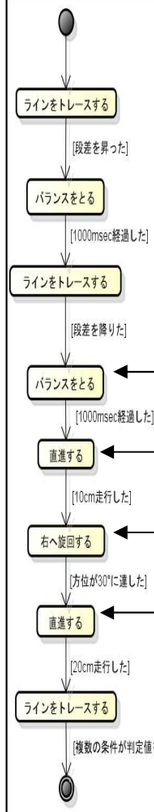
段差を登るごとにすぐ動くのではなく、一旦バランスを調整することで転倒を防ぐ。



段差を昇降する際の衝撃でラインを見失う可能性があるので、昇降後にラインの右側へ移動させることでラインを発見する。

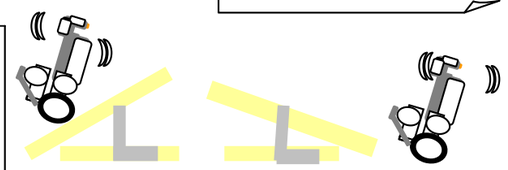


シーソー



リスク	回避策
段差の昇降時に転倒する。	段差を昇降するごとにバランスをとる。【要素技術：振動検知】
昇段・降段時の衝撃で走行ラインを見失う。	昇段・降段後に方位を指定して直進し、ラインに復帰させる(必ずラインの右側に走行体を移動させる)【要素技術：振動検知、方位補正】

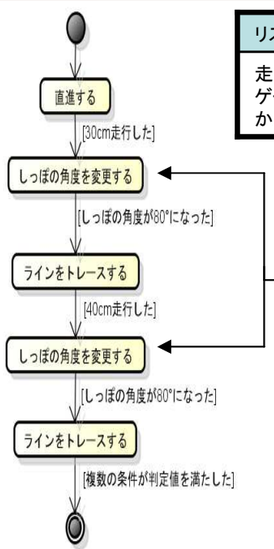
シーソー昇降後にすぐ動くのではなく、一旦バランスを調整することで転倒を防ぐ。



すぐに旋回するとシーソーに当たるため少し前進する

※昇降時のラインを見失う場合への対処は階段時と同様のため詳細説明は省略する。

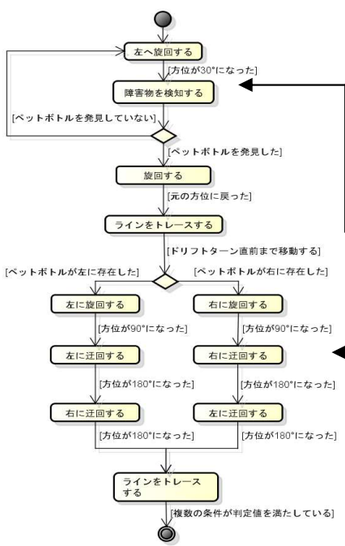
ルックアップゲート



リスク	回避策
走行体がゲートに引っかかる。	しっぽの角度を変える。【関連：走法と姿勢を分離する設計】

ルックアップゲート時にのみしっぽの角度を変更し、元の角度に戻すことで光センサの値を安定させる。

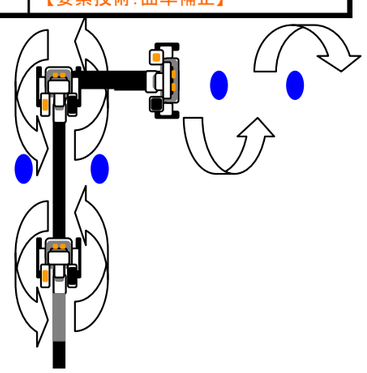
ドリフトターン



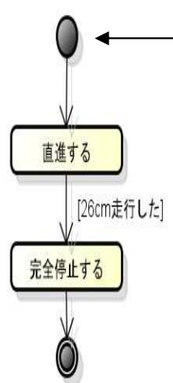
リスク	回避策
ペットボトルを確認する位置がずれ、ペットボトルを発見できない。	左へ30°ごとに旋回してペットボトルの有無を確認することで、360°ペットボトルを確認する。
ペットボトルに衝突してしまう。	ペットボトルを避けて進む。【要素技術：曲率補正】

測定する位置がずれてしまっても360°ペットボトルを確認することで確実に発見する。

ペットボトルの位置によって旋回する方向を変更する。

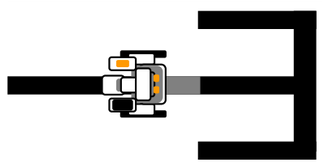


ガレージイン



リスク	回避策
ガレージに入らずに完全停止状態に入ってしまう。	複数の条件からガレージへの進入するタイミングを判断する。【要素技術：複合切替条件】

ガレージインエリアへ切り替わる際に、時間は想定よりも早いが、走行距離が満たされ灰色ラインを検知したためガレージへの進入タイミングであると複数条件から判断する。



5.要素技術・開発支援

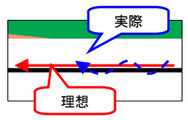
ベーシックステージを速く、難所をより安全・確実に攻略するという要求を満たすために必要な要素技術を検証した。また、設計と実装の整合性を保持するため、ラウンドトリップエンジニアリングを実施した。

オージス総研 芝浦雑伎団

輝度補正

課題

ライントレース時に、路面が「白」か「黒」かの判断のみで固定の旋回量を与えていると、**無駄な旋回が生じる。**



解決手段

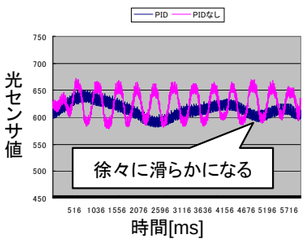
走行閾値(白と黒の間)を目標値としてPD制御を行うことで、ラインから外れた分に応じた旋回量を補正し、**滑らかなライントレースを実現する。**

$$偏差 = (取得した光センサ値 - 走行閾値)$$

$$補正旋回量 = (偏差 \times kp - kd \times (偏差 - 前回の偏差))$$

検証

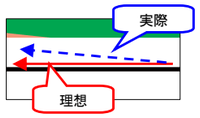
無駄な旋回を無くすことで、**モータの駆動を前進に集中させる**ことができるようになった。



方位補正

課題

左右のモータに同一の駆動量を与えても、**モータによって個体差があるため、直進することができない。**



解決手段

走行体の方位を目標値としてP制御を行うことで、左右モータの個体差に依存しない**一様な走行を実現する。**

以下の式を用いて左右モータの回転角から方位を求める。

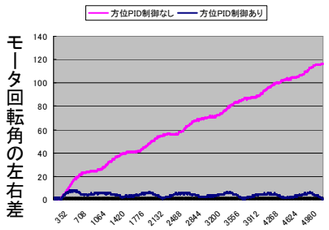
$$左モータ走行距離 = (左モータ回転数 - 前回の左モータ回転数)\frac{2\pi \times 車輪半径}{360}$$

$$右モータ走行距離 = (右モータ回転数 - 前回の右モータ回転数)\frac{2\pi \times 車輪半径}{360}$$

$$走行体の方位 = (左モータ走行距離 - 右モータ走行距離)\frac{360}{2\pi \times 車軸長}$$

検証

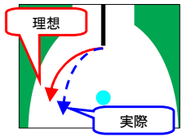
方位0を目標値に指定することで、**正確な直進ができている**ことを確認した。



曲率補正

課題

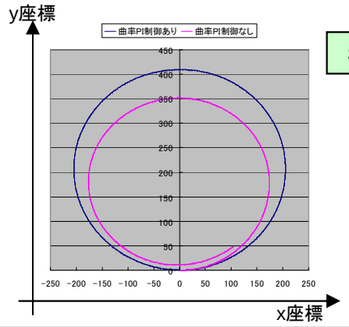
左右のモータに同一の駆動量を与えても、**モータによって個体差があるため、正確な円を描くことができない。**



解決手段

理想とする軌跡の曲率を目標値としてPI制御を行い、左右モータの個体差に依存せずに**一様な円運動を実現する。**

$$走行軌跡の曲率 = \left(\frac{(右モータ回転数 - 左モータ回転数}{左モータ回転数}\right)$$



検証

曲率半径20.5cmの円を目標とする軌跡に指定することで、**正確な円運動ができている**ことを確認した。

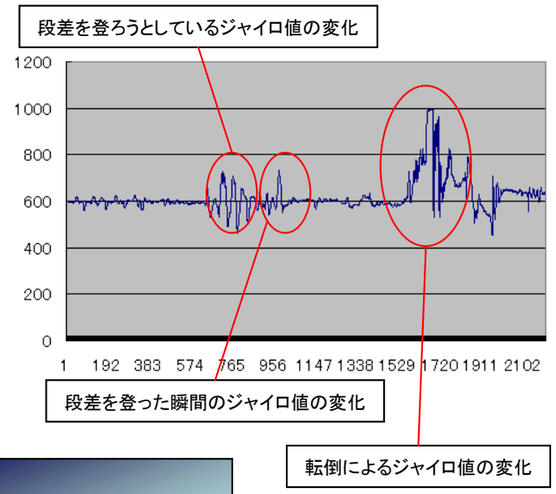
振動検知

課題

階段およびシーソー攻略においては**昇段・降段を検知**することが必要になる。また、転倒時にモータを止める必要がある。

解決手段

ジャイロセンサ値の変化を計測し、**変化の特徴**から昇段・降段および転倒を判断する。



姿勢制御

課題

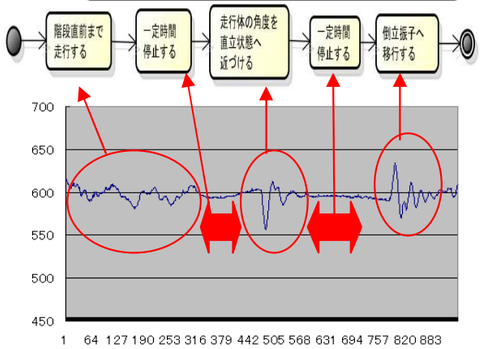
姿勢を切替える際、走行体の振動がすぐには収まらないため、**動作が安定しない。**

解決手段

直立状態へ移行する前と、倒立振子制御へ移行する前に**一時停止し、安定させる。**

検証

ジャイロセンサの値が実際に**安定してから姿勢制御できている**ことを確認した。



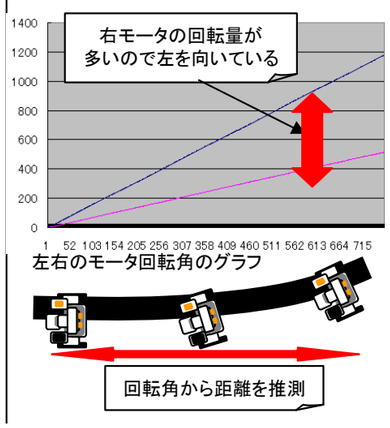
自己位置推定

課題

プロセスを切り替える際に走行体が現在どこを走っているか確認しなければならない。

解決手段

モータ回転角から距離を、**左右のモータの差から走行体の方位**を推測し、走行体の位置を推測する。



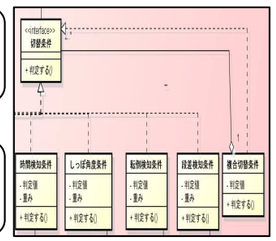
複合切替条件

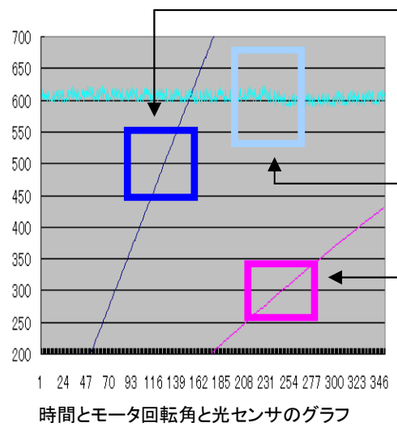
課題

次のプロセスへの切替を判断する際、**基準が1つのみでは切替のタイミングが安定せず**、期待通りに走行しないことが考えられる。

解決手段

複数の条件において重み付けをし、ある一定値を超えたらその条件が満たされたと判断することで、**切替の精度を向上させる。**





電池残量の影響で時間に対する走行距離が異なっている可能性がある。

灰色ラインではなく、黒ラインと白の境界における値である可能性がある。

エンコーダ値が実際の距離とは異なってる可能性がある。

一つの条件だけで判断するのではなく、重みをつけて複数から判断する。

時間とモータ回転角と光センサのグラフ

開発支援

ラウンドトリップエンジニアリング

影響の局所化や再利用性を意識した設計を実装に落とし込むことで、実装でも影響の局所化や再利用性が得られる。
しかし、設計から実装へ落とし込む作業、および設計と実装が合致しているか確認するのは難しい。
そこで、モデルからの自動生成及びソースコードからモデル自動生成を行うことで簡単に整合性を確認する。