プログラミング 101

このリソースでは、FIRST® Robotics Competitionのプログラミングの基礎について説明します。C++、Java/Kotlin、LabVIEWに対応しています。

レベル**1:**ロボットを稼働させる

1. プログラミング言語の選択*:Java*、*C++*、または*LabVIEW*

Javaは、高校で広く教えられ、APCS試験で使用されるテキスト言語です。これは、独自の仮想環境で実行される「安全な」言語です。FIRST Robotics Competition全体には影響しませんが、この仮想環境はJVMとも呼ばれ、Javaプログラムは計算負荷の高いタスクに使用すると、コンパイル型言語よりも著しく遅くなります。Javaは、その使いやすさと相互互換性のためによく選択されます。Java を使用するチームには、254、125、503、4911、 1241 などがあります。

C++ は、高速なテキストプログラミング言語です。そのパワーと効率により、業界ではリアルタイムシステムとして使用されていますが、学習曲線はJavaよりもはるかに急です。C++ は C プログラミング言語から発展したもので、歴史的な機能と最新の機能が混在すると、構文がわかりにくくなったり、予期しない動作が発生したりすることがあります。FIRST Robotics Competitionのチームは、主にその速度、柔軟性、および広範な数学ライブラリのためにこの言語を使用しています。C++ を使用するチームには、971 と 1678 があります。

LabVIEWは、ナショナル インターナショナルが開発したグラフィカルデータフロープログラミング言語です。エンジニアや技術者が使用するための機器(NI)。LEGO WeDo、FIRST LEGO League Jr、FIRST LEGO Leagueで使用されているMindstorms言語は、LabVIEWの派生物です。そのため、これらのプログラムの受講生はLabVIEWに馴染み深いと感じるかもしれません。LabVIEWダイアグラムでは、コードを並行して実行するなど、高度なコンピューティング機能を非常に簡単に活用できます。このような機能は強力ですが、多くの場合、対処すべき新たな問題を引き起こします。ただし、NIでは広範なデバッグツールを提供しています。LabVIEWの環境と言語には、独自の学習曲線と独自の課題があります。FIRST Robotics Competitionチームは、その簡素化されたグラフィカル構文と広範なエンジニアリングライブラリのために、主にこの言語を使用しています。LabVIEWを使用するチームには、33、359、624、1986、2468などがあります。

どの言語を選択するかは、チームにとって最も簡単な言語によって常に異なります。たとえば、プログラミングコンサルタントがその言語の専門家であるため、言語を選択することは理にかなっていることがよくあります。一方、特定の言語の学習のしやすさに基づいて選択することも理にかなっている場合があります。何を決定するにしても、プログラミング言語の選択は作業環境とチームの人々に固有のものであることを忘れないでください。すべての言語は、FIRST Robotics Competitionで使用するのに十分な能力とサポート、および強力です。

2. プログラミング言語を教える

- FIRST Robotics Competitionでプログラミングを教えるとき、教える必要のある科目は2つあります。1つ目はプログラミング言語独自のセマンティクスと構文であり、2つ目はFIRST Robotics Competitionのコンポーネントとのインターフェースです。C++言語学習ガイド 従って、Javaへ 従って およびLabVIEW 従って 達する。

3. スターターコードを選択します

- JavaとC ++の場合、Botとのインターフェースに使用できる4つの異なる「クラス」があります。これらのクラスの比較とその意味 については、こちらをご覧ください。
- LabVIEWの場合、スターターテンプレートには、スケジュール、反復、およびスポーン/キャンセルメカニズムが混在しています。テンプレートについては、こちらで説明します。

4. チームがプログラミング言語を選択し、コーディングを開始したら、***FIRST Robotics Competition Docs***サイトは、開発環境の設定方法とロボットへのコードのアップロード方法に関する優れたリソースです。これらのガイドは、***FIRST Robotics Competition***のプログラミングに非常に役立ちます。

- はじめ
- C++\Java
- LabVIEWの

5. ロボットにコードをアップロード*!*

- Java と C++

-gradleRIOを使用している場合は、VS Codeに「./gradlew deploy」と入力するだけです。コンソールとコードはロボット上にあります。

- しかし、待ってください!コードはまだ何もしていません。ドライバー コードの簡単な例については、こちらを参照してください。スニペット内のコードは Robot.java または Robot.cpp に属しており、これらは gradle/Eclipse プロジェクトで自動的に生成する必要があります。

- LabVIEWの
  - 開発の場合は、ここに示すようにソースから実行する必要があります。
  - 完了したら、ここに示すように作成された実行可能ファイルをデプロイします。

6. メカニズムのコード

- JavaおよびC ++の場合、ここに単純なドライバーコード をコピーして貼り付けることができます 。
- LabVIEWの場合、TeleOp VIのテンプレートでシンプルなドライバコードを使用できます。
- ほとんどのFIRST Robotics Competitionロボットは、ドライブトレインの外部で作動するメカニズムを備えています。これは、回転するフライホイールから空気圧カタパルトまで、何でもかまいません。これらのメカニズムはすべて、自律的または遠隔的に制御できなければなりません。PWM上でスピードコントローラを使用してメカニズムを制御するには、 C++とJavaのガイドがこちらにあり、LabVIEWのガイドがあります。
- CAN経由でスピードコントローラーを使用している場合は、こちらのガイドに従うか、こちらのドキュメントにリンクされているPhoenix APIを使用して 、PWMスピードコントローラーとして扱う必要があります。

7. 自律型

- Java および C++ プログラミングで自律的なアクションを実行する方法に関するガイド は 、こちらにあります。
- Team 1619は、Javaでオートラインを横断するための簡単なコードもまとめ ています。
- LabVIEWテンプレートには、ロボットを所定の位置に振る自律型コードが含まれています。エンジン出力の値とタイミングを変更して、多くのタスクを実行できます。これは、2018 年の 2468 の自律コードの例です。

第 2 レベル: カスタム アーキテクチャと閉ループ モーター制御

1. カスタムアーキテクチャの使用

- 多くの場合、既存のロボットのクラスでは不十分です。たとえば、定期的にテレオペを行い、自律的に順番に実行することができます。その場合は、おそらくカスタムアーキテクチャに移行する時期です。

- カスタムアーキテクチャは、基本的に、カスタマイズされた方法ですべてのコードを構成することです。

- 特定のアーキテクチャの例としては、ここに含まれている 1678 のコードがあります。1678 のコードは、ここでは 971 のコードに基づいています。

●254はまた、特別なアーキテクチャを持っています。彼らの2019年のコードはここにあります。

- 33、624 、 1986 、2468

*2. PID*制御

- PID制御は、電圧ではなく位置に基づいてメカニズムを制御することができます。PIDを使用すると、アームに直接電圧を出力するように指示する代わりに、アームに30度回転するように指示できます。これは、自律システムで特に役立ちます。ロボットに0.5秒間フルパワーではなく5メートルを駆動するように指示できると、再現性が向上します。
- PIDに関する有用なドキュメントを次に示します。
  - ウェスリーのブログ
  - CSIMのダミー用PID

3. Motion Magic(CANのみ)

- TalonSRXスピードコントローラーを使用している場合は、特にレバーやエレベーターなどのメカニズムを制御するために、MotionMagicを使用することをお勧めします。MotionMagicは基本的に、自動的に生成された台形モーションプロファイルをたどる1KHzのPIDループです。これらの言葉が意味をなさなくても、心配しないでください!PIDとここは トランザクションプロファイルを説明するドキュメントがあります。
- Motion Magicのドキュメント はこちらでご覧いただけます 。

第3レベル:アドバンスドライビングパス、MP制御、ユニットテスト

*1.*道路を運転して進みます

- 生のPIDでは、ドライブトレインを自律的に制御できない場合があります。たとえば、ロボットがスイッチを一周して、背面から立方体を拾うようにしたい場合があります。これを行うための巧妙な方法は、走行経路を作成することです。ドライブパスは基本的に、ドライブトレインのPIDループがたどる一連のポイントであり、ポイントは最終目的地につながります。PathWeaver は、PathFinder というライブラリを使用して、このようなパスを作成し、解析可能なファイルに保存するグラフィカル ツールです。PathWeaverの詳細な使用方法については 、こちらをご覧ください。

- ポイントが作成されたら、それらを追跡するためのさまざまな方法があります。これらは、PIDを使用してポイントを直接追跡することから、PIDループにポイントを与える前にポイントを処理するパス追跡アルゴリズムを追加することまで多岐にわたります。このようなパス追従アルゴリズムの例は、こちら(式5.12)にあります。ロードトラッキングに対するその他の一般的なアプローチには、適応型ピュアトラッキング制御があります。アダプティブピュアトラッキングの便利な実装は、254で利用可能であり、ここで見つけることができます。

2. モデルベース制御

- モデルベース制御は、PIDの一歩先を行くものです。これにより、システムの数学的モデルをコードに保持し、モデルをセンサーデータで更新できます。このようなモデルを使用すると、メカニズムの位置、速度、加速度などをはるかに正確に制御できます。モデルベース制御を使用するチームには、1678 や 971 などがあります。
- モデルベースの制御を学習するための有用なリソースには、次のようなものがあります。
  - ウェスリーのブログ
  - このMITパンフレット

3. ユニットテスト

- ほとんどの場合、コードをロボットにデプロイする前にテストする必要があります。これにより、大惨事を防ぐことができます。ユニットテストは、コードの一部をスタンドアロンプログラムとしてテストするために使用される用語です。たとえば、エレベーターを動作させるコードの部分はテストしたいが、いくつかのライトを点滅させる部分はテストしないとします。FIRST Robotics Competitionのメカニズムテストは、モデルベース制御によって大幅に強化され、モデルをメカニズムのシミュレーションとして使用できるため、メカニズム全体を信じられないほどのロバスト性でテストできます。いくつかの便利な単体テストライブラリは次のとおりです。
- GoogleTestの

▽**Compass Alliance**について

Compass Allianceは、FIRST Robotics Competitionチームの存続と成長を支援するために、世界中の10チームによって設立されました。成長を続けるリソースリポジトリと24/7コールセンターは、世界中のどこからでも何か新しいことを学び、より多くのことを学ぶためのツールをあらゆるスキルレベルの人に提供します。メンターがいないリモートチームは、シーズンを通してリモートでガイドするタッグチームにサインアップでき、ヘルプセンターは他のFIRSTチームが提供するローカルサービスへのアクセス場所を決定します。Hear For Youは、チームやボランティアがチームやイベントでメンタルヘルスを改善するのに役立つリソースとツールを提供します。The Compass Alliance の詳細、質の高いヘルプの検索、 www.thecompassalliance.org への参加が可能です。

このリソースについて

このリソースは、FIRSTの支援と概要を受けて、Compass Allianceによって作成されました。このリソースについて質問がある場合は、 thecompassalliance@gmail.com または firstroboticscompetition@firstinspires.org にお問い合わせください。

改訂履歴

改定#	改定日	リビジョンノート
1.0	壊す。2018	初期リリース