Atividades Práticas e o Software *Robomind*

Atividades Práticas

1.1 Sempre em frente: LEGO NXT

Versão Textual

Após aprendermos a linguagem de programação e conhecermos o software responsável por transferir os programas criados no NXT e EV3, vamos fazer algumas práticas a fim de exercitar o pensamento lógico e familiarizar-se com o programa. As tarefas são simples, mas essenciais para quem está começando a programar e conhecendo o funcionamento dos sensores e motores do KIT.

Devemos fazer o nosso veículo seguir em frente por 0,50 metros com 50% da força. Primeiramente, vamos criar um novo programa. Para a movimentação do nosso veículo, precisamos fazer os servo motores girarem em sentido horário ou anti-horário para que, dessa forma, o veículo ande para frente, para trás ou faça uma curva. Vamos então inserir o bloco responsável pelo acionamento dos motores.

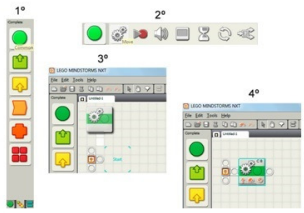


Figura 1: Instruções para inserção do bloco de acionamento dos motores.

Fonte: Autor

Como mostrado na Figura 1, vamos até a palheta de comandos e escolhemos o grupo **Common**, como pode ser visto no primeiro passo. Seguimos para o segundo passo escolhendo o bloco **Move**, responsável por acionar os motores. Após escolhermos o bloco que queremos, precisamos posicioná-lo na sequência correta. Essa é a primeira tarefa que o robô deve realizar, assim, posicionamos o bloco na posição **Start**, como pode ser visto nos terceiro e quarto passos.

Após inserirmos o bloco na posição correta, precisamos especificar como será a movimentação do veículo. Para isso, clicamos com o mouse em cima do bloco **Move** que acabamos de inserir e veremos surgir na parte inferior da janela a palheta de configuração.

Figura 2: Palheta de configuração do bloco **Move**.

Fonte: Autor

Na seção **Port**, podemos selecionar os motores que queremos acionar, e em **Direction**, se o seu giro se dará no sentido horário, anti-horário ou se ele deve parar. Em **Steering**, pode-se distribuir uma porção maior do giro para um dos motores, para assim fazer uma curva. Em **Power**, selecionamos a que potência o motor deve funcionar, e em **Duration**, a duração desse movimento. Na seção **Next Action**, escolhemos a próxima ação do motor ao concluir o seu movimento, podendo parar ou realizar o movimento novamente.

Relembrando, para o nosso primeiro caso, queremos que o motor ande para frente com 50% da força por 0,50 metros. Assim, configuraremos o bloco da seguinte maneira.

Figura 3: Configuração dos motores do bloco **Move**.

Fonte: Autor

Versão Textual

Os motores responsáveis pela movimentação do nosso veículo são os motores A e C, assim, desmarcamos o motor B e marcamos os motores A e C. Como queremos andar para frente, realizaremos o giro no sentido horário. Já que o objetivo é andar em linha reta, distribuímos o giro igualmente entre os motores.

Agora precisamos determinar a duração do movimento do motor. Trabalhar com rotações em movimentos curtos não nos dá muita precisão, assim, vamos mudar a medida da duração do movimento. Temos quatro opções: **Unlimited**, para realizar o movimento sem parar em momento algum; **Degrees**, para trabalhar com a rotação do motor medida em graus; **Rotations**, para contabilizar a quantidade de giros que queremos do motor; e, por fim, **Seconds**, onde definimos o tempo que queremos que o motor fique acionado.

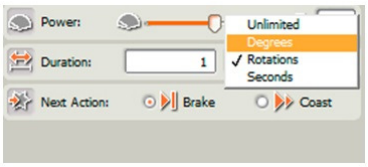


Figura 4: Configuração da medida de duração do movimento do motor.

Fonte: Autor

Vamos mudar então o padrão de medida, que está em rotações, para graus, como mostrado na Figura 5.

Figura 5: Configuração complementar do movimento do motor.

Fonte: Autor

Em **Power**, vamos selecionar a potência do motor em 50, como definido pela tarefa, e em **Duration**, colocamos 1350 graus como a duração do giro do motor. A próxima ação, definimos como parar, visto que não pretendemos que o veículo continue com seu movimento indefinidamente.

Figura 6: Gravação do programa no NXT.

Fonte: Autor

Terminado o programa, vamos passá-lo para o NXT clicando no botão **Download**. Antes de realizar esse passo, conferimos se o NXT está corretamente ligado ao cabo de dados, que também deve estar corretamente ligado à porta USB do computador.

Seguindo os passos descritos, vamos executar o nosso programa no NXT, através do arquivo.

Multimídia

Assista a vídeo "Sempre em frente" utilizando o kit Lego Mindstorms NXT 2.0.

https://www.youtube.com/watch?v=db3N1PYBO1M

Atividades Práticas

1.1.1 LEGO EV3

Para a movimentação do nosso veículo, precisamos fazer os servo motores girarem em sentido horário ou anti-horário para que, dessa forma, o veículo ande para frente, para trás ou faça uma curva. Vamos então inserir o bloco responsável pelo acionamento dos motores.

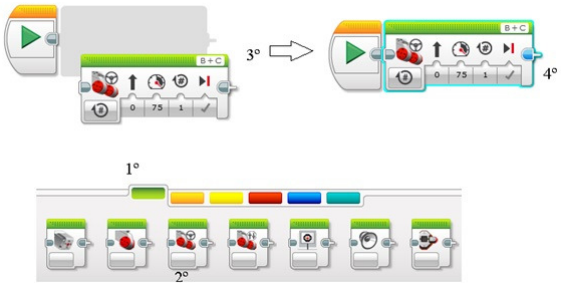


Figura 7: Instruções para inserção do bloco de acionamento dos motores.

Fonte: Autor

Versão Textual

Como mostrado na Figura 7, vamos até a palheta de comandos e escolhemos o grupo **Common**, como pode ser visto no primeiro passo. Seguimos para o segundo passo escolhendo o bloco **Move**, responsável por acionar os motores. Após escolhermos o bloco que queremos, precisamos posicioná-lo na sequência correta. Essa é a primeira tarefa que o robô deve realizar, assim, posicionamos o bloco na posição **Start**, como pode ser visto nos terceiro e quarto passos.

Após inserirmos o bloco na posição correta, precisamos especificar como será a movimentação do veículo. Para isso, clicamos com o mouse em cima do bloco **Move** que acabamos de inserir, e veremos surgir, na parte inferior da janela, a palheta de configuração.

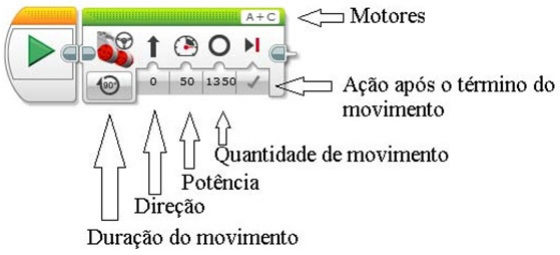


Figura 8: Palheta de configuração do bloco Move.

Na seção **Port**, podemos selecionar os motores que queremos acionar, e em **Direction**, se o seu giro se dará no sentido horário, anti-horário ou se ele deve parar. Em **Steering**, pode-se distribuir uma porção maior do giro para um dos motores para, assim, fazer uma curva. Em **Power**, selecionamos a que potência o motor deve funcionar, e em **Duration**, a duração desse movimento. Na seção **Next Action**, escolhemos a próxima ação do motor ao concluir o seu movimento, podendo parar ou realizar o movimento novamente.

Relembrando, para o nosso primeiro caso, queremos que o motor ande para frente com 50% da força por 0,50 metros. Assim, configuraremos o bloco da seguinte maneira.

Figura 9: Configuração dos motores do bloco **Move**.

Fonte: Autor

Os motores responsáveis pela movimentação do nosso veículo são os motores A e C, assim, desmarcamos o motor B e marcamos os motores A e C. Como queremos andar para frente, realizaremos o giro no sentido horário. Já que o objetivo é andar em linha reta, distribuímos o giro igualmente entre os motores.

Agora precisamos determinar a duração do movimento do motor. Trabalhar com rotações em movimentos curtos não nos dá muita precisão, assim, vamos mudar a medida da duração do movimento. Temos quatro opções: **Unlimited**, para realizar o movimento sem parar em momento algum; **Degrees**, para trabalhar com a rotação do mo-

tor medida em graus; **Rotations**, para contabilizar a quantidade de giros que queremos do motor; e, por fim, **Seconds**, onde definimos o tempo que queremos que o motor fique acionado.

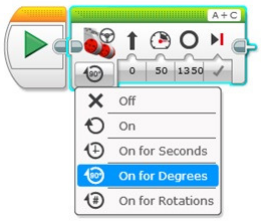


Figura 10: Configuração da medida de duração do movimento do motor.

Fonte: Autor

Vamos mudar então o padrão de medida, que está em rotações, para graus, como mostrado na Figura 11.

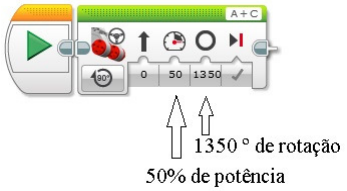


Figura 11: Configuração complementar do movimento do motor.

Fonte: Autor

Em **Power**, vamos selecionar a potência do motor em 50, como definido pela tarefa, e em **Duration**, colocamos 1350 graus como a duração do giro do motor. A próxima ação, definimos como parar, visto que não pretendemos que o veículo continue com seu movimento indefinidamente.

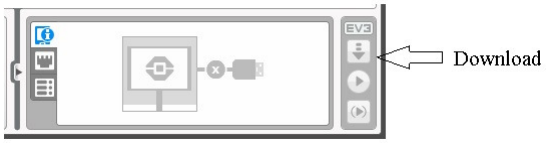


Figura 12: Gravação do programa no EV3

Fonte: Autor

Terminado o programa, vamos passá-lo para o EV3 clicando no botão **Download**. Antes de realizar esse passo, conferimos se o EV3 está corretamente ligado ao cabo de dados, que também deve estar corretamente ligado à porta USB do computador.

Multimídia

Assista a vídeo "Sempre em frente" utilizando o kit Lego EV3.

https://www.youtube.com/watch?v=fOldNkFfz8I

Atividades Práticas

1.2 Vai e Volta – LEGO NXT

Versão Textual

Como já conhecemos o funcionamento dos motores e sabemos calcular a velocidade ou o tempo necessário para ele percorrer um caminho, devemos agora fazer nosso veículo seguir em frente por 0,50 metros e retornar ao ponto inicial de partida com 50% de sua força.

Primeiramente, vamos criar um novo programa e nomeá-lo como **Prática2**.

Para a movimentação inicial do nosso veículo, vamos inserir um bloco para acionamento dos motores. Inserimos então um bloco **Move** na posição **Start**.

Figura 13: Inserção do primeiro bloco de acionamento de motor.

Fonte: Autor

A Figura 13 mostra o passo a passo para inserção do primeiro bloco. Agora, precisamos configurar o funcionamento dos motores. Assim, clicamos com o mouse no primeiro bloco para visualizar a palheta de configuração do bloco **Move**, e configuramos para o funcionamento dos motores A e C, responsáveis pelo movimento do veículo. Adotamos o sentido como sendo horário e distribuímos as rotações igualmente entre os motores a fim de andar para frente e em linha reta. Em **Power**, definimos a força em 50%, e alteramos a medida da duração do movimento para graus. Definimos a próxima ação do motor como **parar**. A palheta deve ficar assim como a mostrada na imagem a seguir.

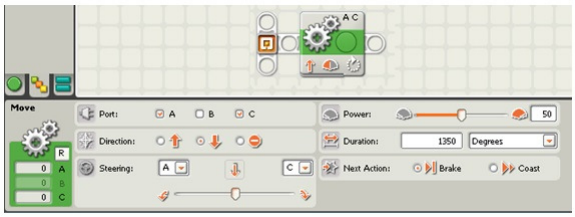


Figura 14: Palheta de configuração do bloco **Move**

Fonte: Autor

⊕ Versão Textual

Aproveitando os dados utilizados na primeira prática, inserir a duração que levou o carrinho a percorrer 0,50 metros com 50% de sua força. Caso não possua esse valor, repeta o processo feito na primeira etapa da prática 1.1 para descobrir.

Terminada essa parte, nosso carrinho está percorrendo 0,50 metros para frente. Precisamos fazê-lo voltar ao ponto de partida. A maneira mais fácil é fazendo-o percorrer o caminho inverso, nesse caso, de ré.

Vamos então adicionar um novo bloco Move inserindo-o após o primeiro. Ao final, temos a seguinte sequência.

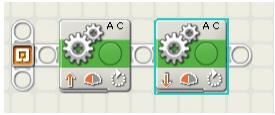


Figura 15: Sequência de blocos **Move**.

Fonte: Autor

Clicando sobre o segundo bloco, vamos fazer os ajustes necessários na palheta de configuração. De início, vamos mudar o sentido de rotação do motor, quanto ao resto das configurações, vamos inserir as mesmas do primeiro bloco.

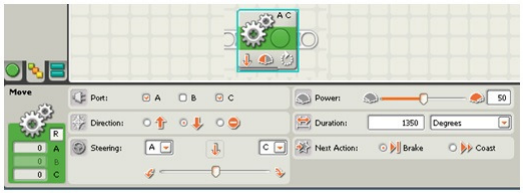


Figura 16: Configuração do segundo bloco **Move**.

Fonte: Autor

Assim, como mostrado na Figura 16, vamos configurar o funcionamento dos motores A e B com os motores girando no sentido anti-horário e as rotações distribuídas igualmente entre os motores, a fim de fazer o veículo an-

dar em linha reta para trás. Em **Power**, definimos a força como sendo 50%, alteramos a medida de duração para funcionamento do motor para graus, e inserirmos o mesmo valor contido no primeiro bloco.

Gravamos então o programa no NXT e, seguindo os passos descritos no tópico 8, executamos o arquivo de nome **Prática2**. Ao fim do movimento, o veículo deve ter sido capaz de percorrer 0,50 metros para frente e 0,50 metros para trás, voltando, assim, para o ponto de partida.

Multimídia

Assista a vídeo "Vai e volta" uutilizando o kit Lego Mindstorms NXT 2.0.

https://www.youtube.com/watch?v=bcmeVP5HGww

Atividades Práticas

1.2.1 LEGO EV3

Para a movimentação inicial do nosso veículo, vamos inserir um bloco para acionamento dos motores. Inserimos então um bloco **Move** na posição **Start**.

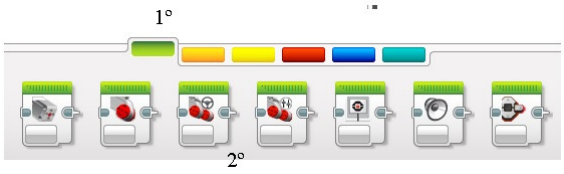


Figura 17: Inserção do primeiro bloco de acionamento de motor.

Fonte: Autor

Versão Textual

A Figura 17 mostra o passo a passo para inserção do primeiro bloco. Agora, precisamos configurar o funcionamento dos motores. Assim, clicamos com o mouse no primeiro bloco para visualizar a palheta de configuração do bloco **Move** e configuramos para o funcionamento dos motores A e C, responsáveis pelo movimento do veículo. Adotamos o sentido como sendo horário e distribuímos as rotações igualmente entre os motores a fim de andar para frente e em linha reta. Em **Power**, definimos a força em 50%, e alteramos a medida da duração do movimento para graus. Definimos a próxima ação do motor como **parar**. A palheta deve ficar assim como a mostrada na imagem a seguir.

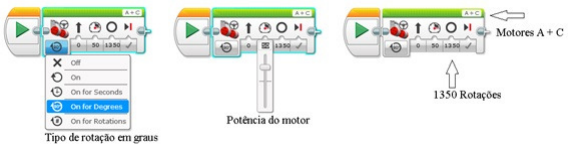


Figura 18: Palheta de configuração do bloco **Move**.

Fonte: Autor

Terminada essa parte, nosso carrinho está percorrendo 0,50 metros para frente. Precisamos fazê-lo voltar ao ponto de partida. A maneira mais fácil é fazendo-o percorrer o caminho inverso, nesse caso, de ré.

Vamos então adicionar um novo bloco **Move** inserindo-o após o primeiro. Ao final, temos a seguinte sequência.

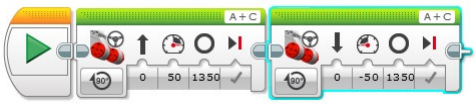


Figura 19: Sequência de blocos **Move**.

Fonte: Autor

Clicando sobre o segundo bloco, vamos fazer os ajustes necessários na palheta de configuração. De início, vamos mudar o sentido de rotação do motor, quanto ao resto das configurações, vamos inserir as mesmas do primeiro bloco

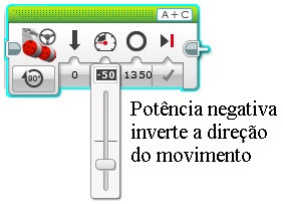


Figura 20: Configuração do segundo bloco **Move**.

Fonte: Autor

Assim, como mostrado na Figura 20, vamos configurar o funcionamento dos motores A e B com os motores girando no sentido anti-horário e as rotações distribuídas igualmente entre os motores a fim de fazer o veículo andar em linha reta para trás. Em **Power**, definimos a força como sendo 50%, alteramos a medida de duração para funcionamento do motor para graus, e inserirmos o mesmo valor contido no primeiro bloco.

Terminado o programa, vamos passá-lo para o EV3, clicando no botão **Download**. Antes de realizar esse passo, conferimos se o EV3 está corretamente ligado ao cabo de dados, que também deve estar corretamente ligado à porta USB do computador.

Multimídia

Assista a vídeo "Sempre em frente" utilizando o kit Lego EV3.

https://www.youtube.com/watch?v=9yU6SGNIxmw

Atividades Práticas

1.3 Pare na faixa

Versão Textual

Iremos conhecer o sensor de luminosidade. Ele identifica cores e linhas claras ou escuras. Nosso objetivo é fazer o veículo seguir em frente e quando encontrar uma linha preta parar imediatamente.

Faremos, pela primeira vez, o uso de críticas e loops. O sensor a ser utilizado vai definir que ação deve ser realizada; como a condição para o carro parar está ligada a uma cor, o uso do sensor de cor é imprescindível.

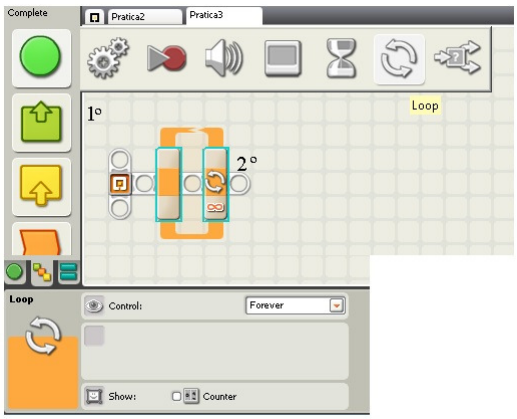


Figura 21: Passos para inserção de um Loop.

Fonte: Autor

Primeiramente, vamos criar um novo programa e nomeá-lo como **Prática3**.

Seguindo os passos da figura 21 inserimos um loop infinito no programa. A finalidade do mesmo é permitir que o sensor permaneça em constante teste e o veículo em movimento até encontrar uma faixa escura. No interior do loop infinito colocaremos uma crítica, o comando Switch. Por padrão, o Switch é associado ao sensor de toque;

com um clique sobre ele, podemos ir à palheta de configurações e mudar o sensor a ser utilizado no campo **Sensor**. Como demonstrado na figura a seguir, devemos mudar o sensor para o tipo *Color*.

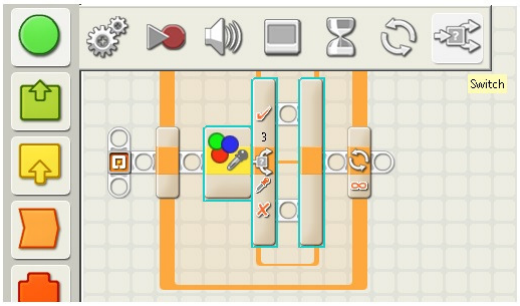


Figura 22: Inserção do comando Switch

Fonte: Autor

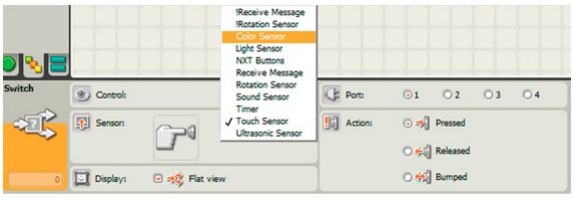


Figura 23: Mudança do tipo de sensor a ser utilizado.

Fonte: Autor

Precisamos agora definir os parâmetros do sensor a serem configurados na parte secundária da palheta de configuração.

Figura 24: Parâmetros do sensor.

Fonte: Autor

Na figura acima, definimos a cor a ser testada, no caso o preto, e vinculamos o sensor à porta que esteja conectado.

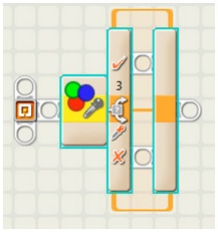


Figura 25: Visualização do bloco *Switch* utilizando o sensor de cor.

Fonte: Autor

Agora devemos definir as ações a serem tomadas. No primeiro campo, onde o sensor julga como a comparação sendo verdadeira, ou seja, como a cor sendo preta, vamos inserir o bloco **Move** com a configuração de parar os motores responsáveis pela movimentação do veículo. Já no segundo campo, onde o sensor julga a comparação como sendo falsa, ou seja, como a cor não sendo preta, vamos inserir o bloco **Move** com a duração do movimento dos motores definida como infinita, pois o sensor vai ser responsável por sua parada.

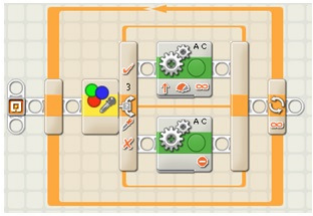


Figura 26: Programa finalizado.

Fonte: Autor

Versão Textual

Resumidamente, o sensor, enquanto não detectar a cor preta, vai informar que o veículo deve andar, e ao encontrar a cor preta, que o mesmo deve parar. Gravamos então o programa no NXT e, seguindo os passos descritos no tópico 8, executamos o arquivo de nome **Prática3**.

Multimídia

Assista a vídeo "Pare na faixa" utilizando o kit Lego Mindstorms NXT 2.0.

https://www.youtube.com/watch?v=9WzI4TeQkjo

Atividades Práticas

1.3.1 LEGO EV3

Primeiramente, vamos criar um novo programa e nomeá-lo **Prática3**.

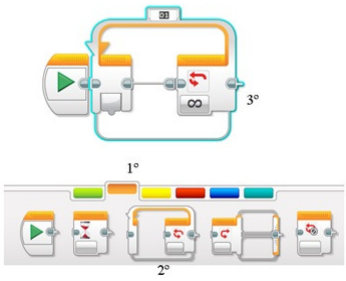


Figura 27: Passos para inserção de um Loop.

Fonte: Autor

Seguindo os passos da figura 27, inserimos um loop infinito no programa. A finalidade do mesmo é permitir que o sensor permaneça em constante teste e o veículo em movimento até encontrar uma faixa escura. No interior do loop infinito, colocaremos uma crítica, o comando *Switch*. Por padrão, o *Switch* é associado ao sensor de toque, com um clique sobre ele, podemos ir a palheta de configurações e mudar o sensor a ser utilizado no campo **Sensor**. Como demonstrado na figura a seguir, devemos mudar o sensor para o tipo *Color*.

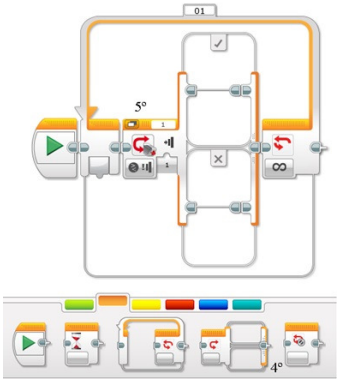


Figura 28: Inserção do comando Switch.

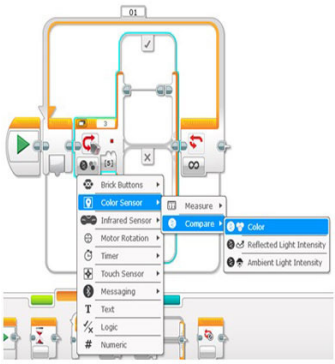


Figura 29: Mudança do tipo de sensor a ser utilizado.

Fonte: Autor

Fonte: Autor

Precisamos agora definir os parâmetros do sensor a serem configurados na parte secundária da palheta de configuração.

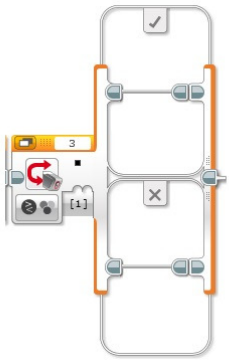


Figura 30: Visualização do bloco *Switch* utilizando o sensor de cor.

Fonte: Autor

Na figura acima, definimos a cor a ser testada, no caso o preto, e vinculamos o sensor à porta que esteja conectado.

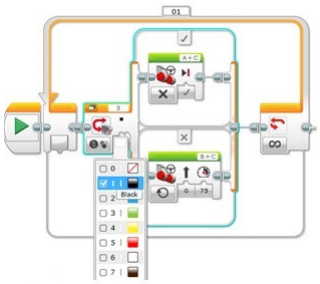


Figura 31: Programa finalizado.

Fonte: Autor

Versão Textual

Agora devemos definir as ações a serem tomadas. No primeiro campo, onde o sensor julga como a comparação sendo verdadeira, ou seja, como a cor sendo preta, vamos inserir o bloco **Move** com a configuração de parar os motores responsáveis pela movimentação do veículo. Já no segundo campo, onde o sensor julga a comparação como sendo falsa, ou seja, como a cor não sendo preta, vamos inserir o bloco **Move** com a duração do movimento dos motores definida como infinita, pois o sensor vai ser responsável por sua parada.

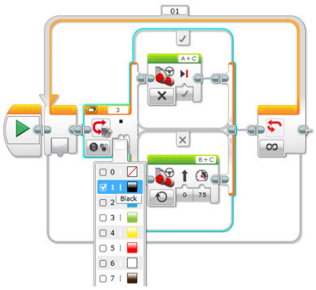


Figura 32: Programa finalizado.

Fonte: Autor

Resumidamente, o sensor, enquanto não detectar a cor preta, vai informar que o veículo deve andar, e ao encontrar a cor preta, que o mesmo deve parar. Gravamos então o programa no NXT e, seguindo os passos descritos no tópico 8, executamos o arquivo de nome **Prática3.**

Multimídia

Assista a vídeo "Pare na faixa" utilizando o kit Lego EV3.

http://www.youtube.com/watch?v=9yU6SGNIxmw

Atividades Práticas

1.4 Olha o muro

Com o sensor ultrassônico, devemos evitar que o nosso veículo colida com obstáculos, a parede por exemplo. Ele deverá seguir em linha reta até encontrar um obstáculo, evitar bater nele e retornar pelo mesmo caminho de ida. Como demonstrado nos exemplos anteriores, todos os procedimentos de criação, gravação e execução do programa se aplicam a esse mesmo exemplo.

Devemos criar um novo programa com o nome de **Prática4**. Para essa prática, utilizaremos um motor, um loop e o sensor de ultrassom. Montaremos o programa como ilustrado na imagem abaixo.

Figura 33: Programa "olha o muro".

Fonte: Autor

Esse programa é bem simples e ilustra de maneira didática o uso dos motores e do sensor de ultrassom. Basicamente, temos um loop infinito controlado pelo sensor, ou seja, o veículo vai se mover para frente indefinidamente até ficar a uma distância menor que vinte centímetros de um obstáculo.

Figura 34: Motor 1 no interior do loop.

Fonte: Autor

Configuramos o loop para ficar monitorando a porta 4 em que está conectado o sensor ultrassônico. Quando ele detectar uma distância menor que vinte centímetros, quebra o loop infinito.

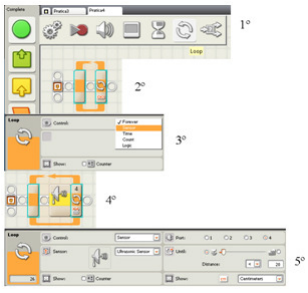


Figura 35: Configuração do loop.

Fonte: Autor

Versão Textual

Devemos ter atenção para alguns detalhes. Devemos mudar a escala de distância de *inch* para centímetros bem como o item *distance* para o símbolo "<", ele que indica se a distância será menor ou maior que a programada. Outro cuidado é observar a porta em que está ligado o sensor.

Quando o veículo se aproximar de algum obstáculo a menos de vinte centímetros, o loop não será mais executado e o próximo bloco da sequência será executado.

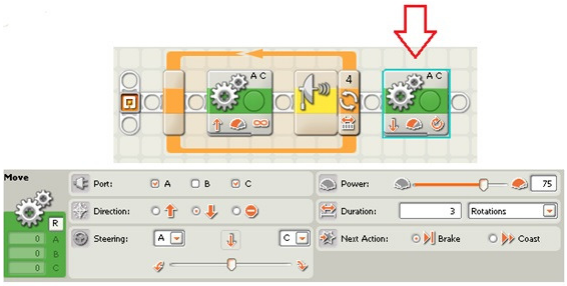


Figura 36: Motor responsável pelo retorno do veículo.

Fonte: Autor

Esse próximo motor é o responsável por nosso veículo retornar pelo caminho inicial. Ele está configurado para acionar os motores A e C, porém com o sentido oposto ao primeiro motor. Ele também possui um número limitado de rotações, ou seja, ele se moverá por um espaço limitado a três rotações. Após retornar pelo caminho de ida, ele encerra o programa com a configuração "Next Action = Break".

Devemos sempre ter atenção em alguns detalhes: verificar os motores e as portas de saída em que eles estão conectados, o sentido de rotação, bem como o tipo de duração do funcionamento dos motores.

Multimídia

Assista ao "Olha o muro" utilizando o kit Lego NXT.

https://www.youtube.com/watch?v=ugL3QquOd4A

Atividades Práticas

1.4.1 LEGO EV3

Com o sensor de infravermelho, devemos evitar que nosso veículo colida com obstáculos, a parede por exemplo. Ele deverá seguir em linha reta até encontrar um obstáculo, evitar bater nele e retornar pelo mesmo caminho de ida. Como demonstrado nos exemplos anteriores, todos os procedimentos de criação, gravação e execução do programa se aplicam a esse mesmo exemplo.

Versão Textual

Devemos criar um novo programa com o nome de **Prática4**. Para essa prática, utilizaremos um motor, um loop e o sensor de infravermelho. Montaremos o programa como ilustrado na imagem abaixo.

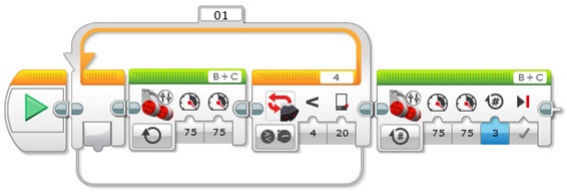


Figura 37: Programa "olha o muro".

Fonte: Autor

Esse programa é bem simples e ilustra de maneira didática o uso dos motores e do sensor de infravermelho. Basicamente, temos um loop infinito controlado pelo sensor, ou seja, o veículo vai se mover para frente indefinidamente até ficar a uma distância menor que vinte centímetros de um obstáculo.

Figura 38: Motor 1 no interior do loop.

Fonte: Autor

Configuramos o loop para ficar monitorando a porta 4 em que está conectado o sensor infravermelho. Quando ele detectar uma distância menor que vinte centímetros, quebra o loop infinito.

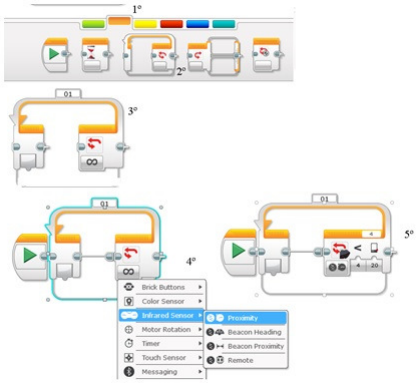


Figura 39: Configuração do loop.

Fonte: Autor

Devemos ter atenção para alguns detalhes: mudar a escala de distância de inch para centímetros bem como o item *distance* para o símbolo "<"; ele que indica se a distância será menor ou maior que a programada. Outro cuidado é observar a porta em que está ligado o sensor.

Quando o veículo se aproximar de algum obstáculo a menos de vinte centímetros, o loop não será mais executado e o próximo bloco da sequência será executado.

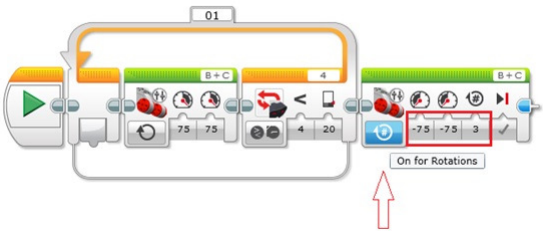


Figura 40: Motor responsável pelo retorno do veículo.

Fonte: Autor

Esse próximo motor é o responsável por nosso veículo retornar pelo caminho inicial. Ele está configurado para acionar os motores A e C, porém com o sentido oposto ao primeiro motor. Ele também possui um número limitado de rotações, ou seja, ele se moverá por um espaço limitado a três rotações. Após retornar pelo caminho de ida, ele encerra o programa com a configuração "Next Action = Break".

Devemos sempre ter atenção em alguns detalhes: verificar os motores e as portas de saída em que eles estão conectados, o sentido de rotação, bem como o tipo de duração do funcionamento dos motores.

Multimídia

Assista ao "Olha o muro" utilizando o kit Lego EV3

https://www.youtube.com/watch?v=iraV0Owwdlo

Software RoboMind

Versão Textual

O *RoboMind* é uma plataforma de programação que provê uma linguagem de programação simples, didática e intuitiva para simular a movimentação de um robô em um ambiente virtual bidimensional. Ao executarmos o software, a tela principal irá abrir conforme a figura 41.

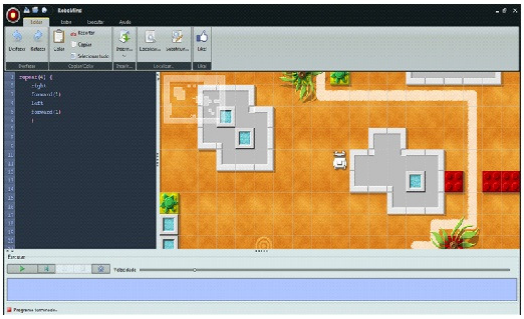


Figura 41. Tela principal do *RoboMind*.

Fonte: Autor

À esquerda, temos a área de programação onde serão escritas as instruções que, brevemente, serão passadas para o robô. À direita, temos o ambiente bidimensional onde o robô executará seus movimentos com base nas instruções dadas na área de programação. Acima, temos um menu onde há alguns comandos importantes, como o comando Inserir na guia Editar. Esse comando é basicamente uma biblioteca com todas as funções do RoboMind, que podem ser inseridas diretamente no código. E abaixo, nós temos um bloco de execução onde são apresentados os erros de sintaxe e de execução por parte do robô virtual.

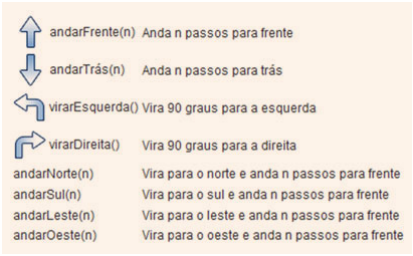
O *RoboMind* é disponível em diversas línguas, dentre elas o português do Brasil e o inglês, que é a linguagem que será usada para permitir a comunicação com o Lego NXT.

O conjunto de instruções do robô é bastante intuitivo, como por exemplo, a *função andarFrente(x)*, que faz o robô se movimentar para a sua própria frente x blocos. O conjunto de linguagens também possui instruções de repetição como *repetir(x) { }*, que repete x vezes as instruções dentro das chaves, e instruções de condição, como por exemplo a função *se(x) { }*, que executa as instruções dentro das chaves se a condição x for satisfeita. Na próxima seção, serão explicadas as instruções básicas desse software.

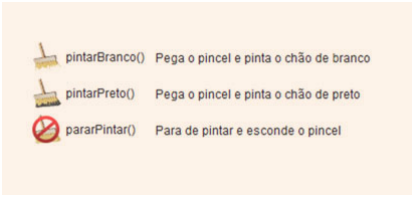
2.1 Instruções Básicas

Como dito na seção anterior, ao abrir o software RoboMind, haverá um ambiente de programação na esquerda; é lá que você irá escrever o seu código. Algumas instruções do RoboMind não fazem parte do objetivo desse curso, portanto, não serão abordadas aqui. As instruções básicas do RoboMind que utilizaremos são:

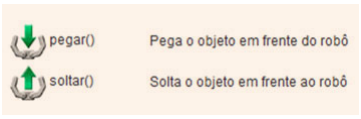
Andar



Pintar



Pegar



Ver

Ver	Esquerda	Frente	Direita
	temObstáculoEsquerda()	temObstáculoFrente()	temObstáculoDireita()
	vazioEsquerda()	vazioFrente()	vazioDireita()
	temObjetoEsquerda()	temObjetoFrente()	temObjetoDireita()
	brancoEsquerda()	brancoFrente()	brancoDireita()
	pretoEsquerda()	pretoFrente()	pretoDireita()

As instruções acima verificam se tem algo em alguma direção. Por exemplo, a função *temObjetoEsquerda()* verifica se tem algum objeto à esquerda do robô. Outro exemplo, a função brancoFrente() verifica se existe cor branca na frente do robô.

2.2 Instruções Avançadas

Além das instruções básicas, você pode utilizar algumas instruções que lhe permitem programar um comportamento mais sofisticado para o robô. Vejamos agora quais são as instruções e alguns exemplos de como utilizá-las.

Laços

1. **repetir(n){instruções}**: Repete as instruções entre chaves n vezes.

Exemplo: Anda para frente 4 passos e vira para a esquerda 6 vezes.

```
repetir(6){andarFrente(4)virarEsquerda()}
```

2. **repetirEnquanto(condição){instruções}:** Repete as instruções enquanto a condição entre parênteses for satisfeita.

Exemplo: Anda 1 passo para frente enquanto ela estiver livre.

```
repetirEnquanto(vazioFrente)
{andarFrente(1)}
```

3. **Sair**: Permite a você sair do laço mais próximo, interrompendo a execução das demais instruções no laço. O robô irá executar as instruções após o fechamento das chaves do laço.

Exemplo: Ao invés de o robô repetir 3 vezes as instruções do laço, ele só irá repetir uma vez, pois temos o comando sair ao final do laço.

```
repetir(3){andarFrente(1) virarDireita()sair}virarEsquerda()
```

Condições

1. **Se(condição){...instruções...}**: Executará as instruções entre as chaves somente se a condição for verdadeira. Caso contrário, o robô pula as instruções entre chaves.

Exemplo: Desvia do local pintado de branco.

```
se(brancoFrente()){virarEsquerda()andarFrente(1)
virarDireita()}
```

2. **Se(condição){...instruções...}senão{...instruções...}**: Executará as instruções entre o primeiro par de chaves somente se a condição for verdadeira. Caso a condição seja falsa, o robô somente executa as instruções compreendidas no segundo par de chaves.

Exemplo: Se tiver branco na frente, o robô desvia, caso contrário, ele continua seguindo em frente.

```
se(brancoEsquerda()){virarEsquerda()andarFrente(1)virarDireita()
senão{andarFrente(1)}
```

- Fim: Fará com que todo o programa pare a execução.
- **Exemplo**: O robô anda para frente até encontrar um obstáculo.

```
repetir() {andarFrente(1) se(temObjetoFrente()){ fim}}
```

2.3 Conexão com o robô da LEGO

A conexão entre o software *RoboMind* e o Kit Lego *Mindstorms* é realizada através do menu principal do *RoboMind*, selecionando a opção export. Após isso, em *Select target*, escolha a opção *NXTBot.nxc*. Feito isso, seu código será enviado para o NXT. No entanto, para que o robô da Lego entenda as instruções enviadas do *RoboMind*, as seguintes condições precisam ser obedecidas:

Condição 01

Condição 02

Condição 03

As condições mostradas acima foram descobertas pelo grupo desenvolvedor do projeto através de uma semana inteira de testes com o software *RoboMind* e o kit de robôs da Lego *Mindstorms NXT*.

Referência

BACAROGLO, M. **Robótica Educacional**: uma metodologia educacional. Dissertação. UEL, Londrina, 2005.

BUCKHAULTS, C. Increasing computer science participation in the FIRST robotics competition with robot simulation. In: PROCEEDINGS OF THE 47TH ANNUAL SOUTHEAST REGIONAL CONFERENCE, Clemson, South Carolina, 2009.

BENITTI, F.B. VAVASSORI; VAHLDICK, A; URBAN, D.L; KRUEGER, M.L; HALMA, A. Experimentação com Robótica Educativa no Ensino Médio. In: CSBC - XXIX CONGRESSO DA SOCIEDADE BRASILEIRA DE COMPUTAÇÃO, UFRGS, 2009

SILVA, F.A. **Uma Metodologia de Aprendizado com Robótica Educacional**, 2009. Tese. Pós-Graduação em Engenharia Elétrica, RoboEduc, Universidade Federal do Rio Grande do Norte, 2099.

INOVA Engenharia "Proposta para modernização da educação em engenharia no Brasil", 2006.

NASCIMENTO, Paulo C. Inteligência Artificial. Disponível em: . Acesso em: 30 maio. 2013.

NUNES, J.E.A; DA SILVA, J.H.J; BOUÇAS, M.V; DE OLIVEIRA, S.R. **Apostila de Robótica Educacional**, Versão 1.0. TEE – Departamento de Engenharia Elétrica - Programa de Educação Tutorial da Engenharia Elétrica – PET-Elétrica, Universidade Federal Fluminense, Niterói, Rio de Janeiro, 2013. Disponível em: http://www.peteletrica.uff.br/wp-content/uploads/2013/08/Apostila-Rob%C3%B3tica-Educacional.pdf . Acesso em: 30 maio. 2013.

SOUSA. L; LIMA. R; PERREIRA.V; DE CARVALHO. R.N; CORREIA. B.W; MACIEL DIAS. B.N. Utilização do kit lego *mindstorms* para motivar e atrair alunos para os cursos de engenharia da UFC campus de Sobral. In: xli CONGRESSO BRASILEIRO DE ENSINO EM ENGENHARIA, UFRGS, 2013.

MINDSTORMS education, Manual do NXT 2.0. Disponível em: http://www.inf.furb.br/~aureliof/downloads/ManualNXT.pdf. . Disponível em: http://education.lego.com/downloads/?q={DC0CE993-6544-45A1-8680-B2A547D1EEB6 Acessado em: 01 Jan. 2014.

MINDSTORMS EV3 . Guia do Usuário Disponível em: http://www.lego.com/en-us/mindstorms/downloads/software/ddsoftwaredownload/ Acessado em: 01 de maio. 2014.

http://robolab.inf.furb.br/index.php?option=com_content&view=article&id=12&Itemid=20

http://robolab.inf.furb.br/index.php?option=com_content&view=article&id=12&Itemid=20