09

Sorteando posição dos servos

Transcrição

Logo abaixo do que já foi feito, criaremos algumas funções auxiliares. A primeira delas é `posiciona_motores()`, responsável por posicionar os motores:

```
// FUNÇÕES AUXILIARES

void posiciona_motores() {

}
```

Para de fato posicionarmos o motor, realizaremos um cálculo que determina sua próxima posição. Para isso, utilizaremos a função `random()` que sorteia um número aleatório entre dois valores. Os valores que utilizaremos são `0` (zero) e o número do intervalo do eixo, ou seja, `12` e `4` para X e Y, respectivamente.

Depois que o número for sorteado, multiplicaremos o mesmo por `10` e adicionaremos o valor mínimo ao resultado. Assim, para X e Y teremos:

```
int posicaoX = (random(0, (X_INTERVALO)) * 10 + X_MINIMO);
int posicaoY = (random(0, (Y_INTERVALO)) * 10 + Y_MINIMO);
```

Mas por que dividimos antes por `10` e agora estamos fazendo o inverso? Isso está relacionado ao intervalo das movimentações que desejamos ter. Quando usamos a função `random`, será sorteado um número aleatório entre o intervalo indicado, mas se definirmos um intervalo grande, talvez, a variação da movimentação do motor fique ruim.

Por exemplo, se indicássemos que os intervalos será entre `30` e `150` para o `random`, teríamos como resultado variações como `40`, `41`, `45` que são números muito próximos. Quando dividimos e multiplicamos o intervalo por `10`, aumentaremos o intervalo das variações por `10`, pelo menos. Vejamos a simulação abaixo:

Para a **primeira posição**:

1. `random` sorteia 4;
2. Multiplicamos por 10 e temos 40;
3. adicionamos 30 e temos 70 como posição final.

Para a **segunda posição**:

1. `random` sorteia 5;
2. Multiplicamos por 10 e temos 50;
3. adicionamos 30 e temos 80 como posição final.

Com a posição calculada, precisamos apesar escrever essa posição no servo motor. Nossa função de posicionamento completa fica da seguinte forma:

```
// FUNÇÕES AUXILIARES

void posiciona_motores() {
    int posicaoX = (random(0, (X_INTERVALO)) * 10 + X_MINIMO);
    int posicaoY = (random(0, (Y_INTERVALO)) * 10 + Y_MINIMO);
    servoX.write(posicaoX);
    servoY.write(posicaoY);
    delay(2000);
}
```

Note que adicionamos um `delay` de `2000` apenas para o uso em testes. Com ele, teremos uma pausa de `2` segundos, depois de cada posição ter sido configurada. Por último, chamaremos nossa função dentro do bloco `loop` que executará nosso código.

```
void loop() {
    posiciona_motores();
}
```

Vamos testar nosso projeto?