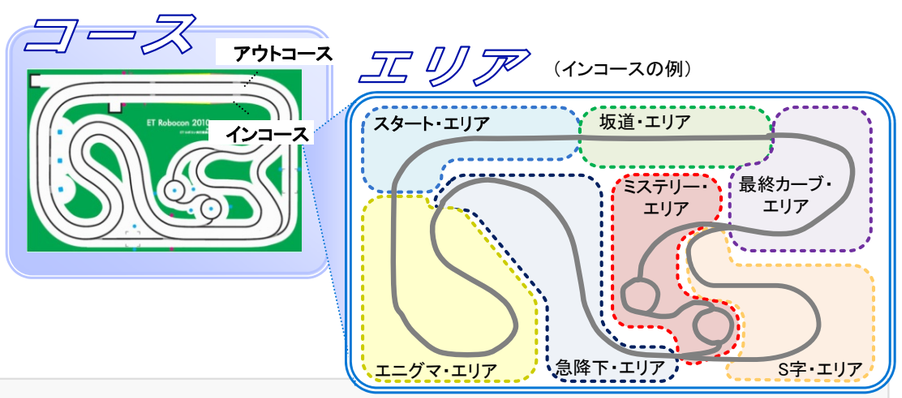
要求分析

エリアごとに分割することで最適な走りを検討する

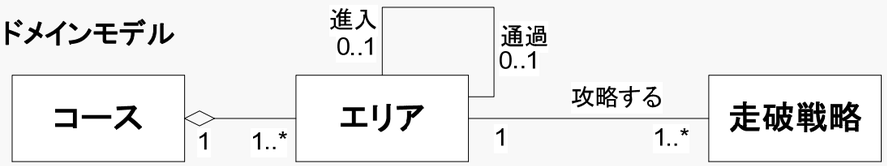
ロボットが走行するコースは直線、カーブ、坂道、難所といった様々な特徴をもつエリアにより構成されています。各エリアを走破するためには、**エリアの特徴に応じた実現可能な走破戦略を立案**する必要があります。2010年度のコースでは、エニグマデコーディング結果に応じた**動的な経路の選択**や、シーソー、階段といった**立体的な難所**が存在しており、これらをどのように攻略するかが、競技で好成績を残すために重要であると考えています。

①ドメイン分析

コース上の特徴のある場所を1つのエリアと捉えると、コースは複数のエリアが繋がったものと考えられます。**エリアごとに分割することで並行して最適な走行戦略を検討**することができます。



ドメインモデル

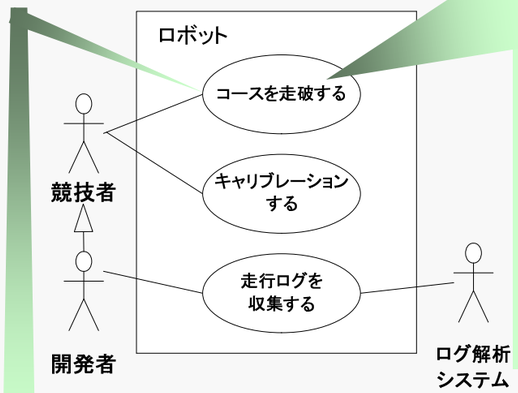


エリアごとに最適な走破戦略を検討する。

②ユースケース分析

開発対象をユースケースモデルで整理しました。**「コースを走破する」ユースケース**が最重要ユースケースです。

ユースケースモデル



ユースケース名	コースを走破する
アクター	競技者
事前条件	ロボットがラインのエッジ上に倒立状態で置かれている ロボットのキャリブレーションが終了している
事後条件	ロボットは倒立状態にある ロボットはガレージにインしている
基本系列	1. アクタは、システムに対して走行開始の指示をする（※1） 2. システムは、エリアを走行する 3. システムは、エリアの終了を認識する （ガレージインするまで2~3を繰り返す） 4. システムは、ガレージの中で停止する
代替系列	2a. エリアに複数経路が存在する場合 2a1. システムは、進路を選択する 2a2. システムは、選択した進路を走行する 2a3. 基本系列3に戻る
備考	（※1）タッチセンサーを押下する

※主要なユースケースのみ記述します

③走破戦略の検討

要求図を使って、「コースを走破する」ユースケースから「各エリアを走破するための基本的な機能要件」、「考慮すべき設計制約」を整理し、取りうる最適な技術要素を抽出しました。

「コースを走破する」ためには、エリアの特徴を正しく認識し、その特徴にあった走行（駆動）をすることが重要です。そこで、**認識系**と**駆動系**の2つの視点からエリアを走破するための基本機能を抽出しました。

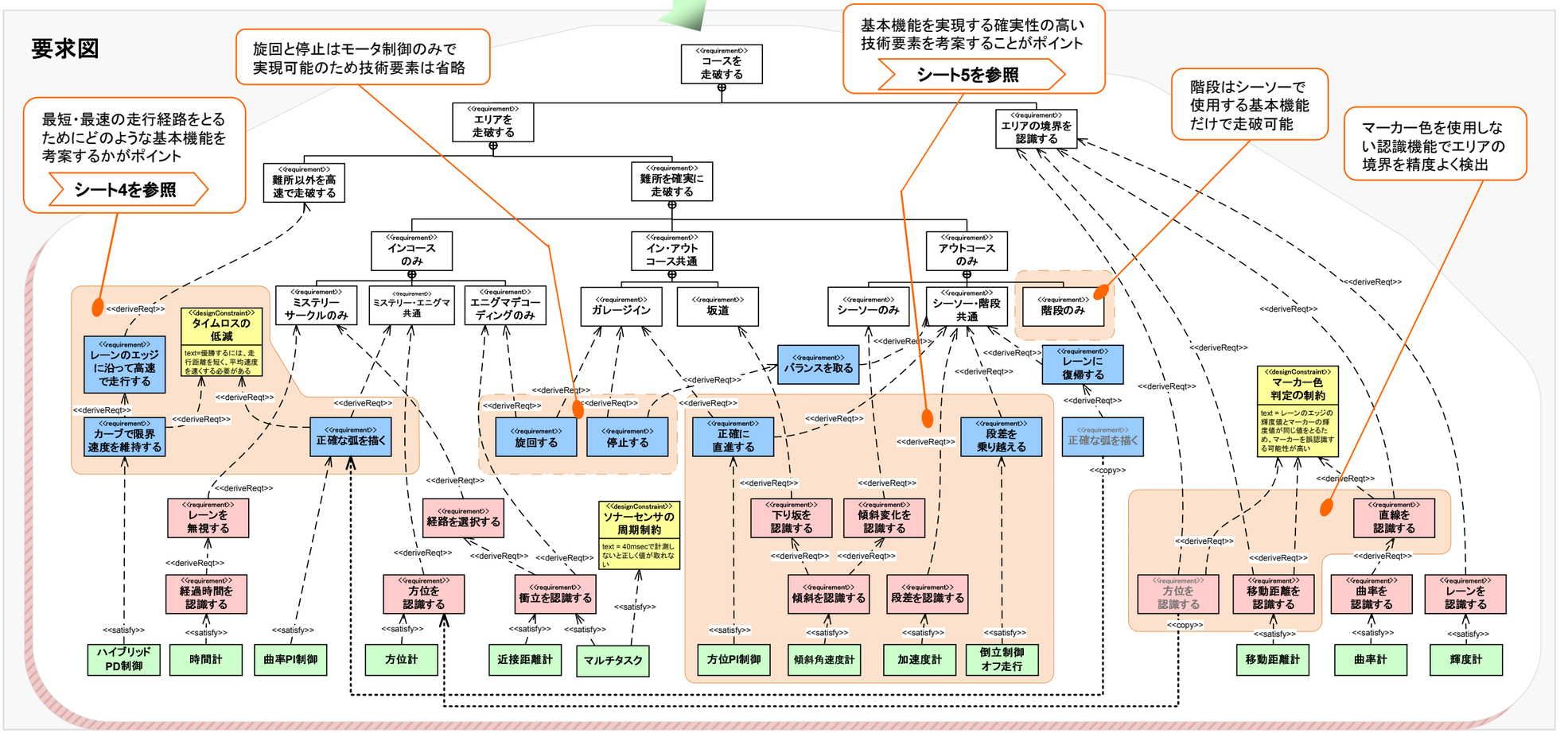
基本機能（駆動系）
エリアの特徴に合った最適な走りを実現する機能です。

基本機能（認識系）
エリアの特徴を正しく認識する機能です。

これら基本機能の組み合わせで全エリアが走破できることを検討しています。

基本機能の実現を支える**技術要素**

要求図

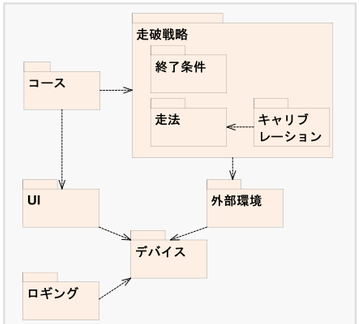


特定のエリア攻略に特化しない拡張性の高い基本構造を構築する

要求分析で洗い出したユースケースを実現するための構造を検討しました。
エリアごとに走破戦略の比較検討がし易くなるように、**変更や追加が想定される個所の切り分け**を行いました。
分岐があるエリアを走破するために**動的に経路を選択する仕組み**を基本構造に取り入れました。

①パッケージ構成

コースを走破するために必要となる様々な概念をパッケージで分類しました。ドメインモデルの走破戦略クラスは構造が複雑になるため、終了条件や走法などのサブパッケージで階層化しました。

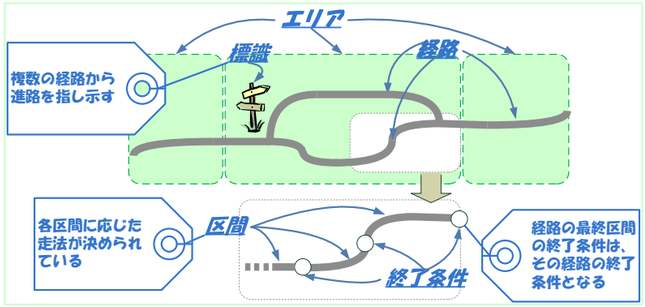


パッケージ	概要
コース	コースの特徴に応じて分割した概念
走破戦略	エリアを走破するための骨格
終了条件	認識系の基本機能を実現
走法	駆動系の基本機能を実現
キャリブレーション	キャリブレーションを実現
UI	利用者とのやりとりを抽象化
外部環境	外部環境を抽象化
デバイス	実行環境が提供する入出力用のデバイス
ロギング	走行ログの取得

②基本構造

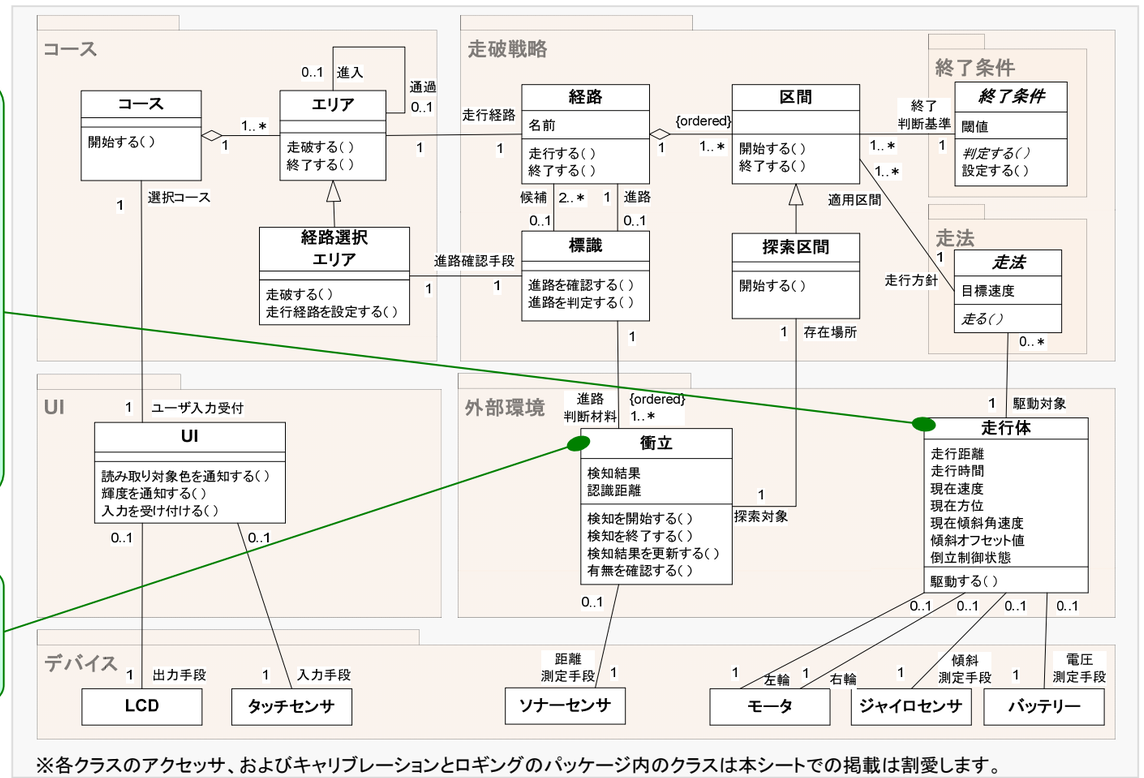
将来的な拡張性を高めるため、特定のエリア攻略に特化しない基本構造を検討しました。
様々なエリアに対して走破戦略を細かく設定できるように、右図のような構成要素に分解しました。「区間」毎に「走法」と「終了条件」があり、それらを切り替えることで**区間単位で比較検討が可能**です。また、分岐があるエリアを走破できるように「標識」を用意し、**候補となる経路の保持と進路を決定する責務**を割り当てました。

基本構造の概念図



実現する技術要素
- 移動距離計
- 傾斜角速度計
- 時間計
- 方位計
- 加速度計

実現する技術要素
- 近接距離計

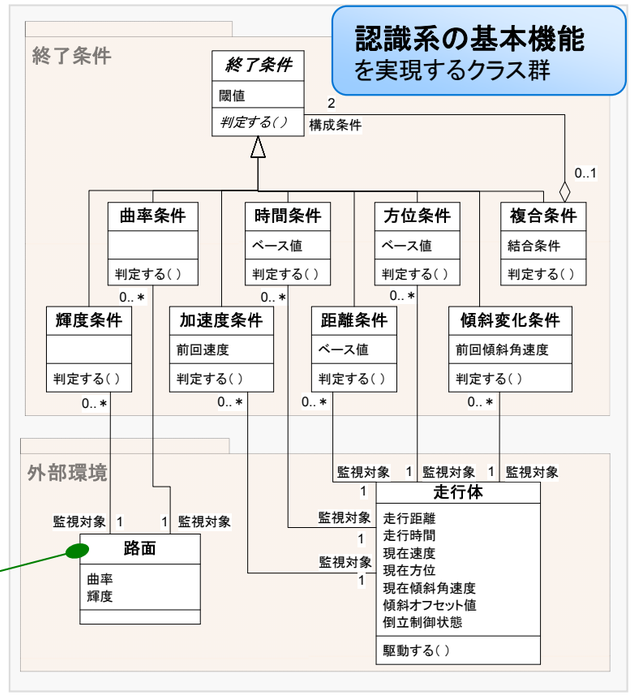


※各クラスのアクセッサ、およびキャリブレーションとロギングのパッケージ内のクラスは本シートでの掲載は割愛します。

③終了条件詳細

区間の終了は「走行体（ロボット自身）」や「路面（ロボットが通過する場所）」が持つ走行開始時を基準とする絶対的な値や現在値で判定できます。また、区間開始時の値（ベース値）からの相対的な変化量で判定することもできます。
区間の終了条件の変更や追加は、サブクラスを定義することで**基本構造を変えることなく**対応可能です。

実現する技術要素
- 輝度計
- 曲率計

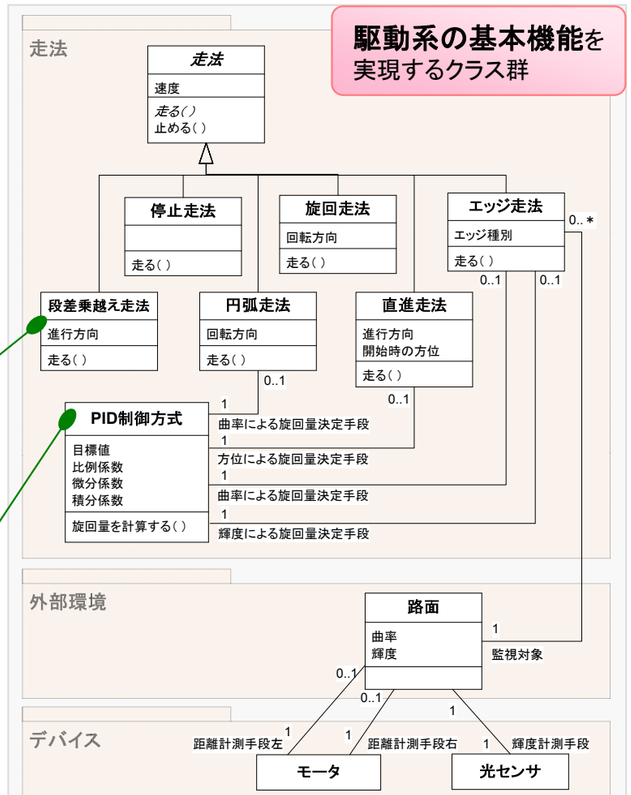


④走法詳細

区間ごとの走り方は直進や旋回といった基本的な動作で実現します。
区間の走法の変更や追加は、サブクラスを定義することで**基本構造を変えることなく**対応可能です。

実現する技術要素
- 倒立制御
- オフ走行

実現する技術要素
- 方位PI制御
- 曲率PI制御
- ハイブリッドPD制御



振る舞い分析

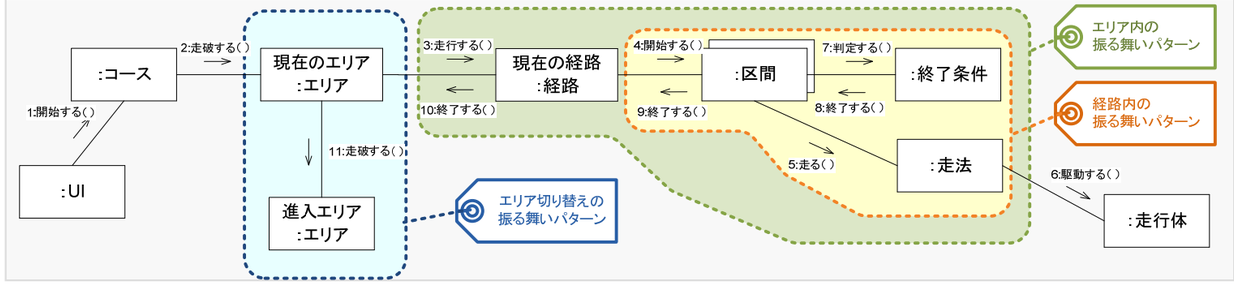
オージス総研 TAMACHI RACING 田町レーシング

基本的な振る舞いパターンで、すべてのエリアを走破できることを検証する

構造分析で洗い出した基本構造で「コースを走破する」機能の実現可能性を検証し、**基本的な振る舞いパターンとして整理**しました。
基本的な振る舞いパターンにより**衝立の探索**や**動的な経路選択が実現**できることを検証しました。
分析終了後の設計の方針として、エニグマデコーディングの際にマルチタスクで処理を行うことの妥当性についても検証しました。

① 基本的な振る舞い

右図は、走行体がすべてのエリアを走破するための基本的な振る舞いです。エリアの切り替えとエリア内の振る舞いは、すべてのエリアに適用される基本パターンになります。経路内に複数の区間がある場合、**経路内の振る舞いパターンを繰り返して**走破します。

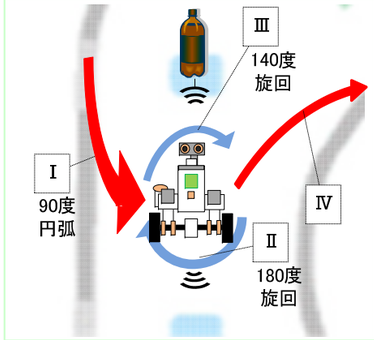


「コースを走破する」ユースケースの基本系列

基本系列	1. アクタは、システムに対して走行開始の指示をする 2. システムは、エリアを走行する 3. システムは、エリアの終了を認識する （ゴール地点に到達するまで2~3を繰り返す） 4. システムは、ガレージの中で停止する

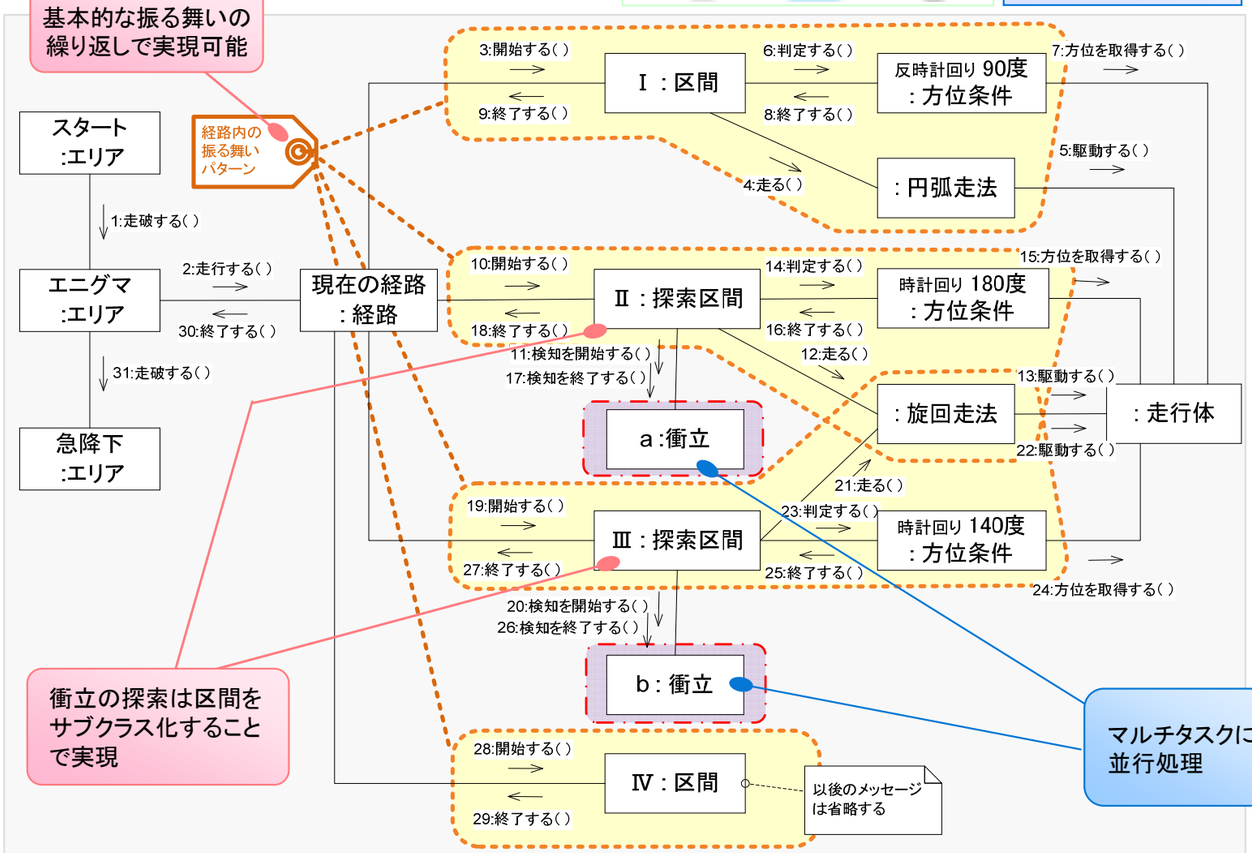
② エニグマエリアの振る舞い

エニグマエリアのように経路内に複数の区間が存在する場合、走行体は基本的な振る舞いを繰り返すことで走破します。また衝立の探索が必要な場合にも、探索する区間を特化することで**基本的な振る舞いを変えることなく**エリアを走破できます。



走行体はそれぞれの**衝立**の有無を判定して結果を記録します。
衝立のデコーディング結果はミステリーサークルの通過ルートを決定する材料になります。

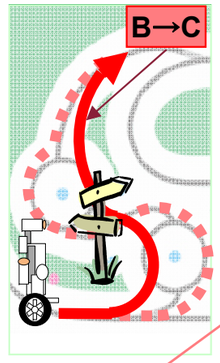
基本的な振る舞いの繰り返しで実現可能



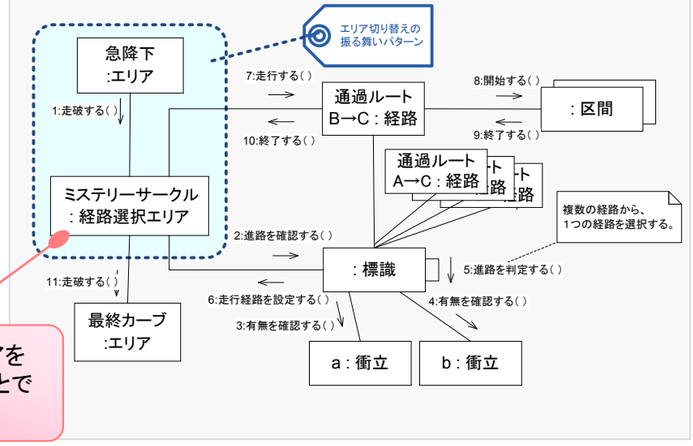
③ ミステリー エリアの振る舞い

動的な経路選択が必要となるミステリーエリアでは、進路を示す標識を見るようにエリアを特化することで、**基本的な振る舞いを変えることなく**走破できます。

ミステリーエリアのように複数の経路が存在する場合、走行体は**標識**を見て進路を決め走行を開始します。
右図はエニグマのデコーディング結果より、通過ルート「B-C」に進路が決定した場合の振る舞いを示しています。

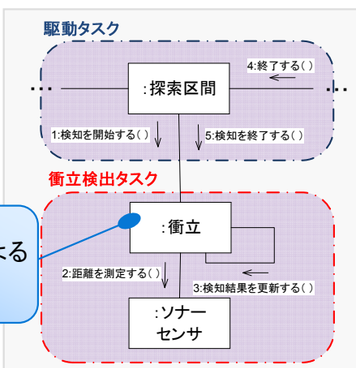


標識の確認はエリアをサブクラス化することで実現



＜タスクを分割した理由＞

- 走行体の倒立制御に必要な周期(4msec)と、ソナーセンサが衝立を正確に読みとる制御に必要な周期(40msec)が異なる。
- 両制御間に関係性はない。そこで、制御ごとにタスクを分割し周期を独立させることで、一方の周期の変更が他方に影響を及ぼさないようにする。



＜タスクの優先度＞

- 衝立の検出モレを防ぐため、タスクの優先度は下記とする。

衝立検出タスク　＞　駆動タスク

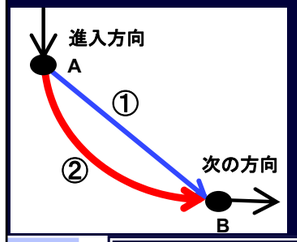
- 衝立検出タスクの処理時間は極めて短いため、駆動タスクによる倒立制御に影響を及ぼすことはない。

＜共有資源の排他制御＞

- タスク間に共有資源はないため、排他制御について今回は検討する必要はないと判断した。

最速を目指す走破戦略・技術要素

走行タイム＝移動距離／走行速度

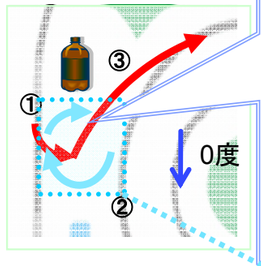


2地点間を移動する際のタイムは、移動距離が短いほど、平均移動速度が速いほど短くすることができます。しかしながら、インコースのエニグマエリアとミステリーエリアは、直進を駆使して走破しようとすると頻繁に方向転換を繰り返さなければならなくなるため移動距離を短くするメリットよりも方向転換によるタイムロスのデメリットの方が大きくなります。そのため、できる限りタイムロスが発生しない走行を実現するため、滑らかに曲線を描くコース取りを採用し、その動きを正確に制御するための技術要素を検討しました。

走破戦略

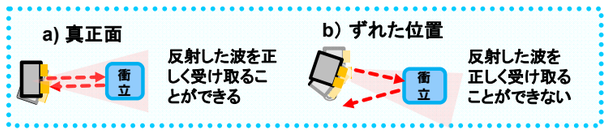
エニグマエリア

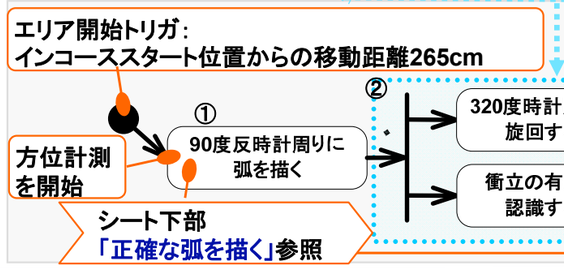
エニグマ・エリアでは1本目と2本目の衝立の間から両衝立の有無を判断し、その後ショートカットしてコースに復帰します。その際、解読地点への進入と退出は、**滑らかに弧を描く走行**でタイムロスをできるだけ減らします。



衝立を真正面から読む

近接距離計には、衝立の正面から判別しないと正確に検知できない制約があります。旋回しながら2本の衝立を正しく検知するためには、走行体を**2本の衝立を結ぶ直線上に停止させる**必要があります。

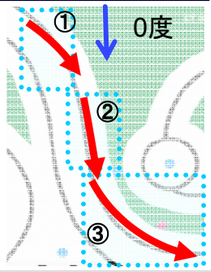


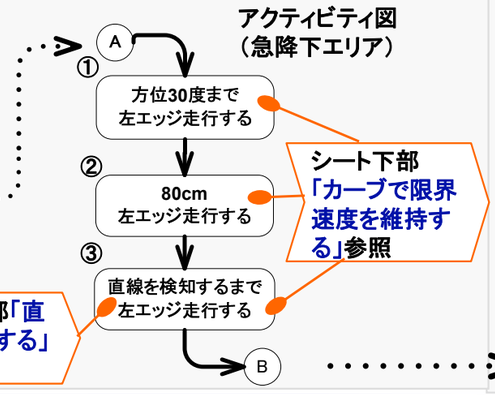


アクティビティ図（エニグマエリア）

急降下エリア

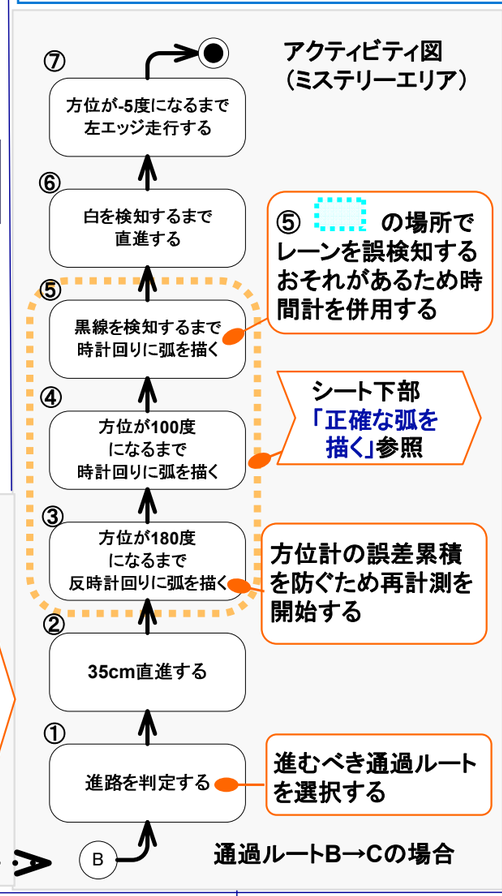
レーンに復帰した後はミステリーサークルの開始地点までライントレースで移動します。その際はカーブも高速走行し、タイムロスを減らします。

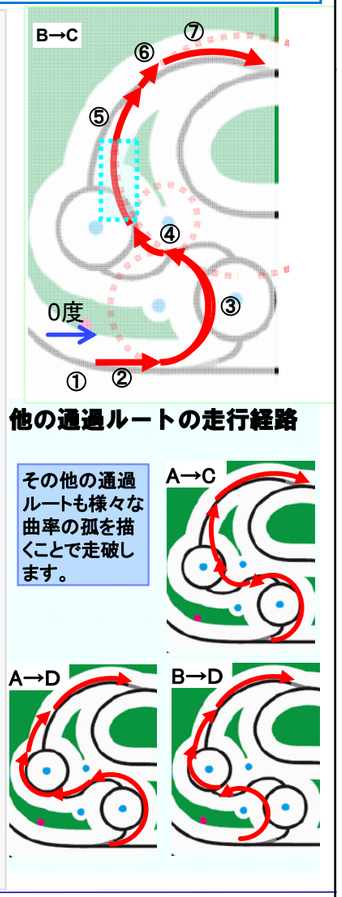




ミステリーエリア

ミステリー・エリアでは、全通過ルートを、レーンを無視して滑らかな曲線を描きながら、停止・減速することなく走行し、**今大会最速突破**を目指します。





他の通過ルートの走行経路

その他の通過ルートも様々な曲率の孤を描くことで走破します。

技術要素

曲率を認識する

曲率計

$$曲率\ {}^{1}\!/_{R} = \begin{cases} \dfrac{(e_r - e_l)}{e_l} & (e_r > e_l) \quad 右カーブ \\ -1 \times \dfrac{(e_l - e_r)}{e_r} & (e_l > e_r) \quad 左カーブ \end{cases}$$

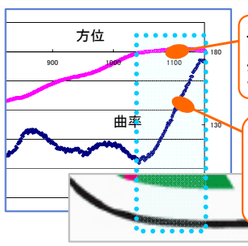
e_l ：左モータ回転角度変化量
e_r ：右モータ回転角度変化量

下記3つの基本機能を実現するためには曲率が重要です。左記の計算式により曲率計を実装しています。

<<requirement>> 直線を認識する
<<requirement>> 正確な弧を描く
<<requirement>> カーブで限界速度を維持する

直線を認識する

ミステリーサークルの開始地点を判定するためにレーンの曲率を調べて直線を検知します。レーンの曲率はコースの形状に基づく情報であるため、灰色マーカーと比べて非常に安定した検知が行えます。



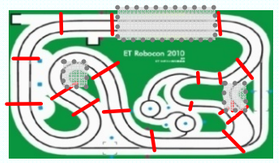
直線に入ると方位は変化しなくなる

曲率値が0に向かって急激に変化することを閾値で判定する

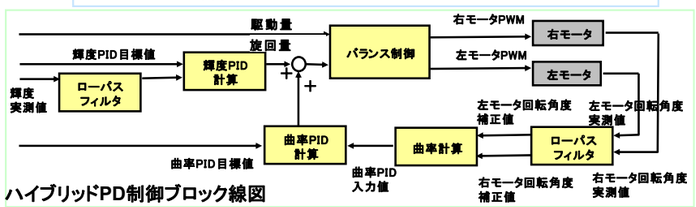
カーブで限界速度を維持する

カーブで限界速度を維持するために輝度PD制御と曲率PD制御を合わせた**ハイブリッドPD制御**を用いました。曲率PD制御は、**「走行軌跡の曲率を、カーブの曲率に近づけるように旋回量を調節する」**働きがあります。この制御によってカーブでのレーンの追従性が高まりました。その結果従来より速い速度でカーブで走行させることができ、ライントレースでの走行タイムを大幅に短縮することができました。

目標値設定
コースをレーンの曲率がほぼ同じ部分に区切り、目標曲率を設定した。



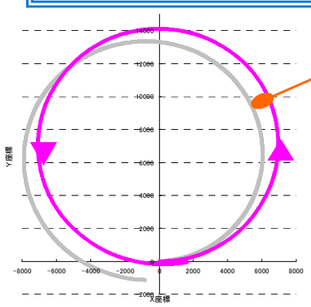
右上図記の ▒ を除いて**約48cm/秒**を維持して走行可能



ハイブリッドPD制御ブロック線図

正確な弧を描く

弧を描く走行の精度を上げるために、曲率PI制御を用いました。**「一定曲率半径の円（弧）を描くための操作量」**を走行体に指示することで、レーン外での正確な動きを実現しています。特に、**積分制御**を導入することで比例制御だけに比べて、大幅に精度を上げることができました。



曲率PI制御を行ったときの走行軌跡

曲率半径70cm の円を反時計周りに走行させた場合

比例（P）制御のみ

比例（P）制御＋積分（I）制御

約40cm/秒を維持して走行可能

確実さを追求する走破戦略・技術要素

シーソー・階段・ガレージインは確実に走破する戦略と技術要素が重要

シーソーと階段は段差を乗り越えるため、他に比べてコースアウトや転倒のリスクが高くなります。また、ガレージインの走破にはcm単位の精密な制御が必要です。
これらの難所に共通して要求されるのは速さよりも確実さ。そこで、各難所で失敗するリスクを洗い出し、リスクを低減するための戦略と技術要素を検討しました。

走破戦略

シーソー＆階段

大会規定の倒立制御APIは平面上の走行で姿勢を維持するように設計されており、立体的な難所の段差を乗り越えるには対策が必要です。

私たちは、開発の中で発生頻度の多かった以下の3つの失敗例（リスク）から、その原因を洗い出し、対策を検討しました。

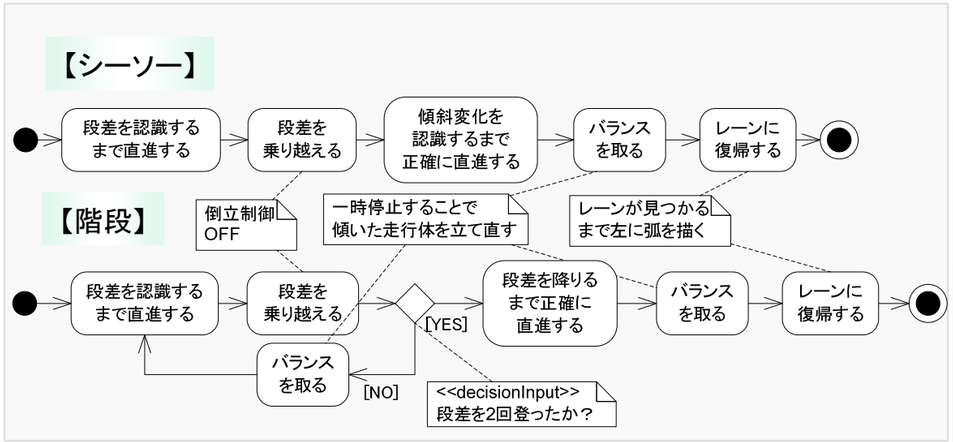
1. **段差を乗り越える時にコースアウトする**
2. **段差乗り越える時に前のめりに倒れる**
3. **シーソーを降りるときに前のめりに倒れる**

結果として難所走破の成功率向上を達成しました。

リスク対策後のアクティビティ図は右のとおりです。

段差にアプローチする方法、段差を乗り越える方法、乗り越えた後にバランスを取ったり、レーンに復帰したりする方法などは共通しています。

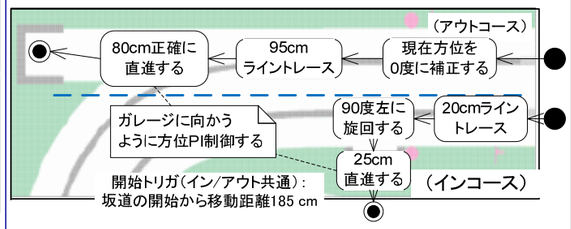
特に、階段エリアは段差が連続するため、次の段を乗り越え始める前に走行体のバランスを取ることが重要です。



ガレージイン

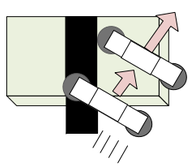
ゴール後の難所のため、10秒以内に走破できればよく、タイムを競う必要はありません。しかし、レーンから離れた位置にあるガレージへアプローチするため、以下のリスクに対処する必要があります。

4. 坂道～ガレージ間で進行方位がずれる



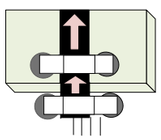
技術要素

【リスク1】段差を乗り越える時にコースアウトする

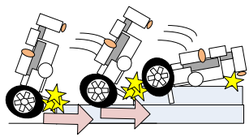


失敗例
ラインレースしながら段差へ斜めから進入してしまうと片輪だけ段差に乗り上げてしまい、片輪が引っかかったまま右方向にコースアウトする。

対策
方位PI制御で段差に垂直に進入するように方位を補正する。また、段差の直前から乗り越えるまでラインレースをOFFする（⇒「正確に直進する」参照）。

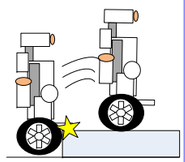


【リスク2】段差を乗り越える時に前のめりに倒れる

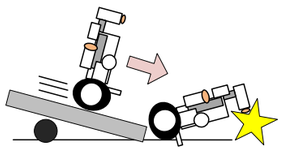


失敗例
段差に衝突した衝撃で前傾になると、倒立制御で重心を前に移す力が働いて急加速する。階段エリアでは特に1段目後に体勢を立て直せないと、車輪が段差に引っかかり、前のめりに倒れる。

対策
段差の衝突を加速度計で即座に認識する（⇒「段差を認識する」参照）。また、段差を乗り越えるまで倒立制御をOFFし、意図しない急加速を防ぐ（⇒「段差を乗り越える」参照）。

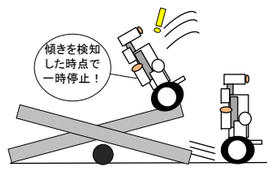


【リスク3】シーソーを下る時に前のめりに倒れる

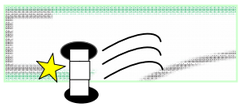


失敗例
シーソーが急激に傾くと、タイヤへの摩擦が弱くなり、重力の影響力を受けて走行体が滑り、一度後ろに傾く。その後、重心を前に戻す力、シーソーを下る勢い、重力により、そのまま前のめりに倒れる。

対策
シーソーの傾き変化を傾斜角速度計で検知した時点で一時停止し、車体を立て直す（⇒「傾斜変化を認識する」参照）。急加速で前のめりに倒れるのを防ぐ。

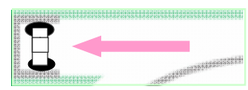


【リスク4】坂道～ガレージ間で進行方位がずれる



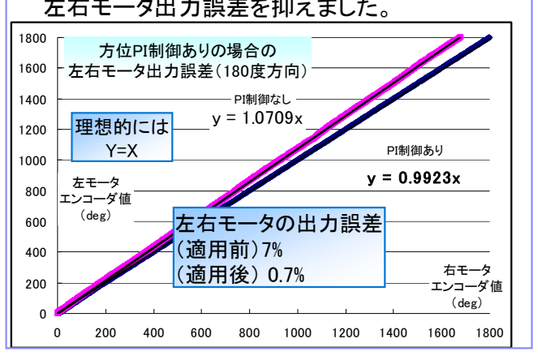
失敗例
左右モータに対して同じ制御量を与えてもモータ個体差などの要因により直進せず、途中で進行方位がずれてしまう。

対策
ガレージにアプローチする際は方位PI制御で進行方位を補正する。また、現在方位の誤差蓄積防止のため、坂道直後の直線で現在方位を0度に補正する（⇒「正確に直進する」参照）。



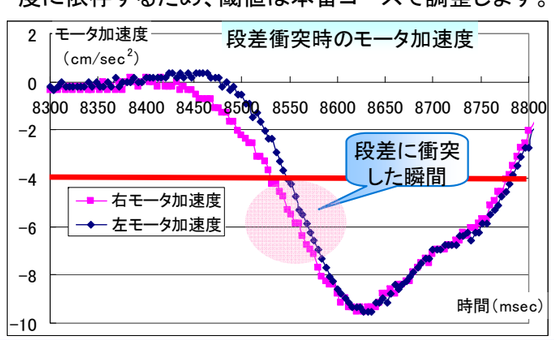
正確に直進する

正確に直進するため、方位PI制御で方位計値を目標値に近づけるよう旋回速度を算出し、左右モータ出力誤差を抑えました。



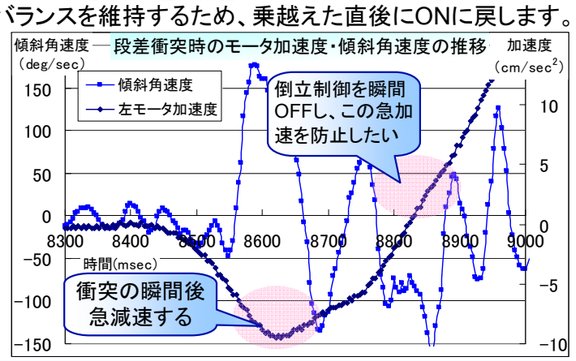
段差を認識する

タイヤが段差に衝突し、加速度が急減少する瞬間を加速度計で認識します。加速度の変化量はモータ速度に依存するため、閾値は本番コースで調整します。



段差を乗り越える

段差を乗り越える際に倒立制御を瞬間的にOFFし、倒立制御による急加速（減速制御不能）を防止します。ただし、バランスを維持するため、乗越えた直後にONに戻します。



傾斜変化を認識する

シーソーが傾いた後、走行体が後ろに傾いた瞬間を傾斜角速度計で認識します。