TOPPERS 活用アイデア・アプリケーション開発コンテスト

部門　：　アプリケーション開発部門

作品のタイトル　：　SPIKE-RT を用いた競技大会向け自律型ロボットプログラミング

作成者　：　平山 結也 (プログラボ 四条烏丸)

共同作業者　：　堀留 拓人 (プログラボ 四条烏丸), 尾﨑 貴俊(プログラボ 四条烏丸)

対象者　：　LEGO SPIKE Prime 使用者

使用する開発成果物　：　SPIKE-RT

目的・狙い

- LEGO SPIKE Prime で SPIKE-RT を用いて，C 言語を使用，習得する．
  また，自律型ロボットによる国際的なロボットコンテストである WRO に出場する．

アイデア/アプリケーションの概要

- SPIKE-RT を用いて SPIKE Prime で作られた自律型ロボットを自動制御し，
  与えられた課題をクリアする．

1. 背景

2022 年の年末頃に，名古屋大学 組込みリアルタイムシステム研究室 朱 義文氏より LEGO® Education SPIKE™ Prime（以下，SPIKE Prime）向けソフトウェアプラットフォーム，SPIKE-RT がリリースされた．その後 2023 年 1 月 10 日に，株式会社アフレル（以下，アフレル）より SPIKE Prime 用 C 言語学習教材「SPIKE™ プライム C 言語 プログラミングブック」が発売された．同時期，毎年開催されている WRO Japan 公認京都予選会への出場に向け準備を始める段階であった．Scratch ベースか Python でのみプログラミングが可能だった SPIKE Prime において，C 言語によるプログラミングが可能になり，新たな挑戦ということで取り組んだ．

2. SPIKE-RT の優位性

純正の SPIKE Prime は，MicroPython のランタイム（インタプリタ）で動作している．一方 SPIKE-RT は事前にコンパイルを行う OS である．WRO などの競技では，動作の正確性や安定性が求められ，リアルタイム OS である SPIKE-RT は純正のランタイムに対して，応答性などの観点において優位性がある．

3. 動作，開発環境

ハードウェア

・SPIKE Prime

SPIKE Prime は，小学校高学年から中高生向けの STEAM 学習セットである[1]．SPIKE LHub と呼ばれるプログラミング可能なデバイスには，モーターや各種センサーを最大 6 個接続でき，様々なロボットやデバイスなどを作ることが可能である．SPIKE Prime は，アフレル[2]から購入が可能である．

ソフトウェア

・SPIKE-RT

・WSL2

・Ubuntu 22.04 LTS

・Docker

・Visual Studio Code + WSL 拡張機能

4. 環境構築

SPIKE™ プライム C 言語 プログラミングブック[3]に従い環境構築後，Visual Studio Code に WSL 拡張機能をインストール．SPIKE-RT の GitHub リポジトリ[4]より，2023 年 5 月 8 日時点までの更新を適応．

5. WRO とは

WRO®（World Robot Olympiad）は，小学生から高校生までが，2 名もしくは 3 名でチームを組んで競技に参加する自律型ロボットによる国際的なロボットコンテストである．WRO には，いくつかの競技カテゴリーと年齢区分，競技難易度が存在している．

年齢区分は 3 種類存在しており，競技開催年に 8 歳から 12 歳ならエレメンタリー競技，11 歳から 15 歳ならジュニア競技，14 歳から 19 歳ならシニア競技と区分される．

競技難易度には，それぞれ“ミドル”，“エキスパート”と呼ばれる 2 種類の難易度が存在している．ミドルは初学者向け，エキスパートは中上級者向けとなっており，ミドルに比べエキスパートは競技の難易度が高くなっている．

今回は「仲間と共にロボットを組立て、与えられたミッションをクリアしながら速さを競う競技（ROBO MISSION）[5]」のジュニアエキスパート競技に中学生 2 名のチームが，同じく ROBO MISSION のシニアエキスパート競技に高校生 2 名のチームの計 2 チーム 4 名が SPIKE-RT を使用し競技に参加した．

なお便宜上，以降はジュニアエキスパート競技を中学生競技，シニアエキスパート競技を高校生競技と呼称する．

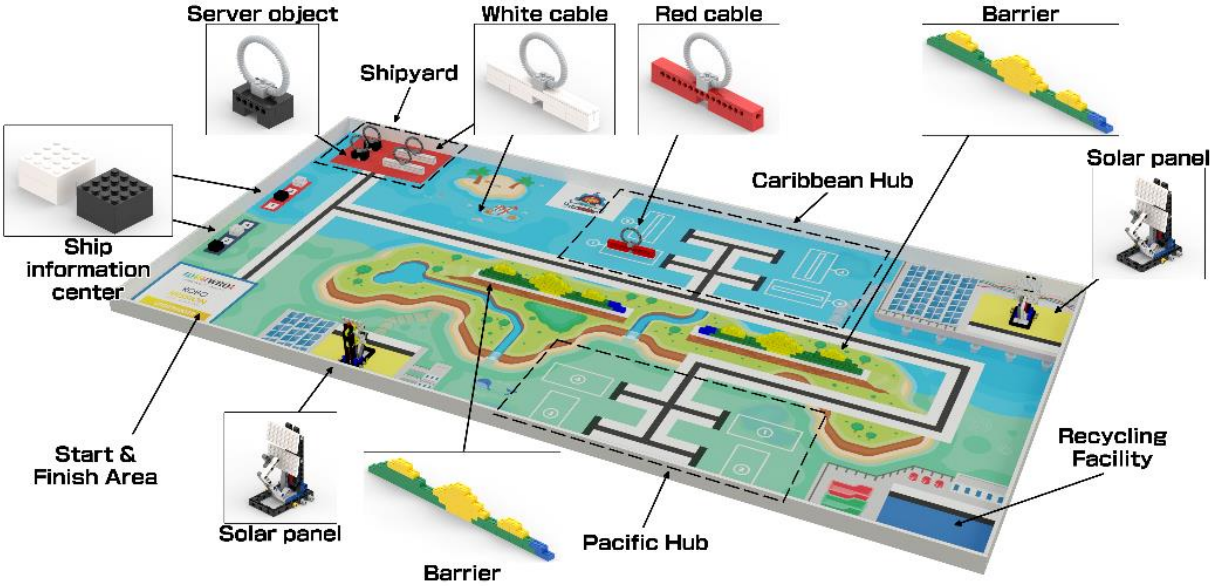
6.　WRO のルール説明

設計の導入にあたり，WRO の競技ルールを説明する．

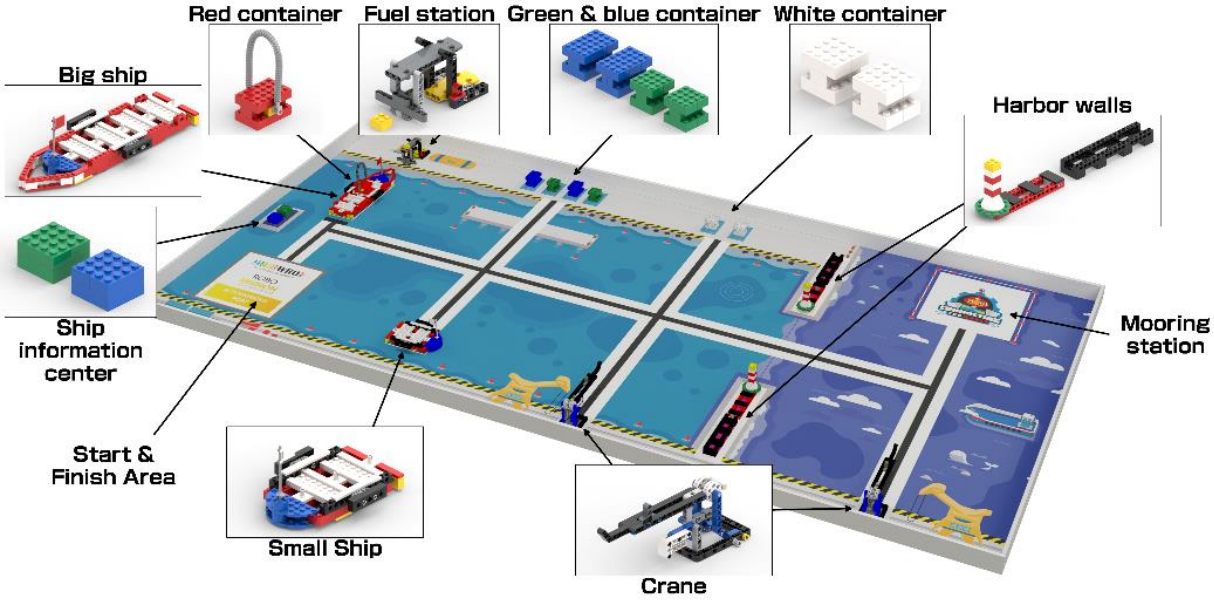
なお，共通の競技ルールとして，ロボットは Start & Finish Area から動き出すこと，競技時間は 2 分以内であること，筐体の大きさは W25cm × D25cm × H25cm であることが挙げられる．

- 中学生競技（画像 1 参照）では，予め配置された赤ケーブルをリサイクル施設に運び，ヤードに置かれている白ケーブルとサーバーを船舶情報センターでランダムに示された接続口(Caribbean / Pacific Hub)に運ぶ．　2 箇所のソーラーパネルを起動状態にし，ロボットを Start & Finish Area で停止させる.[6]

- 高校生競技（画像 2 参照）では，予め大型船に積み込まれている赤コンテナをクレーンに降ろし，船舶情報センターでランダムに示される色と一致するコンテナを 2 つ，白コンテナを 1 つ積み込み，燃料を補給し外洋エリアに出航，小型船には色を問わずコンテナを 2 つ積み込み，外洋エリアに出航，ロボットを係留エリアで停止させる.[7]



画像 1 ： 中学生競技コース



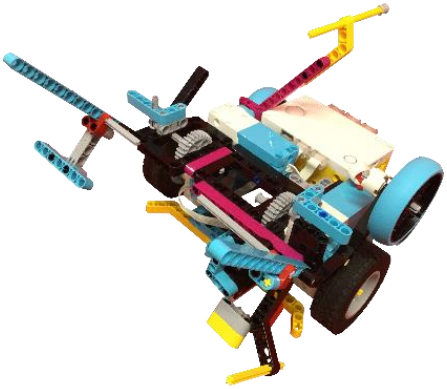
画像 2 ： 高校生競技コース

7.　設計

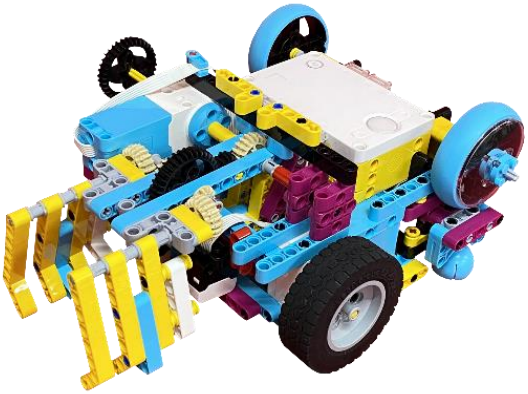
前述した競技ルールのクリアを目標として，ロボット及びプログラムを設計，開発した．

完成したロボットは，中学生競技用が画像 3，高校生競技用が画像 4 を参照．

プログラムのフローチャートは，中学生競技用が図 1，高校生競技用は図 2 を参照．



画像 3 ： 中学生競技用ロボット



画像 4 ： 高校生競技用ロボット

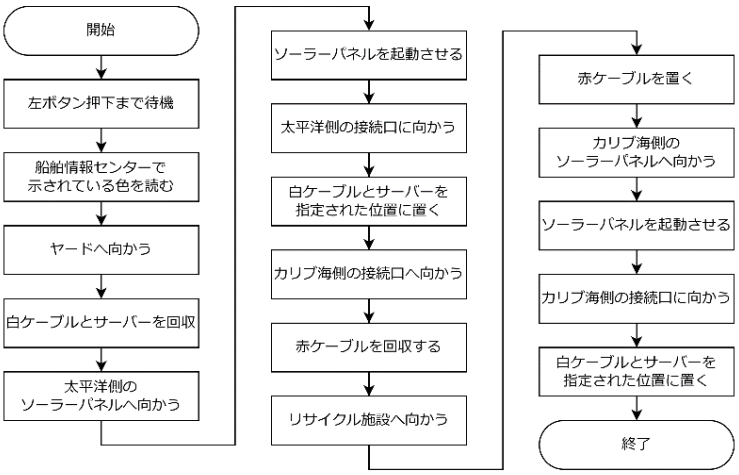


図 1 ： 中学生競技フローチャート

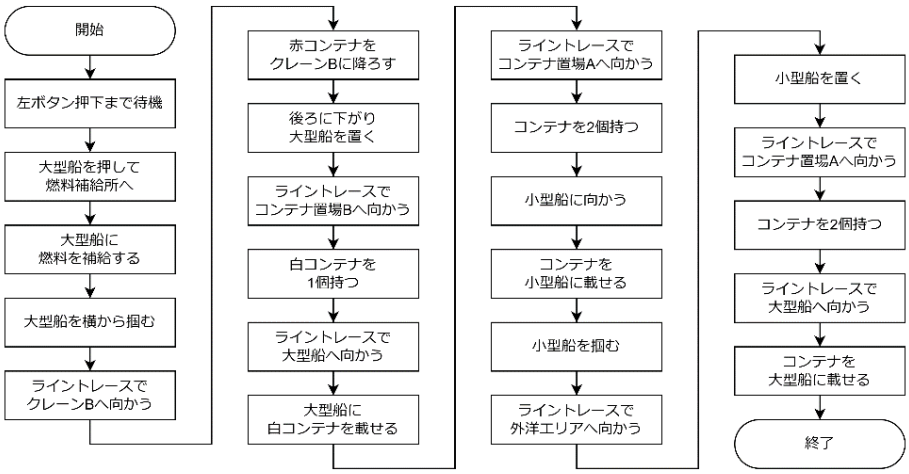


図 2 ： 高校生競技フローチャート

7-1.　ハードウェア

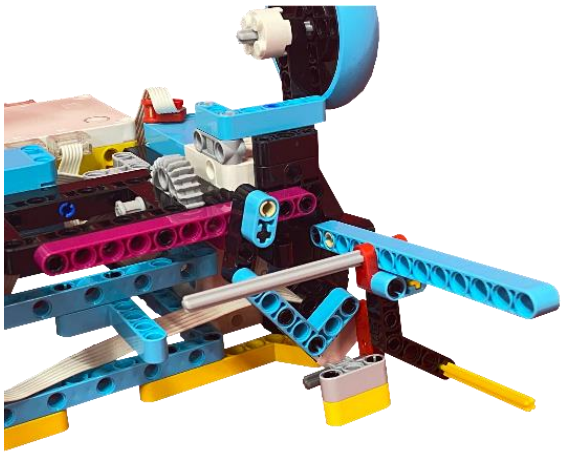
中学生競技用と高校生競技用のそれぞれについて，ロボット設計で工夫した点を説明する．

なお，SPIKE-RT で動作させるにあたり，SPIKE Prime に搭載されているモーターなどを接続する外部ポート 6 個のうち 1 個（ポート F）が使用不可の仕様であるため，ポートが 1 個使えない状態であっても，競技が行えるようにロボット設計を工夫する必要があった．実際の動作映像は中学生競技が“junior_run.mp4”，高校生競技は“senior_run.mp4”を参照．

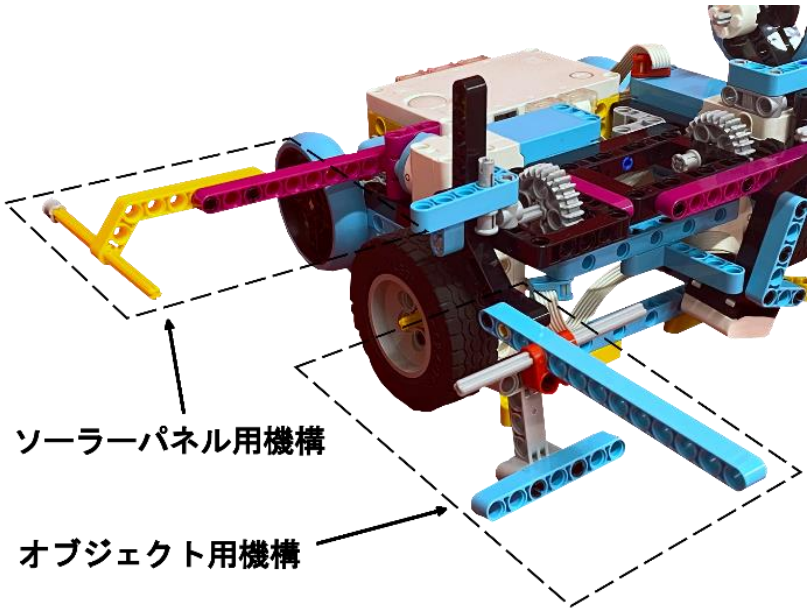
- 中学生競技用では，主にアームについて説明する．

ポート F が使用できないため，左右両方のアームに複数の機能を搭載した．左アームには，カラーセンサー1 つでライントレースとオブジェクトの色読み取りを行うため，アームとカラーセンサーを連動して動作させるようにした(画像 5，画像 6 参照)．右アームには，ソーラーパネルを起動させるための機構とオブジェクトを持ち上げるための機構を搭載した(画像 7 参照)．ロボットの重心を低く設定し，操作性と走行性を高め，オブジェクトを持ち上げた際にロボットが浮き上がらないように後方に重りを搭載した．またアーム後方にも重りを載せオブジェクトとの重量バランスを取れるようにした．

画像 5 ： 左アームとカラーセンサーの動き 1



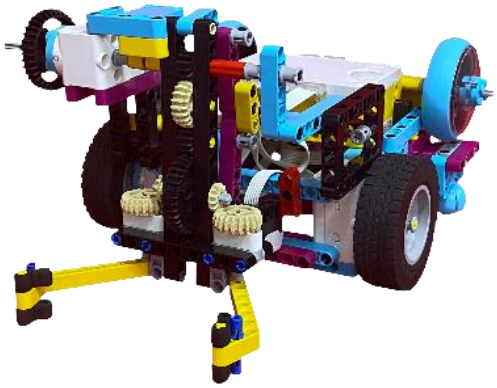
画像 6 ： 左アームとカラーセンサーの動き 2



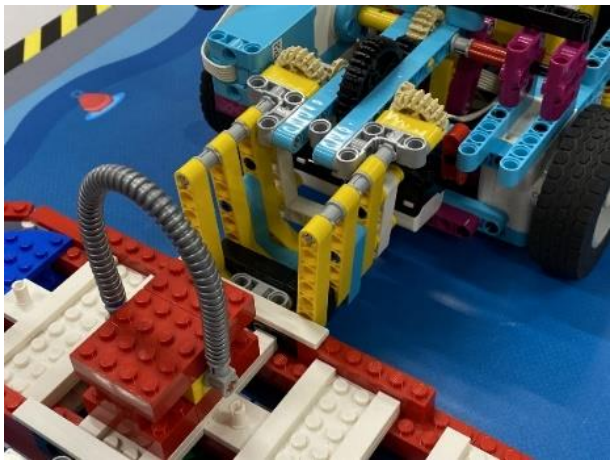
画像 7 ： 右アームの機構

－ 高校生競技用では，アームについて説明する．

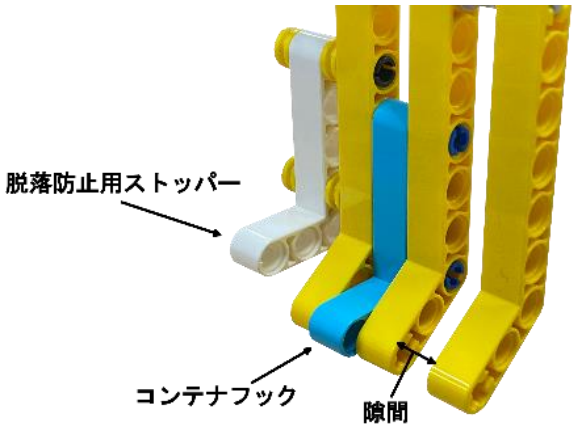
アームは，同一形状で大型船と小型船を掴めること，コンテナを持つことができるように設計する必要があった．一次設計では，走行路面と平行にしたアームを使用していたが（画像 8 参照），大型船と小型船を掴んだ状態で移動する際に，アームが船の重さに負けてしまい，船がアームから脱落する事象が発生した．二次設計では，アームを走行路面と垂直にし，大型船と小型船の側面フック部分に引っ掛けるようにして掴むように変更を行った（画像 9 参照）．アームをすのこ状にし，先端部分には船の側面フックに引っ掛けるための隙間を設けた一方，手前部分には，コンテナの溝に適合するフックとコンテナを持ち上げた際の脱落を防止するストッパーを取り付けた（画像 10，画像 11 参照）．また，アームとカラーセンサーをリンク機構により連動させ，ライントレース時は走行路面に指向し，コンテナなどの色を見る際は，後方に 45°程度傾き，前方に指向するようになっている（画像 12，画像 13 参照）．



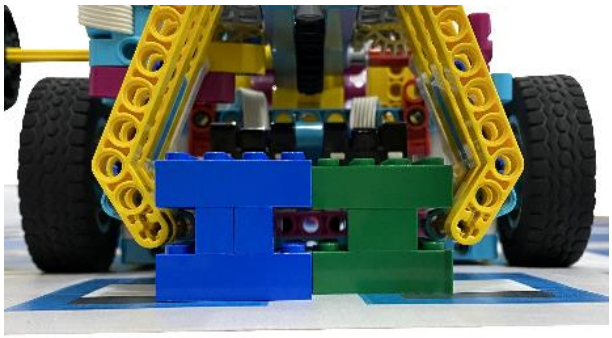
画像 8 ： 一次設計のアーム



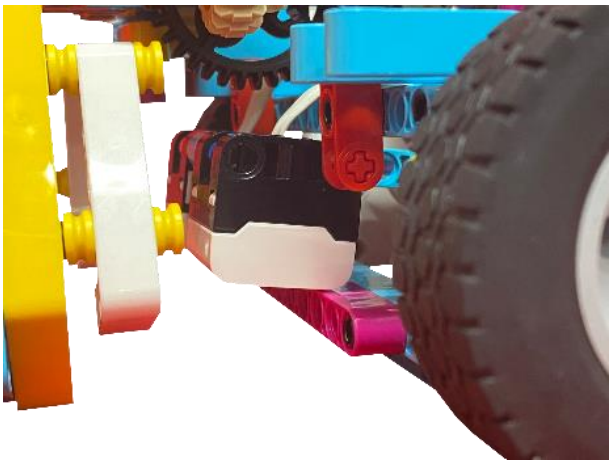
画像 9 ： 船をつかむ様子



画像 10 ： アーム構造



画像 11 ： コンテナの溝に引っ掛ける様子



画像 12 ： カラーセンサーの動き 1

画像 13 ： カラーセンサーの動き 2

8.　ソフトウェア

中学生競技用と高校生競技用のそれぞれについて，プログラム内で工夫した点を説明する．

- 中学生競技用では，条件分岐と台形制御について説明する．

オブジェクトを接続口に置くための条件分岐では，ロボットがオブジェクトを置きやすい位置から順に置けるように，分岐を工夫することで動作の簡略化を行い，無駄な動きを減らし成功率が上がるようにした．

条件分岐のフローチャートは図 3 を参照．ソースコードは“Junior_ObjectPut.h”を参照．

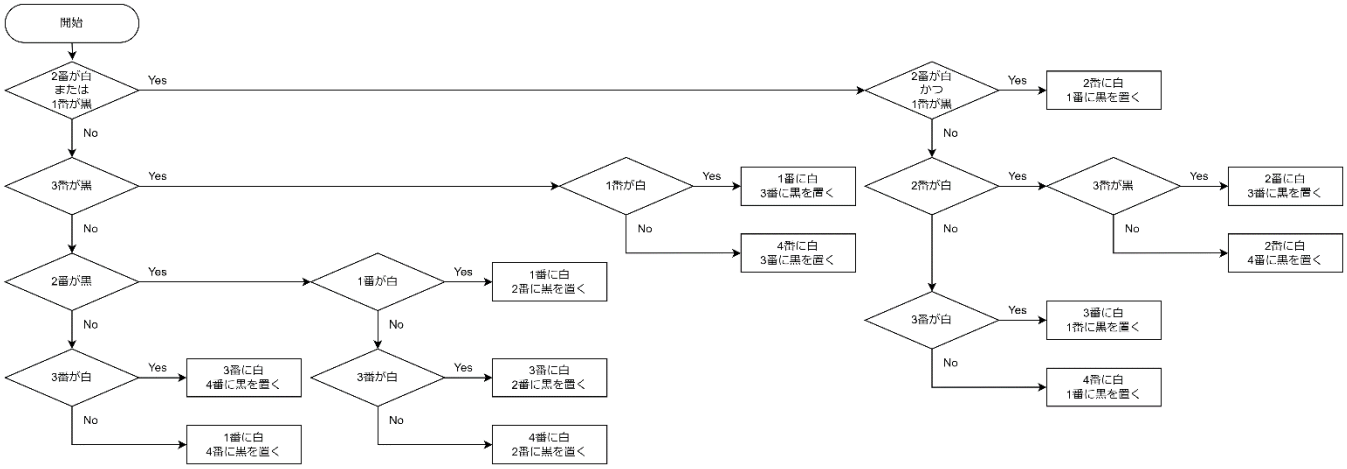


図 3 ： 条件分岐のフローチャート

台形制御は，一般に加速に要した時間を基に制御を行うが，今回はモーターエンコーダの回転角を基に制御を行った．

まず，開始速度(V_s)から巡航速度(V_m)への加速に要した回転角(x)を取得し，以下の式で巡行を行う区間(M_2)を求める．M_Aは全体の移動距離，V_eは終了速度である．$V_s < V_e$ならxより小さい回転角から，$V_s > V_e$ならxより大きい回転角から減速を開始する．

$$a = 0.2,\quad t_A = \frac{V_m - V_s}{a},\quad t_B = \frac{(V_m - V_e) \times 0.9}{a},\quad M_2 = M_A - \frac{x \times \{(V_m + V_e) \times t_B\}}{\{(V_m + V_s) \times t_A\}} - 50$$

減速後は，速度V_eで目標値まで前進する．加速部分のプログラムは画像 14，巡行/減速部分は画像 15 を参照．

また，左右のモーターに個体差があり，出力数値が同じでも直進はできなかった．そこで左右のモーターエンコーダの差を基に，元々パワーが大きいモーターの出力を調整し，直進できるよう工夫を行った．ジャイロセンサーを使用する場合に比べ容易に実装ができる事も利点である．

台形制御のソースコードは“Junior2_MotorBase.h”内の tADJMotor 関数を参照．

```
startAngle = pup_motor_get_count(motorA);
accelTime = (MotorPower - startPower) / accel;
brakeTime = (MotorPower - finPower) * 0.9f / accel;

for (int i = 0; i < accelTime; i++)
{
  basePower = (startPower + (accel * i)) * FrontBack;

  leftAngle = pup_motor_get_count(motorA);
  rightAngle = pup_motor_get_count(motorB);
  motorDiff = leftAngle - rightAngle;
  rightPower = basePower + (motorDiff * ADJMOTOR_GAIN);

  pup_motor_set_power(motorA, basePower);
  pup_motor_set_power(motorB, rightPower);

  dly_tsk(1000);
}
```

画像 14 ： 加速部分のプログラム

```
accelAngle = abs(pup_motor_get_count(motorA) - startAngle);
brakeAngle = accelAngle * ((MotorPower + finPower) * brakeTime) / ((MotorPower + startPower) * accelTime);
moveAngle = MotorAngle - brakeAngle - 50;

ADJMotor(MotorPower * FrontBack, moveAngle, 0, false);

for (int i = 0; i < brakeTime; i++)
{
  basePower = (MotorPower - (accel * i)) * FrontBack;

  leftAngle = pup_motor_get_count(motorA);
  rightAngle = pup_motor_get_count(motorB);
  motorDiff = leftAngle - rightAngle;
  rightPower = basePower + (motorDiff * ADJMOTOR_GAIN);

  pup_motor_set_power(motorA, basePower);
  pup_motor_set_power(motorB, rightPower);

  dly_tsk(1000);
}

ADJMotor(finPower * FrontBack, MotorAngle, stop, false);
```

画像 15 ： 巡行 / 減速部分のプログラム

- 高校生競技用では，ジャイロターンとライントレースについて説明する．

ロボット旋回時の安定性と精度を高めるため，SPIKE Prime に内蔵されているジャイロセンサーを使用した．ジャイロセンサーから取得した角速度を積分し，現在のロボットの角度を割り出す．積分し割り出した角度と実際の角度で差が生じるため，積分した結果に一定の数値を加算し差を補正しているほか，ジャイロターン関数の引数に対しても，固定数値と，走行路面に応じた可変数値の加減算を行っている（以下の式，及び画像 16 参照）．

積分式: *積分値* += *z軸角速度* × *取得間隔* + *補正値*　　対引数式: *旋回角* = *指示角度* ± *固定値* ± *可変値*

ソースコードは“Senior_Lib.h”内の gyro 関数を参照．

```
while (true) {
  hub_imu_get_angular_velocity(gx); //角速度を格納

  deg += gx[2] * time + 0.024; //動いていなくても軸がマイナスに動くためプラス側に補正
  dlyms(time * 1000); //z軸を積分格納して待機

  //条件ヒットしたらループを抜ける
  if (deg <= set_deg + 12 + correction && set_deg < 0) { //負旋回かつ指定旋回角を超えたらループを抜ける
    break;
  } else if (deg >= set_deg - 12 - correction && set_deg > 0) { //正旋回かつ指定旋回角を超えたらループを抜ける
    break;
  }
}
```

画像 16 ： ジャイロターンのメインプログラム

ライントレースでは，比例制御(P)を採用した．比例・積分・微分を行う PID 制御には劣るが，直線のライントレースのみであることや，ライントレースを行う際に低速から高速に，高い係数から低い係数になるよう関数を呼び出すことで，ロボットをラインと平行に近い状態にし，高速走行ができるよう工夫した．また，カラーセンサーを 2 個搭載しているため，両方のセンサーを使うライントレースと，片方のカラーセンサーをライン読み取り用，もう 1 つを特定の色を読み取ったときに動作を停止する用と関数内で引数に応じて使い分けることによって，2 種類のライントレースが行えるようにした（画像 17 参照）．

ソースコードは“Senior_Lib.h”内の linetrace 関数を参照．

```
while (true) {
  //モーターA, Bの回転角度を格納
  int motor_1_deg = pup_motor_get_count(motor_A);
  int motor_2_deg = pup_motor_get_count(motor_B);

  //反射光に応じてモーターパワーを指定
  if (duble) {
    pup_motor_set_power(motor_A, power*1.2 - ((pup_color_sensor_reflection(cs_B) - 5) - (pup_color_sensor_reflection(cs_A) - 5) * 2) * gein);
    pup_motor_set_power(motor_B, power*1.2 + ((pup_color_sensor_reflection(cs_B) - 5) - (pup_color_sensor_reflection(cs_A) - 5) * 2) * gein);
  } else {
    pup_motor_set_power(motor_A, power*1.2 - (50 - (pup_color_sensor_reflection(cs_A) - 5) * 3) * gein);
    pup_motor_set_power(motor_B, power*1.2 + (50 - (pup_color_sensor_reflection(cs_A) - 5) * 3) * gein);
  }

  //条件ヒットしたらループを抜ける
  if (count * 360 <= (motor_1_deg + motor_2_deg) / 2) {
    break;
  } else if (pup_color_sensor_color_name(cs_A, true) == color && pup_color_sensor_color_name(cs_B, true) == color && !duble) {
    break;
  } else if (color == 'w' && pup_color_sensor_color_name(cs_B, true) == color) {
    break;
  }
}
```

画像 17 ： ライントレースのメインプログラム

9. 動作状況など

ロボットの動作状況を撮影した動画や 6 面図，細部の写真などの補足写真，本文書内で使用した画像の高解像度版（ソースコード画像を除く）を GitHub Pages にて公開している．

https://hira-yu.github.io/toppers-contest/

10. ソースコード

ロボットのソースコードについても GitHub にて公開している．

https://github.com/hira-yu/toppers-contest/tree/main/source

11. 課題点

2023 年 9 月時点で，SPIKE-RT には次の 3 つの課題点が存在する．

11-1. 180 秒経過でのタイムアウト

SPIKE-RT はプログラムの実行開始から 180 秒が経過すると，実行中のプログラムを終了するようである．純正の SPIKE Prime は少なくとも 60 分以上連続動作する．タイムアウトまでの時間，あるいはタイムアウト機能の有効無効を設定できるようなれば，長時間連続動作するデバイスなどを制作することが可能となり，SPIKE-RT が使用される場面も増えるものと思われる．

11-2. プログラムの中断操作

SPIKE-RT において，実行中のプログラムを中断するには中央丸ボタンを 3 秒間押下する必要がある．純正の SPIKE Prime では，単に中央丸ボタンを押下するだけである．今回制作した様な移動型ロボットの場合，移動中に瞬時に停止することができず，モーターなどへの負荷も考慮し一度持ち上げてから中央丸ボタンを押下する必要があった．また，プログラムの並列化による改良を試みたが，実装までは至らなかった．

11-3. 充電時の課題

SPIKE Prime は，バッテリー駆動であるため充電が必須である．しかし，充電ケーブルを挿すとプログラムが自動的に実行されてしまうため，充電の都度前述のプログラム中断を行う必要がある．

12.　参考文献

[1]. レゴ エデュケーションの SPIKE Prime 製品ページ:

https://education.lego.com/ja-jp/products/-spike-/45678/

[2]. アフレルによる SPIKE Prime 販売ページ:

https://afrel-shop.com/view/item/000000000569

[3]. SPIKE Prime 用 C 言語学習教材:

https://afrel.co.jp/product/spike/c-language/

[4]. SPIKE-RT GitHub リポジトリ:

https://github.com/spike-rt/spike-rt

[5]. WRO JAPAN 公式サイト WRO とは:

https://www.wroj.org/action/wro

[6]. WRO Association Junior Game Rules:

https://wro-association.org/wp-content/uploads/WRO-2023-RoboMission-Junior.pdf

[7]. WRO Association Senior Game Rules:

https://wro-association.org/wp-content/uploads/WRO-2023-RoboMission-Senior.pdf

以上