Trabalho Controle Remoto do robô

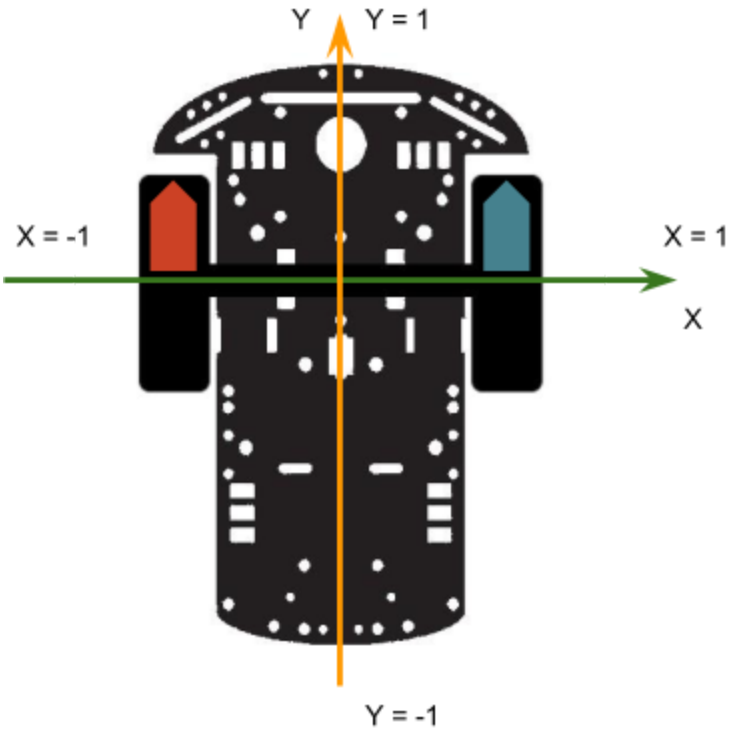
Descrição:

Utilizar da atividade anterior e implementar o suporte ao controle do robô com o protocolo criado.

O trabalho deve ser feito em duplas, aonde um dos integrantes deve criar um pequeno resumo sobre como foi criar a aplicação e submeter o mesmo no moodle e o outro integrante deve enviar o arquivo zipado gerado pelo github, contendo a aplicação criada pela equipe, é importante que o arquivo seja gerado através da plataforma do github.

Especificação:

A aplicação deve se utilizar do protocolo informado pelo Exercício 03 e mapear as posições ditas antes reservadas que são as posições de 0 a 40, para as funções de movimentação do robô utilizando o código base de movimentação do robô.



Primeiro suponha que exista um vetor de um tamanho qualquer e esse vetor se chama MOV.

MOV[0] / MOV[1] - Para as posições 0 e 1 teremos as direções das rodas, sendo MOV[0] para direção da roda esquerda e MOV[1] para direção da roda direita, 1 indica que a roda deve rotacionar para frente e -1 para trás, sendo que 0 indica freio.

MOV[2] / MOV[3] - Para o controle de velocidade deve-se usar a mesma ideia da direção, a velocidade máxima para os dois casos é de 255. A posição 2 representa a roda esquerda do robô e a posição 3 representa a roda direita do robô.

MOV[4] {Opcional} - Define a distância de movimento.

MOV[5] - MOV[10] {Reservado} - Sensores de refretância

Exemplo:

OBS.: Códigos começados com # são apenas comentários do exemplo.

Ande para frente por 2 segundos:

```
# Direção roda esquerda, para frente
S0=1
# Direção roda direita, para frente
S1=1
# Velocidade é a mesma em ambas as rodas
S2=255
S3=255
# Espera-se 2 segundos ...
# Ativa o freio das rodas e para o robô
S2=0
S3=0
```

Observações:

A aplicação final preferencialmente deve ser uma aplicação de terminal que lide com as mensagens no formato especificado na descrição deste trabalho, porém não se restringe a isso, podendo também ser uma aplicação com interface gráfica, desde que ela siga os seguintes critérios que também vale para a aplicação de terminal:

- Seja implementada utilizado alguma das seguintes linguagens:
    - C/C++
    - C# (Utilizando o Mono)
    - Ruby
    - Python
    - Java
    - Javascript (Utilizando o Node.js)
    - Processing
- A aplicação final deve ser multiplataforma, não devendo ter erros de execução no Linux
- É obrigatório o uso do Git e da plataforma GitHub para disponibilização do código final
- É obrigatório que quaisquer adições na linguagem devem ser documentadas
- É obrigatório que sejam utilizadas ferramentas de gerenciamento de dependências/compilação e.g.:
    - GCC Make (C/C++)
    - XBuild (C# com o Mono)
    - Bundler (Ruby)
    - VirtualEnv + PIP (Python)
    - Maven (Java)
    - Ant (Java)
    - NPM (Javascript com o Node.js)
    - Processing IDE (Modelo de projeto do Processing, com o header definido)
    - PlatformIO (Código Arduino)
- Para a aplicação do arduino deve ser providenciado um header bem descritivo com o formato na pa ǵ ina a seguir
- As aplicações para Arduino podem ser feitas também no PlatformIO
- Deve ser utilizado o modelo de repositório, ver o link para mais informações

O TRABALHO DEVE FUNCIONAR DE ALGUMA FORMA NO LABORATÓRIO

Modelo de cabeçalho de Projetos Arduino e Processing:

Aplicações feitas utilizando a IDE do Processing ou Arduino, não tem ferramentas de gerenciamento de dependências por essa razão, é necessário que os projetos que utilizem essa ferramenta possuam um cabeçalho bem definido a fim de não haver problemas de replicação, este cabeçalho deve ser utilizado no arquivo principal do projeto do Arduino/Processing, não serão aceitos trabalhos com dependências que precisam ser baixadas de repositórios e instaladas manualmente, só serão aceitas dependências registradas na interface do Arduino através do **LibraryManager**. Não esquecer de usar o comando CTRL+T para formatar o código a ser entregue ;)

```
/*
  Project Name: NomeDoProjeto
  Authors:
    * Fulano de Tal
    * Cicrano de Alguma Coisa
  Version: 1.0.0
  Boards Supported:
    * Arduino *
    * NodeMCU
  Dependencies:
    * NomeDaDependencia1, 1.0.0
    * NomeDaDependencia2, 2.0
  Description:
    Descrição do projeto ..........................................
  Continuação da descrição do projeto.
*/
```