Programação 101

Este recurso abordará os fundamentos da programação na FIRST® Robotics Competition. Abrange C++, Java/Kotlin e LabVIEW.

Nível um: Coloque seu robô em funcionamento

1. Escolha de uma linguagem de programação: Java, C++ ou LabVIEW

Java é uma linguagem textual que é amplamente ensinada em escolas de ensino médio e usada em exames AP CS. É uma linguagem "segura" que roda em seu próprio ambiente virtual. Embora não afete a FIRST Robotics Competition em geral, este ambiente virtual, também conhecido como JVM, significa que os programas Java são visivelmente mais lentos do que as linguagens compiladas quando usados para tarefas computacionalmente intensivas. Java é frequentemente escolhido por sua facilidade de uso e compatibilidade cruzada. As equipes que usam Java incluem 254, 125, 503, 4911 e 1241.

C++ é uma linguagem de programação textual rápida. É usado na indústria para sistemas em tempo real devido à sua potência e eficiência, mas a curva de aprendizagem é muito mais íngreme do que o Java. C++ evoluiu a partir da linguagem de programação C, e a mistura de recursos históricos e modernos às vezes leva a sintaxe confusa e/ou comportamento inesperado. As equipas da FIRST Robotics Competition usam esta linguagem principalmente devido à sua velocidade, flexibilidade e extensas bibliotecas matemáticas. As equipes que usam C++ incluem 971 e 1678 .

O LabVIEW é uma linguagem de programação gráfica de fluxo de dados desenvolvida pela National. Instrumentos (NI) para uso por engenheiros e técnicos. As linguagens Mindstorms usadas para LEGO WeDo e FIRST LEGO League Jr e FIRST LEGO League são derivadas do LabVIEW; assim, os alunos desses programas podem achar o LabVIEW familiar. Em um diagrama do LabVIEW, é muito fácil tirar proveito de recursos avançados de computação, como executar partes de código em paralelo. Embora poderosos, esses recursos muitas vezes representam novos problemas para lidar. No entanto, a NI fornece

extensas ferramentas de depuração. O ambiente e a linguagem do LabVIEW vêm com seu próprio conjunto de curvas de aprendizado e desafios únicos. As equipas da FIRST Robotics Competition utilizam principalmente esta linguagem devido à sua sintaxe gráfica simplificada e extensas bibliotecas de engenharia. As equipes que usam o LabVIEW incluem 33, 359, 624, 1986 e 2468.

O idioma que você escolhe sempre depende do que é mais fácil para sua equipe. Por exemplo, muitas vezes pode fazer sentido escolher uma linguagem porque um consultor de programação é um especialista nessa linguagem. Por outro lado, também pode fazer sentido escolher com base na facilidade de aprender uma determinada língua. Seja qual for a sua decisão, lembre-se que a escolha da linguagem de programação é específica para o ambiente de trabalho e para as pessoas da sua equipa. Todas as linguagens são capazes, bem suportadas e poderosas o suficiente para serem usadas na FIRST Robotics Competition.

2. Ensino da Linguagem de Programação

- Ao ensinar programação para a FIRST Robotics Competition, há duas disciplinas diferentes que precisam ser ensinadas. O primeiro é a semântica e sintaxe da própria linguagem de programação, e o segundo é a interface com os componentes da FIRST Robotics Competition. Um guia para aprender a linguagem C++ daí, para Java daí e LabVIEW daí Alcance.

3. Escolha o seu código inicial

- Para Java e C++, existem quatro "classes" diferentes que podem ser usadas na interface com o Bot. Uma comparação dessas classes e o que elas significam pode ser encontrada aqui.
- Para o LabVIEW, os modelos iniciais incluem uma combinação de mecanismos programados, iterativos e de desova/cancelamento. Os modelos estão descritos aqui.

4. Uma vez que sua equipe escolheu uma linguagem de programação e começou a codificar, o site FIRST Robotics Competition Docs é um bom recurso sobre como configurar seu ambiente de desenvolvimento e fazer upload de código para seu robô. Estes guias são inestimáveis para a programação da FIRST Robotics Competition.

- Primeiros passos

- C++\Java
- LabVIEW

5. ***Carregue o código para o seu robô!***

- Java e C++

  Se você estiver usando -gradleRIO, basta digitar "./gradlew deploy" no VS Code. O console e seu código estarão no robô.

  — Mas espere! Seu código ainda não está fazendo nada. Alguns exemplos simples de código de driver podem ser encontrados aqui. O código em trechos pertence a Robot.java ou Robot.cpp, que devem ser gerados automaticamente com o projeto gradle/Eclipse.

- LabVIEW

  - Para o desenvolvimento, você deve executá-lo a partir do código-fonte, como mostrado aqui.

  - Uma vez feito, você implantará um executável que foi criado como mostrado aqui.

6. ***Códigos dos mecanismos***

- Para Java e C++, o código de driver simples pode ser copiado e colado aqui.
- Para o LabVIEW, o código do driver simples está disponível no modelo em TeleOp VI.
- A maioria dos robôs FIRST Robotics Competition tem mecanismos que são acionados fora do conjunto propulsor. Pode ser qualquer coisa, desde um volante giratório até uma catapulta pneumática. Todos estes mecanismos devem poder ser controlados autonomamente ou teleopicamente. Para controlar mecanismos usando controladores de velocidade sobre PWM, há um guia aqui para C++ e Java, e aqui está um guia para o LabVIEW.
- Se você estiver usando controladores de velocidade sobre CAN, você deve seguir o guia aqui ou usar a API Phoenix, cuja documentação está vinculada aqui, para tratá-los como controladores de velocidade PWM.

7. Autónomo

- Um guia sobre como executar ações autônomas em programação Java e C++ pode ser encontrado aqui.
- Team 1619 também compilou alguns códigos simples para cruzar a autoline em Java, que você pode encontrar aqui.
- Os modelos do LabVIEW contêm código autônomo para agitar o robô no lugar. Você pode alterar os valores de potência do motor e o tempo para executar muitas tarefas. Aqui está um exemplo do código autônomo de 2468 de 2018.

Segundo Nível: Arquitetura Personalizada e Controle de Motor de Circuito Fechado

1. Usando uma arquitetura personalizada

- Muitas vezes, as classes de robôs existentes não são suficientes. Por exemplo, você pode querer teleop periodicamente e executar autonomamente sequencialmente. Se esse for o caso, provavelmente é hora de passar para uma arquitetura personalizada.
- A arquitetura personalizada é essencialmente a configuração de todo o código de forma personalizada.
- Alguns exemplos de arquitetura específica incluem o código de 1678, que está incluído aqui. O código de 1678 é baseado no código de 971 aqui.
- O 254 também tem uma arquitetura especial. O código de 2019 pode ser consultado aqui.
- 33, 624 e 1986 e 2468

2. Controlo PID

- PID Control permite controlar um mecanismo com base na posição em vez de tensão. Usando PID, você pode dizer a um braço para girar 30 graus em vez de dizer-lhe diretamente para emitir uma tensão. Isto é especialmente útil em sistemas autónomos. Ser capaz de dizer a um robô para conduzir 5 metros em vez da potência total durante 0,5 segundos proporciona uma repetibilidade melhorada.
- Aqui estão alguns documentos úteis para PID:
  - Blog do Wesley
  - PID da CSIM para Dummies

3. Magia de movimento (apenas CAN)

- Se você estiver usando um controlador de velocidade TalonSRX, é recomendável que você use o MotionMagic, especialmente para controlar mecanismos como alavancas ou elevadores. O MotionMagic é essencialmente um loop PID de 1KHz que segue perfis de movimento trapezoidal gerados automaticamente. Se estas palavras não fizerem sentido, não se preocupe! Ver acima para informações sobre PID e aqui Há um documento que descreve os perfis de transação.

- A documentação para Motion Magic pode ser encontrada aqui.

Terceiro Nível: Percursos de Condução Avançados, Controlo MP e Testes Unitários

1. Dirija estradas e siga-as

- Às vezes, o PID bruto não é suficiente para controlar autonomamente o conjunto propulsor. Por exemplo, você pode querer que o robô contorne o interruptor e pegue um cubo na parte de trás. Uma boa maneira de fazer isso é criar um caminho de condução. O caminho da unidade é essencialmente uma série de pontos que o loop PID do trem de força seguirá, e os pontos levarão ao destino final. O PathWeaver é uma ferramenta gráfica que usa uma biblioteca chamada PathFinder que cria esses caminhos e os salva em um arquivo analisável. Instruções detalhadas sobre como usar o PathWeaver podem ser encontradas aqui.

- Uma vez que os pontos são criados, existem várias maneiras de acompanhá-los. Isso vai desde o uso de PID para rastrear pontos diretamente até a adição de um algoritmo de rastreamento de caminho para processar pontos antes de fornecê-los ao loop PID. Um exemplo de tal algoritmo de seguimento de caminho pode ser encontrado aqui (eq. 5.12). Outras abordagens populares para o rastreamento rodoviário incluem o controle de rastreamento puro adaptativo. Uma implementação prática de rastreamento puro adaptativo está disponível no 254 e pode ser encontrada aqui.

2. Controlo baseado em modelos

- O controle baseado em modelos é um passo além do PID. Ele permite que um modelo matemático do sistema seja mantido em código e o modelo seja atualizado com dados do sensor. Usando tal modelo, a posição, velocidade, aceleração, etc., de um mecanismo pode ser controlada com muito mais precisão. Algumas equipes que usam controle baseado em modelo incluem 1678 e 971.
- Recursos úteis para o controle baseado em modelos de aprendizagem incluem:
  - Blog do Wesley
  - Esta brochura do MIT

3. Ensaios unitários

- Na maioria das vezes, você deseja testar seu código antes de implantá-lo no robô. Isso poderia evitar uma catástrofe. Teste de unidade é um termo usado para testar partes do código como programas autônomos. Por exemplo, você pode querer testar a parte do código que faz o elevador funcionar, mas não a parte que faz algumas luzes piscarem. O teste de mecanismos para a FIRST Robotics Competition é muito melhorado pelo controle baseado em modelos, pois o modelo pode ser usado como uma simulação do mecanismo, o que significa que todo o mecanismo pode ser testado com incrível robustez. Algumas bibliotecas de teste de unidade úteis são:
- GoogleTeste

Sobre a Compass Alliance

A Compass Alliance foi fundada por 10 equipas de todo o mundo para ajudar as equipas da FIRST Robotics Competition a sobreviver e a crescer. Um repositório de recursos em crescimento e um call center 24 horas por dia, 7 dias por semana, fornecem a qualquer pessoa de qualquer nível de habilidade as ferramentas para aprender algo novo ou aprender mais em qualquer lugar do mundo. As equipas remotas sem um mentor podem inscrever-se numa Tag Team que irá orientar remotamente durante toda a temporada, e os Centros de Ajuda determinam onde aceder aos serviços locais oferecidos por outras equipas FIRST. A Hear For You fornece recursos e ferramentas para ajudar equipas e voluntários a melhorar a sua saúde mental nas suas equipas e em eventos. Você pode saber mais sobre a The Compass Alliance, encontrar ajuda de qualidade e participar da www.thecompassalliance.org.

Sobre este recurso

Este recurso foi produzido pela Compass Alliance com o apoio e visão geral da FIRST. Se você tiver dúvidas sobre este recurso, entre em contato com thecompassalliance@gmail.com ou firstroboticscompetition@firstinspires.org.

Histórico de Revisões

Revisão #	Data de revisão	Notas de revisão
1.0	Intervalo. 2018	Versão inicial