

Departamento de Sistemas e Computação – FURB
Curso de Ciência da Computação
Trabalho de Conclusão de Curso – 2018/1

Solução para locomoção da cadeira de rodas através dos movimentos da cabeça

Acadêmico: Gustavo Klabunde
guga_klabunde@hotmail.com

Orientador: Prof. Miguel Alexandre Wisintainer
maw@furb.br



Roteiro

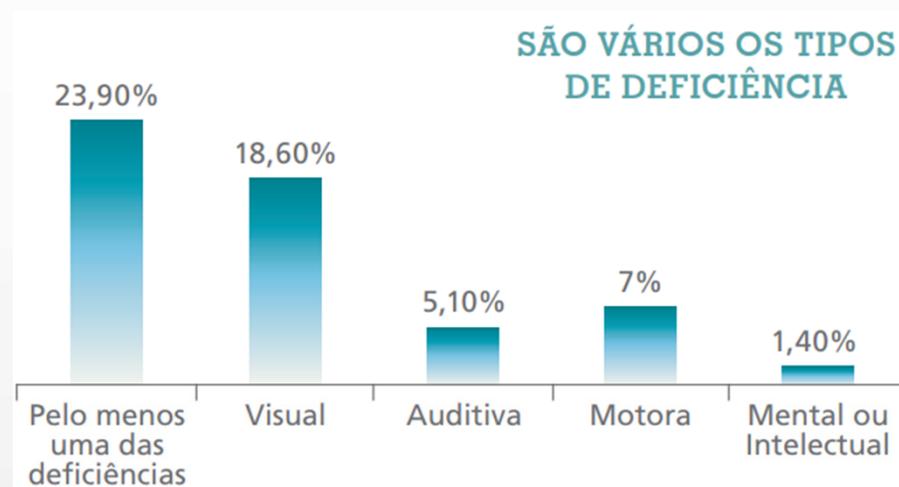
- Introdução
- Objetivos
- Fundamentação teórica
- Trabalhos correlatos
- Requisitos
- Especificação
- Implementação
- Operacionalidade da implementação
- Resultados e discussões
- Conclusão
- Sugestões

Introdução

45.606.048 de brasileiros, 23,9% da população total, têm algum tipo de deficiência – visual, auditiva, motora e mental ou intelectual.

25.800.681 (26,5%) são mulheres e 19.805.367 (21,2%) são homens.

38.473.702 pessoas vivem em áreas urbanas e 7.132.347 em áreas rurais.



Fonte: IBGE

Introdução

45.606.048
pessoas com
deficiência

1,6%
são totalmente
cegas

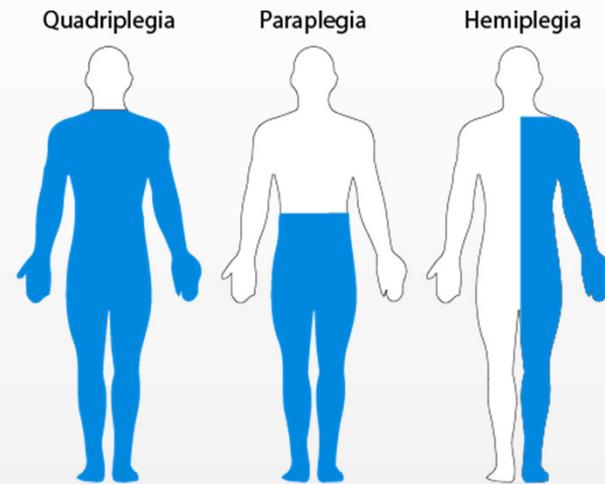
7,6%
são totalmente
surdas

1,62%
não conseguem
se locomover

Fonte: IBGE

Introdução

Tetraplegia (Quadriplegia) é a perda da função motora e ou sensitiva nos segmentos cervicais da medula espinhal



Objetivos

Objetivo geral:

Desenvolver uma solução para locomover a cadeira de rodas através dos movimentos da cabeça

Objetivos específicos:

- a) identificar os movimentos da cabeça através de um acelerômetro e um giroscópio;
- b) transmitir os movimentos identificados para a cadeira de rodas através da radiofrequência;
- c) movimentar a cadeira de rodas com base nos movimentos identificados;
- d) identificar objetos atrás da cadeira de rodas para evitar colisões.

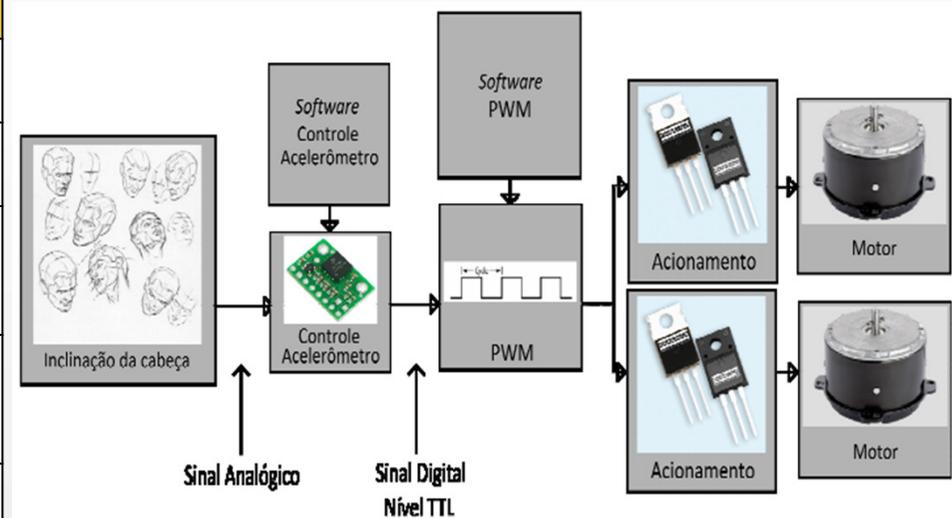
Fundamentação Teórica

- Dificuldades na locomoção
 - Cadeira de rodas
 - Muletas
 - Idosos
 - Gestantes
 - Obesos
- Módulos e sensores
 - Microcontrolador NodeMCU ESP8266
 - Módulo MPU6050
 - Sensor HC-SR04
- Filtro complementar
- Pulse Width Modulation

Trabalhos Correlatos

Título: Acionamento de uma cadeira de rodas através de um acelerômetro bi-axial inclinômetro

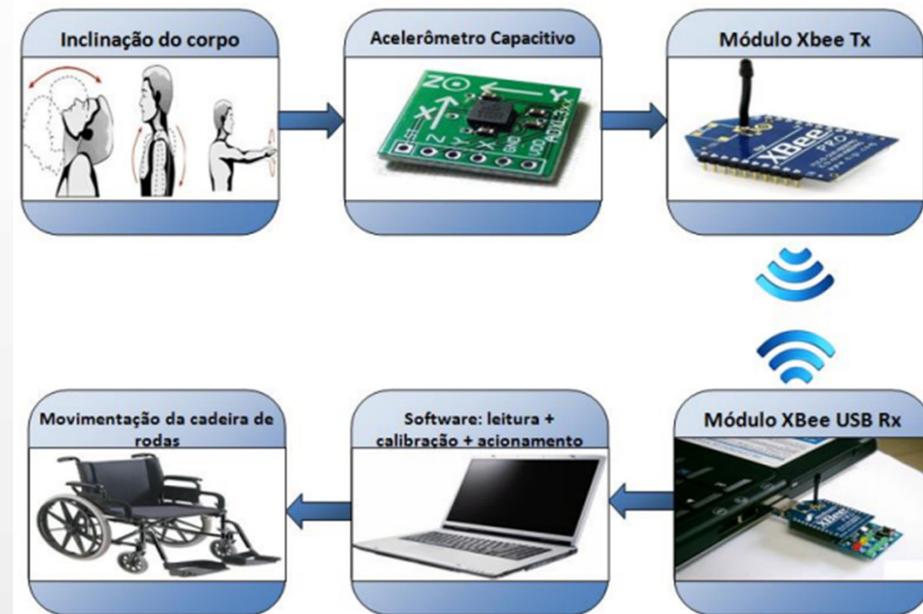
| Características | Fusco (2010) |
|---|----------------------|
| Acionamento | Movimentos da cabeça |
| Comunicação sem fio | Não |
| Adaptação mecânica da cadeira de rodas | Sim |
| Detecta objetos atrás da cadeira de rodas | Não |
| Calibração dos movimentos | Sim |



Trabalhos Correlatos

Título: Rede de acelerômetros para tecnologia assistiva

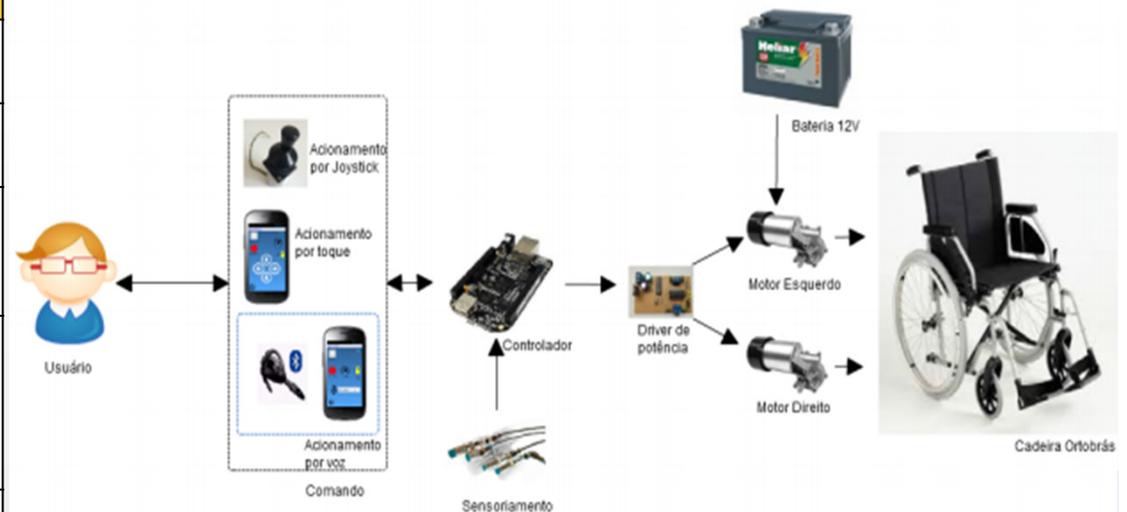
| Características | Dellagostin (2011) |
|---|---------------------|
| Acionamento | Movimentos do corpo |
| Comunicação sem fio | Sim |
| Adaptação mecânica da cadeira de rodas | Não |
| Detecta objetos atrás da cadeira de rodas | Não |
| Calibração dos movimentos | Sim |



Trabalhos Correlatos

Título: Sistema de controle de cadeira de rodas motorizada para usuários portadores de tetraplegia

| Características | Ivo (2016) |
|---|--------------------------|
| Acionamento | Voz / joystick / web-app |
| Comunicação sem fio | Parcialmente |
| Adaptação mecânica da cadeira de rodas | Sim |
| Detecta objetos atrás da cadeira de rodas | Sim |
| Calibração dos movimentos | Não |



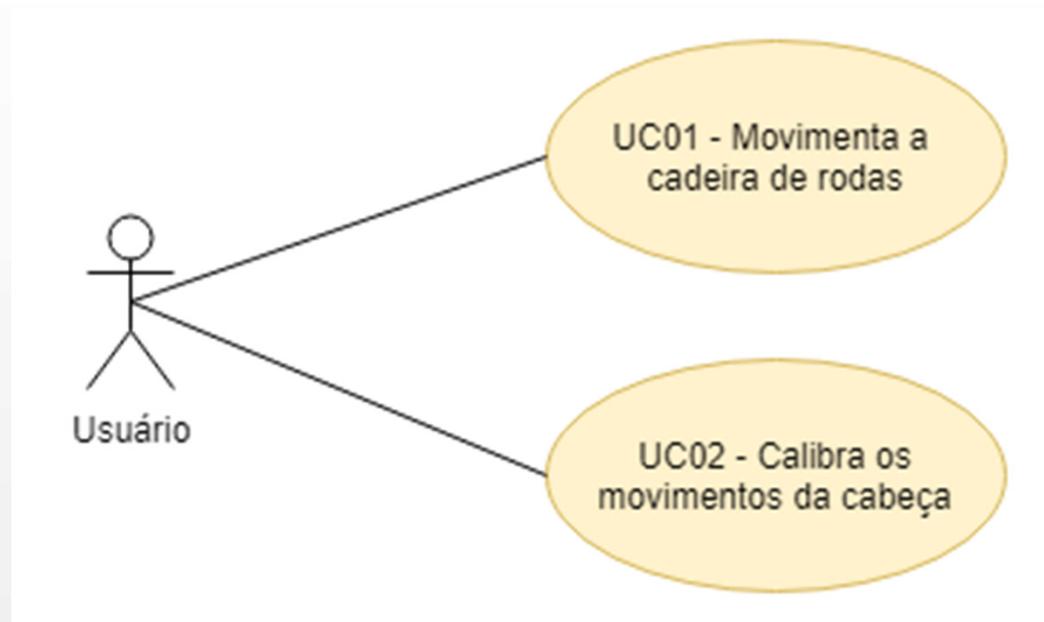
Requisitos Funcionais

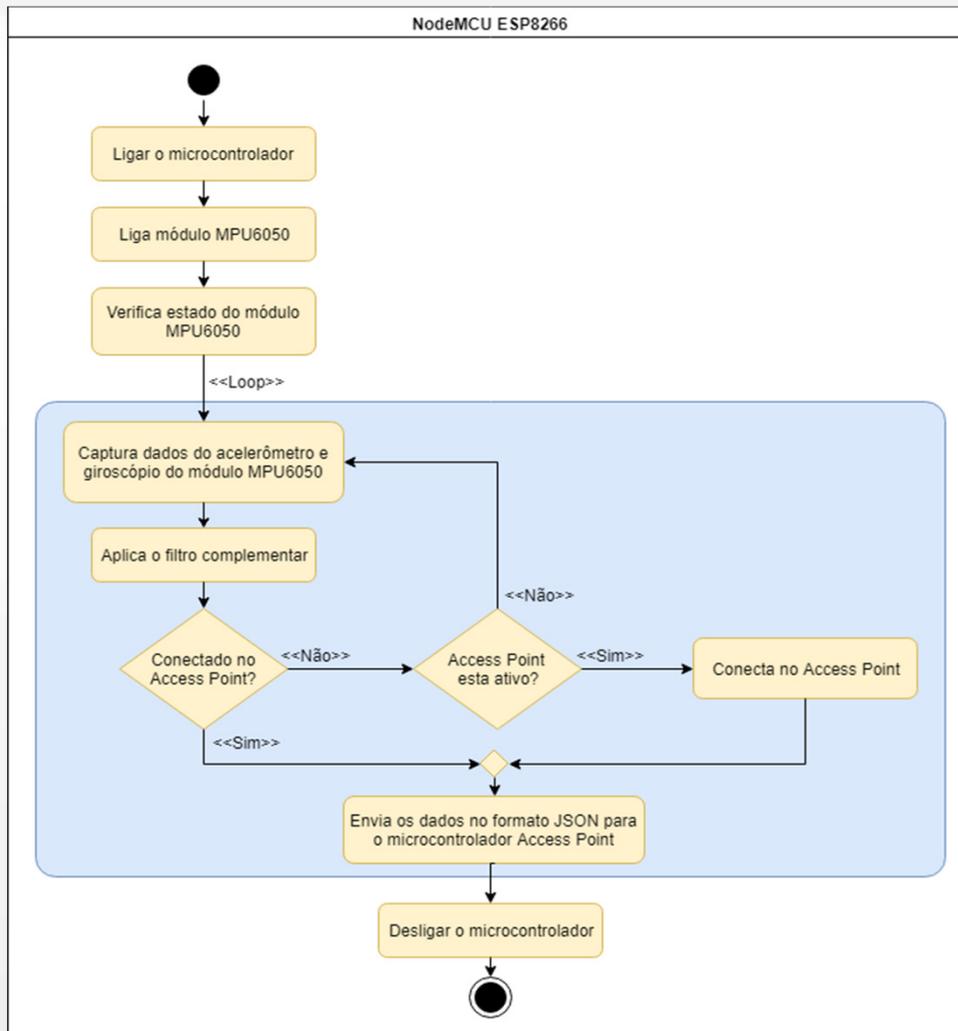
| Requisito | Descrição |
|-----------|--|
| RF01 | movimentar a cadeira de rodas através dos movimentos da cabeça |
| RF02 | identificar objetos na parte de trás da cadeira de rodas |
| RF03 | permitir a calibração dos ângulos de movimento da cabeça |
| RF04 | movimentar suavemente a cadeira de rodas |
| RF05 | identificar e eliminar impossíveis movimentos da cabeça |

Requisitos Não Funcionais

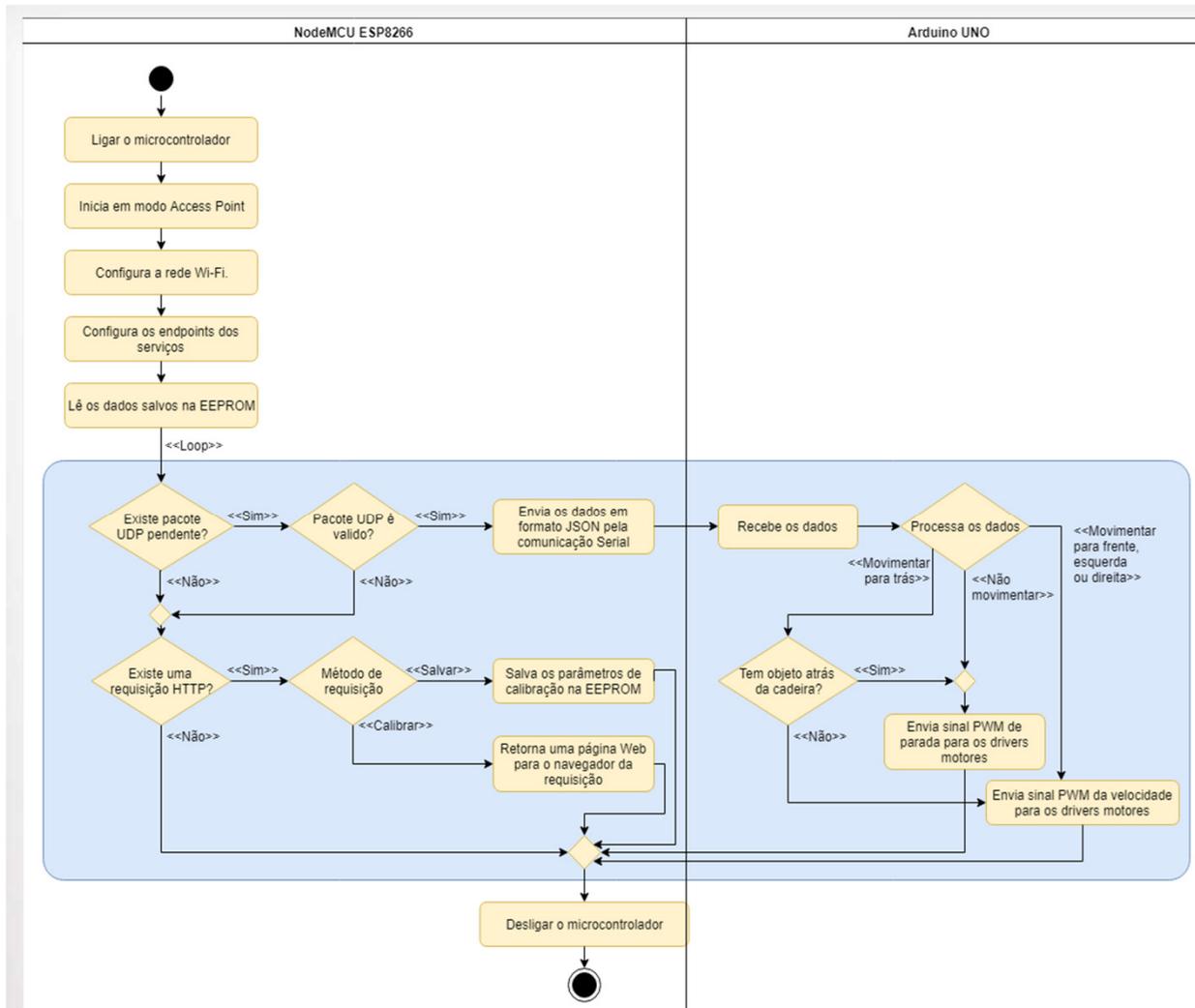
| Requisito | Descrição |
|-----------|--|
| RNF01 | utilizar o microcontrolador NodeMCU ESP8266 |
| RNF02 | realizar a comunicação sem fio entre os microcontroladores NodeMCU ESP8266 |
| RNF03 | utilizar o acelerômetro e giroscópio MPU6050 |
| RNF04 | enviar os dados do acelerômetro e giroscópio MPU6050 para o microcontrolador NodeMCU ESP8266 |
| RNF05 | utilizar um motor de limpador de para-brisas em cada roda para locomoção da cadeira de rodas |
| RNF06 | utilizar o driver motor IBT-2 para controlar os motores |
| RNF07 | ser desenvolvida na linguagem C++ através da Arduino IDE |

Especificação



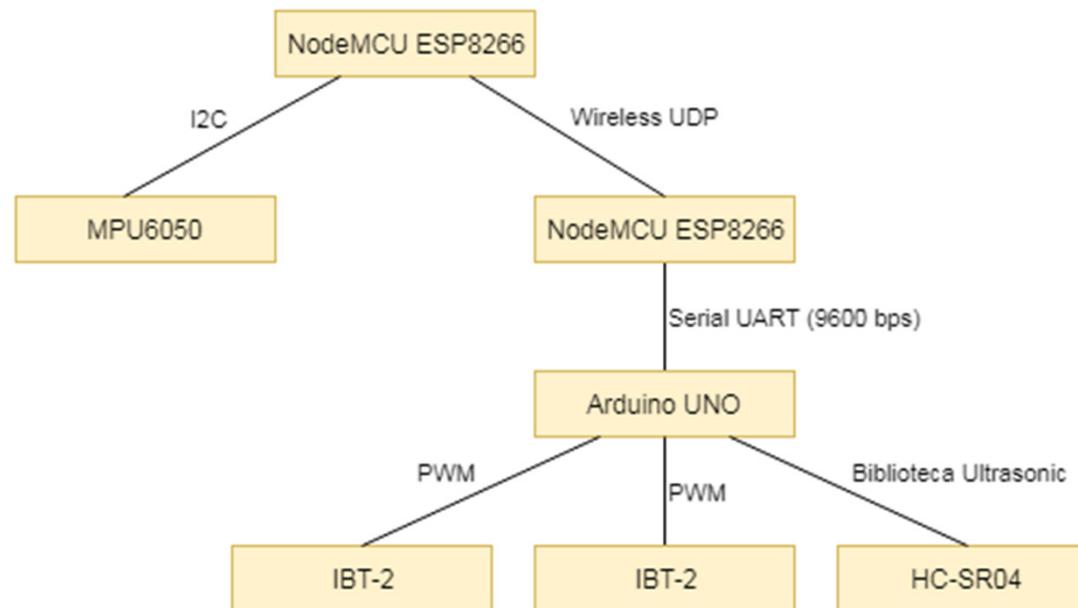


Especificação

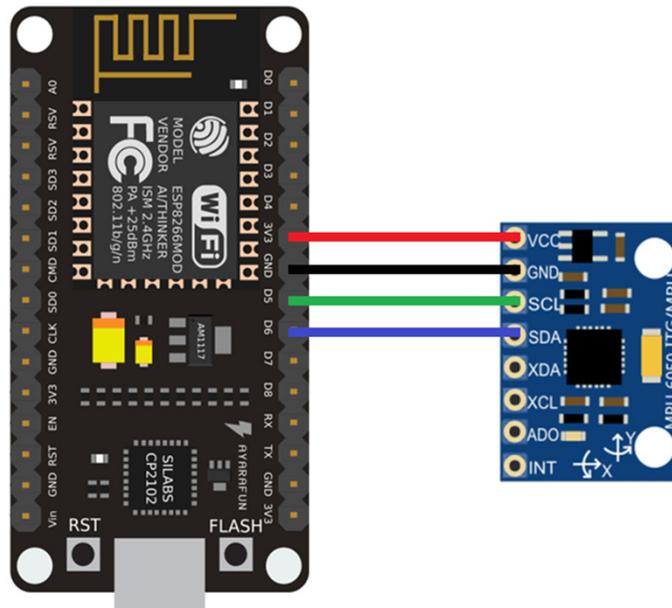


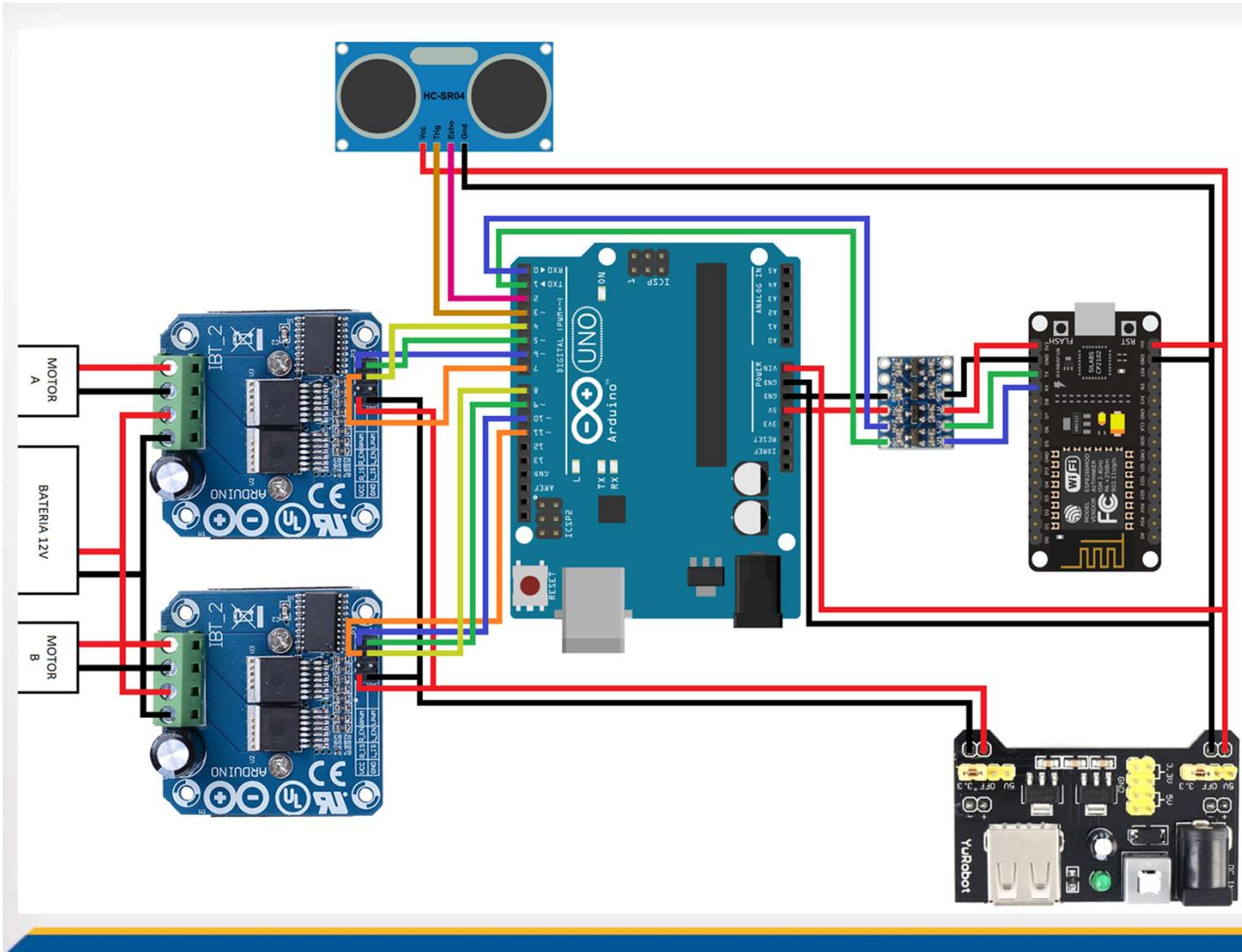
Especificação

Especificação



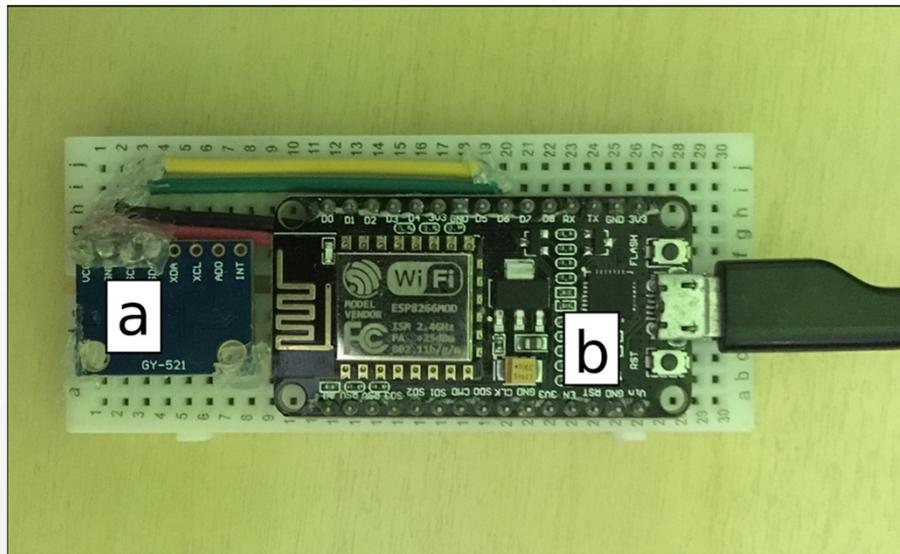
Especificação





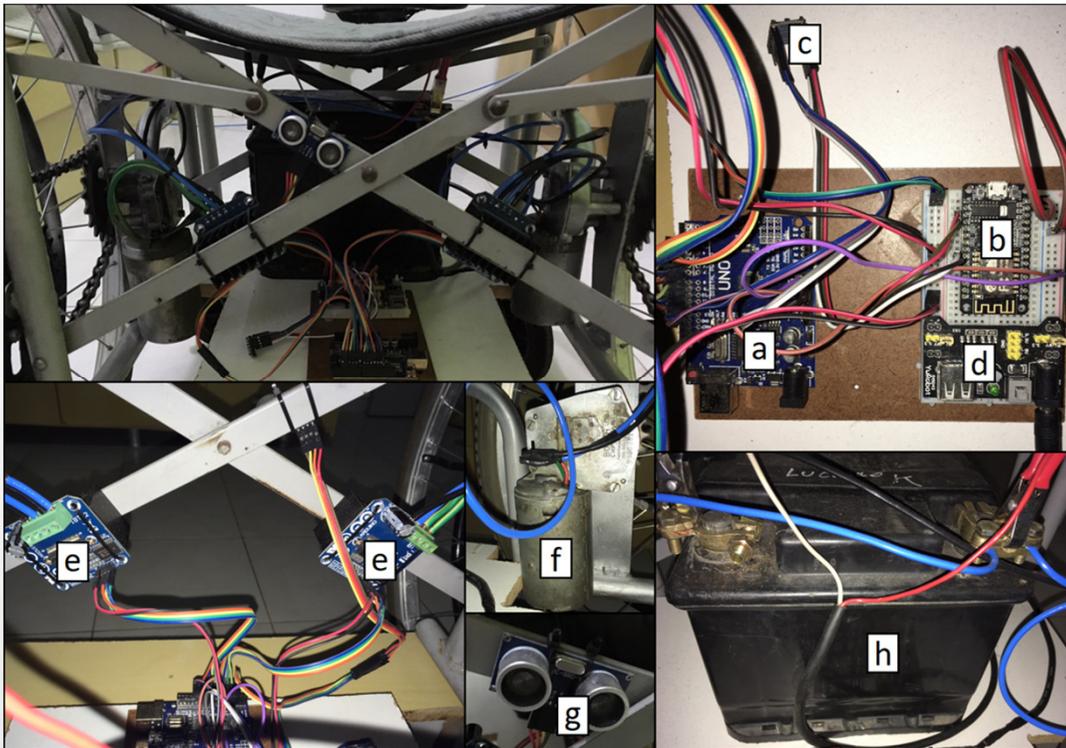
Especificação

Implementação



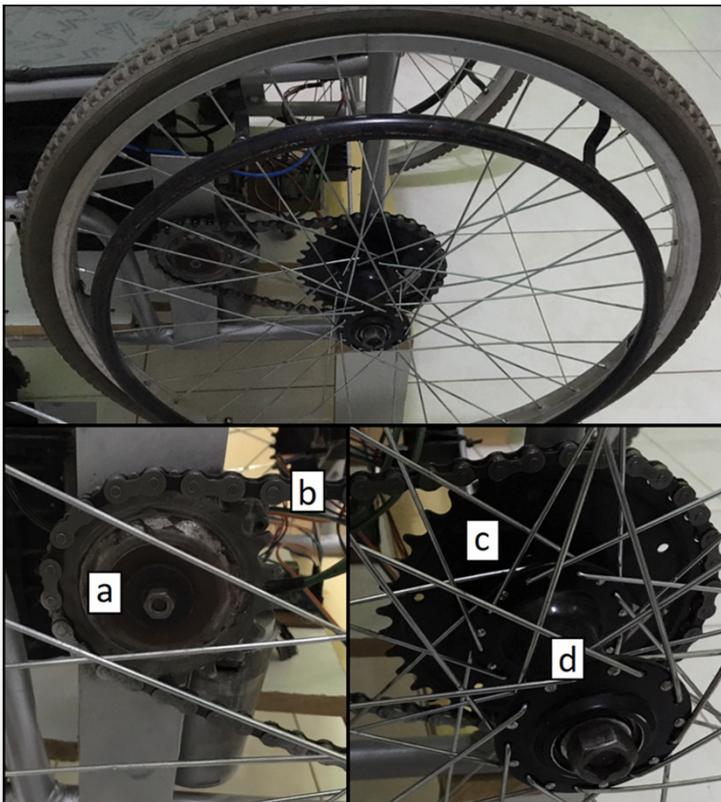
- a) módulo MPU6050: acelerômetro e giroscópio;
- b) microcontrolador NodeMCU ESP8266.

Implementação



- a) microcontrolador Arduino UNO;
- b) microcontrolador NodeMCU ESP8266;
- c) conversor de nível lógico bidirecional;
- d) fonte ajustável 3.3V/5V
- e) driver motor IBT-2;
- f) motor de limpador de para brisa;
- g) sensor HC-SR04;
- h) bateria 12V automotiva.

Implementação



- a) pinhão de bicicleta;
- b) corrente de bicicleta;
- c) coroa de bicicleta;
- d) cubo trocado.

```

1 void filter(unsigned long now) {
2 // conversão dos valores brutos do giroscópio para ângulos
3 float gyro_x = (GyX - base_x_gyro) / FS_SEL;
4 float gyro_y = (GyY - base_y_gyro) / FS_SEL;
5 float gyro_z = (GyZ - base_z_gyro) / FS_SEL;
6 // conversão os valores brutos do acelerômetro para ângulos
7 float accel_angle_x = convertAccelToAngle(AcY, AcX, AcZ);
8 float accel_angle_y = convertAccelToAngle(AcX, AcY, AcZ);
9 float accel_angle_z = convertAccelToAngle(AcZ, AcX, AcY);
10 /* calcula o ângulo do giroscópio com base no tempo delta, no ângulo
11 * do giroscópio lido e nos últimos valores filtrados */
12 float dt = (now - last_read_time) / 1000.0;
13
14 float gyro_angle_x = gyro_x * dt + last_x_angle;
15 float gyro_angle_y = gyro_y * dt + last_y_angle;
16 float gyro_angle_z = gyro_z * dt + last_z_angle;
17 /* aplica o filtro complementar, com o fator de confiança de 0.96
18 * para o giroscópio e 0.04 para o acelerômetro */
19 float alpha = 0.92;
20 float angle_x = alpha * gyro_angle_x + (1.0 - alpha) * accel_angle_x;
21 float angle_y = alpha * gyro_angle_y + (1.0 - alpha) * accel_angle_y;
22 float angle_z = alpha * accel_angle_y + (1.0 - alpha) * gyro_angle_z;
23 // atualiza os últimos valores filtrados
24 set_last_read_angle_data(now, angle_x, angle_y, angle_z);
25 }

```

```

1 // conversão radiano em graus
2 float RAD_TO_DEGREE = 180/3.14159;
3 float FS_SEL = 131; // sensibilidade do giroscópio
4
5 float convertAccelToAngle(int a, int b, int c) {
6 return RAD_TO_DEGREE * atan(a / sqrt(pow(b, 2) + pow(c, 2)));
7 }

```

Implementação

```
1 void setMotor(int motor, char side, byte pwm) {
2   if(motor == 1 && side == 'R') {
3     analogWrite(MOTOR_1_RPWM,pwm);
4   }
5   if(motor == 1 && side == 'L') {
6     analogWrite(MOTOR_1_LPWM,pwm);
7   }
8   if(motor == 2 && side == 'R') {
9     analogWrite(MOTOR_2_RPWM,pwm);
10  }
11  if(motor == 2 && side == 'L') {
12    analogWrite(MOTOR_2_LPWM,pwm);
13  }
14 }
```

Implementação

Acesso restrito para calibrar os movimentos de limite para inicio de locomoção da cadeira.

Ângulo de acionamento p/ frente:

Ângulo de acionamento p/ atrás:

Ângulo de acionamento p/ esquerda:

Ângulo de acionamento p/ direita:

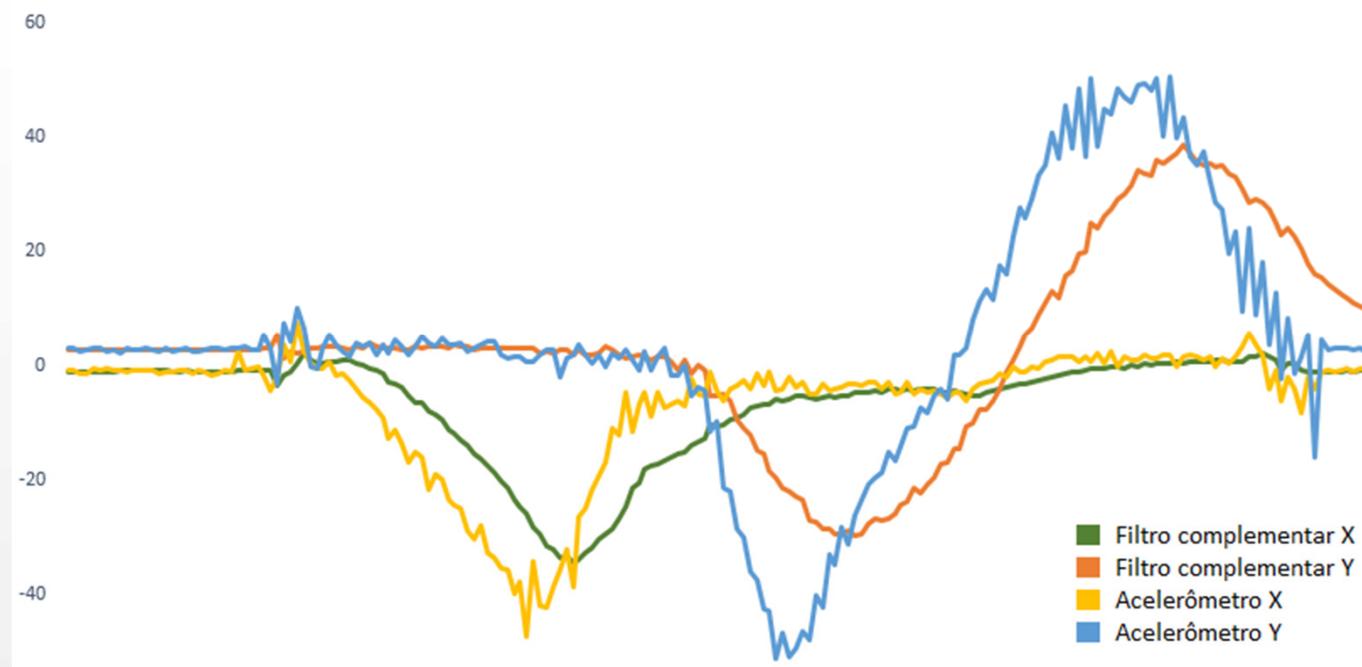
Distância máxima dos objetos:

Salvar

Operacionalidade da implementação



Resultados e discussões



Resultados e discussões

- Driver motor H-Bridge DC Mosfet IRF3205 / IBT-2
- Utilização de um Arduino UNO
- Comunicação por radiofrequência
- Acúmulo na quantidade de pacotes
- Nomes de identificação abreviados no JSON
- Velocidade na transmissão dos dados
- Força de impulsão na cadeira de rodas
- Controle da velocidade dos motores
- Objetos atrás da cadeira de rodas

Resultados e discussões

| Características | Fusco (2010) | Dellagostin (2011) | Ivo (2016) | Trabalho desenvolvido |
|---|----------------------|---------------------|--------------------------|-----------------------|
| Acionamento | Movimentos da cabeça | Movimentos do corpo | Voz / joystick / web-app | Movimentos da cabeça |
| Comunicação sem fio | Não | Sim | Parcialmente | Sim |
| Adaptação mecânica da cadeira de rodas | Sim | Não | Sim | Sim |
| Detecta objetos atrás da cadeira de rodas | Não | Não | Não | Sim |
| Calibração dos movimentos | Sim | Sim | Não | Sim |

Conclusão

- Objetivos alcançados
- Desempenho do protótipo
- Calibração dos ângulos de acionamento
- Custo do protótipo

Sugestões

- a) corrigir possíveis problemas com ondulações no piso onde está a cadeira de rodas, como rampas e descidas que poderiam aumentar a velocidade ultrapassando os limites de segurança;
- b) controlar a velocidade de movimentação da cadeira de rodas conforme o ângulo de inclinação da cabeça (quanto maior o ângulo, maior a velocidade);
- c) comprimir os dados dos movimentos, diminuindo o tamanho do pacote de envio dos dados pela radiofrequência;
- d) implementar uma camada de segurança no envio dos dados dos movimentos através da radiofrequência;
- e) possibilitar ao usuário reposicionar o módulo MPU6050, deixando o usuário escolher onde ele quer colocar o protótipo de captura dos movimentos;

Sugestões

- f) alterar o processo de calibração dos ângulos de acionamento, para que eles sejam capturados diretamente pelos movimentos e não por um formulário;
- g) colocar uma caixa de redução que possibilite maior torque ao motor, para corrigir o problema da força de movimentação da cadeira de rodas;
- h) utilizar uma bateria mais potente ou duas baterias automotivas, deixando os motores em velocidades equivalentes quando necessário, sem perdas de desempenho por carga de energia;
- i) movimentar a cadeira de rodas na diagonal através do controle da velocidade das rodas;
- j) adicionar um verificador de carga de bateria;
- k) realizar o controle de temperatura dos drivers motores, evitando problemas com aquecimento.

Demonstração do trabalho