テクニカルドキュメント

こんぶ畑

2024 年 3 月 1 日

1　チーム

1.1　チームについての説明

チーム名：**こんぶ畑**
メンバーの 1 人が所属していた部活の愛称と顧問への敬意を込めて名付けた。
○○ (リーダー) は高 1、高 2 とレスキューラインに参加してきた。今年度は受験の年で参加は半ば諦めていたが、幸運にも年内に志望校合格を果たし、もう 1 人を連れてくることでチームを結成することができた。

1.2　チームメンバー

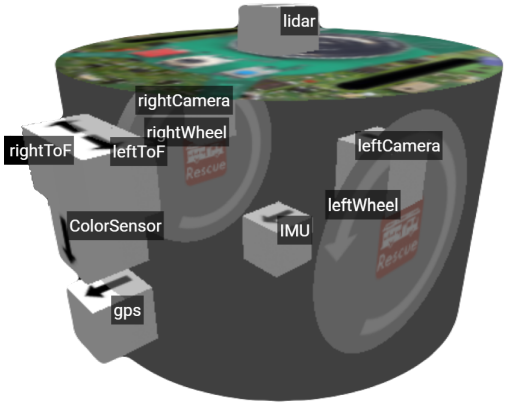
○○○○ (●●●●)
プログラムの作成をした。
今年こそ最後の大会。悔いのないように頑張りたい。

□□□□ (■■■■)
ほげほげふがふが

—以下予想される内容—

2　ロボットの構造など全般の説明

2.1　使用しているセンサーと使用目的



- 左右モータ + エンコーダ
  標準のものをそのまま使用している。
- IMU

角度を取得するために使用している。

- GPS
  位置情報を取得するために使用している。
- カラーセンサ
  床の色情報を取得するために使用している。床との距離を近づけすぎないことで、チェックポイントタイルと通常タイルの判別をしやすくしている。
- 距離センサ
  くっついているだけの飾りである。いつか取ろうと思っていつまでたっても取っていない。
- LiDAR
  障害物の検知、壁の認識に使用している。
- 左右カメラ
  被災者の検知に使用している。

3　使用しているプログラミング言語・ソフトウェア

主な開発環境

- Webots 2023.a
- Erebus v23.0.5
- C++ 14 (メイン言語)
- Python 3.12.1
- Visual Studio 2022

使用した言語は C++ である。殊レスキューシミュレーションにおいては Python の方がポピュラーであることは何となしに知っていたが、C++ の方が好みなので C++ を採用した。
もちろん、画像認識の選択肢が狭まることは承知の上である。

Visual Studio の環境構築には苦労した。なにせ、レスキューシミュレーション公式サイトにも Discord にも情報がないのである。結果的には、Webots の公式サイトの情報を基に作成した。このままではよくないと思い、簡単にだが環境構築方法をまとめたページを作成した。`https://qiita.com/kikou0517/items/f13045b6b97767e7d0f0`

使用ライブラリ

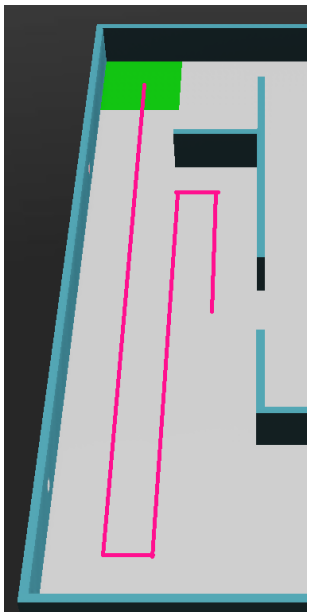
- OpenCV 4.8.0

4　被災者・ハザードマップ

考えたくない泣

5　探索アルゴリズム

深さ優先探索を使用する。探索は 6cm 単位で行う。
オリジナルの工夫点も加えた。



上図のマップでは、ピンク線で示したように探索済みのマスを半分跨ぎながら進むという効率の悪い探索を行う可能性がある。これは機体のサイズは 12×12cm を基準とするのに対して、6cm 単位で探索を行うことが原因である。そこで深さ優先探索における「未探索」「探索済み」という 2 つのパラメーターに「部分的に探索済み」を加え、進行方向の候補が複数あった場合に「部分的に探索済み」の優先度を下げることで、上図のような探索を防いでいる。

5.1　壁の認識

LiDAR による壁の認識は 6cm 進むごと、つまり深さ優先探索の進路選択の一部として行う。識別する方向は機体の前後左右のマスである。
まず、ロボットが毎回完璧な位置にいるとは限らない。その影響を最小限に抑えるために 2 種類の補正を行う。

1. 東西南北方向に対しての傾きを補正する (下図)。修正には回転行列を用いる。

$$\begin{pmatrix} X \\ Y \end{pmatrix} = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x\cos\theta - y\sin\theta \\ x\sin\theta + y\cos\theta \end{pmatrix}$$

2. 目標位置とのズレを補正する。こちらはズレの分だけ全ての座標をずらすだけである。

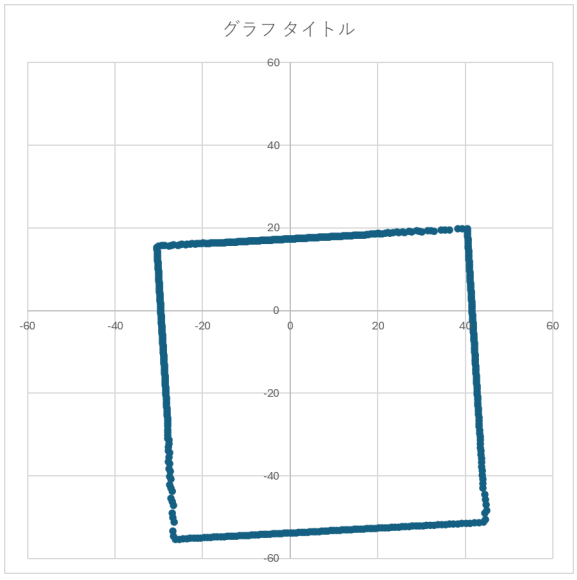


図 1　修正前

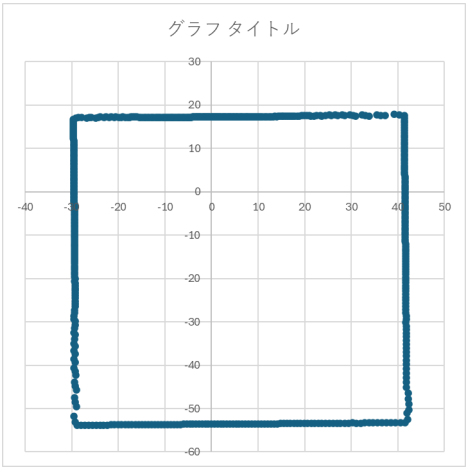


図 2　修正後

次に、壁の認識を行うのに必要な範囲のデータを切り取る。切り取る範囲はそれぞれの方向の 12cm である。

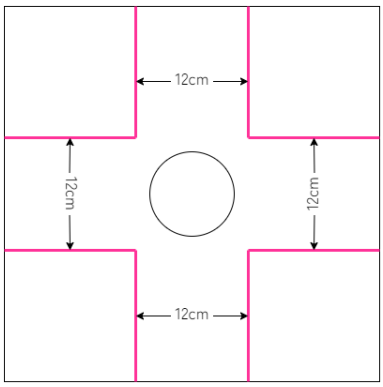


図 3

切り取ったデータはそれぞれの方向を向いており扱いづらいので、機体前向きに回転させる。

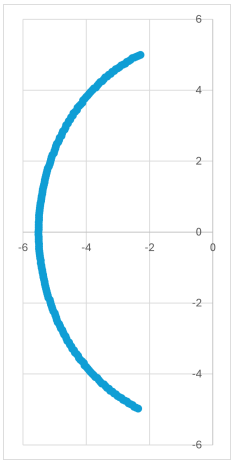


図 4　修正前

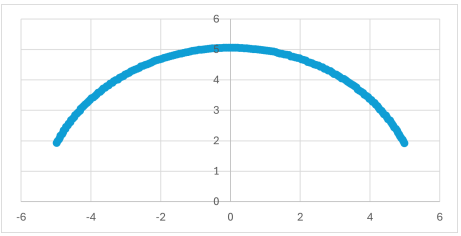


図 5　修正後

5.2　障害物の検知

6　アピールポイント

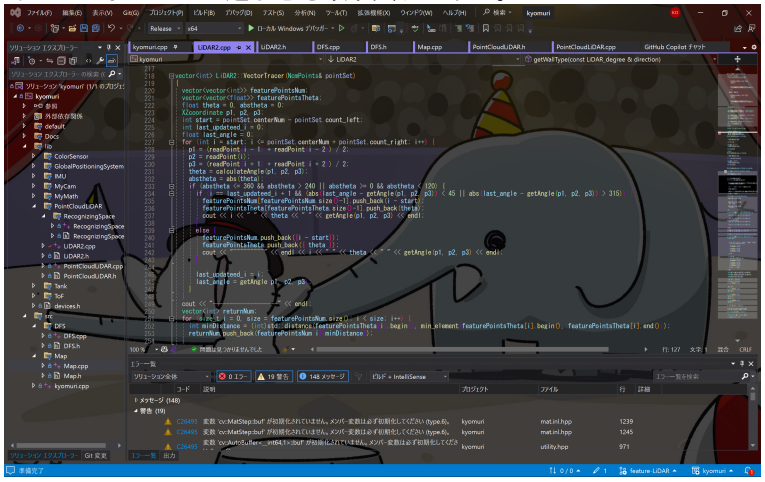
6.1　情報公開

情報公開はロボカップジュニアの国是のようなもだ。
しかし、シミュレーションリーグでコードをオープンにすることは、実機リーグのそれとは持つ意味合いが少し違うと考える。良くも悪くも、たった 1 つの exe ファイルが文字通り「全て」なのだから。だが、RCJJ のコミュニティに何か少しでも貢献できることがあればと思い、今年もソースコードを大会終了後に公開することにした。
実際にこのソースコードを公開することで直接誰かの参考になるとはあまり思っていない。それよりも、自分を含めより多くの人が情報公開をすることで、よりコミュニティが活発になることに期待している。

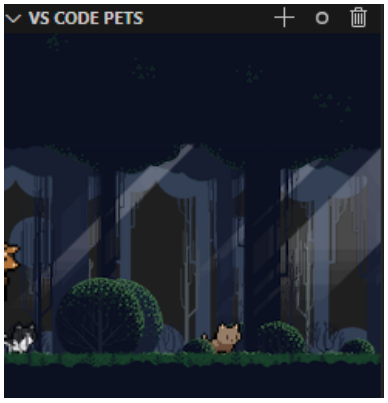
去年のリポジトリも公開状態だが、ポスターに明記していなかったため、アクセスする方法が限られていた。今年はポスターに載せているのでそのようなことにはならないはずである。

6.2　アソビゴコロ

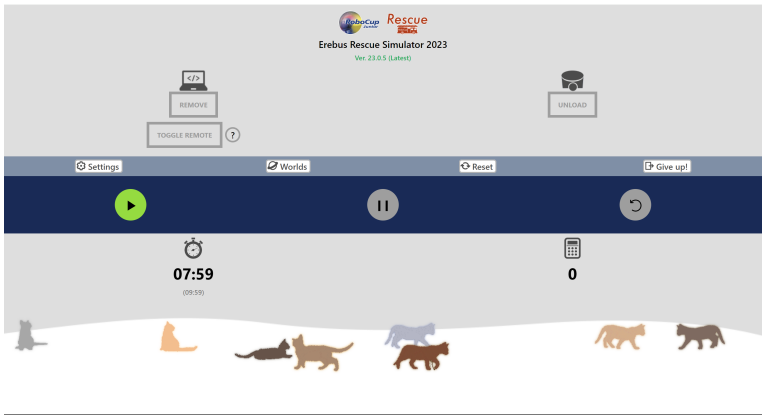
　ずっとカラフルな文字と黒い背景ににらめっこしていると、目だけじゃなくて心まで疲れてくる。
そこでちょっとした遊び心を取り入れてみた。



Visual Studio の画面である。見ての通り、背景にキャラクターの画像を設定している。



Viaual Studio Code では vscode-pets という拡張機能を使用して、いつでも猫を眺めることができる。



ブラウザを開けば、やはり猫がいる。”ネッコサーフィン”という拡張機能だ。

　こうやって、たまに癒されいている。