

チーム紹介、目標、意気込み

KatLabは、ソフトウェア工学の研究室に所属する学生のチームです。ETロボコンには毎年、研究室の有志メンバーで参加しています。今年は学部生3人、修士課程の学生6人で参加しています。主にモデル作成を行う設計チーム、実装や要素技術の検証を行う実装チーム、開発の効率化を目的とした生産性向上チームの3チームで開発を行っています。

今年の目標は、**CS大会の総合で3位以内に入賞**することです。KatLabは、2018年から7年連続CS大会に出場しており、年々成績は良くなっているものの目標とする総合入賞を果たせたことはありません。今年こそは、競技とモデルの両方で良い結果を出し、総合での入賞を目指します！！

モデルの概要

※主にロボコンスナップNEOについて記載

信頼性と保守性を重視したシステム

- 走行体の競技動作を実現するためにCommandパターンを採用することで、**システムの拡張性が上がり、効率的な開発ができる**ようにした。
- 今年の競技固有の機能と、汎用的な機能を別のパッケージに分離し、レイヤー構造を採用することで、**来年以降も再利用しやすい**構造にした。
- 走行時に算出したPWM値や取得したRGB値などを可視化するツールを開発することで、**調整が必要な箇所の特定を容易**にした。
- コースアウトを検知し、線上に復帰するライントレース復帰動作を実装することで、**走行精度が25%向上**した。
- 配置エリアAにおける撮影に機械学習および走行体の向き補正を用いることで、**ベストショット成功率が20%向上**した。

モデルの構成

1. 要求分析

目標リザルトポイント **61pt以上** を達成するために、主に以下の要求を導出した。

- LAPゲートまでの走行で17pt以上獲得する。
- ロボコンスナップNEOで22pt獲得する。
- ダブルループNEOで12pt獲得する。
- スマートキャリーで5pt獲得する。
- ゴールまで走行し、5pt獲得する。

2. 分析モデル

- リアルタイムな制御が必要な走行体の動作に関する処理はC++で実装し、画像に関する処理は優れたライブラリが豊富なPythonで実装する。
- 走行体内で画像処理を完結することによって、無線通信デバイスとの通信にかかる時間の削減を図る。

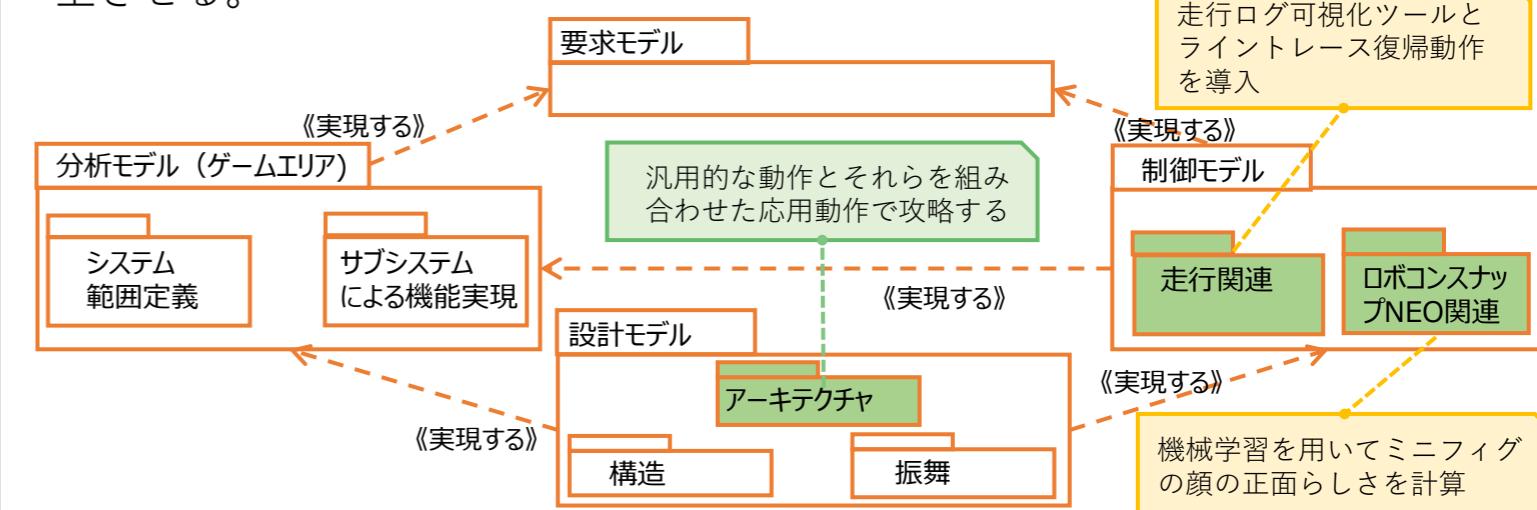
3. 設計モデル

- 今年の競技固有の機能と、汎用的な機能を別のパッケージに分離し、複雑なパッケージでは、Facadeパターンを採用することによって拡張性および保守性を向上させる。
- Commandパターンを採用し、拡張性を向上させる。

4. 制御モデル

以下の事項を実現するために、制御戦略を立てた。

- 走行に関する制御戦略により、走行精度を向上させる。
- ロボコンスナップNEOに関する制御戦略により、ベストショット成功率を向上させる。



1.1 開発の目標

開発目標：CS大会で総合3位以内に入賞する

今年は、昨年以上の成績を残すために、CS大会で総合3位以内に入賞することを目標とする。地区大会の上位3チームのリザルトポイントを参考に、各目標ポイントを検討する。

目標リザルトポイント：61pt

地区大会の上位3チームの平均リザルトポイントは59.1ptであった。地区大会では、KatLabの目標リザルトポイントを、昨年CS大会上位3チームの平均リザルトポイントから、56ptに設定した。CS大会では、さらにスマートキャリーで5ptを獲得した61ptの獲得を目標に設定する。

目標走行ポイント：15pt

地区大会では、LAPゲートを確実に通過することを優先した。KatLabの走行タイムは14.6sであった。CS大会でも地区大会と同じ速度で走行し、LAPゲートを確実に通過することを目標とし、15pt獲得を目標に設定する。

目標ボーナスポイント：46pt

目標リザルトポイントから目標走行ポイントを引くと、46pt獲得する必要がある。中間ゲート通過とLAPゲート通過で1ptずつ、ゴールで5pt獲得を前提として、ロボコンスナップNEOで22pt、ダブルループNEOで12pt、スマートキャリーで5pt獲得することを目標に設定する。デブリリムーバルは攻略しない。

表1.1-1. 目標ポイント

ポイント獲得対象	目標ポイント
中間ゲート通過	1pt
LAPゲート通過	1pt
走行ポイント	15pt
ダブルループNEOチェックポイント通過	3pt × 4箇所
ロボコンスナップNEO	
橋円	7pt
背景	5pt
プラレール	8pt
パーフェクトショット	2pt
デブリリムーバル	0pt
スマートキャリー	5pt
ゴール	5pt
合計	61pt

1.3 要求分析

1.2節のユースケース分析を基に導出した、開発目標を満たすための要求を、図1.3-1に示す。さらに、紙面の都合から、ロボコンスナップNEOの攻略に関する要求を中心に記載する。

非機能要求は、ISO/IEC25010が定義する製品品質モデルの品質特性に基づいて分類した。品質特性を、表1.3-1に示す。なお、セキュリティと使用性、互換性の品質特性については、今回の競技では使用しないため省略する。

ユースケースとは別に、開発時間に限りがあるという制約から、開発効率に関する下位要求を導出した。開発効率に関する要求図を図1.3-2に、ダブルループNEOに関する要求図を図1.3-3に、ロボコンスナップNEOに関する要求図を図1.3-4に、それぞれ示す。

また、本モデル内の関連を表す凡例を図1.3-5に、要求図の凡例を図1.3-6に、それぞれ示す。

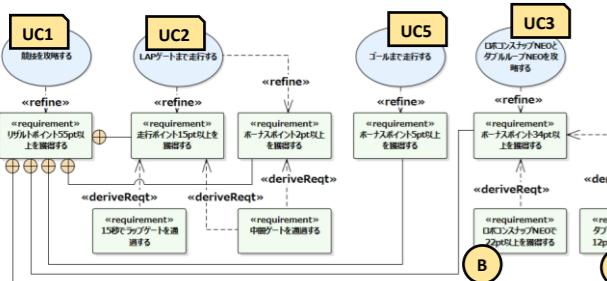


図1.3-1. ユースケースを基に導出した要求

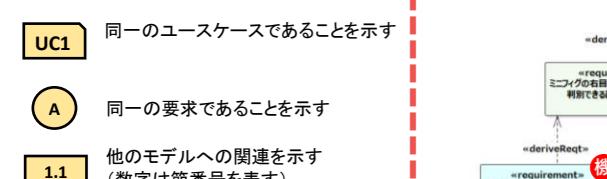


図1.3-5. モデル内の関連を表す凡例

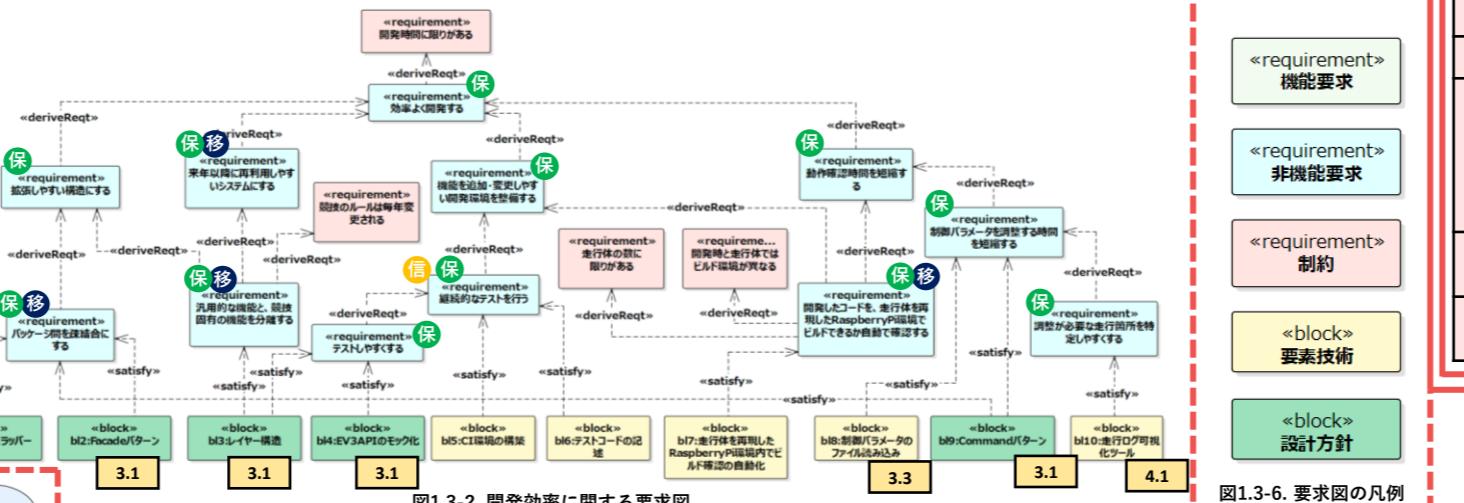


図1.3-2. 開発効率に関する要求図

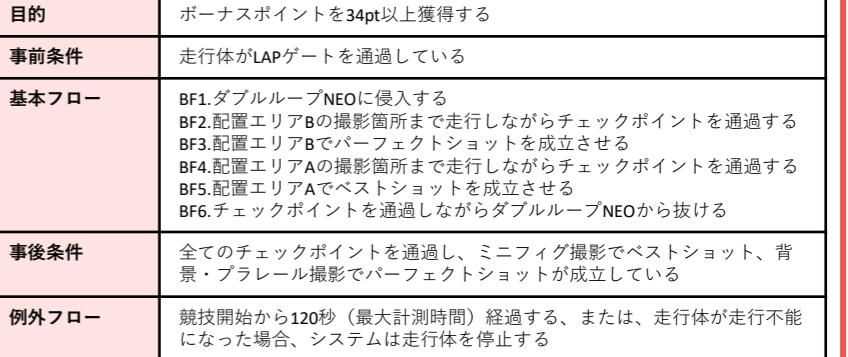


表1.3-1. 品質特性

品質特性	概要	アイコン
機能適合性	目的に対する機能の適切さ	機
性能効率性	システムの処理能力やリソース活用の適切さ	性
信頼性	正しく機能できる度合、異常発生時の回復しやすさ	信
保守性	保守・修正のしやすさ	保
移植性	異なる環境にシステムを移す時の効率化	移

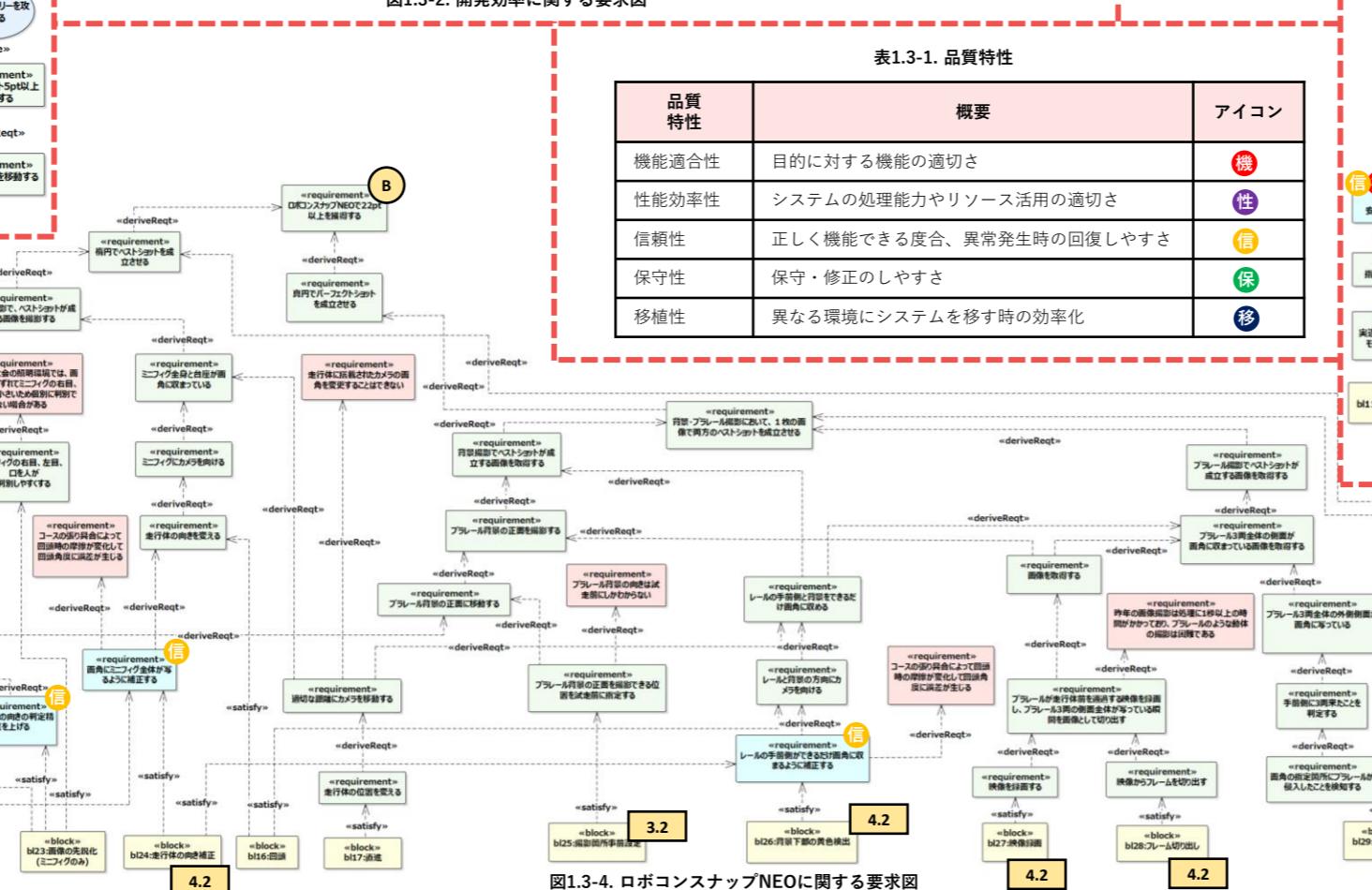


図1.3-4. ロボコンスナップNEOに関する要求図

1.2 ユースケース分析

開発目標を実現するためのユースケースを、図1.2-1に示す。「LAPゲートまで走行する」「ゴールまで走行する」および「スマートキャリーを攻略する」については、紙面の都合から省略する。ロボコンスナップNEOとダブルループNEO攻略後、デブリリムーバルでデンジャーボトルを避け、スマートキャリーでキャリーボトル移動を成立させてゴールまで走行することで開発目標を実現する。

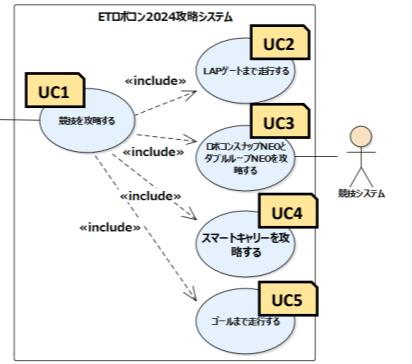


図1.2-1. 目標を達成するために必要な機能

表1.2-1. 「競技を攻略する」の詳細

ユースケース名	競技を攻略する
アクター	スタート
目的	リザルトポイントを61pt以上獲得する
事前条件	システムの初期設定が終了している
トリガー	スタート者が走行体のスタート操作を行う
基本フロー	BF1.走行体はLAPゲートまで走行する BF2.システムはロボコンスナップNEO、ダブルループNEO、およびスマートキャリーを攻略する BF3.走行体はゴールまで走行する BF4.システムは走行体を停止する
事後条件	リザルトポイントを61pt以上獲得した状態で走行体が停止している
例外フロー	競技開始から120秒（最大計測時間）経過する、または、走行体が走行不能になった場合、システムは走行体を停止する

表1.2-2. 「ロボコンスナップNEOとダブルループNEOを攻略する」の詳細

ユースケース名	ロボコンスナップNEOとダブルループNEOを攻略する
アクター	競技システム
目的	ボーナスポイントを34pt以上獲得する
事前条件	走行体がLAPゲートを通過している
基本フロー	BF1.ダブルループNEOに侵入する BF2.配置エリアBの撮影箇所まで走行しながらチェックポイントを通過する BF3.配置エリアBでバーフェクトショットを成立させる BF4.配置エリアAの撮影箇所まで走行しながらチェックポイントを通過する BF5.配置エリアAでベストショットを成立させる BF6.チェックポイントを通過しながらダブルループNEOから抜ける
事後条件	全てのチェックポイントを通過し、ミニフィグ撮影でバーフェクトショット、背景・プラレール撮影でバーフェクトショットが成立している
例外フロー	競技開始から120秒（最大計測時間）経過する、または、走行体が走行不能になった場合、システムは走行体を停止する

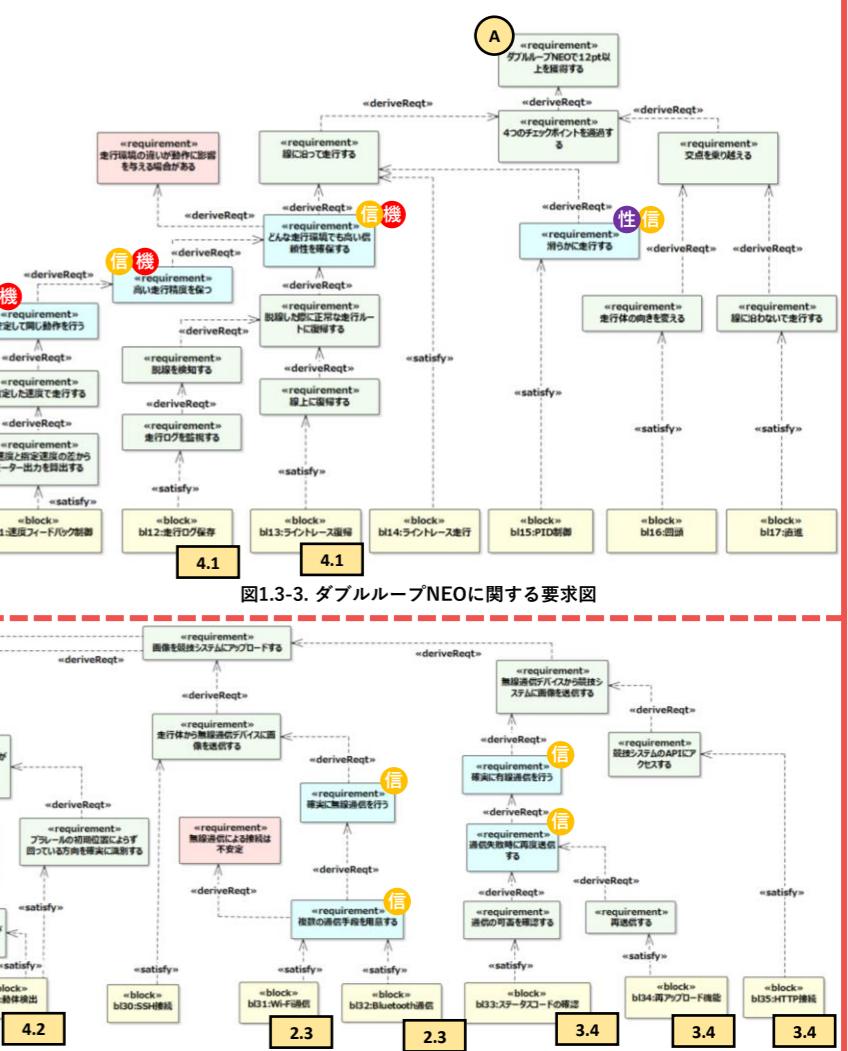
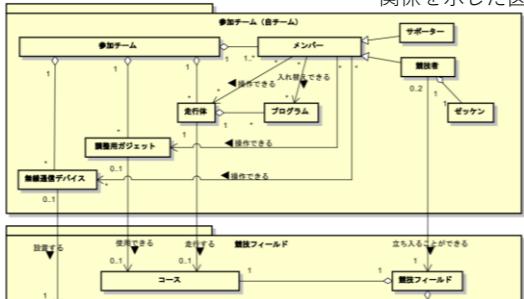


図1.3-3. ダブルループNEOに関する要求図

2.1 静的構造の概要

システム分析による静的構造の概要

競技規約を踏まえ、ETロボコンを構成する要素を抽出した。ETロボコン2024競技規約より、競技チームと競技フィールドの関係を示した図を、図2.1-1に示す。また、2.3節に後述するシステム範囲定義を行い、各サブシステムをシステムに配置した。システム範囲定義による静的構造の概要を表すブロック定義図を、図2.1-2に示す。



競技規約より
導出

図2.1-2. システム分析による静的構造の概要

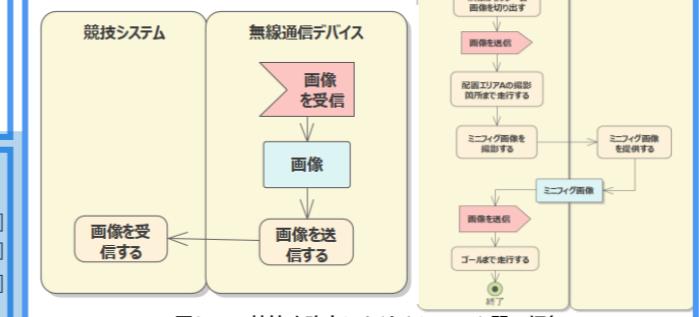
2.4 競技全体の攻略フロー

スタート合図後の各要素間の振舞

競技攻略中における各システム間の振舞を概要レベルで表すアクティビティ図を、図2.4-1に示す。

プラレール・背景撮影箇所の設定

競技攻略開始前に、2.3節で定義した撮影箇所指定システムを用いて、スタート者が配置エリアBでのプラレール・背景撮影箇所を指定する。



2.5 サブシステムによる機能実現

ロボコンスナップNEOにおけるシステム間の振舞

2.3節にて定義したサブシステムによるロボコンスナップNEOの攻略における振舞を表すアクティビティ図を、図2.5-1に示す。

ロボコンスナップNEO攻略の役割分担

図2.5-1より、ロボコンスナップNEOの攻略において、動作に関するものを走行システムが、画像処理に関するものをフロントカメラシステムが分担して担うことで、ロボコンスナップNEO攻略を実現する。

2.2 コンテキスト分析

システム間の関係

図2.1-2より、コンテキスト分析を行い、それぞれの関係性を分析した。ETロボコン2024における各システム間の関係性を表す内部ブロック図を、図2.2-1に示す。さらに、図2.2-1の走行体ブロック内部のシステム間の関係性を図2.2-2に示す。図2.2-1より、走行体内部におけるソフトウェア開発対象を詳細に絞り込むことができる。

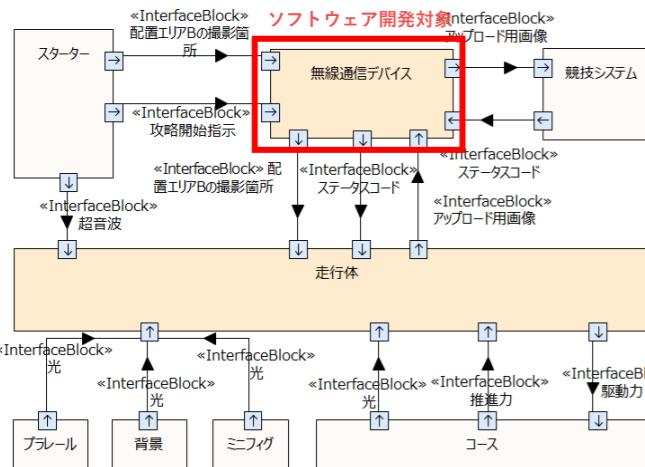


図2.2-1. ETロボコン2024におけるシステム間の関係性

無線通信デバイスで画像判定を行う場合、無線通信デバイスへ全ての画像を送信する必要がある。そのため、Raspberry Piで画像処理等を行うことで、アップロード用画像のみを送信できる。これにより、画像送信失敗のリスクを最小限にする。

サブシステム配置

フロントカメラシステムおよび走行システムは、走行体固有の機能（車輪を回転、カメラ撮影など）を実現する。さらに、これらの機能を組み合わせて今年の競技を攻略するための機能を実現する。

考慮すべき事項②

Raspberry Piの内、EV3APIはソフトウェア開発対象外

2.3 システム範囲定義

サブシステムの定義

1.3節で導出した要求を基に、サブシステムを定義した。各サブシステムの基になる要素技術のブロック番号と、各サブシステムの役割を、表2.3-1に示す。

サブシステムの静的振舞

2.2節のコンテキスト分析および表2.3-1で定義したサブシステムを基に、定義したサブシステムを走行体と無線通信デバイスのそれぞれに配置した。ロボコンスナップNEOの攻略に関わるサブシステムに限定し、走行体、および無線通信デバイスに配置するサブシステム間の関係を表す内部ブロック図を、それぞれ図2.3-1、図2.3-2に示す。

考慮すべき事項③

画像アップロードの為に、無線通信デバイスを経由する必要がある。

画像処理などに優れたライブラリが豊富に存在するPythonで実装。

画像処理などに優れたライブラリが豊富に存在するPythonで実装。

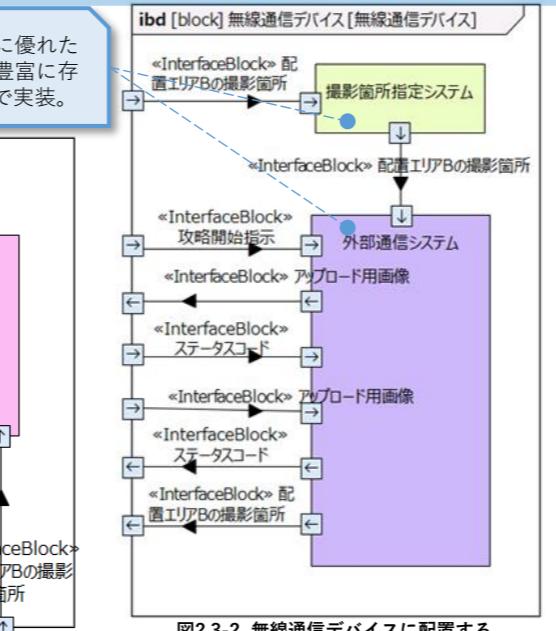


図2.3-1. Raspberry Piに配置するサブシステム間の関係

表2.3-1. サブシステムの定義

基になる要素技術の ブロック番号	サブシステム名	役割
bl11 bl13 bl14 bl15 bl16 bl17	走行システム	コースを走行する
bl18 bl19 bl20 bl21 bl22 bl23 bl24 bl26 bl27 bl28 bl29	フロントカメラシステム	フロントカメラを用いた競技攻略の為の画像処理等を行う
bl30 bl31 bl32 bl33	外部通信システム (走行体)	無線通信デバイスとの通信を行う
bl34 bl35	外部通信システム (無線通信デバイス)	走行体および競技システムとの通信を行う
bl25	撮影箇所指定システム	スタートからの撮影箇所の入力を受け付ける

サブシステムと実行環境の対応

図2.3-1と図2.3-2から、走行システムをC++で、それ以外のシステムをPythonで実装する。そこで、C++でPythonコマンドを呼び出す際には、シェルファイルを用いる。また、Raspberry Piと無線通信デバイス間の通信手段として、BluetoothおよびWi-Fiの2種類を用意し、本番の通信環境に応じて選択する。これらを踏まえ、走行体と無線通信デバイスについての配置図を、図2.3-3に示す。

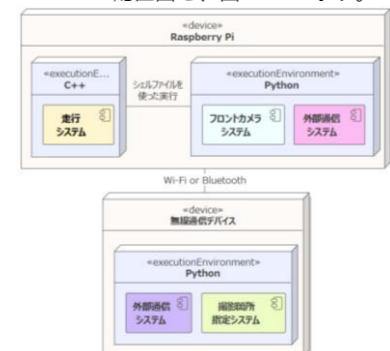
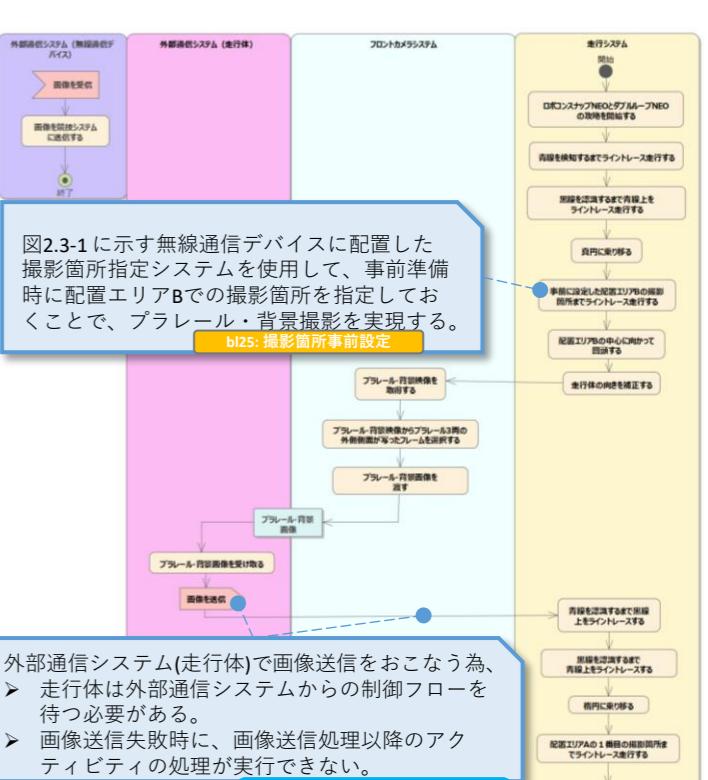


図2.3-2. 無線通信デバイスに配置するサブシステム間の関係



- 外部通信システム(無線通信デバイス)で画像送信をおこなう為、走行体は外部通信システムからの制御フローを待つ必要がある。
- 画像送信失敗時に、画像送信処理以降のアクティビティの処理が実行できない。

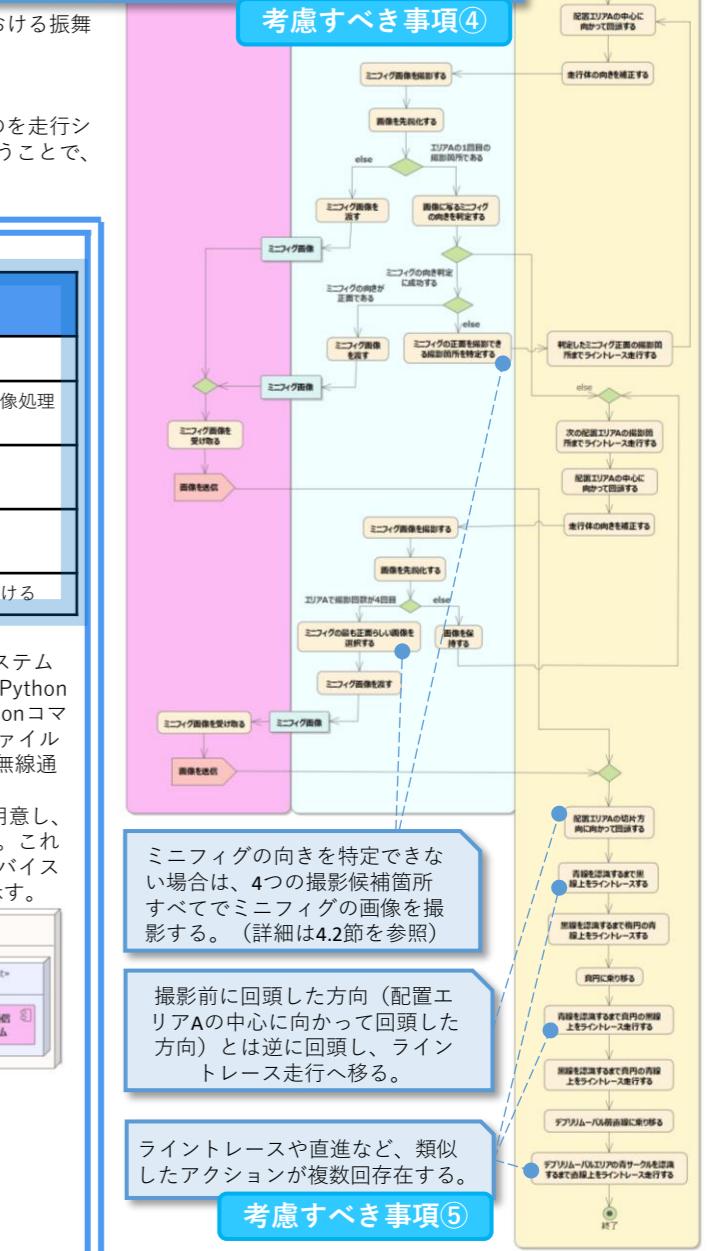


図2.5-1. ロボコンスナップNEO攻略におけるサブシステム間の振舞

3 設計モデル（1）

KatLab

3.1 アーキテクチャ

設計方針②

1章の要求モデルで抽出したソフトウェア設計に関する要求、および、2章のシステム全体の分析において考慮すべき事項を基に、ロボコンスナップNEOとダブルループNEOを攻略する上で検討すべき事項を抽出し、設計方針を決定した。決定した設計方針を、表3.1-1に示す。

表3.1-1に記載した設計方針に基づき、アーキテクチャ設計を行った。開発するシステム全体のパッケージ図を図3.1-1に示す。また、各パッケージの責務を、表3.1-2に示す。

表3.1-1. システムの設計方

根拠となる要求・分析	検討すべき事項	設計方針
bi9: Commandパターン bi8: 制御パラメータのファイル読込 考慮すべき事項① 考慮すべき事項⑤	競技を攻略する上で、共通する処理や動作が複数回存在する。 大会当日の限られた時間で目標リザルトポイントを達成できるよう調整する必要がある。しかし、走行システムにC++を採用したため、変更を加えるたびにビルドする必要があり、調整に時間がかかる。	①競技動作をクラスとして実装し、各競技の攻略は競技動作インスタンスを組み合わせることで実現する。これにより、システムの拡張性を向上させる。また、競技動作を表す「動作マンド」を定義し、C++ソースコードから独立した動作コマンドリストのCSVファイルを実行時に読み込み、競技動作に変換する。これにより、制御パラメータを調整する度にビルドする必要がなくなり、調整時間を短縮できる。
bi3: レイヤー構造 bi2: Facadeパターン bi1: EV3APIラッパー 考慮すべき事項②	各年共通の走行体が持つ汎用的な機能と、今年の競技において実現したい機能の両機能を含むフロントカメラシステムおよび走行システムは、保守性と拡張性を高くしたい。	②各年共通の汎用的な機能を集約したパッケージと、今年の競技特有の機能を実現するパッケージに分割することで、サブシステムの再利用性と拡張性を向上させる。 ③フロントカメラシステムにおいて、競技攻略で必要となるカメラを用いた複数の処理を、シンプルな1つの窓口から利用できるようにすることで、保守性と拡張性を向上させる。 ④運営が提供する外部パッケージであるEV3APIのラッパーを実装することで、影響範囲を最小限に集約する。
bi31: Wi-Fi接続 bi32: Bluetooth接続 bi34: 再アップロード機能 考慮すべき事項③ 考慮すべき事項④	ロボコンスマナップNEO攻略時、画像は走行体で撮影し、無線通信デバイスを経由して競技システムにアップロードする。そのため、会場の通信環境によっては、画像の送信に失敗する可能性がある。また、画像の送信失敗が、その後の競技攻略に影響を与える可能性がある。	⑤Wi-Fi接続による通信とBluetooth接続による通信を用意して本番環境に応じて通信手段を選択する。 ⑥ステータスコードを確認し、画像送信に失敗した際には再送信することで、信頼性を向上させる。 ⑦フロントカメラシステムと外部通信システム（走行体）間の画像送信を非同期処理で実装することで、他競技を攻略する時間を確保し、与える影響を小さくする。
bi13: ライントレース復帰	走行環境の違いから、大会本番では高い信頼性の走行を実現できず、最低限のリザルトポイントも獲得できない可能性がある。	⑧走行体が意図しない脱線を検知した場合、ライントレース復帰によって線上に復帰する。これによりシステムの信頼性を向上させる。詳細は4.1.5節に後述する。紙面の都合上、3章ではライントレース復帰に関する記述を省略する。

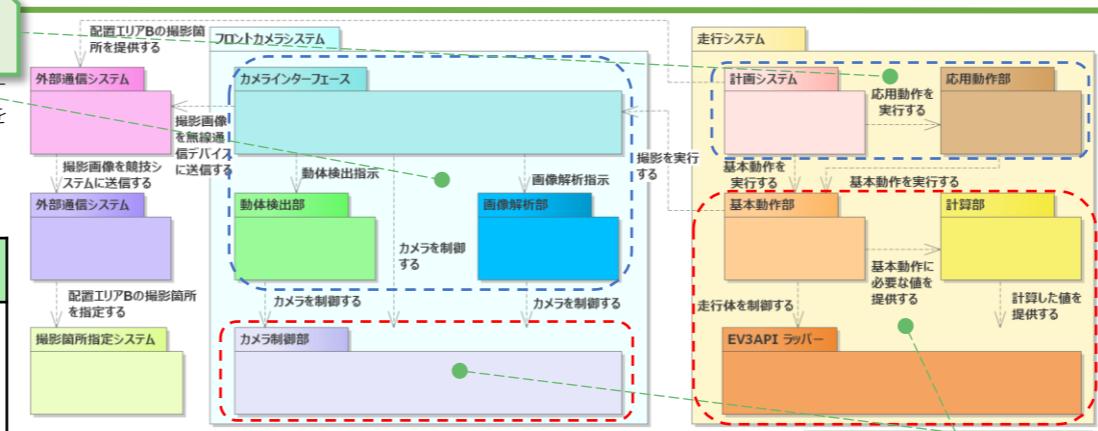


図3.1-1. システムの依存関係を表したパッケージ図

3.2 ユースケースの実現

各年共通の汎用的な機能
を集約したレイヤー

表3.2-1. コマンドと競技動作の対応表

コマンド	基本動作	説明
DL	距離指定ライントレース	指定距離走行するまでライントレースする
CL	色指定ライントレース	指定色を認識するまでライントレースする
DS	距離指定直進	指定距離走行するまで直進する
CS	色指定直進	指定色を認識するまで直進する
PR	回頭	指定角度回転するまで回頭する
EC	エッジ変更	指定したエッジに切り替える
コマンド	応用動作	説明
AC	ミニフィグ撮影	配置エリアAでミニフィグ画像を撮影し、競技システムへアップロードする
BC	プラレール・背景撮影	配置エリアBでプラレール・背景画像を撮影し、競技システムへアップロードする

3.3 構造設計

3.1節のアーキテクチャを踏まえて、ソフトウェア構造を設計した。走行体内のソフトウェア構造を図3.3-1に、無線通信デバイス内のソフトウェア構造を図3.3-2(次ページに記載)に、それぞれ示す。ソフトウェア構造は、以下のルールの下で記述する。

設計方針①
競技動作はコマンドで表現し、コマンドファルにはコマンドと制御パラメーターを記述する。これにより、大会当日の調整時間を短縮する。

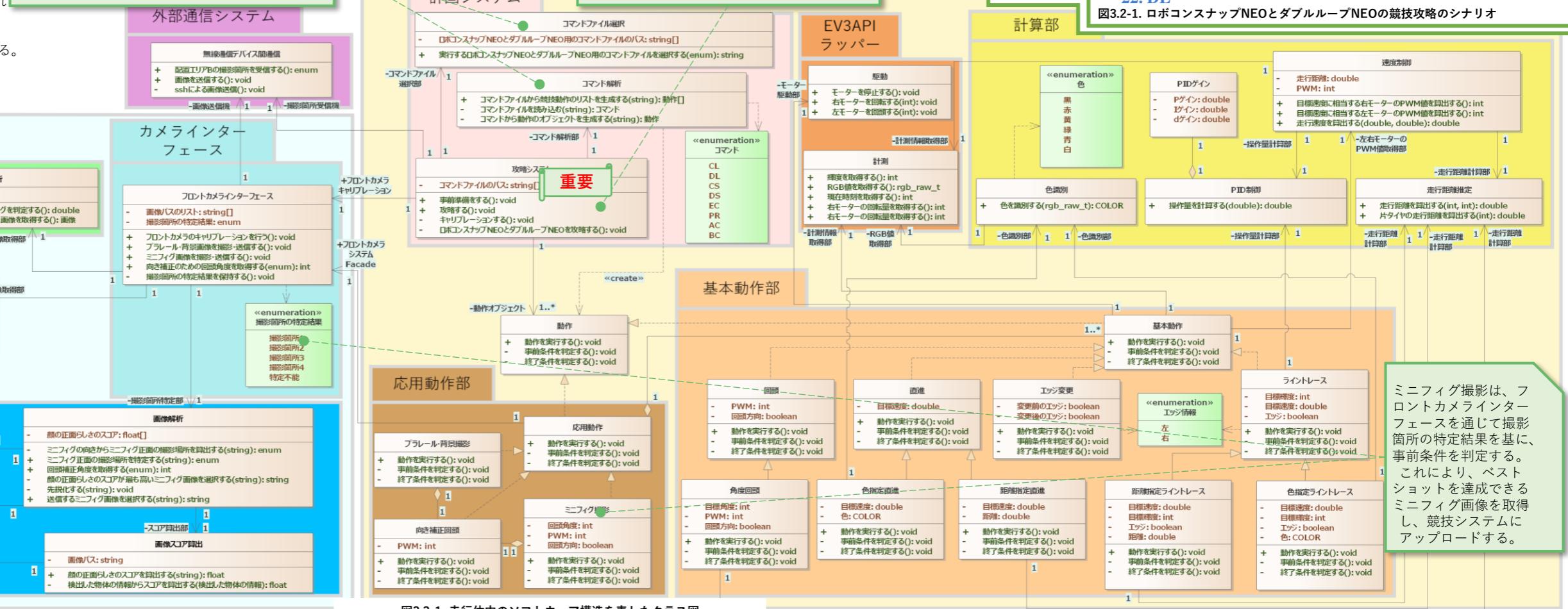
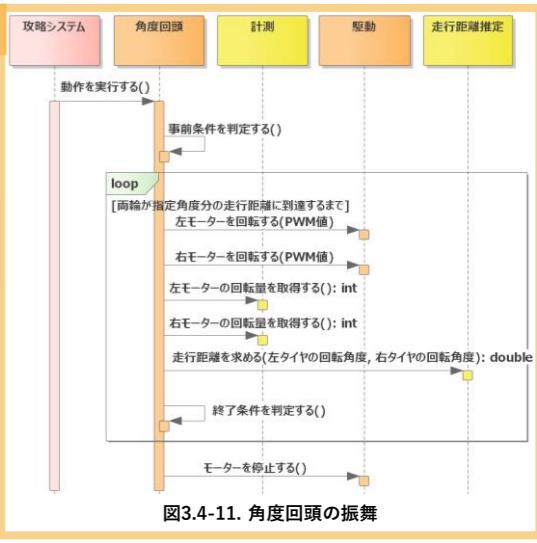
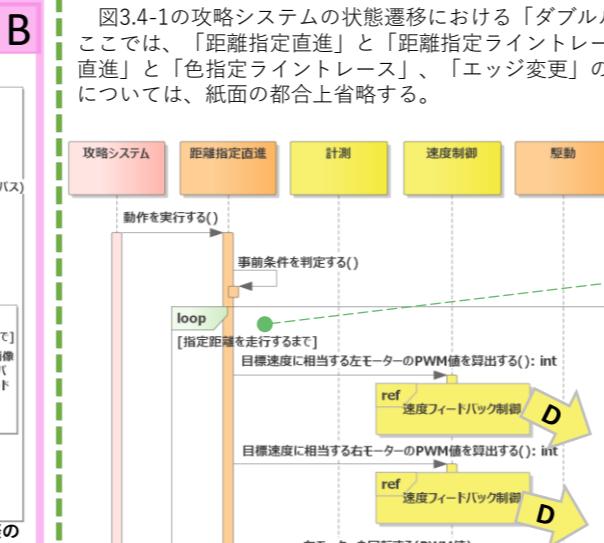
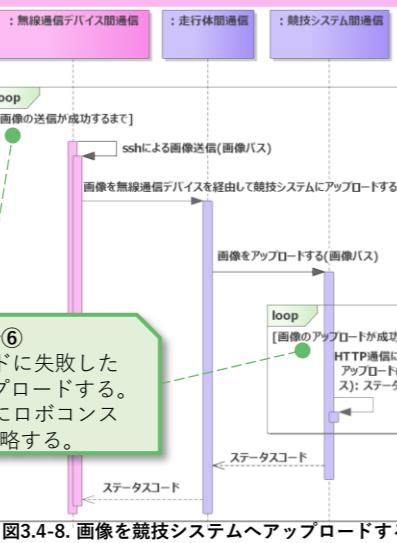
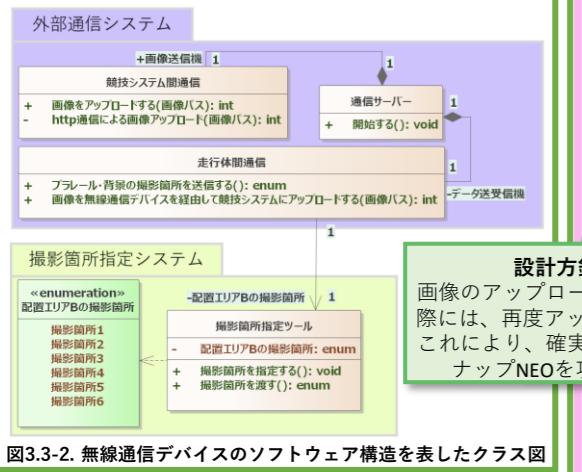


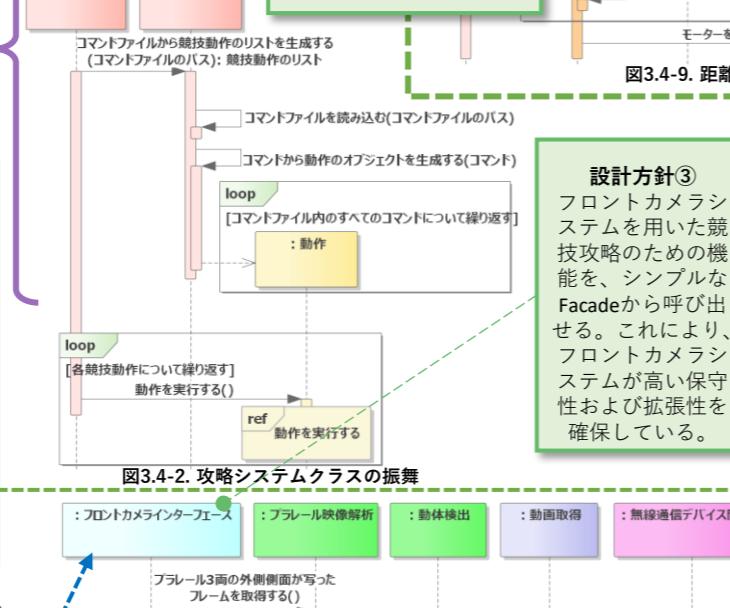
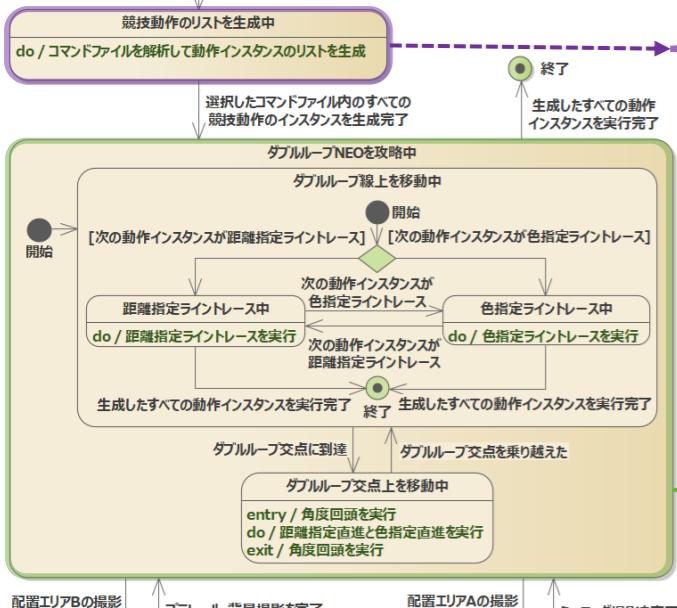
図3.3-1. 走行体内のソフトウェア構造を表したクラス図

3.3 構造設計～続き～

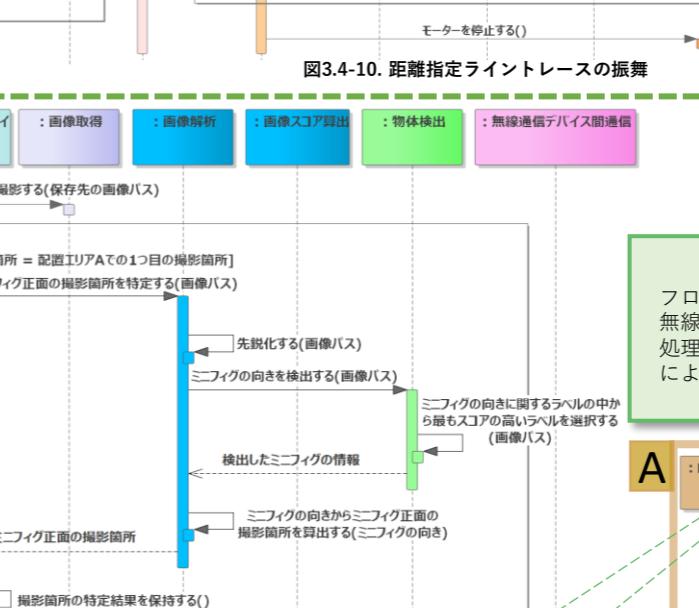


3.4 振舞設計

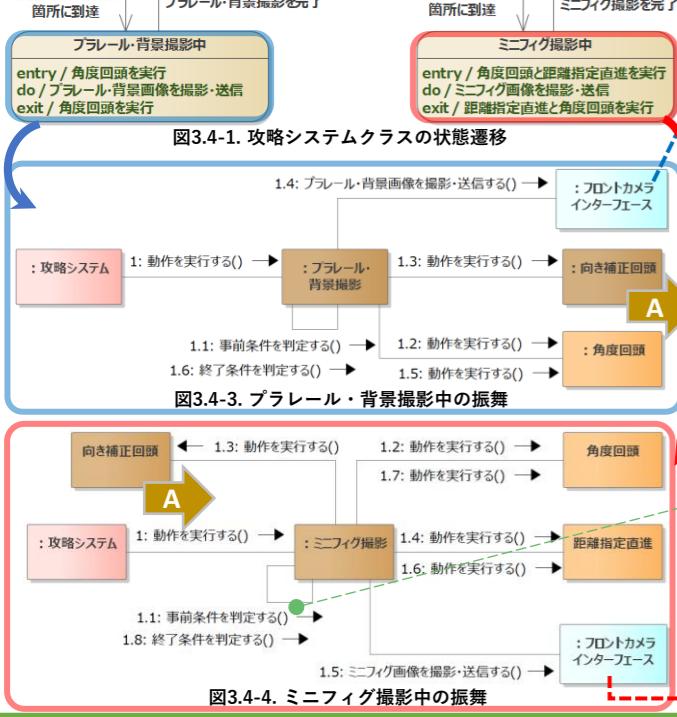
3.3節で定義したソフトウェアの構造設計において、走行システムの**攻略システム**クラスは、コマンドファイルに存在する動作コマンドのリストから競技動作のインスタンスを生成し、順次実行することで、ETロボコンの競技攻略を主導する。そこで、ロボコンスナップNEOとダブルループNEOの競技攻略における**攻略システム**クラスの状態遷移を図3.4-1に、攻略システムの振舞を図3.4-2に、それぞれ示す。



設計方針③ フロントカメラシステムを用いた競技攻略のための機能を、シンプルなFacadeから呼び出せる。これにより、フロントカメラシステムが高い保守性および拡張性を確保している。



配置エリアAではミニフィグ台を、配置エリアBでは背景下部の黄色を、それぞれ検出することで、撮影対象をできるだけカメラの中心に収められるよう補正のための回頭を行う。
(詳細は4.2節を参照)



すべての撮影候補箇所で同じミニフィグ撮影の競技動作を実行する。実際に撮影・送信を行うかは、事前条件の判定によって決定することで、配置エリアAでのベストショットを達成する。

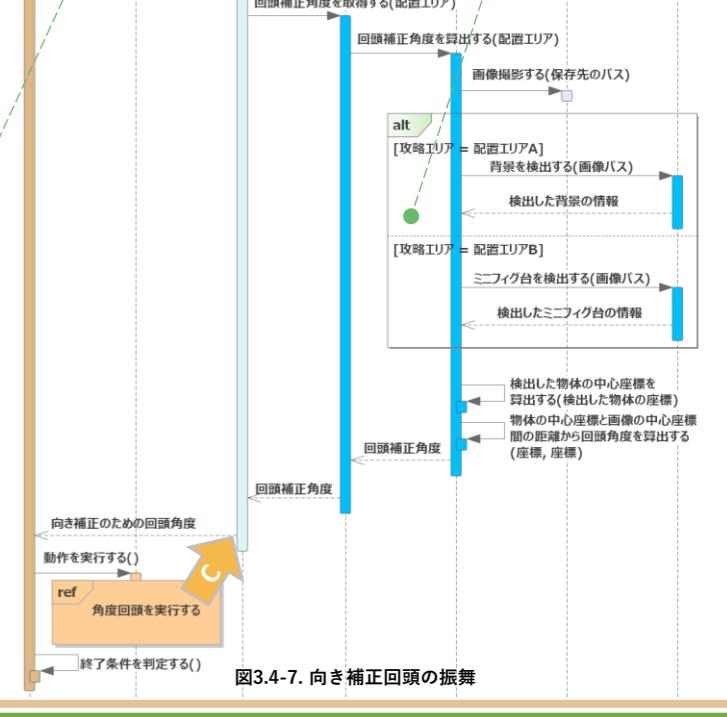
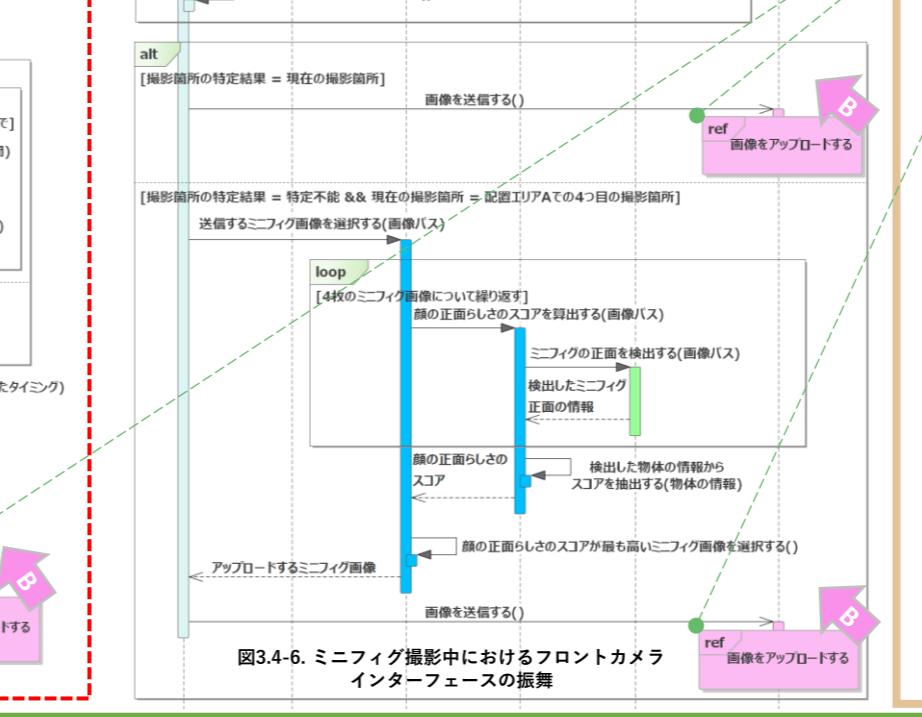


図3.4-4. ミニフィグ撮影中の振舞

図3.4-7. 向き補正回頭の振舞

4.1 走行に関する制御戦略

4.1.1 制御戦略

bl9: Commandパターン
bl14: ライントレース走行
bl15: PID制御
bl17: 直進

走行を細かく分割し、走行体の動作や制御パラメーターを変更することで、走行経路や黒線の形状に応じて動作を実行する。それら動作をコマンドとして表現し、コマンドファイルとして管理することで、各エリアにおける走行を実現する。例として、LコースのLAPゲートまで走行する際の動作の組み合わせであるコマンドファイルを、図4.1-1に示す。

また、走行経路や黒線の形状に対応して滑らかに走行するために、PID制御や速度制御を利用したライントレース走行と直進を行なう。

```
DL,2730,420,0,0.12,0.10,0.03,第一直線ライントレース
DL,460,200,-7,0.90,0.50,0.20,第一カーブライントレース
DL,1675,420,6,0.12,0.10,0.03,第二直線ライントレース
DL,450,200,-7,0.90,0.50,0.20,第二カーブライントレース
DL,200,300,-5,-0.12,0.10,0.03,第三直線ライントレース
CL,BLUE,200,-10,0.15,0.05,0.1,青まで指定色ライントレース
```

図4.1-1. LコースのLAPゲートまで走行する際のコマンドファイル

4.1.4 走行ログ可視化ツール

bl10: 走行ログ可視化ツール
bl12: 走行ログ保存



図4.1-2. 走行ログ可視化ツールの外観

機能

- 走行ログ（PWM値、輝度値、RGB値）のグラフ描画
- グラフ描画の対象（PWM値、輝度、RGB値）選択機能
- データポイントの値と対応コマンド表示機能（図4.1-3）
- 過去の走行ログ確認機能

走行ログの可視化手順

- 走行体から得た走行ログをCSV形式で保存
- CSVファイルをサーバーにアップロード
- JSON形式に変換し、グラフを描画

検証

実験において、LAPゲートまでの走行とダブルループNEOの走行について、調整メンバーにより、それぞれ10回分の調整を行った。その結果、以下に示すフィードバックが得られた。

- グラフにより意図しないPWM出力を一目で特定できる
 - 対応コマンド表示機能によって、PWM値に対応するコマンドを素早く特定できる
 - 走行後すぐにグラフを確認できる
 - 可視化により、走行動作失敗の原因を特定できた
- フィードバックより、走行ログ可視化ツールを使用することで課題が解決できたと言える。

動作失敗時に、グラフの傾きから取得輝度値の大幅な変化が原因だと素早く特定でき、対応コマンドの輝度値に関するパラメータを調整することで、安定した走行を実現できた。
よって、走行ログ可視化ツールを使用することで、走行精度の向上が期待できる。



図4.1-3. マウスホバーした際の様子

4.1.5 ライントレース復帰動作

bl13: ライントレース復帰
bl12: 走行ログ保存

実現方法

4.1.4節の走行ログ可視化ツールにより、ライントレースで意図せず脱線した場合、走行区間に応じて、走行体が取得する輝度値やRGB値に特定の傾向が存在することを確認した。そこで、走行区間ごとに脱線を検知するトリガーと、線上に復帰するライントレース復帰動作を実装する（表4.1-2、図4.1-4）。

脱線してしまった走行区間は、最低限のポイントを獲得し、次の走行区間に繋げることを目的とする

本番の調整環境と異なる走行環境では、走行ログ可視化ツールを用いて、これらの数値は適切な値に再設定する



図4.1-4. ライントレース復帰動作の例

検証結果

ライントレース復帰動作を用いる場合と用いない場合で、LAPゲートまでの走行およびダブルループNEOの走行を10回ずつ行った。それぞれの走行成功率を表4.1-3に示す。

表4.1-3より、以下の効果を確認できた。

ライントレース復帰動作	LAPゲート通過の成功率	ダブルループNEOの成功率
有	100%	70%
無	80%	40%

LAPゲート通過：成功 rate 80% ⇒ 100% !!
ダブルループNEO：成功 rate 40% ⇒ 70% !!

走行精度 25% アップ！

4.1.2 2つの課題

制御戦略を用いた走行精度向上には、以下に示す課題がある。

- 調整が必要な箇所の特定が難しい
走行精度向上に向けて、動作コマンドごとにPID制御の各パラメーターを調整する必要がある。しかし、パラメータだけでなく、走行体が取得する輝度値やRGB値・左右輪のPWM出力が走行に影響を与える場合もあり、走行失敗の原因特定が難しい。
- 高い走行精度を保つことが難しい
大会本番では、取得できる輝度値やRGB値が調整環境とは異なることで、ライントレースで脱線する恐れがある。

4.1.3 課題に対する解決策

- 走行ログ可視化ツールの開発
調整が必要な箇所の特定が難しいという課題解決のために環境による差を受けにくいWebアプリとして開発する。加えて、ログを可視化することで、走行動作失敗時に走行ログの傾向分析を支援する。
- ライントレース復帰動作の実装
大会本番での、輝度値やRGB値が調整環境とは異なることによる高い走行精度を保つことが難しいという課題を解決するために実装する。

4.2 ロボコンスナップNEOに関する制御戦略

4.2.1 制御戦略

配置エリアBにおける戦略 プラレール・背景

配置エリアBに設置された背景の向きは、競技本番前にかかる。よって、背景の撮影でベストショットを成立させるために、背景正面方向を撮影できる撮影候補箇所を6箇所、図4.2-2のように設定し、本番走行前に設定した6箇所の中から撮影箇所を1つ指定する。指定した箇所で撮影動作を行い、競技システムに画像をアップロードする。撮影候補箇所の決定にあたり、検証を行い、以下のことを確認した。

- 図4.2-1. より、撮影にあたって行う回頭の精度との関係上、背景のベストショットを達成するには、1周を60度ずつ分割できる最低限6箇所からの撮影が必要である
- プラレールの周回方向は、走行開始まで分からず。よって、周回方向に影響を受けず、確実にプラレールのベストショット画像を取得する必要がある。

配置エリアAにおける戦略 ミニフィグ

配置エリアAに設置されたミニフィグの向きは、走行開始まで分からず。そのため、ミニフィグ撮影でベストショットを成立させるために、ミニフィグの正面方向を撮影できる撮影候補箇所を4箇所、図4.2-4のように設定し、撮影候補箇所から必要な要素（ミニフィグの両目、口）を含んでいる画像を取得する撮影動作を行い、競技システムに画像をアップロードする。

撮影候補箇所の決定にあたり、検証を行い、以下のことを確認した。

- 顔の向きが分からないミニフィグの両目と口を写した画像を撮影するには、最低限4箇所からの撮影が必要である。
- 黒線上からはミニフィグの全身が画角に収まらない場合があるため、走行体をミニフィグに近づける必要がある
- 撮影動作にはラインを外れることによるコースアウトのリスクがある

以上を踏まえ、リスクを最小限にしつつ、確実にベストショットを達成する。

パーフェクトショット 成功率 90% 達成！

ミニフィグベストショット 成功率 85% 達成！



図4.2-1. 撮影動作のイメージ

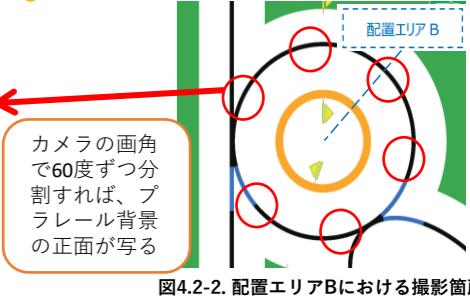


図4.2-2. 配置エリアBにおける撮影箇所

配置エリアAのミニフィグは走行開始まで向きが分からない

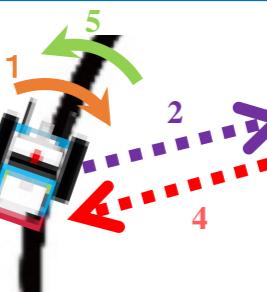


図4.2-3. 撮影動作のイメージ

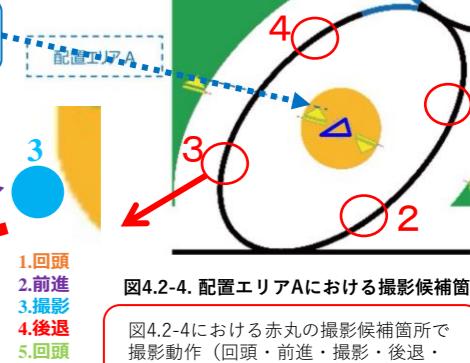


図4.2-4における赤丸の撮影候補箇所で撮影動作（回頭・前進・撮影・後退・回頭）を行う。（詳細は3.4節を参照）

4.2.2 プラレール・背景画像取得

画像取得方法 プラレール・背景

- フロントカAMERAで映像を録画する。

bl26: 背景下部の黄色検出
bl27: 映像録画

bl28: フレーム切り出し
bl29: 動体検出

検証結果

黄色検出→動体検知用バウンディングボックス配置→中間フレーム切り出しに分割し、順次処理することで、周回方向に影響を受けず、確実にプラレールのベストショット画像を取得することができる。

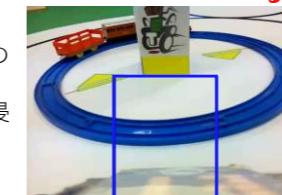


図4.2-5 動体検知用バウンディングボックス配置画像



図4.2-6. ボックス侵入・ボックス退出・中間フレームの画像

4.2.3-1 ミニフィグの向き判定

ミニフィグ

ミニフィグの正面(front)・右面(right)・背⾯(back)・左面(left)を学習させたYOLOv5を用いてミニフィグ画像を物体検出し、ミニフィグの向きを判定する（図4.2-7）。

bl22: ミニフィグの向き判定
bl19: ミニフィグ検出

検証結果

図4.2-4の1番目の赤丸で向き判定成功時、撮影回数を4回から最大1回に減らすことができる。これにより、コースアウトのリスク減少を期待できる。



図4.2-7. 正面・右面・背面・左面を検出した画像

4.2.4 走行体の向き補正

プラレール・背景 ミニフィグ

撮影対象を画角中心に収めるための回頭補正角度を算出し、回頭補正角度の分だけ回頭する。

4.2.3-2 ミニフィグの正面らしさ算出・比較

ミニフィグ

図4.2-8より、正面方向の特定に失敗した場合に発動する。4箇所から撮影したミニフィグ画像に対して、ミニフィグ(Fig)、ミニフィグの正面顔(Frontal Face)、横顔(Profile)を3つのラベルを学習させたYOLOv5を用い、物体検出したラベルの信頼度スコア(0から1までの数値)を算出する。

bl20: ミニフィグの正面らしさ算出
bl18: ミニフィグの正面らしさ比較

検証結果

図4.2-9のように、ミニフィグの顔を正面顔と横顔の2種類のラベルに分けた効果で、ベストショットに適した画像をより正確に選択することができる。

[1] 正面顔を検出した画像例
[2] 横顔を検出した画像例

図4.2-9. 表4.2-3のラベルを用いて物体検出した検証結果

4.2.4 走行体の向き補正

プラレール・背景 ミニフィグ

撮影対象を画角中心に収めるための回頭補正角度を算出し、回頭補正角度の分だけ回頭する。

検証結果 1

向き補正無しと向き補正有りのそれぞれで、20回の検証をおこなった結果、パーフェクトショット成功率：55% ⇒ 90%にUP !!



図4.2-10. 向き補正前のプラレール・背景画像(左)と向き補正後のプラレール・背景画像(右)

検証結果 2

向き補正無しと向き補正有りのそれぞれで、20回の検証をおこなった結果、ベストショット成功率：70% ⇒ 85%にUP !!

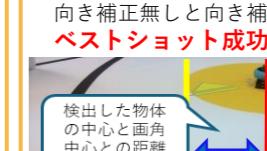


図4.2-11. 向き補正前のミニフィグ画像(左)と向き補正後のミニフィグ画像(右)