概要

私たちのチームは、レスキューラインのすべてのタスクを完遂するために、一番重要なことは早く正確にライントレースをすることだと考えています。そのために私たちは、ラインの未来位置を予測しその位置に正確にロボットを動かすことが必要であると考え、25個のラインセンサ及びカメラによってラインをベクトルで検知することによりラインの未来位置を予測することができます。これによりメカナムホイールの運動方程式にベクトルを代入することによって正確にラインをトレースすることができます。またレスキューゾーンにおいては、被災者を効率よく回収するために、８個のレーザー距離センサを用いることで三次元の地図を作り、短時間でボールを回収することができます。そして軽く頑丈にするためにフレームは炭素繊維強化プラスチック(CFRP)を使用したロボットになっています。

このようにして私たちは制限時間内にすべてのタスクを終えることができ、故障の少ないロボットを作ることができました。

1 はじめに

A チーム

私たちのチームは、海陽学園海陽中等教育学校ロボット部レスキュー部門の５年生で構成されたチームであり、今までにロボカップジュニアワールドリーグ東海大会に二回出場しています。私たちのチームにはウェブサイトなどはありませんが、Githubのリンクを添付しておきます。

B チームメンバー

Mitsuyoshi Sugaya

ソフト、ハード、電子回路担当

Ryouta Obara

ソフト、デバッグ担当

2 プロジェクト計画

A 全体的なプロジェクト計画

A-1 コンテストの目的

私たちのチームがこのコンテストに出場する理由は、私たちがこの一年間で学んできたロボットの制御アルゴリズム、設計、電子回路などの成果をこのコンテストでロボットに組み込み、これまで学んできたことをより良いものにし、未来につなげるためです。また他のチームの良いところを学ぶことによって私たちのチームの知識を深め、自分たちの技術などを部活の後輩に伝承するためでもあります。

A-2 プロジェクトの計画

私たちのチームではプロジェクト計画が三つあります。一つ目はハードの計画です。私たちのチームではこれまでレゴマインドストームEV3で大会に出ていましたが、ルールの改変によりEV3ですべてのタスクをこなすことは困難であると考え、一から設計することにしました。設計要件として私たちは、「大きい衝撃を加えられても壊れず、メンテナンスがしやすく、軽い」という３つを目標に設計をしました。２つ目は電子回路の計画です。これまでEV3でしかRCJに出場してこなかったので、電子回路とは無縁でしたが、ロボットを一から設計するようになって電子回路を設計する必要が出てきました。初めのころは、ユニバーサル基盤にジャンパワイヤで配線をしていましたが、今回のロボットからはプリント基板を自分で設計し企業に発注することで信頼性の高い電子基板をロボットに乗せることができるようになりました。三つ目はソフトの計画です。EV3からシフトしたためにメインマイコンをArduinoにする必要がありました。ArduinoはEV3とは違いC言語でプログラミングする必要があり、まずはC言語を学ぶ必要がありました。しかしArduinoに移行することによって比較的自由なプログラムを書くことができ、プログラムの誤作動やバグを減らすことができるようになりました。

A-3 マイルストーン

私たちのチームでは次のようなマイルストーンを設定しています。

１、設計要件、システム要件の決定及び各種センサ、部品の選定

２、機体設計

３、回路設計

４、メカナムホイールの運動方程式の算出

５、カメラの機能を理解

６、プログラム設計

７、機体、回路の不具合の修正およびプログラムの各変数の調整

８、最終調整

A-4 タスクと制約の分析及びプロジェクトへの影響

2018、2019年のルールから2020年のルールに変わったことによりタスクにシーソー、レスキューキットなどが追加されレスキューゾーンの改変も行われました。これにより私たちのチームでは、EV3による全タスクの完遂は非常に難しいと考え、EV3から機体は3DCadで設計しマイコンをArduino mega、Arduino nano、esp32に変更し、EV3からのシフトにより電子回路をKiCadで設計しプリント基板をロボットに搭載しています。またレスキューキットを距離センサ、カラーセンサなどで判別することは難しいと考え、カメラを搭載することにしました。レスキューゾーンの改変については、8個のレーザー距離センサを搭載することによって三次元の地図を作り対応しています。

B 統合計画

私たちのロボットでは3個のマイコン、37個の各種センサ、9個のモータ、1個のカメラを三個のマイコンの中のメインマイコンであるArduino megaで一括制御しています。マイコン同士はI2C通信でデータを交信し、すべてのセンサの値はメインマイコンか読み取り、カメラのデータはもう一つのマイコン(Arduino nano)で必要な情報だけをピックアップした後、メインマイコンにデータを送信しています。このようにして私たちのロボットは、各マイコンやセンサがバラバラに動いているのではなく連携して動いています。

C テスト

私たちのチームではこのロボットの他にBluetooth双方向シリアル通信システムを開発しています。このシステムを私たちのロボットに搭載することによってパソコン上ですべてのセンサーの値やカメラのデータ、モーターの動きを確認することができます。これにより得られた走行、センサ、モータ、カメラのデータを分析することによって、プログラムの中の変数やモータなどの誤差修正値を正確に算出することができます。

3 ソフトウェア

A 一般的なソフトウェアアーキテクチャ

私たちのロボットに搭載されているソフトウェアは主に三つの部分に分かれています。一つ目はライントレースのプログラムです。今まではライントレースをするのにラインセンサまたはカラーセンサしか使わなかったのに対し、今回のロボットにはカメラを搭載しているので、ラインセンサ及びカメラのデータを同時に処理できるようになっています。二つ目は、レスキューキットを回収するためのプログラムです。レスキューキットの判別にはカメラを使い、回収にはジャイロセンサを用いてアームの角度を常時監視しながら慎重に回収することができます。三つめは、避難者の回収のプログラムです。レスキューゾーンでは8個のレーザー距離センサとカメラが常時動いており、ロボットの位置を常に把握しています。また3次元の地図を生成することにより、どこにどの種類のボールがあるかを常に把握しているため、効率的に被災者及び死者を回収することができます。

B 革新的なソリューション

B-1 メカナムホイールの運動方程式を応用したモーター制御関数

私たちのロボットのホイールはメカナムホイールを採用しています。メカナムホイールの詳細や、採用理由については「ハードウェア」で説明します。このメカナムホイールは前を向いたまま横方向に移動でき、超信地旋回を行えるといった戦車のような機動ができる特性を持っており、これを応用することによって全方向に移動することができます。この特徴を活用するために私たちはメカナムホイールの運動方程式を算出しました。この方程式に動きたい方向のベクトルや、回りたい角度を代入することで、直線上を進みながら90度回転するというような動きをすることができます。

B-2 ラインセンサとカメラによるラインの読み取り

これまでのライントレースで使うセンサはラインセンサだけでした。ラインセンサでライントレースを正確に行うため、僕たちのチームは今までPID制御を用いていましたが、ゲイン値の調整などがうまくいかず、まともにライントレースができていたとしても精度はあまりいいものではありませんでした。PID制御の場合、ゲイン値の調整を正確に行わなければ、ラインの未来位置の予測は困難であるため、僕たちはカメラを用いてラインの未来位置を予測するのではなく、未来位置を未来ではないようにする、つまりロボットがライントレースする前にトレースするラインをカメラで検出すればいいのではないかと考えました。このようにすれば不正確なデータでモーターを動かさなくて済むようになります。

B-1で示した通りメカナムホイールを制御するためには、運動方程式に動きたい方向ベクトルを代入しなければなりません。そのために、25個のラインセンサのデータをそれぞれ分析するのでは方向ベクトルを出すのにとても複雑なアルゴリズムが必要になるため、25個のデータを最小二乗法を用いてラインを一次関数に変換し、方向ベクトルに直しています。しかし、最小二乗法で正確な値が出るときは、ロボットが交差点や鋭角のラインの上に位置していないときであり、最小二乗法は実質、数値の平均であるため、ラインが複数本検知されていると不正確な値が出力されます。カメラは、２本や３本に分かれているラインも識別でき、それぞれ方向ベクトルとして出力できるためロボットが直線や曲線のライン上に位置していない間は、カメラのデータから得られた方向ベクトルでモーターを制御します。このようにしてラインが直線や曲線の時はラインセンサとカメラ、ラインが多数存在するときにはカメラだけというような使い方をすることによって正確なライントレースをすることができます。

B-3 レーザー距離センサによる効率的な被災者の回収

私たちのロボットには、レーザー距離センサが８個搭載されています。ロボットはレスキューゾーンに入った後、一回転します。これは一回転することによって8個の距離センサで避難ゾーン、被災者、ロボットの位置を極座標を用いて三次元の地図に目標をプロットするためです。しかし、被災者と被災者が重なっているときは同じ目標として捉えられてしまうため、最低でも2回回転して地図に目標をプロットするようにしています。ロボットは、目標を回収するときは壁にボールを当てないと回収できないため、ロボット自身の位置を知ることが重要になってきます。この距離センサの分解能は1㎜なのですが、1㎜単位の地図を作ってしまうとオーバーフローしてしまうので1cm単位の地図でロボットの位置を管理しています。また、どのボールから回収すれば一番効率がいいのかという問題は、ボールの数はそこまで多いわけではないので、すべての経路の移動距離を算出し、各ルートの移動時間を分析することによって、最短ルートを通って回収できるようにしています。このようにして効率的に被災者の回収をすることができます。

B-4 プログラム実行時間の最小化

Arduino系列のマイコンをプログラムするにあたって、ArduinoIDEが指定している関数、つまりIOピンなどの入出力関数は、ArduinoのAPIを見る限りプログラムの実行時間を短くするようには作られていません。ラインセンサを1サイクル動かすにはIOピンの入出力関数を10回以上使用することになりますが、この入出力関数は私たちが期待するほど高速には動かないので、時間当たりに何サイクル動かせられるかが重要になるラインセンサのプログラムにはあまり向いていないと私たちのチームは考えました。そこで、Arduinoのレジスタを直接操作しIOピンの入出力を行うことで既存の関数よりもプログラムの実行時間を短縮することができ、時間当たりで実行できるプログラムの回数を増やすことができます。しかしレジスタのピンの番号はマイコンに搭載されているチップの種類によって異なるため、他のマイコンでレジスタを直接操作しているプログラムが動かせなくなってしまうというデメリットも存在します。このようにマイコンのレジスタを直接操作することによってプログラムの実行時間を最小化することができます。

4 ハードウェア

A ハードウェア設計の概要

私たちのロボットのハードウェア設計は「プロジェクトの計画」で示したように「大きな衝撃を加えても壊れず、メンテナンスがしやすく、軽い」ということを目標にして設計しています。