07

Finalizando e rodando o nosso projeto

Transcrição

Ainda há um ajuste a ser feitos. Dentro da pasta **mjpg-streamer** há uma subpasta, a **mjpg-streamer-experimental**, que é onde fica o código do programa. Para não ter que ficar entrando nesse subdiretório para acessar o programa, vamos mover o seu conteúdo para a pasta **mjpg-streamer**, executando os seguintes comandos:

```
cd ~/mjpg-streamer/mjpg-streamer-experimental/
mv * ../
```

Agora todo código do programa fica na pasta **/home/pi/mjpg-streamer**.

Analisando os arquivos do projeto

Dentro da pasta **/home/pi/pibot**, vamos analisar alguns dos novos arquivos que foram baixados, para entender como iremos fazer o Flask rodar. No arquivo **pibot-web.py**, logo no começo importamos alguns módulos, dentre eles o módulo do Flask, e também o template, que irá montar a nossa página de controle.

Do programa `distancia.py` , importamos as suas funções:

```
from distancia import setup_sensor, roda_medicao, get_distancia
```

Fazemos o mesmo com o programa `controle.py` :

```
from controle import setup_motor, move_frente, move_direita, move_esquerda, move_tras, move_parar
```

A primeira coisa que fazemos é realizar o *setup* do sensor e do motor, dentre outras coisas, com o seguinte código:

```
@app.before_first_request
def _run_on_start():
    setup_sensor()
    setup_motor()
    t = Thread(target=roda_medicao)
    t.start()
```

O restante do arquivo já foi detalhado no primeiro vídeo deste capítulo. Vamos agora ver o template, a página **pibot/templates/form.html**:

Aqui temos o HTML da nossa página de controle. Vale destacar o seguinte trecho de código:

```
<div class="container">
    <img src="http://192.168.1.60:8080/?action=stream">
    <p>
        Distância: <span id="distancia">0</span> cm
```

```
    </p>
</div>
```

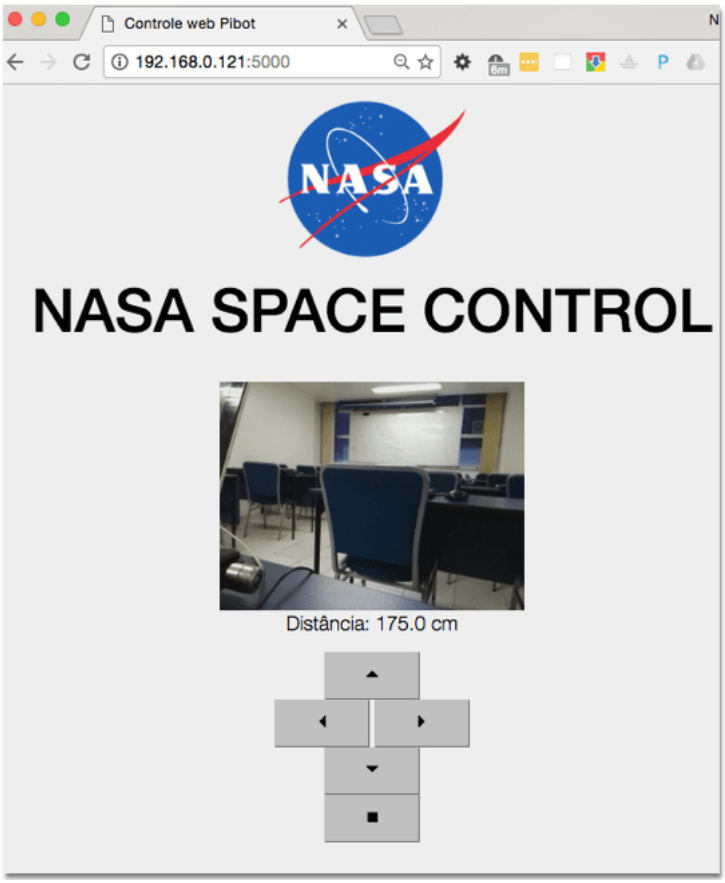
É preciso ter atenção no atributo `src` da tag `<img>`. Você deve colocar o IP do seu Raspberry Pi, no caso a página já vem com o IP **192.168.1.60**, altere de acordo com o seu. Essa URL é importante, pois é a URL que disponibilizará dentro da nossa página o *streaming* da câmera do carrinho.

No mais, há o HTML dos botões, cada um com os atributos `id` e `value` que representam a sua tecla. Para estilizar a página, utilizamos o Bootstrap, além de um CSS customizado, chamado **estilos.css**. Também utilizamos o jQuery e um arquivo JavaScript customizado, o **main.js**. Ele se encontra na pasta **pibot/static/js** e as suas funcionalidades estão bem comentadas e descritas no arquivo.

Finalmente, podemos rodar o nosso projeto. Temos um script para isso, o **pibot/pibot.sh**. Esse arquivo sobe o **mjpg-streamer** e executa o **pibot-web.py**. Podemos rodá-lo com os seguintes comandos:

```
cd ~/pibot
./pibot.sh
```

Agora abrimos a página na seguinte URL [http://IP-DO-RASPBERRY-PI:5000 (http://IP-DO-RASPBERRY-PI:5000)](http://IP-DO-RASPBERRY-PI:5000):



Vemos a nossa página plenamente funcional. Temos acesso ao *streaming* da câmera e podemos mover o carrinho clicando nos botões! Com isso temos acesso e controle totalmente do carrinho, tudo concentrado em apenas uma página.

Esperamos que vocês tenham gostado do projeto. Ele é um pouco complexo, dá um pouco de trabalho na hora de montar, mas o resultado é bem legal e satisfatório!