Universidade do Estado de Minas Gerais

Instituto Superior de Ensino e Pesquisa de Ituiutaba
Curso de Bacharelado em Engenharia de Computação

BÍPEDE:
Com funções programadas em um Microcontrolador

Rodrigo Ferreira Malta

2014
Ituiutaba – MG

Rodrigo Ferreira Malta

BÍPEDE:
Com funções programadas em um Microcontrolador

Monografia de Conclusão de Curso apresentada à Coordenação do Curso de Engenharia da Computação, da Universidade do Estado de Minas Gerais, sob a orientação do Prof. Me. Walteno Martins Parreira Jr., como pré-requisito para obtenção do grau de Bacharel em Engenharia da Computação.

Orientador: Walteno Martins Parreira Jr.

Ituiutaba, 2014

Rodrigo Ferreira Malta

BÍPEDE:
Com funções programadas em um Microcontrolador

Monografia de Conclusão de Curso apresentada à Coordenação do Curso de Engenharia da Computação, da Universidade do Estado de Minas Gerais, sob a orientação do Prof. Me. Walteno Martins Parreira Jr., como pré-requisito para obtenção do grau de Bacharel em Engenharia da Computação.

LOCAL E DATA DA APROVAÇÃO

Banca Examinadora

_____
Prof(a). Nome da Instituição de Origem

_____
Prof(a). Nome da Instituição de Origem

_____
Prof. Orientador. Me. Walteno Martins Parreira Jr.

Dedico este trabalho a minha família, é em especial minha namorada e aos colegas de serviço, que ambos sempre me apoiaram.

AGRADECIMENTOS

Agradeço aos meus pais Nivaldo Teodoro Malta e Joana D’arc Ferreira Malta que sempre me apoiaram, ajudaram e incentivaram, no decorrer de todos esses anos de curso, e por sempre me darem bons conselhos. Aos meus irmãos Douglas e Tiago que sempre me apoiaram. A minha namorada Carina Alves da Silva que sempre me apoiou e insistiu para que eu estudasse mais. Aos professores por compartilharem os seus conhecimentos comigo.

Sou grato amigos e colegas da universidade, que carregarei para toda minha vida porque com os quais compartilhei bons momentos, e também momentos difíceis em que, unidos, conseguimos superá-los.

Agradeço ao meu orientador Walteno Martins Parreira Jr., que compartilhou sua experiência e conhecimento para que eu pudesse desenvolver o projeto com sucesso.

Agradeço aos meus colegas de serviço Eduardo, André e Olegário que sempre me ajudaram com ideias e motivações.

À Universidade do Estado de Minas Gerais, ao departamento de Engenharia de Computação, aos diretores e funcionários.

Agradeço também aquelas pessoas que, de forma direta ou indireta, contribuíram para que eu chegasse até aqui.

“Jamais sofra antecipadamente. Pense positivo. Acredite nos seus sonhos. Nunca desista de lutar. A vida é generosa para aqueles que acreditam nela.”
(Vitoria Cirilo)

Resumo

A tecnologia tem avançado muito, é e com essa tecnologia que nos permite colocar em pratica várias ideias e projetos que antes não poderiam ser colocados em práticas. Um bípede pode ser usado de várias maneiras como, por exemplo: brinquedos, um protótipo para estudar movimentos de um ser vivo que anda sobre duas pernas. Um bípede possui seis motores-servos, que controlam os eixos da perna, onde será feito o aceso no momento em que o programa for executado através de um Microcontrolador. O Microcontrolador irá mandar o comando necessário para a porta na qual os servos estão conectados, fazendo com que o mesmo começa a se movimentar.

Palavras-chave: TCC, Projeto de Conclusão, Trabalho de Conclusão, Bípede, Microcontrolador, Motor-Servo.

Abstract

Technology has advanced so much, and it is with this technology that allows us to put into practice many ideas and projects that previously could not be put into practice. A biped can be used in many ways, such as: toys, a prototype for studying movements of a living being that walks on two legs. A biped has six servo-motors that control the axis of the leg, which will be lit at the time the program is run through a microcontroller. The microcontroller will send the appropriate command to the door where the servos are connected, so that it begins to move.

Password: Conclusion Project, Conclusion Work , Biped, Microcontroller, Motor-Servo.

Sumário

1. Introdução

Bípede, de acordo com o dicionário de português online Léxico: “Bípede: Que anda em dois pés. M, Animal que anda sobre dois pés. (Lat. Bipes)”. [3]

No caso deste projeto, o Bípede será um conjunto de pernas robóticas com articulações na cintura e joelho que movimenta para frente, para trás e para os dois lados.

A alimentação do Bípede é feita de forma com que cada um do servo-motores possa ser acionado a qualquer momento de acordo com o circuito.

> Os servo-motores são usados em várias aplicações quando se deseja movimentar algo de forma precisa e controlada. Sua característica mais marcante é a sua capacidade de movimentar os seu braço Até uma posição e mantê-lo, mesmo quando sofre uma força em outra direção. [6]

Os servo-motores tem 3 fios, um de tensão (que vai a energia), um terra e um fio de controle. O fio de controle é o fio que passa o comando do computador para o servo, sendo assim o controle de cada um dos motores e feito por um Microcontrolador, que permite controlar todos os motores dizendo a angulação e o caminho para onde deve se mover.

A tecnologia proposta por esse trabalho já existe, porém não está muito documentada, existe alguns protótipos para se comprar em alguns sites da China que já vem com um sistema embarcado e fica restrito os seus movimentos.

A vantagem desse projeto e que o Bípede terá um controle independente, ou seja, será controlado pelo usuário através de um cabo. Esse cabo passará os comando, e o usuário poderá escolher se o Bípede irá se movimentar para direita, esquerda, frente ou para traz.

1.1. Objetivos

Os objetivos definidos para o projeto de pesquisa são os seguintes:

Adquirir os conhecimentos disponíveis sobre o tema;

Promover a interdisciplinaridade e favorecer a construção do conhecimento através da pesquisa e troca de informações entre os integrantes do projeto e também professores e alunos que atuam na área de robótica;

Pesquisar e aprender a manipular softwares que permitam a produção do bípede;

Desenvolvimento do protótipo

Desenvolvimento do código;

1.2. Justificativa

A construção do Bípede passará por algumas etapas, onde serão simples e outras mais complexas. O objetivo de dividir o trabalho em etapas é fazer com que os problemas que venha a surgir com o tempo sejam solucionados o mais rápido possível, de modo que não atrase o cronograma.

Inicialmente será desenvolvida uma pesquisa bibliográfica sobre o assunto do projeto e a capacitação teórica e prática nos softwares a serem utilizados durante o projeto.

Em uma segunda etapa, será feito a escolha do material a fim de desenvolver o protótipo do bípede, para estudar e planejar os seus movimentos de forma a não perder o equilíbrio e executar as situações propostas.

Após a construção do protótipo será feita uma avaliação do funcionamento para o seu aprimoramento e correção das falhas identificadas.

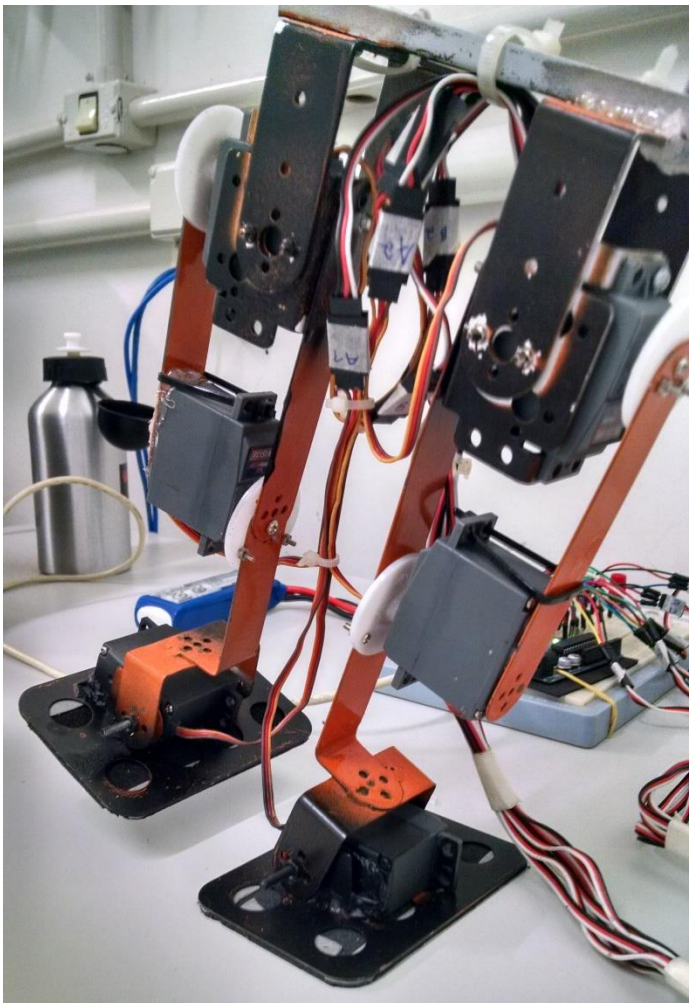

Imagem 15 – Bípede completo, Fonte próprio Autor

2. Fundamentação Teórica

O que é Arduino?

> O Arduino é o que chamamos de plataforma de computação física ou embarcada, ou seja, um sistema que pode interagir com seu ambiente por meio de hardware e software. Por exemplo, um uso simples de um Arduino seria para acender uma luz por certo intervalo de tempo, digamos, 30 segundos, depois que um botão fosse pressionado. Nesse exemplo, o Arduino teria uma lâmpada e um botão conectados a ele. O Arduino aguardaria pacientemente ate que o botão fosse pressionado; uma vez pressionado o botão, ele acenderia a lâmpada e iniciaria a contagem. Depois de contados 30 segundos, apagaria a lâmpada e aguardaria um novo apertar do botão. [11]

Os projetos utilizando Arduino, PIC e outros Microcontroladores são bastante utilizados em várias áreas, por exemplo, Automação Comercial, Automação Residencial, Robótica e várias outras. Para o funcionamento do Bípede é necessário utilizar um Microcontrolador para controlar os servo-motores e ter conhecimento na linguagem de programação necessária para desenvolver o código, necessita-se também ter um conhecimento básico em física e várias outras matérias relacionadas ao curso para calcular o equilíbrio a força e tudo que envolve essa área.

> *Robô* Bípede é um pequeno *robô* que possui duas pernas, que pode ser controlado pelo usuário através de um programa rodando num computador. Ele possui vários motores, que serão utilizados na sua locomoção. [5]

O Bípede não é apenas um objeto que se movimenta, muito menos um brinquedo com intenção de chamar atenção. O Bípede terá um circuito complexo, com uma montagem bastante interessante, e deverá estar pronto para receber o comando do usuário a hora que ele quiser. Ao usar um Microcontrolador ATMEGA em sua plataforma Arduíno, será possível controlar os seis motores que o Bípede utilizará.

O Bípede será desenvolvido com objetivo de apresentar a facilidade de manuseio e de movimentação que se aproxima a de um ser humano para atuação em atividades insalubres ou que necessita de esforço físico.

2.1 Matérias Relacionadas

Esse trabalho tem uma relação com várias disciplinas que com o decorrer do curso foram ensinadas com um proposito, que agora será utilizado para se desenvolver o mesmo. Algumas dessas disciplinas são:

- Eletrônica
- Aplicação Industrial de Microprocessadores
- Circuitos Elétricos
- Eletricidade
- Algoritmos e Estrutura de Dados

- Engenharia de *Software*
- Robótica

2.1.1 Eletrônica

O conhecimento teórico e prático para usar os circuitos integrados, diodos, e outros dispositivos eletrônicos foi adquirido durante as aulas da disciplina de Eletrônica. Durante o ano em que estudamos a disciplina aprendemos sobre eletrônica analógica e digital.

2.1.2 Aplicação Industrial de Microprocessadores

Essa disciplina trabalha em conjunto com a Robótica, sendo que na disciplina de Aplicação Industrial de Microprocessadores é uma disciplina voltada para a aula prática, tornando o aluno apto a montar circuitos com microprocessadores e programa-los. Também aprende a utilização de Microcontrolador, como manipula-lo como fazer desenvolver o código e muito mais.

> Um Microcontrolador possui um **dispositivo dedicado de entrada** (mas nem sempre) e geralmente possui um pequeno ***LED*** **ou visor** ***LCD*** **de saída**. Um Microcontrolador também obtém a entrada do dispositivo que está controlando e o controla enviando sinais a diferentes componentes desse dispositivo. [1]

2.1.3 Circuitos Elétricos

O projeto requer um conhecimento para a montagem do circuito elétrico, onde será usado desde uma fonte de tensão às resistências necessárias para proteger o circuito contra uma tensão maior do que a desejada.

A disciplina de Circuitos Elétricos influi bastante para qualquer aluno que deseja montar um circuito, pois durante o ano em que a mesma foi aplicada obtive um conhecimento sobre a análise de circuitos, quedas de tensão, Leis de *Kirchoff* e *Ohm*, entre outros tópicos que poderão ser necessários não apenas no projeto, mas futuramente.

2.1.4 Eletricidade

Semelhante a Circuitos Elétricos, a disciplina de Eletricidade é aplicada um ano antes com o intuito de ensinar ao graduando o que são os componentes elétricos e como funcionam. O escopo maior dessa disciplina é abrir as portas do pensamento para o aluno começar a calcular tensões, correntes, resistências e etc.

Após o termino da disciplina o graduando sente mais confiante para poder criar desde um pequeno circuito elétrico, como acendimento de uma única lâmpada, a um circuito maior, porém simples e sem componentes que exijam um maior conhecimento.

2.1.5 Algoritmos e Estruturas de Dados

A disciplina de Algoritmos e Estrutura de dados terá total influencia para o projeto ser finalizado com êxito.

Durante o ano em que a disciplina é aplicada, o aluno aprende a criar um algoritmo para poder implementá-lo em uma linguagem de programação, sendo que no ano em que estudamos a disciplina voltamos mais para a linguagem de programação C ou C++.

Para o desenvolvimento do código que será executado no microprocessador, poderá usar apenas uma de duas linguagens de programação, C ou Assembly. Porém será usado C por ser uma linguagem de fácil acesso a livros para pesquisa e com um código mais claro para futuramente poder explicar ou corrigir erros e *bugs*.

2.1.6 Engenharia de *Software*

Engenharia de *software* é uma área da computação voltada à especificação, desenvolvimento e manutenção de sistemas de *software*, com aplicação de tecnologias e práticas de gerência de projetos e outras disciplinas, visando organização, produtividade e qualidade.

Engenharia de *software* também é uma derivação da engenharia de sistemas e de hardware. Ela abrange um conjunto de três elementos fundamentais - métodos, ferramentas e procedimentos - que possibilita ao gerente o controle do processo de desenvolvimento do software e oferece ao profissional uma base para a construção de software de alta qualidade produtivamente.

2.1.7 Robótica

Por último, não menos importante, a disciplina de Robótica, onde se adquire o conhecimento suficiente para poder desenvolver projetos relacionados a robótica. EX: Bípede.

> Um robô é um dispositivo que permite realizar trabalhos mecânicos, normalmente associados a seres humanos, de uma maneira muito mais eficiente e sem a necessidade de pôr em risco a vida humana [8]

Em robótica estuda-se muito os Microcontroladores, um em especifico é o Arduino. O Arduino é um *software* e também um *hardware* que se utiliza como Microcontrolador.

> Arduino é uma placa de controle de entrada de dados (*IN*), como sensores, e saída de dados (*OUT*), como motores e *leds*, com cristal oscilador de 16 Mhz, um regulador de tensão de 5 V, botão de *reset*, plugue de

> alimentação, pinos conectores, e alguns *LEDs* para facilitar a verificação do funcionamento. [8]

De acordo com o livro Robótica do autor J.A.M Felippe de Souza ele explica muito bem o significado de robótica.

> A diversidade de tipos de *robôs* que existem impedem que haja uma definição de *robô* que seja universalmente aceite. No entanto há um conjunto comum de componentes que essa diversidade de *robôs* partilha, como por exemplo: Sistemas de locomoção; sensores; sistemas de processamento; etc.; [9]

Esses componentes que em conjunto forma a definição de *robô* nada mais é como a mistura de todas as matérias acima citado ou seja robótica e uma subdivisão do curso de engenharia da computação que abrange uma área de alcance inimaginável.

Para o funcionamento ideal do Bípede é necessário uma serie de cálculos de grandezas envolvidas como potência, velocidade, torque e várias outras. Essas grandezas são essenciais na escolha dos motores ideias para qualquer tipo de projeto, de acordo com o Quadro 01 mostra as equações.

> Potência mecânica (Pm) no eixo do motor é igual a potência elétrica (Pe) fornecida multiplicada pelo rendimento do mesmo. Unidade: watts [W] = Joule/segundo [J/s]. [7]
>
> Torque é a fração da força aplicada sobre um objeto que é efetivamente utilizada para fazê-lo girar em torno de um eixo. Unidade: Newton x metro [n.m]. [7]
>
> Velocidade angular ou de rotação (w) é equivalente ao ângulo que gira o eixo por unidade de tempo. Unidade: [rpm] , [o/s] , [rad/s]. [7]

	Equações	**Unidades**
Potência Mecânica	Pm = n.Pe	Joule/segundo [J/s]
Torque	|T|= |r|.|f|.sen □ □	Newton x metro [n.m]
Velocidade	Pm = T . □	[rpm] , [o/s] , [rad/s]

Quadro 01 – Grandezas Físicas Fonte: do próprio autor

2.2 Sistema Locomotor

O sistema locomotor do ser-humano e responsável pelas funções do andar. O sistema locomotor do Bípede será bastante parecido com o do ser-humano.

> O sistema locomotor é responsável pelas funções do movimento, locomoção e deslocamento dos seres vivos. O conjunto de ossos, músculos e elementos das articulações compreende a locomoção na espécie humana. O sistema esquelético sustenta, protege os órgãos internos, armazena minerais e íons e produz células sanguíneas. [4]

O sistema esquelético do Bípede será sustentado pelo chassi que irá comprometer os movimentos. O chassi se equivale ao sistema esquelético dos seres humanos, porém o chassi do Bípede só possui pernas. Os seres humanos possuem as articulações que proporciona os movimentos, no Bípede essas articulações serão junção de movimentos ligando duas partes do chassi, por exemplo o Joelho e a Perna como mostra na Imagem 01.

> O corpo humano é capaz de realizar diversos movimentos, graças à **articulação** encontrada em nosso esqueleto. O responsável por dar esta mobilidade entre ossos e estabilizar as zonas de união entre os vários segmentos do esqueleto é o **Sistema Articular**. [4]

Na Imagem 01 abaixo mostra uma comparação entre o Bípede e o Sistema esquelético, você pode observar que o Bípede tem pês, perna, joelho, e quadril igual o sistema esquelético a diferença e que no Bípede o sistema esquelético dele pode chamar-se de Chassi.

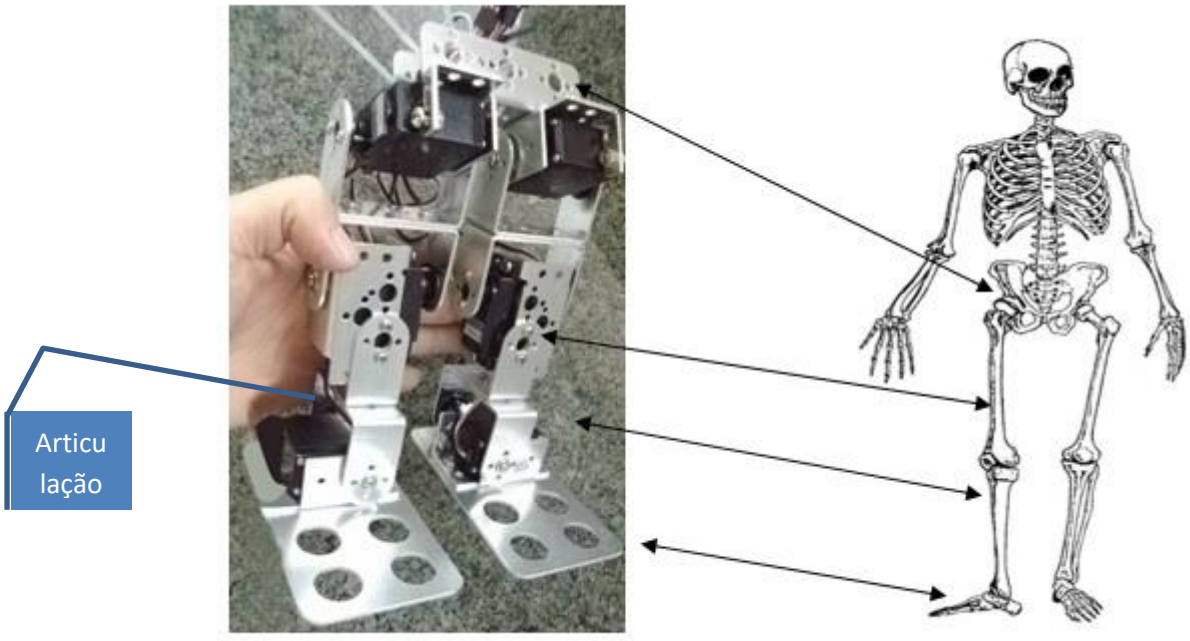

Imagem 01 – Diferença do Bípede para o sistema esqueletico humano. Fonte: do próprio autor

3. Trabalhos Relacionados

Os robôs podem ter diversas características, e cada uma dessas características tem uma aplicação adequada a um determinado tipo de atividade. Existem robôs que se locomovem sobre rodas, patas, e pernas que e o caso do Bípede para se deslocar. A quantidade de rodas, patas ou pernas de um robô para o outro pode variar.

A robótica hoje está bastante evoluída por conta da utilização de sensores que podem ser utilizados para auxiliar na movimentação e nas atividades realizadas pelos robôs.

Alguns robôs utilizam sensores ou são mais simples, mais ambos os modelos apresentam uma semelhança grande, exemplo:

3.1 ASIMO

O ASIMO é um *robô* da Honda, com aproximadamente 1,3 metros de altura, que é capaz de caminhar, chutar bola, subir escadas e correr a uma velocidade de até 6 km/h.

> Com um sistema de controle total de recém-desenvolvido que controla todas as funções do ASIMO, ASIMO pode agir de forma autônoma como recepcionista, ou mesmo entregar bebidas em uma bandeja.(Traduzido pelo Autor)[2]

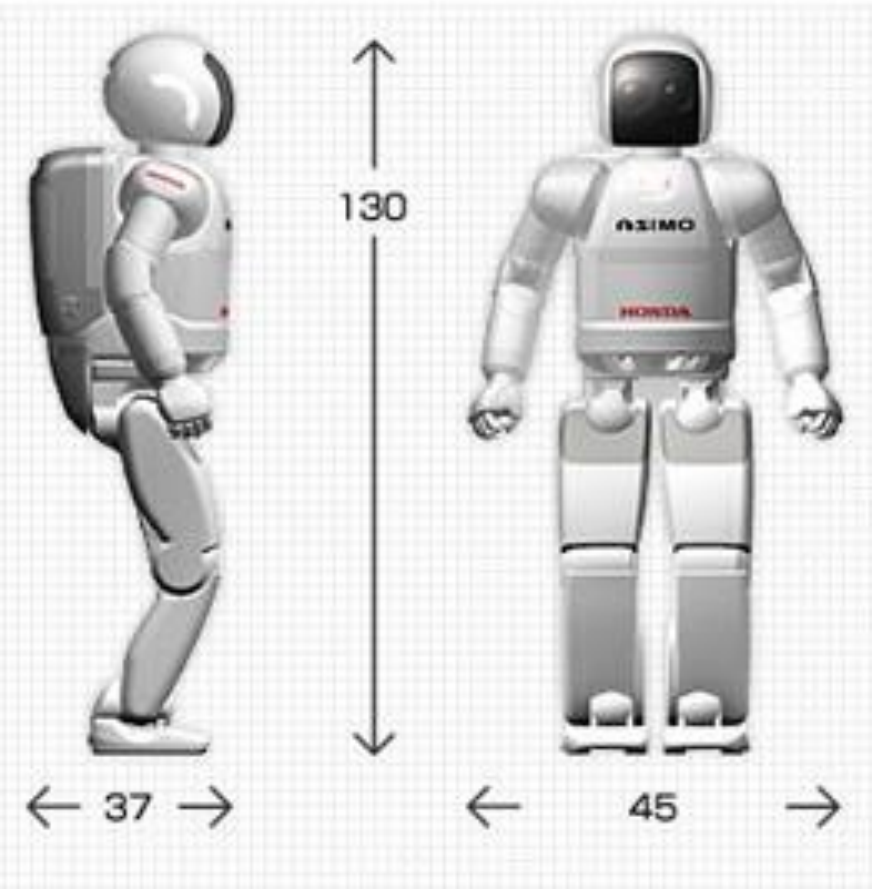

Imagem 02 – ASIMO. Fonte: Honda Worldwide[2]

3.2 Omnibot i-SODoG

Omnibot e um cão-*robô* criado pela empresa de brinquedos Takara Tomy, ele se parece bastante com um animal de verdade. Ele possui 15 servo-motores que utiliza-se para uma melhor performance.

> O novo cão robô, chamado de Omnibot i-SODOG, deverá ter movimento realista de um cão utilizando 15 servo-motores de design personalizado. Ele contará com reconhecimento de voz e responder a sinais manuais por meio de sensores de detecção de movimento. Além de um controle remoto dedicado que é semelhante à unidade i-sobot controle humanoide, i-SODOG pode ser controlado através de um smartphone. Estamos supondo que a interface será Bluetooth. (Traduzido pelo autor) [10]

Imagem 03 - *Omnibot i-SODOG.* Fonte: Robots Dreams[10]

Na imagem 03 mostra como é o modelo do Omnibot i-SODOG, e também apresenta como o controle dele é realizado, e pode ser tanto através de um smartphone quanto de um controle remoto tradicional.

3.3 RiSE

De acordo com **J. A. M. Felippe De Souza,** o *Robô* RiSE foi desenvolvido pelas empresa Boston Dynamics, foi um projeto desenvolvido por várias universidades como Universidade da Pennsylvania, de Carnegie Mellon, de Berkeley, deStanford, e de Lewis

and Clark. E um *robô* com a habilidade de subir paredes, arvores ou qualquer superfície em vertical, como mostra a Imagem 04.

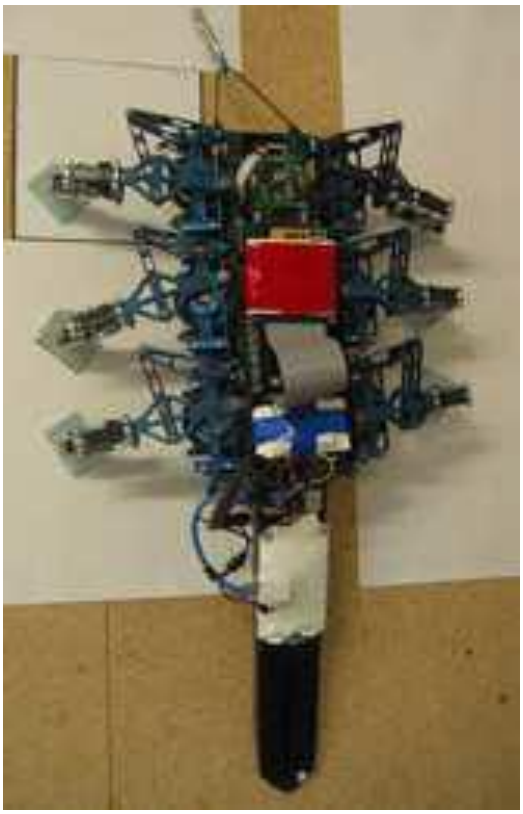

Imagem 04 – RiSE subindo uma parede. Fonte: Robótica[9]

Ele possui 6 patas com um material que gruda, para facilitar a escalada. Cada uma das patas possui dois motores elétricos.

RiSE foi utilizados em vários filmes como Runaway para atacar o ator Tom Selleck como mostra a Imagem 05.

Imagem 05 – Filme Runaway **aranha – RiSE.** Fonte: Robótica[9]

3.4 Semelhança e diferença entre os Trabalhos Relacionados é o Bípede

Todos os trabalhos relacionados acima mostram como é a estrutura de cada um, nenhum deles são iguais porem tem várias semelhanças que podem ser utilizadas no desenvolvimento do Bípede.

Projetos	Diferença com o Bípede
ASIMO	1. E um *robô* completo, com corpo. 2. E tem uma inteligência que lhe permite ter controle total.
Omnibot i-SODoG	1. Possui quarto pernas 2. Pode ser controlado via *bluetooth*, por um smartphone. 3. É um cachorro, e não se compara com a maneira de andar.
RiSE	1. Possui 6 patas. 2. Tem uma programação definida, que só faz o que foi programado 3. É uma aranha, e não se compara com a manheira de andar.

Quadro 02 –Diferenças de Trabalhos Relacionados.
Fonte: do próprio autor

Projetos	Semelhança com o Bípede
ASIMO	1. Ele anda sobre duas pernas. 2. Possui um Microcontrolador 3. Possui motores para fazer os movimentos de locomoção.
Omnibot i-SODoG	1. Possui um Microcontrolador. 2. Possui um controle que lhe permite controlar todos os movimentos. 3. Possui motores para fazer os movimentos de locomoção.
RiSE	1. Possui um Microcontrolador. 2. Possui Motores para fazer os movimentos de locomoção.

Quadro 06 –Diferenças de Trabalhos Relacionados.
Fonte: do próprio autor

Existe várias semelhanças e diferenças entre os trabalhos relacionados como foi citado no Quadro 02, porem são essas semelhanças e diferença que podemos utilizar de modelo para o desenvolvimento do Bípede. O Bípede possui 2 pernas como o **ASIMO,** possui um Microcontrolador como o **Omnibot i-SODoG** e o **RiSE,** e possui um controle remoto como o **Omnibot i-SODoG** para poder ser controlado a distância e fazer todos os movimentos que o usuário desejar.

Um dos principais pontos que podemos utilizar no desenvolvimento do Bípede é como o Microcontrolador é utilizado em relação aos motores e em relação ao controle, já que alguns possui controle remoto e outros possui uma programação definida. Porém o Microcontrolador que foi utilizado nos trabalhos relacionados não e igual ao Microcontrolador que está sendo utilizado no desenvolvimento do Bípede, no Bípede será utilizado o Microcontrolador ATMega na plataforma do Arduino como já foi mencionado acima.

4. Desenvolvimento

O desenvolvimento do projeto foi dividido em várias partes, como, Analise dos Servo-motores que se consiste em qual seria o melhor servo que se adaptaria no protótipo, a Analise e seleção dos componentes eletrônicos, construção do protótipo, o desenvolvimento do código.

Para que possamos comandar os motores, responsáveis pela movimentação foi feito um estudo onde se determinou uma sequência de comandos, que e responsável pelo movimento do Bípede, como mostra o subitem 4.4. Programação.

4.1 Analise dos Servo-motores

Foram analisados dois modelos de servo motores para selecionar o mais viável para o projeto. Os testes realizados com o servo-motor descritos a seguir, foram realizados no início do período de montagem da estrutura física do Bípede.

Standart Analog Servo Motor HK15138

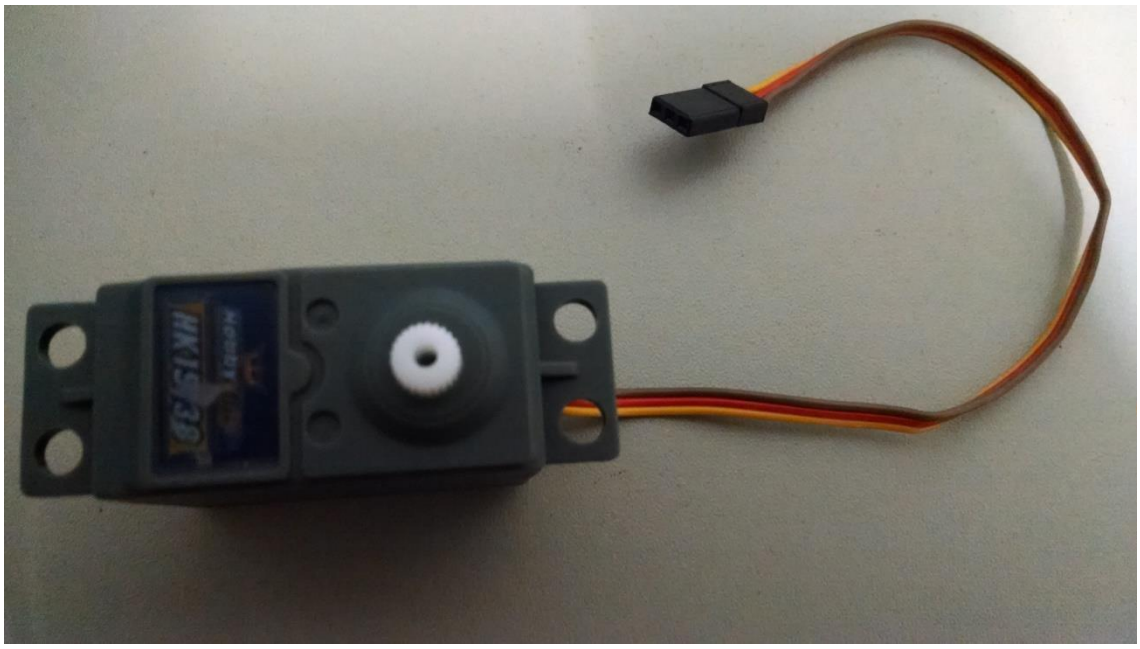

Imagem 06 - Standart Analog Servo Motor HK15138. Fonte: do próprio autor

Standart Analog Servo Motor HK15138 foi o primeiro modelo analisado, este servo motor possui torque de 4,3 kg/cm, trabalha em 6V e pesa apenas 38g, seu custo é em torno de R$25,00 (vinte e cinco reais) no mercado brasileiro. Este modelo pode ser visto na imagem 06.

4.2 Analise e seleção de componentes eletrônicos

Foi feita uma análise para identificar quais componentes eletrônicos seriam necessários para o desenvolvimento do projeto.

Como o Bípede possui seis junções, e cada junção possui seu movimento atuado individualmente, foi necessária a utilização de seis servo motores Standart Analog Servo Motor HK15138. Para o desenvolvimento do circuito de controle, foram usados um *protoboard*, um "Arduino", alguns fios extensores e fios jumper. Para a alimentação dos servos motores foi necessária a utilização de uma bateria *Lipo* "Turnigy" de 11.1V e 2200 mAh, 1 reguladore de tensão L7806CV, 1 capacitore cerâmico de 0.33µF (microfarads), um dissipador de calor e um *cooler*.

A Imagem 07 mostra o diagrama do circuito de alimentação dos servo motores.

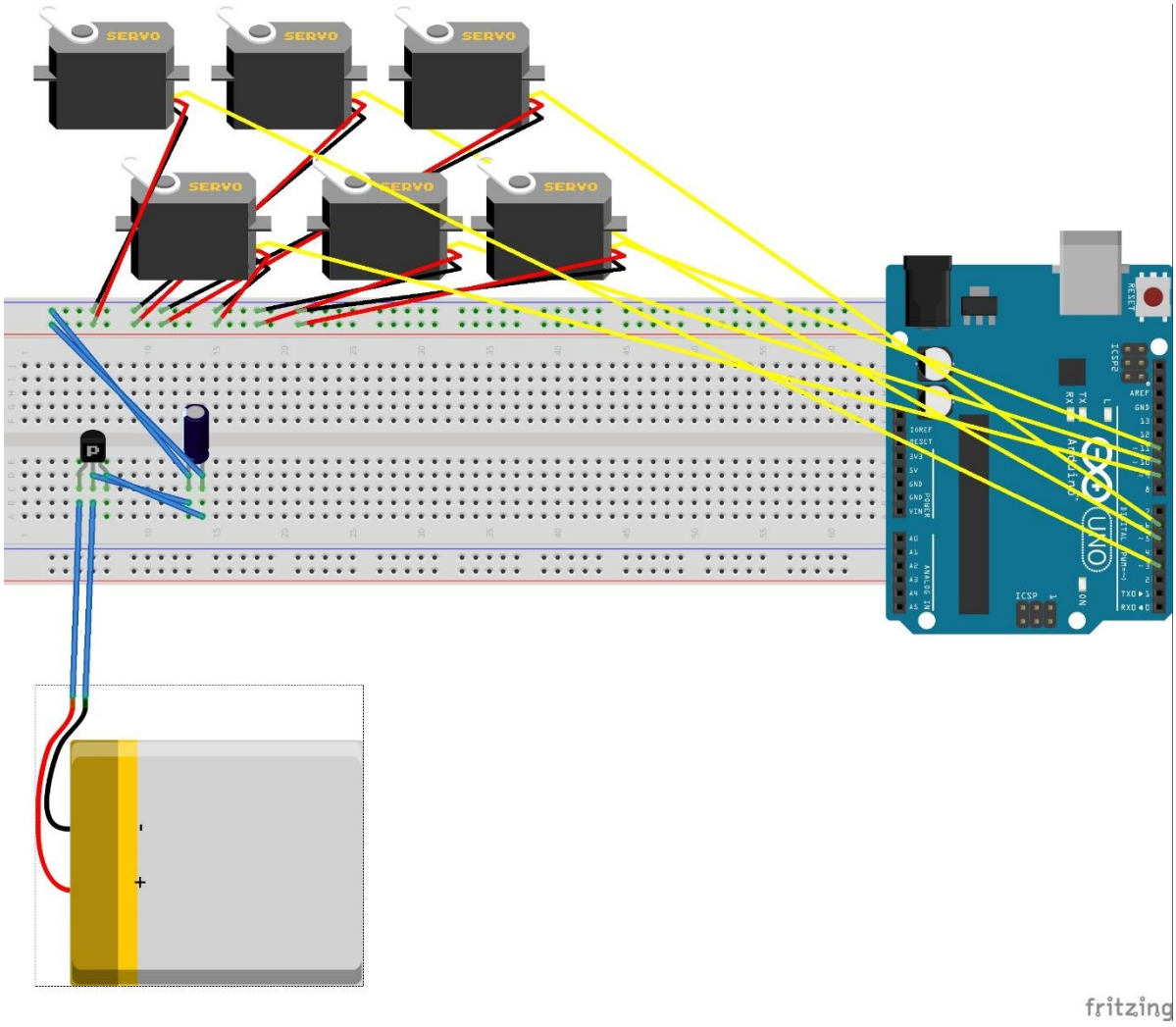

Imagem 07 - Diagrama do circuito de alimentação. Fonte: do próprio autor

4.2.1 Aquisição de materiais e componentes eletrônicos

Devido ao seu custo elevado no Brasil, o “Arduino”, o *protoboard*, os servo motores e os fios *jumper*, foram importados da China.

No mercado brasileiro foram adquiridos a bateria, os resistores, os capacitores e os reguladores de tensão. As ferramentas necessárias para o desenvolvimento e testes do circuito como: multímetro, solda, ferro de solda e sugador de solda. Os materiais que foram utilizados para o desenvolvimento o Bípede: acrílico, alumínio, parafusos, cola quente, arame de perca e braçadeiras. As ferramentas de corte, perfuração e acabamento como: furadeira, alicate, serra de corte, lixadeira, pistola de cola, chaves de fenda e *philips*.

4.3 Circuito

Houve um problema no desenvolvimento do circuito de alimentação dos servo motores, pois inicialmente utilizou-se uma bateria de 9V comum. A bateria tinha uma corrente muito baixa para alimentar os 6 Standart Analog Servo Motor HK15138, que consomem 140mA (miliampère) cada um. A bateria fornecia cerca de apenas 30mA conforme a medição do multímetro.

Para resolver esse problema utilizou-se uma bateria “Lipo” de 2200mA e 11.1V, essa bateria atendeu às necessidades do circuito, mas houve outro problema. A bateria era ligada a um regulador de tenção que aquecia muito e parava de funcionar, devido a esse regulador de tensão suportar apenas 1,5A enquanto a bateria fornece 2.2A, para resolver esse problema utilizou-se dois regulares de tensão ligados em paralelo, assim conseguiu-se uma capacidade total de operação de 3A pelos dois reguladores. Além disso utilizou-se um dissipador de calor juntamente com um *cooler* de *notebook* para refrigerar os reguladores de tensão.

4.4 Construção do Protótipo

A construção do protótipo foi feita com chapa de ferro n° 20, devido a precisão do corte da chapa, ela teve que ser feito em uma máquina de corte a laser.

Imagem 08 – Peças cortadas ao laser, Fonte do próprio autor

A imagem 08 mostra as peças que foram cortadas no laser, todas essas peças foram utilizadas para a construção do protótipo, o valor médio das peças foi no valor total de R$ 450,00.

A imagem 09 apresenta como foi montada a primeira perna do Bípede, onde possui uma braçadeira azul significa que o motor precisa ficar fixado naquela parte, ou seja, precisa ficar colado ou parafusado, alguma maneira que fique bem firme.

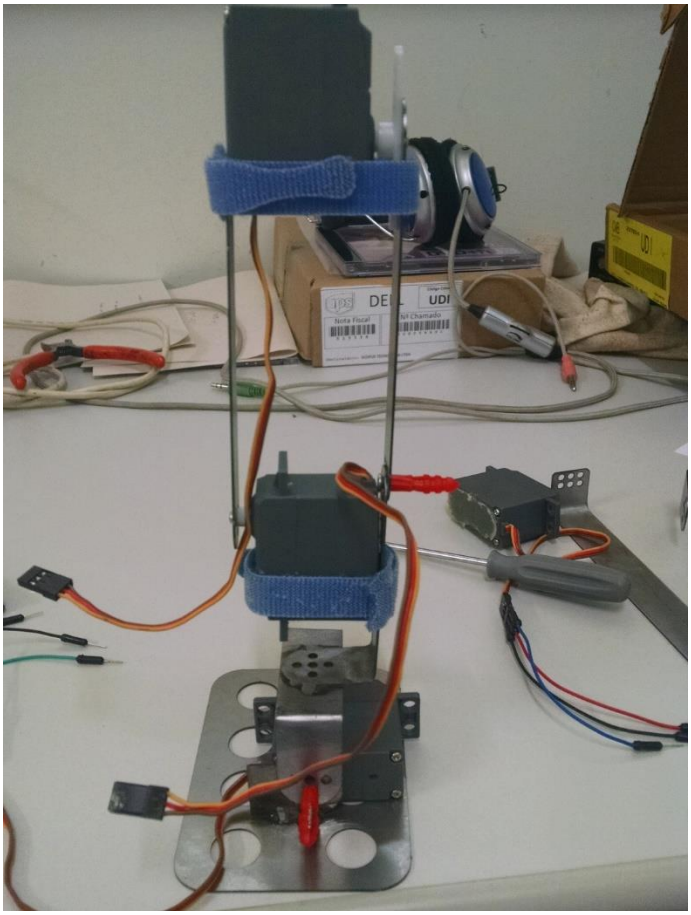

Imagem 09 – Primeira perna Montada, Fonte do próprio autor

A construção do protótipo foi desenvolvida para se adequar aos servo-motores, foi construída sob medida. A construção do protótipo demorou mais que o esperado pelo fato ser sob medida e precisar de um corte mais preciso, como a máquina de corte e uma máquina de um auto custo e não se encontra em qualquer lugar, teve que se descolar até a cidade de Uberlândia para poder fazer o corte. A imagem 10 mostra o Bípede todo montado.

Imagem 10 – Bípede Completo. Fonte do próprio autor

4.5 Programação

A programação escolhida foi a própria do Arduino, sua programação é simples e de fácil entendimento, no código primeiramente deve-se declarar os motores que vão utilizar, as portas que vão ser ligadas os motores devem ser de PWM para poder delimitar o pulso que

vai para o motor. Como mostra o quadro 03, os servos estão declarados já com suas portas ou seja o pino que vai ser ligado na porta do próprio Arduino.

```
void setup() {
  servo_1.attach(3); //Pino do arduino do Servo 1  (D3)
  servo_2.attach(5); //Pino do arduino do Servo 2  (D5)
  servo_3.attach(6); //Pino do arduino do Servo 3  (D6)
  servo_4.attach(9); //Pino do arduino do Servo 4  (D9)
  servo_5.attach(10); //Pino do arduino do Servo 5  (D10)
  servo_6.attach(11); //Pino do arduino do Servo 6  (D11)
  Serial.begin(9600); // Abre conexão serial e define a taxa de transmissão em 9.600kbps
}
```

Quadro 03 – Declaração dos servo-motores. Fonte: do autor

Depois de declarado os motores, deve-se lembrar que os servo-motores só trabalham com 180° e com isso é necessário limitar os movimentos do servo para que não force muito e não queime. Como mostra o quadro 04, o valor do incremento que está de vermelho significa que ainda não foi decido que o valor a ser usado vai ser o mesmo que está colocado no quadro.

```
int minPulse_1     =  0;   // Posição (angular) mínima do servo
int maxPulse_1     =  180; // Posição (angular) máxima do servo
int incrementoMotor_1      =   10;   // Incremento do servo (Maior valor = maior velocidade, Menor valor = Mais precisão)

int minPulse_2     =  0;   // Posição (angular) mínima do servo
int maxPulse_2     =  180; // Posição (angular) máxima do servo
int incrementoMotor_2      =   10;   // Incremento do servo (Maior valor = maior velocidade, Menor valor = Mais precisão)

int minPulse_3     =  0;   // Posição (angular) mínima do servo
int maxPulse_3     =  180; // Posição (angular) máxima do servo
int incrementoMotor_3      =   10;   // Incremento do servo (Maior valor = maior velocidade, Menor valor = Mais precisão)

int minPulse_4     =  0;   // Posição (angular) mínima do servo
int maxPulse_4     =  180; // Posição (angular) máxima do servo
int incrementoMotor_4      =   10;   // Incremento do servo (Maior valor = maior velocidade, Menor valor = Mais precisão)

int minPulse_5     =  0;   // Posição (angular) mínima do servo
int maxPulse_5     =  180; // Posição (angular) máxima do servo
int incrementoMotor_5      =   10;   // Incremento do servo (Maior valor = maior velocidade, Menor valor = Mais precisão)

int minPulse_6     =  0;   // Posição (angular) mínima do servo
int maxPulse_6     =  180; // Posição (angular) máxima do servo
int incrementoMotor_6      =   10;   // Incremento do servo (Maior valor = maior velocidade, Menor valor = Mais precisão)

int pulseWidth_1   = 0;        // Largura do Pulso do servo (PWM)
int pulseWidth_2   = 0;        // Largura do Pulso do servo (PWM)
int pulseWidth_3   = 0;        // Largura do Pulso do servo (PWM)
int pulseWidth_4   = 0;        // Largura do Pulso do servo (PWM)
int pulseWidth_5   = 0;        // Largura do Pulso do servo (PWM)
int pulseWidth_6   = 0;        // Largura do Pulso do servo (PWM)
```

Quadro 04 – Limitação dos servo-motores. Fonte: do autor

O comando do Bípede será feito através do computador ligado por um fio, o comando será feito com leitura do teclado em um monitor serial do próprio Arduino, o quadro 05 apresenta como e feita o movimento do motor, já a Imagem 11 apresenta como é o monitor serial.

Quando pressionar a tecla “a” no monitor serial o motor 1 vai movimentar para frente, quando pressionar a tecla “q” o motor 1 vai movimentar para trás, e do mesmo modo que os outros motores funcionam.

```
Void loop() {
 if (Serial.available() > 0) { // verifica serial

  int data = Serial.read();     // lê o byte na serial
  switch(data)
  {
  case ‘a’ :
   {
    pulseWidth_1 = pulseWidth_1 + incrementoMotor_1;

    break;
   }
  case ‘q’ :
   {
    pulseWidth_1 = pulseWidth_1 – incrementoMotor_1;
    break ;
   }
  case ‘s’ :
   {
    pulseWidth_2 = pulseWidth_2 – incrementoMotor_2;
    break;
   }
  case ‘w’ :
   {
    pulseWidth_2 = pulseWidth_2 + incrementoMotor_2;
    break ;
   }
```

Quadro 05 – Comando para o Bípede. Fonte: do autor

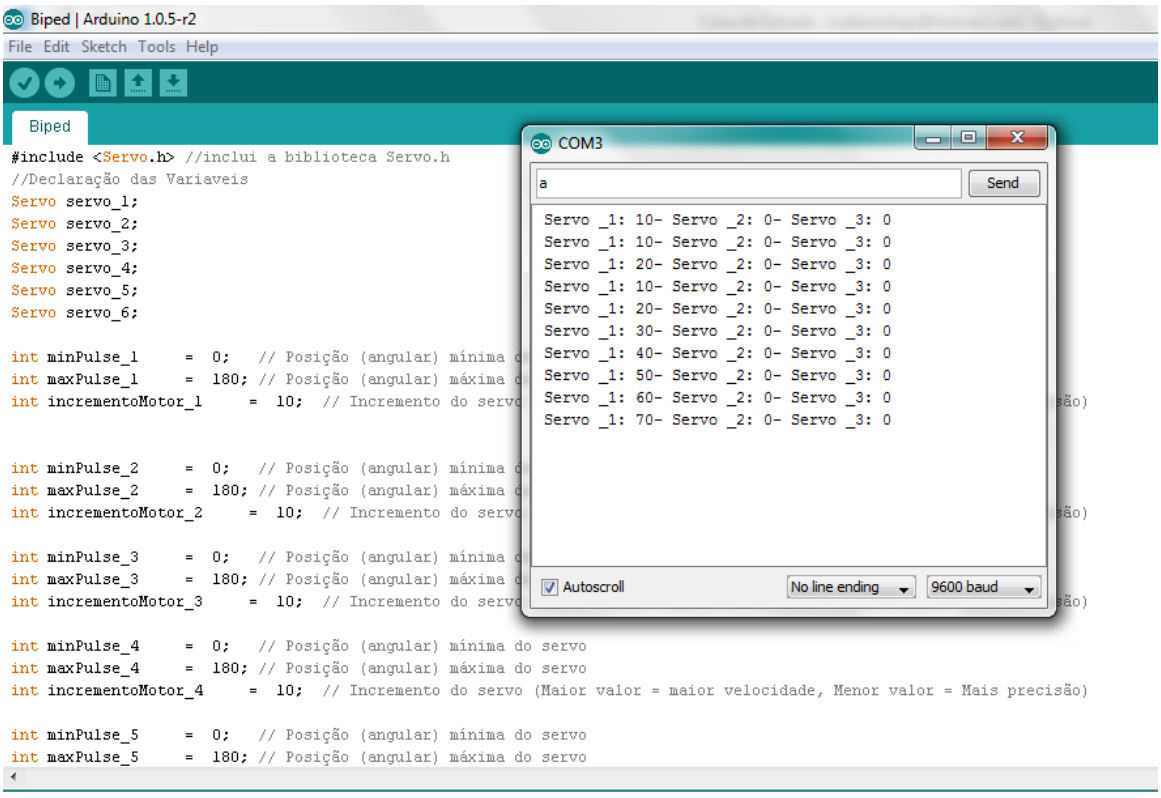

Imagem 11 – Monitor serial arduino. Fonte: do autor

Como mostra a imagem 11, quando se lê do teclado a tecla “a” e apertar a tecla send, o servo_1 irá se movimentar 10° para frente, quando se lê a tecla “q” o servo_1 irá movimentar 10 para a esquerda.

4.6 Teste

Após a conclusão da montagem da parte estrutural, eletrônica e da programação do Bípede, foram feitos testes para eventuais ajustes.

O código desenvolvido controlou perfeitamente o bípede, A única observação que foi constatada através dos testes de funcionalidade, que quando a perna se movimentava elas se juntavam então terá que colocar uma chapa no meio dela para não se juntar.

A imagem 12 mostra uma parte do teste, que é testando o motor e como ele vai se movimentar. Na imagem 12 você pode ver que o motor este movimenta toda estrutura ligada a ele.

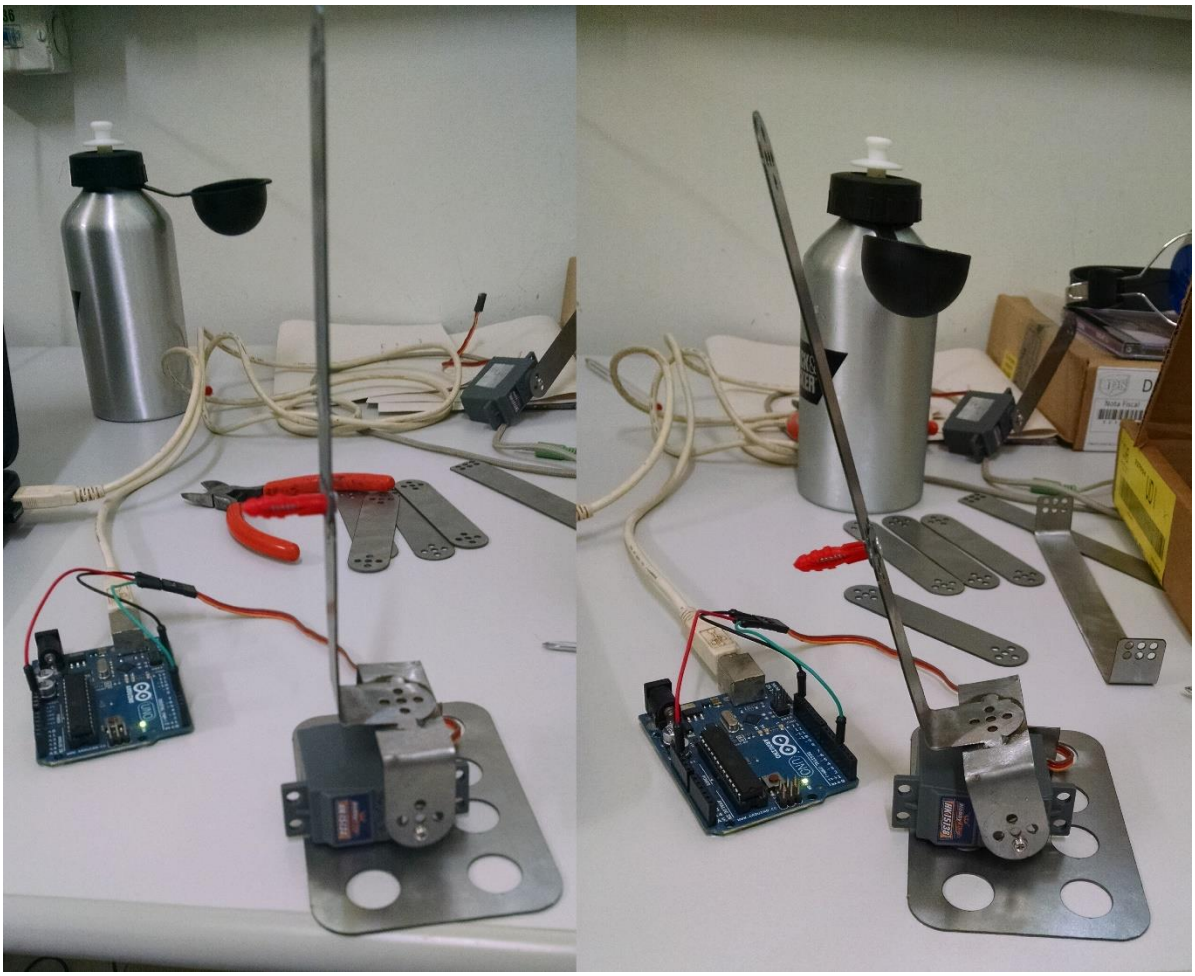

Imagem 12 – Movimento pelo motor. Fonte do próprio autor

Para fazer esse movimento a parte do código utilizado e a parte do quadro 05 que mostra como incrementa.

Foram feitos testes com o Bípede para identificar se ela seria capaz de se movimentar, como pode ser observado nas figuras.

Imagem 13 – Movimentação. Fonte do próprio autor

Imagem 14 – Movimentação. Fonte do próprio autor

A velocidade e a intensidade da força exercida pelos motores podem ocasionar algum problema como: queimar os motores, forçar para um lado e perder o equilíbrio, e demais coisas que o acaso por ocasionar.

5. Conclusões

Com o avanço constante da tecnologia, a robótica vem sendo cada vez mais desenvolvida, mas a maioria das tecnologias robóticas não são acessíveis para grande parte da sociedade. Pelo fato de ser uma plataforma de *hardware* livre, a utilização do "Arduino" no desenvolvimento de tecnologias robóticas, faz com que os projetos tenham custo mais viável, podendo assim expandir ainda mais o surgimento de novas ideias e novas tecnologias que auxiliem a sociedade.

Com o desenvolvimento desse projeto expõe-se uma alternativa mais viável, que pode ser utilizada em carregamento de pesos, como por exemplo, em atividades de construção, pode-se também utilizar como brinquedo, um robô equipado com uma perna robótica como essa poderia ser utilizado para fazer movimentos que uma roda não consegue, ou chegar a lugares que outro dispositivo não chegaria.

Inicialmente foi realizada uma pesquisa bibliográfica sobre os robôs bípedes e desenvolvido o planejamento para o início dos testes com os componentes que serão utilizados na sua construção. A proposta do projeto foi construir a perna robótica (Bípede) que possui 6 juntas que são duas para os Pés, duas para os Joelhos e duas para a Cintura humana, o Bípede é controlado por um usuário através do computador ligado por um fio, o usuário utilizará o programa Arduino, e dentro desse programa ele irá abrir o centro de comandos que enviaria ao "Arduino" os comandos que por sua vez acionaria um dos 6 servos motores, executando o movimento do Bípede que o usuário desejar.

Para o desenvolvimento do projeto vários componentes eletrônicos tiveram que ser importados da China e dos Estados Unidos, pois não existiam no Brasil ou por terem um custo elevado no mercado nacional.

Para o teste dos servo-motores foi necessário desenvolver uma parte do código principal onde os 6 servo-motores estão ligados diretamente no Arduino. Para o desenvolvimento do chassi foi necessário a compra de algumas peças, e o desenvolvimento de outras.

O chassi do Bípede já está sendo customizado é alguns objetos essenciais para o funcionamento do Bípede já foram adquiridos, como os 6 servo-motores, o Arduino, fios *jumper*.

Os servo-motores possuem 3 fios, um de tensão (que conduz a energia), um terra e um fio de controle. O fio de controle é o fio que transmite o comando do computador para o servo, sendo assim o controle de cada um dos motores e feito por um Microcontrolador que permite controlar todos os motores dizendo a angulação e o caminho para onde deve se mover.

Com o protótipo customizado viu-se que e necessário mais massa para o Bípede se movimente sozinho, foi encontrada duas soluções para a solução do problema:

5.1. A longo prazo: melhorar a estrutura, aumentando a massa colocando uma chapa de ferro mais grossa.
5.2. Imediata: Impulsionar uma perna para poder movimentar a outra.

6. Trabalhos Futuros

Para trabalhos futuros, pode-se, fazer várias projetos como:

6.1. Implementação de um sistema wifi ou um sistema a radio, para controle a distância. Para essa implementação e necessária uma melhoria do protótipo com uma massa ideia suficiente para carregar todo o peso da estrutura juntamente com os componentes.
6.2. Implementação do restante do robô. Implementar cintura, braço, cabeça, pescoço, o resto do corpo, imitando o ser humano.

Referências

1. BRAIN, M. UOL. **comotudofunciona**, 1998. Disponivel em: <http://eletronicos.hsw.uol.com.br/microcontroladores1.htm>. Acesso em: 15 out. 2013.

2. HONDA Worldwide. **Honda Worldwide**, 2014. Disponivel em: <http://world.honda.com/ASIMO/new/>. Acesso em: 14 mar. 2014.

3. LÉXICO. Léxico: dicionário de português online. **Léxico**, 2009. Disponivel em: <http://www.lexico.pt/bipede/>. Acesso em: 15 out. 2013.

4. LOPES, D. Sistema Locomotor. **Mundo Educação**, 2014. Disponivel em: <http://www.mundoeducacao.com/biologia/sistema-locomotor.htm>. Acesso em: 16 fev. 2014.

5. MAGNO , C. F. D. S.; BORINI, L. L.; FERNANDO, L. P. TUTORIAL ROBÔ BÍPEDE. **Greenitbrasil**, 2006. Disponivel em: <http://www.greenitbrasil.com.br/downloads/documentos/projeto%20-robo.pdf>. Acesso em: 14 mar. 2014.

6. MANTOVANI, S. C. A.; OKI, N. Unesp. **Unesp**, 2013. Disponivel em: <http://www.feis.unesp.br/Home/departamentos/engenhariaeletrica/aula-4---servo-motor-13-03-2013-final.pdf>. Acesso em: 20 out. 2013.

7. NOGUEIRA, K. L. **Introdução a Robotica**. Fundação Educacional de Ituiutaba, Universidade do Estado de Minas Gerais. Ituiutaba. 2013.

8. RIOS, J. et al. Destacom. **Destacom**, 2013. Disponivel em: <http://destacom.ufms.br/mediawiki/images/9/9f/Arduino_Destacom.pdf>. Acesso em: 20 out. 2013.

9. SOUZA, J. A. M. F. D. J. A. M. Felippe De Souza. **J. A. M. Felippe De Souza**, 2013. Disponivel em: <http://webx.ubi.pt/~felippe/main_pgs/mat_didp.htm>. Acesso em: 10 out. 2013.

10. TAKARA Tomy Developing Robot Dog – Omnibot i-SODOG. **Robots Dreams**, 2011. Disponivel em: <http://www.robots-dreams.com/2012/06/breaking-news-takara-tomy-developing-robot-dog-omnibot-i-sodog.html>. Acesso em: 02 mar. 2014.

11. MCROBERTS, Michel. **Arduino Básico.** Novatec Editora Ttda 2011

Apêndice A

```
#include <Servo.h> //inclui a biblioteca Servo.h
//Declaração das Variaveis
Servo servo_1;
Servo servo_2;
Servo servo_3;
Servo servo_4;
Servo servo_5;
Servo servo_6;

int minPulse_1    =  0;   // Posição (angular) mínima do servo
int maxPulse_1    =  180; // Posição (angular) máxima do servo
int incrementoMotor_1    =  5;  // Incremento do servo (Maior valor = maior velocidade, Menor valor = Mais precisão)

int minPulse_2    =  0;   // Posição (angular) mínima do servo
int maxPulse_2    =  180; // Posição (angular) máxima do servo
int incrementoMotor_2    =  5;  // Incremento do servo (Maior valor = maior velocidade, Menor valor = Mais precisão)

int minPulse_3    =  0;   // Posição (angular) mínima do servo
int maxPulse_3    =  180; // Posição (angular) máxima do servo
int incrementoMotor_3    =  5;  // Incremento do servo (Maior valor = maior velocidade, Menor valor = Mais precisão)

int minPulse_4    =  0;   // Posição (angular) mínima do servo
int maxPulse_4    =  180; // Posição (angular) máxima do servo
int incrementoMotor_4    =  5;  // Incremento do servo (Maior valor = maior velocidade, Menor valor = Mais precisão)

int minPulse_5    =  0;   // Posição (angular) mínima do servo
int maxPulse_5    =  180; // Posição (angular) máxima do servo
int incrementoMotor_5    =  5; // Incremento do servo (Maior valor = maior velocidade, Menor valor = Mais precisão)

int minPulse_6    =  0;   // Posição (angular) mínima do servo
int maxPulse_6    =  180; // Posição (angular) máxima do servo
int incrementoMotor_6    =  5;  // Incremento do servo (Maior valor = maior velocidade, Menor valor = Mais precisão)


int pulseWidth_1   = 100;       // Largura do Pulso do servo (PWM)
int pulseWidth_2   = 90;        // Largura do Pulso do servo (PWM)
int pulseWidth_3   = 80;        // Largura do Pulso do servo (PWM)
int pulseWidth_4   = 90;        // Largura do Pulso do servo (PWM)
```

```
int pulseWidth_5   = 90;        // Largura do Pulso do servo (PWM)
int pulseWidth_6   = 100;        // Largura do Pulso do servo (PWM)

void setup() {
  servo_1.attach(3); //Pino do arduino do Servo 1  (D3)
  servo_2.attach(5); //Pino do arduino do Servo 2  (D5)
  servo_3.attach(6); //Pino do arduino do Servo 3  (D6)
  servo_4.attach(9); //Pino do arduino do Servo 4  (D9)
  servo_5.attach(10); //Pino do arduino do Servo 5  (D10)
  servo_6.attach(11); //Pino do arduino do Servo 6  (D11)
  Serial.begin(9600); // Abre conexão serial e define a taxa de transmissão em 9.600kbps
}
void loop() {
  if (Serial.available() > 0) { // verifica serial

    int data = Serial.read();     // lê o byte na serial
    switch(data)
    {
    case 'a' :
      {
       pulseWidth_1 = pulseWidth_1 + incrementoMotor_1;
       break;
      }
    case 'q' :
      {
       pulseWidth_1 = pulseWidth_1 - incrementoMotor_1;
       break ;
      }
    case 's' :
      {
       pulseWidth_2 = pulseWidth_2 + incrementoMotor_2;
       break;
      }
    case 'w' :
      {
       pulseWidth_2 = pulseWidth_2 - incrementoMotor_2;
       break ;
      }
    case 'd' :
      {
       pulseWidth_3 = pulseWidth_3 + incrementoMotor_3;
       break;
      }
    case 'e' :
```

```
  {
   pulseWidth_3 = pulseWidth_3 - incrementoMotor_3;
   break;
  }
  case 'f' :
  {
   pulseWidth_4 = pulseWidth_4 + incrementoMotor_4;
   break;
  }
case 'r' :
  {
   pulseWidth_4 = pulseWidth_4 - incrementoMotor_4;
   break;
  }
case 'g' :
  {
   pulseWidth_5 = pulseWidth_5 - incrementoMotor_5;
   break;
  }
case 't' :
  {
   pulseWidth_5 = pulseWidth_5 + incrementoMotor_5;
   break;
  }
case 'h' :
  {
   pulseWidth_6 = pulseWidth_6 - incrementoMotor_6;
   break;
  }
case 'y' :
  {
   pulseWidth_6 = pulseWidth_6 + incrementoMotor_6;
   break;
  }
case 'z' :   //levantar pra esquerda
  {
   pulseWidth_4 = pulseWidth_4 - incrementoMotor_4;
   delay(40);
   servo_4.write(pulseWidth_4);
   pulseWidth_4 = pulseWidth_4 - incrementoMotor_4;
   delay(40);
   servo_4.write(pulseWidth_4);
   pulseWidth_4 = pulseWidth_4 - incrementoMotor_4;
   delay(40);
```

```
servo_4.write(pulseWidth_4);
pulseWidth_4 = pulseWidth_4 - incrementoMotor_4;
delay(40);
servo_4.write(pulseWidth_4);
pulseWidth_4 = pulseWidth_4 - incrementoMotor_4;
delay(40);
servo_4.write(pulseWidth_4);
pulseWidth_4 = pulseWidth_4 - incrementoMotor_4;
delay(40);
servo_4.write(pulseWidth_4);
pulseWidth_4 = pulseWidth_4 - incrementoMotor_4;
delay(40);
servo_4.write(pulseWidth_4);
pulseWidth_4 = pulseWidth_4 - incrementoMotor_4;
delay(40);
servo_4.write(pulseWidth_4);
pulseWidth_4 = pulseWidth_4 - incrementoMotor_4;
delay(40);
pulseWidth_1 = pulseWidth_1 + incrementoMotor_1;
delay(40);
servo_1.write(pulseWidth_1);
pulseWidth_1 = pulseWidth_1 + incrementoMotor_1;
delay(40);
servo_1.write(pulseWidth_1);
pulseWidth_1 = pulseWidth_1 + incrementoMotor_1;
delay(40);
servo_1.write(pulseWidth_1);
pulseWidth_1 = pulseWidth_1 + incrementoMotor_1;
delay(40);
servo_1.write(pulseWidth_1);
pulseWidth_1 = pulseWidth_1 + incrementoMotor_1;
delay(40);
servo_4.write(pulseWidth_4);
pulseWidth_4 = pulseWidth_4 + incrementoMotor_4;
delay(40);
servo_4.write(pulseWidth_4);
pulseWidth_1 = pulseWidth_1 + incrementoMotor_1;
delay(40);
servo_1.write(pulseWidth_1);
pulseWidth_4 = pulseWidth_4 + incrementoMotor_4;
delay(40);
servo_4.write(pulseWidth_4);
pulseWidth_4 = pulseWidth_4 + incrementoMotor_4;
delay(40);
```

```
    servo_4.write(pulseWidth_4);
    pulseWidth_4 = pulseWidth_4 + incrementoMotor_4;
    delay(40);
    servo_4.write(pulseWidth_4);
    pulseWidth_4 = pulseWidth_4 + incrementoMotor_4;
    delay(40);
    servo_4.write(pulseWidth_4);
    pulseWidth_4 = pulseWidth_4 + incrementoMotor_4;
    delay(40);
    servo_4.write(pulseWidth_4);
    pulseWidth_4 = pulseWidth_4 + incrementoMotor_4;
    delay(40);
    servo_4.write(pulseWidth_4);
    pulseWidth_4 = pulseWidth_4 + incrementoMotor_4;
    delay(40);
    servo_4.write(pulseWidth_4);
    pulseWidth_4 = pulseWidth_4 + incrementoMotor_4;
    delay(40);
    servo_4.write(pulseWidth_4);
    break;
  }
case 'x' :    //voltar pra esquerda
  {
    pulseWidth_1 = pulseWidth_1 - incrementoMotor_1;
    delay(100);
    servo_1.write(pulseWidth_1);
    pulseWidth_1 = pulseWidth_1 - incrementoMotor_1;
    delay(100);
    servo_1.write(pulseWidth_1);
    pulseWidth_1 = pulseWidth_1 - incrementoMotor_1;
    delay(100);
    servo_1.write(pulseWidth_1);
    pulseWidth_1 = pulseWidth_1 - incrementoMotor_1;
    delay(100);
    servo_1.write(pulseWidth_1);
    pulseWidth_1 = pulseWidth_1 - incrementoMotor_1;
    delay(100);
    servo_1.write(pulseWidth_1);
    pulseWidth_1 = pulseWidth_1 - incrementoMotor_1;
    delay(100);
    break;
  }
case 'c' :   //levantar pra direita
  {
```

```
pulseWidth_1 = pulseWidth_1 - incrementoMotor_1;
delay(40);
servo_1.write(pulseWidth_1);
 pulseWidth_1 = pulseWidth_1 - incrementoMotor_1;
delay(40);
servo_1.write(pulseWidth_1);
 pulseWidth_1 = pulseWidth_1 - incrementoMotor_1;
delay(40);
servo_1.write(pulseWidth_1);
 pulseWidth_1 = pulseWidth_1 - incrementoMotor_1;
delay(40);
servo_1.write(pulseWidth_1);
pulseWidth_1 = pulseWidth_1 - incrementoMotor_1;
delay(40);
servo_1.write(pulseWidth_1);
pulseWidth_1 = pulseWidth_1 - incrementoMotor_1;
delay(40);
servo_1.write(pulseWidth_1);
pulseWidth_1 = pulseWidth_1 - incrementoMotor_1;
delay(40);
servo_1.write(pulseWidth_1);
pulseWidth_4 = pulseWidth_4 - incrementoMotor_4;
delay(40);
servo_4.write(pulseWidth_4);
pulseWidth_4 = pulseWidth_4 - incrementoMotor_4;
delay(40);
servo_4.write(pulseWidth_4);
pulseWidth_4 = pulseWidth_4 - incrementoMotor_4;
delay(40);
servo_4.write(pulseWidth_4);
pulseWidth_4 = pulseWidth_4 - incrementoMotor_4;
delay(40);
servo_4.write(pulseWidth_4);
pulseWidth_4 = pulseWidth_4 - incrementoMotor_4;
delay(40);
servo_4.write(pulseWidth_4);
pulseWidth_1 = pulseWidth_1 + incrementoMotor_1;
delay(40);
servo_1.write(pulseWidth_1);
pulseWidth_1 = pulseWidth_1 + incrementoMotor_1;
delay(40);
servo_1.write(pulseWidth_1);
pulseWidth_1 = pulseWidth_1 + incrementoMotor_1;
delay(40);
```

```
    servo_1.write(pulseWidth_1);
    pulseWidth_1 = pulseWidth_1 + incrementoMotor_1;
    delay(40);
    servo_1.write(pulseWidth_1);
    pulseWidth_1 = pulseWidth_1 + incrementoMotor_1;
    delay(40);
    servo_1.write(pulseWidth_1);
    pulseWidth_1 = pulseWidth_1 + incrementoMotor_1;
    delay(40);
    servo_1.write(pulseWidth_1);
    pulseWidth_1 = pulseWidth_1 + incrementoMotor_1;
    delay(40);
    servo_1.write(pulseWidth_1);
    break;
  }
case 'v' :   //voltar pra direita
  {
    pulseWidth_4 = pulseWidth_4 + incrementoMotor_4;
    delay(100);
    servo_4.write(pulseWidth_4);
    pulseWidth_4 = pulseWidth_4 + incrementoMotor_4;
    delay(100);
    servo_4.write(pulseWidth_4);
    pulseWidth_4 = pulseWidth_4 + incrementoMotor_4;
    delay(100);
    servo_4.write(pulseWidth_4);
    pulseWidth_4 = pulseWidth_4 + incrementoMotor_4;
    delay(100);
    servo_4.write(pulseWidth_4);
    pulseWidth_4 = pulseWidth_4 + incrementoMotor_4;
    delay(100);
    servo_4.write(pulseWidth_4);
    pulseWidth_4 = pulseWidth_4 + incrementoMotor_4;
    delay(100);
    servo_4.write(pulseWidth_4);
    break;
  }
 case 'm' :   //passo completo
  {
    pulseWidth_1 = pulseWidth_1 + incrementoMotor_1;
    delay(60);
    servo_1.write(pulseWidth_1);
    pulseWidth_1 = pulseWidth_1 + incrementoMotor_1;
    delay(60);
```

```
servo_1.write(pulseWidth_1);
pulseWidth_1 = pulseWidth_1 + incrementoMotor_1;
delay(60);
servo_1.write(pulseWidth_1);
pulseWidth_1 = pulseWidth_1 + incrementoMotor_1;
delay(60);
servo_1.write(pulseWidth_1);
pulseWidth_1 = pulseWidth_1 + incrementoMotor_1;
delay(60);
servo_1.write(pulseWidth_1);
pulseWidth_4 = pulseWidth_4 + incrementoMotor_4;
delay(60);
servo_4.write(pulseWidth_4);
pulseWidth_4 = pulseWidth_4 + incrementoMotor_4;
delay(60);
servo_4.write(pulseWidth_4);
pulseWidth_4 = pulseWidth_4 + incrementoMotor_4;
delay(60);
servo_4.write(pulseWidth_4);
pulseWidth_4 = pulseWidth_4 + incrementoMotor_4;
delay(60);
servo_4.write(pulseWidth_4);
pulseWidth_4 = pulseWidth_4 + incrementoMotor_4;
delay(60);
servo_4.write(pulseWidth_4);
pulseWidth_4 = pulseWidth_4 + incrementoMotor_4;
delay(60);
servo_4.write(pulseWidth_4);
pulseWidth_4 = pulseWidth_4 + incrementoMotor_4;
delay(60);
servo_4.write(pulseWidth_4);
pulseWidth_4 = pulseWidth_4 + incrementoMotor_4;
delay(60);
servo_4.write(pulseWidth_4);
pulseWidth_4 = pulseWidth_4 + incrementoMotor_4;
delay(60);
servo_4.write(pulseWidth_4);
pulseWidth_4 = pulseWidth_4 + incrementoMotor_4;
delay(60);
servo_4.write(pulseWidth_4);
pulseWidth_4 = pulseWidth_4 - incrementoMotor_4;
```

```
delay(60);
servo_4.write(pulseWidth_4);
pulseWidth_4 = pulseWidth_4 - incrementoMotor_4;
delay(60);
servo_4.write(pulseWidth_4);
pulseWidth_4 = pulseWidth_4 - incrementoMotor_4;
delay(60);
servo_4.write(pulseWidth_4);
pulseWidth_4 = pulseWidth_4 - incrementoMotor_4;
delay(60);
servo_4.write(pulseWidth_4);
pulseWidth_4 = pulseWidth_4 - incrementoMotor_4;
delay(60);
servo_4.write(pulseWidth_4);
pulseWidth_4 = pulseWidth_4 - incrementoMotor_4;
delay(60);
servo_4.write(pulseWidth_4);
pulseWidth_4 = pulseWidth_4 - incrementoMotor_4;
delay(60);
servo_4.write(pulseWidth_4);
pulseWidth_4 = pulseWidth_4 - incrementoMotor_4;
delay(60);
servo_4.write(pulseWidth_4);
pulseWidth_4 = pulseWidth_4 - incrementoMotor_4;
delay(60);
servo_4.write(pulseWidth_4);
pulseWidth_4 = pulseWidth_4 - incrementoMotor_4;
delay(60);
servo_4.write(pulseWidth_4);
pulseWidth_3 = pulseWidth_3 + incrementoMotor_3;
delay(60);
servo_3.write(pulseWidth_3);
pulseWidth_3 = pulseWidth_3 + incrementoMotor_3;
delay(60);
servo_3.write(pulseWidth_3);
pulseWidth_3 = pulseWidth_3 + incrementoMotor_3;
delay(60);
servo_3.write(pulseWidth_3);
pulseWidth_3 = pulseWidth_3 + incrementoMotor_3;
delay(60);
servo_3.write(pulseWidth_3);
pulseWidth_3 = pulseWidth_3 + incrementoMotor_3;
delay(60);
servo_3.write(pulseWidth_3);
```

```
pulseWidth_2 = pulseWidth_2 + incrementoMotor_2;
delay(60);
servo_2.write(pulseWidth_2);
pulseWidth_2 = pulseWidth_2 + incrementoMotor_2;
delay(60);
servo_2.write(pulseWidth_2);
pulseWidth_2 = pulseWidth_2 + incrementoMotor_2;
delay(60);
servo_2.write(pulseWidth_2);
pulseWidth_1 = pulseWidth_1 - incrementoMotor_1;
delay(60);
servo_1.write(pulseWidth_1);
pulseWidth_1 = pulseWidth_1 - incrementoMotor_1;
delay(60);
servo_1.write(pulseWidth_1);
pulseWidth_1 = pulseWidth_1 - incrementoMotor_1;
delay(60);
servo_1.write(pulseWidth_1);
pulseWidth_1 = pulseWidth_1 - incrementoMotor_1;
delay(60);
servo_1.write(pulseWidth_1);
pulseWidth_1 = pulseWidth_1 - incrementoMotor_1;
delay(60);
servo_1.write(pulseWidth_1);
pulseWidth_1 = pulseWidth_1 - incrementoMotor_1;
delay(60);
servo_1.write(pulseWidth_1);
pulseWidth_1 = pulseWidth_1 - incrementoMotor_1;
delay(60);
servo_1.write(pulseWidth_1);
pulseWidth_4 = pulseWidth_4 - incrementoMotor_4;
delay(60);
servo_4.write(pulseWidth_4);
pulseWidth_4 = pulseWidth_4 - incrementoMotor_4;
delay(60);
servo_4.write(pulseWidth_4);
pulseWidth_4 = pulseWidth_4 - incrementoMotor_4;
delay(60);
servo_4.write(pulseWidth_4);
pulseWidth_4 = pulseWidth_4 - incrementoMotor_4;
delay(60);
servo_4.write(pulseWidth_4);
pulseWidth_1 = pulseWidth_1 - incrementoMotor_1;
delay(60);
```

```
servo_1.write(pulseWidth_1);
pulseWidth_1 = pulseWidth_1 - incrementoMotor_1;
delay(60);
servo_1.write(pulseWidth_1);
pulseWidth_1 = pulseWidth_1 - incrementoMotor_1;
delay(60);
servo_1.write(pulseWidth_1);
pulseWidth_1 = pulseWidth_1 - incrementoMotor_1;
delay(60);
servo_1.write(pulseWidth_1);
pulseWidth_1 = pulseWidth_1 - incrementoMotor_1;
delay(60);
servo_1.write(pulseWidth_1);
pulseWidth_1 = pulseWidth_1 - incrementoMotor_1;
delay(60);
servo_1.write(pulseWidth_1);
pulseWidth_4 = pulseWidth_4 - incrementoMotor_4;
delay(60);
servo_4.write(pulseWidth_4);
pulseWidth_4 = pulseWidth_4 - incrementoMotor_4;
delay(60);
servo_4.write(pulseWidth_4);
pulseWidth_3 = pulseWidth_3 - incrementoMotor_3;
delay(60);
servo_3.write(pulseWidth_3);
pulseWidth_3 = pulseWidth_3 - incrementoMotor_3;
delay(60);
servo_3.write(pulseWidth_3);
pulseWidth_3 = pulseWidth_3 - incrementoMotor_3;
delay(60);
servo_3.write(pulseWidth_3);
pulseWidth_1 = pulseWidth_1 + incrementoMotor_1;
delay(60);
servo_1.write(pulseWidth_1);
pulseWidth_1 = pulseWidth_1 + incrementoMotor_1;
delay(60);
servo_1.write(pulseWidth_1);
pulseWidth_1 = pulseWidth_1 + incrementoMotor_1;
delay(60);
servo_1.write(pulseWidth_1);
pulseWidth_1 = pulseWidth_1 + incrementoMotor_1;
delay(60);
servo_1.write(pulseWidth_1);
pulseWidth_4 = pulseWidth_4 + incrementoMotor_4;
```

```
  delay(60);
  servo_4.write(pulseWidth_4);
  pulseWidth_4 = pulseWidth_4 + incrementoMotor_4;
  delay(60);
  servo_4.write(pulseWidth_4);
  pulseWidth_4 = pulseWidth_4 + incrementoMotor_4;
  delay(60);
  servo_4.write(pulseWidth_4);
  pulseWidth_4 = pulseWidth_4 + incrementoMotor_4;
  delay(60);
  servo_4.write(pulseWidth_4);
  pulseWidth_1 = pulseWidth_1 + incrementoMotor_1;
  delay(60);
  servo_1.write(pulseWidth_1);
  pulseWidth_1 = pulseWidth_1 + incrementoMotor_1;
  delay(60);
  servo_1.write(pulseWidth_1);
  pulseWidth_1 = pulseWidth_1 + incrementoMotor_1;
  delay(60);
  servo_1.write(pulseWidth_1);
  pulseWidth_1 = pulseWidth_1 + incrementoMotor_1;
  delay(60);
  servo_1.write(pulseWidth_1);
  pulseWidth_2 = pulseWidth_2 - incrementoMotor_2;
  delay(60);
  servo_2.write(pulseWidth_2);
  pulseWidth_2 = pulseWidth_2 - incrementoMotor_2;
  delay(60);
  servo_2.write(pulseWidth_2);
  pulseWidth_2 = pulseWidth_2 - incrementoMotor_2;
  delay(60);
  servo_2.write(pulseWidth_2);
  pulseWidth_3 = pulseWidth_3 - incrementoMotor_3;
  delay(60);
  servo_3.write(pulseWidth_3);
  pulseWidth_3 = pulseWidth_3 - incrementoMotor_3;
  delay(60);
  servo_3.write(pulseWidth_3);
  pulseWidth_4 = pulseWidth_4 + incrementoMotor_4;
  delay(60);
  servo_4.write(pulseWidth_4);
  break;
 }
case '1' :
```

```
 {
  pulseWidth_3 = pulseWidth_3 - incrementoMotor_3;
  pulseWidth_6 = pulseWidth_6 + incrementoMotor_6;
  pulseWidth_3 = pulseWidth_3 - incrementoMotor_3;
  pulseWidth_6 = pulseWidth_6 + incrementoMotor_6;
  pulseWidth_3 = pulseWidth_3 - incrementoMotor_3;
  pulseWidth_6 = pulseWidth_6 + incrementoMotor_6;
  pulseWidth_3 = pulseWidth_3 - incrementoMotor_3;
  pulseWidth_6 = pulseWidth_6 + incrementoMotor_6;
  pulseWidth_3 = pulseWidth_3 - incrementoMotor_3;
  pulseWidth_6 = pulseWidth_6 + incrementoMotor_6;
  pulseWidth_3 = pulseWidth_3 - incrementoMotor_3;
  pulseWidth_6 = pulseWidth_6 + incrementoMotor_6;
  pulseWidth_3 = pulseWidth_3 - incrementoMotor_3;
  pulseWidth_6 = pulseWidth_6 + incrementoMotor_6;
  pulseWidth_3 = pulseWidth_3 - incrementoMotor_3;
  pulseWidth_6 = pulseWidth_6 + incrementoMotor_6;
  pulseWidth_3 = pulseWidth_3 - incrementoMotor_3;
  pulseWidth_6 = pulseWidth_6 + incrementoMotor_6;
  pulseWidth_3 = pulseWidth_3 - incrementoMotor_3;
  pulseWidth_6 = pulseWidth_6 + incrementoMotor_6;
  pulseWidth_3 = pulseWidth_3 - incrementoMotor_3;
  pulseWidth_6 = pulseWidth_6 + incrementoMotor_6;
  pulseWidth_3 = pulseWidth_3 - incrementoMotor_3;
  pulseWidth_6 = pulseWidth_6 + incrementoMotor_6;
  break;
 }
case '2' :
 {
  pulseWidth_3 = pulseWidth_3 + incrementoMotor_3;
  pulseWidth_6 = pulseWidth_6 - incrementoMotor_6;
  pulseWidth_3 = pulseWidth_3 + incrementoMotor_3;
  pulseWidth_6 = pulseWidth_6 - incrementoMotor_6;
  pulseWidth_3 = pulseWidth_3 + incrementoMotor_3;
  pulseWidth_6 = pulseWidth_6 - incrementoMotor_6;
  pulseWidth_3 = pulseWidth_3 + incrementoMotor_3;
  pulseWidth_6 = pulseWidth_6 - incrementoMotor_6;
  pulseWidth_3 = pulseWidth_3 + incrementoMotor_3;
  pulseWidth_6 = pulseWidth_6 - incrementoMotor_6;
  pulseWidth_3 = pulseWidth_3 + incrementoMotor_3;
  pulseWidth_6 = pulseWidth_6 - incrementoMotor_6;
  pulseWidth_3 = pulseWidth_3 + incrementoMotor_3;
  pulseWidth_6 = pulseWidth_6 - incrementoMotor_6;
  pulseWidth_3 = pulseWidth_3 + incrementoMotor_3;
```

```
    pulseWidth_6 = pulseWidth_6 - incrementoMotor_6;
    pulseWidth_3 = pulseWidth_3 + incrementoMotor_3;
    pulseWidth_6 = pulseWidth_6 - incrementoMotor_6;
    pulseWidth_3 = pulseWidth_3 + incrementoMotor_3;
    pulseWidth_6 = pulseWidth_6 - incrementoMotor_6;
    pulseWidth_3 = pulseWidth_3 + incrementoMotor_3;
    pulseWidth_6 = pulseWidth_6 - incrementoMotor_6;
    pulseWidth_3 = pulseWidth_3 + incrementoMotor_3;
    pulseWidth_6 = pulseWidth_6 - incrementoMotor_6;
    break;
  }
case '3' :
  {
    pulseWidth_2 = pulseWidth_2 + incrementoMotor_2;
    pulseWidth_5 = pulseWidth_5 - incrementoMotor_5;
    delay(100);
    servo_2.write(pulseWidth_2);
    servo_5.write(pulseWidth_5);
    pulseWidth_2 = pulseWidth_2 + incrementoMotor_2;
    pulseWidth_5 = pulseWidth_5 - incrementoMotor_5;
    delay(100);
    servo_2.write(pulseWidth_2);
    servo_5.write(pulseWidth_5);
    pulseWidth_2 = pulseWidth_2 + incrementoMotor_2;
    pulseWidth_5 = pulseWidth_5 - incrementoMotor_5;
    delay(100);
    servo_2.write(pulseWidth_2);
    servo_5.write(pulseWidth_5);
    pulseWidth_2 = pulseWidth_2 + incrementoMotor_2;
    pulseWidth_5 = pulseWidth_5 - incrementoMotor_5;
    delay(100);
    servo_2.write(pulseWidth_2);
    servo_5.write(pulseWidth_5);
    pulseWidth_2 = pulseWidth_2 + incrementoMotor_2;
    pulseWidth_5 = pulseWidth_5 - incrementoMotor_5;
    delay(100);
    servo_2.write(pulseWidth_2);
    servo_5.write(pulseWidth_5);
    pulseWidth_2 = pulseWidth_2 + incrementoMotor_2;
    pulseWidth_5 = pulseWidth_5 - incrementoMotor_5;
    delay(100);
    servo_2.write(pulseWidth_2);
    servo_5.write(pulseWidth_5);
    pulseWidth_2 = pulseWidth_2 + incrementoMotor_2;
```

```
    pulseWidth_5 = pulseWidth_5 - incrementoMotor_5;
    servo_2.write(pulseWidth_2);
    servo_5.write(pulseWidth_5);
    break;
  }
case '4' :
  {
    pulseWidth_2 = pulseWidth_2 - incrementoMotor_2;
    pulseWidth_5 = pulseWidth_5 + incrementoMotor_5;
    delay(100);
    servo_2.write(pulseWidth_2);
    servo_5.write(pulseWidth_5);
    pulseWidth_2 = pulseWidth_2 - incrementoMotor_2;
    pulseWidth_5 = pulseWidth_5 + incrementoMotor_5;
    delay(100);
    servo_2.write(pulseWidth_2);
    servo_5.write(pulseWidth_5);
    pulseWidth_2 = pulseWidth_2 - incrementoMotor_2;
    pulseWidth_5 = pulseWidth_5 + incrementoMotor_5;
    delay(100);
    servo_2.write(pulseWidth_2);
    servo_5.write(pulseWidth_5);
    pulseWidth_2 = pulseWidth_2 - incrementoMotor_2;
    pulseWidth_5 = pulseWidth_5 + incrementoMotor_5;
    delay(100);
    servo_2.write(pulseWidth_2);
    servo_5.write(pulseWidth_5);
    pulseWidth_2 = pulseWidth_2 - incrementoMotor_2;
    pulseWidth_5 = pulseWidth_5 + incrementoMotor_5;
    delay(100);
    servo_2.write(pulseWidth_2);
    servo_5.write(pulseWidth_5);
    pulseWidth_2 = pulseWidth_2 - incrementoMotor_2;
    pulseWidth_5 = pulseWidth_5 + incrementoMotor_5;
    delay(100);
    servo_2.write(pulseWidth_2);
    servo_5.write(pulseWidth_5);
    pulseWidth_2 = pulseWidth_2 - incrementoMotor_2;
    pulseWidth_5 = pulseWidth_5 + incrementoMotor_5;
    servo_2.write(pulseWidth_2);
    servo_5.write(pulseWidth_5);
    break;
  }
}
```

```
 // Verifica os limites do servo 1
 if (pulseWidth_1 > maxPulse_1) {
   pulseWidth_1 = maxPulse_1;
 }
 if (pulseWidth_1 < minPulse_1) {
   pulseWidth_1 = minPulse_1;
 }

 // Verifica os limites do servo 2
 if (pulseWidth_2 > maxPulse_2) {
   pulseWidth_2 = maxPulse_2;
 }
 if (pulseWidth_2 < minPulse_2) {
   pulseWidth_2 = minPulse_2;
 }

// Verifica os limites do servo 3
 if (pulseWidth_3 > maxPulse_3) {
   pulseWidth_3 = maxPulse_3;
 }
 if (pulseWidth_3 < minPulse_3) {
   pulseWidth_3 = minPulse_3;
 }

// Verifica os limites do servo 4
 if (pulseWidth_4 > maxPulse_4) {
   pulseWidth_4 = maxPulse_4;
 }
 if (pulseWidth_4 < minPulse_4) {
   pulseWidth_4 = minPulse_4;
 }

// Verifica os limites do servo 5
 if (pulseWidth_5 > maxPulse_5) {
   pulseWidth_5 = maxPulse_5;
 }
 if (pulseWidth_5 < minPulse_5) {
   pulseWidth_5 = minPulse_5;
 }
// Verifica os limites do servo 6
 if (pulseWidth_6 > maxPulse_6) {
   pulseWidth_6 = maxPulse_6;
 }
```

```
    if (pulseWidth_6 < minPulse_6) {
      pulseWidth_6 = minPulse_6;
    }

    //envia o "comando" para os servos
    servo_1.write(pulseWidth_1);
    servo_2.write(pulseWidth_2);
    servo_3.write(pulseWidth_3);
    servo_4.write(pulseWidth_4);
    servo_5.write(pulseWidth_5);
    servo_6.write(pulseWidth_6);

    // imprime na serial os angulos dos servos
    Serial.print(" Servo _1: ");
    Serial.print(pulseWidth_1);
    Serial.print("- Servo _2: ");
    Serial.print(pulseWidth_2);
    Serial.print("- Servo _3: ");
    Serial.print(pulseWidth_3);
    Serial.print(" Servo _4: ");
    Serial.print(pulseWidth_4);
    Serial.print("- Servo _5: ");
    Serial.print(pulseWidth_5);
    Serial.print("- Servo _6: ");
    Serial.print(pulseWidth_6);
    Serial.println();
  }
}
```