SOLUÇÃO DE LOCOMOÇÃO PARA CADEIRANTES ATRAVÉS DE INTEGRAÇÃO ARDUINO-ANDROID

Mirella Melo*, João Marcelo Teixeira°, Marcílio Feitosa*

*Universidade de Pernambuco, °Universidade Federal de Pernambuco
Recife, Pernambuco, Brasil

Email: mspm@cin.ufpe.br, jmxnt@cin.ufpe.br, marcilio@poli.br

Abstract— This project intends to contribute to accessibility, specifically about assistive technology. The presented project consists of a wheelchair prototype guided by a mobile control application. The mobile app includes the projection of the wheelchair on an interactive 2D map. The prototype was built on Arduino, while the control application used MIT App Inventor as software development platform. The mobile app communicates to the wheelchair via Bluetooth and sends to it a 2D vector calculated from analytic geometry equations. Once the user drags the wheelchair icon to any part of the map, updating the new location in the map at the mobile device, the wheelchair goes to that corresponding real position. The motivation for this work is the possibility to make the life of those who depend on a wheelchair more comfortable.

Keywords— Accessibility, assistive technology, Arduino, Android application, wheelchair.

Resumo— O projeto busca contribuir para o tema da acessibilidade, especificamente sobre tecnologia assistiva. O trabalho apresentado consiste em um protótipo de cadeira de rodas guiado por uma aplicação móvel de controle. O aplicativo móvel inclui a projeção da cadeira de rodas em um mapa 2D interativo. O protótipo foi desenvolvido usando a plataforma Arduino, enquanto o aplicativo de controle usou o ambiente MIT App Inventor como plataforma de desenvolvimento de software. O aplicativo se comunica com o protótipo através de Bluetooth, enviando para o mesmo um vetor 2D calculado a partir de equações de geometria analítica. Quando o usuário desloca o ícone da cadeira de rodas no aplicativo para uma parte específica do mapa, a plataforma dirige-se até a posição real correspondente. É motivação para o desenvolvimento deste trabalho a possibilidade de facilitar a vida daqueles que dependem de uma cadeira de rodas para sua locomoção, tornando-a mais confortável, como por exemplo, reduzindo o estresse sofrido pelos ombros e braços para se movimentar.

Palavras-chave— Acessibilidade, tecnologia assistiva, Arduino, aplicação Android, cadeira de rodas.

1 Introdução

Tecnologia Assistiva (TA) se refere a toda ferramenta direcionada a serviços que contribuem e ampliam habilidades funcionais de pessoas com deficiência, e proporcionam uma vida de maior independência e inclusão (Bersch, 2008).

É possível perceber a TA presente em aplicativos de celulares como o *HandTalk*[1], destinado a fazer tradução simultânea de conteúdos em português para a língua brasileira de sinais (libras) e no *Weelmap*[2], que permite que o usuário encontre lugares com acessibilidade baseado na avaliação de outras pessoas com deficiência.

As cadeiras de rodas podem ser convencionais ou motorizadas. No entanto, arrastá-la manualmente exige esforço constante nos braços e, portanto, há momentos de exaustão em que é fundamental a ajuda de terceiros, trazendo o inconveniente da dependência. Já as cadeiras com rodas motorizadas, controladas por botões ou *joysticks*, não passam por essa problemática. Há também, ainda que em fase experimental, modelos controlados por sinais de ondas cerebrais (Velasco et al., 2017), ou ainda, implementações como a de controle por comando de voz (de Azevedo et al., 2016), além de estudos com soluções mais complexas capazes de atender até mesmo a demanda de pessoas paraplégicas, pois associam a direção do olhar com o movimento da cadeira (Purwanto et al., 2009).

No âmbito da TA, este artigo aborda as etapas envolvidas na construção de uma cadeira de rodas elétrica de baixo custo, controlada via celular e adequada à realidade brasileira. O objetivo é que, em um mapa 2D, que representa um ambiente real e fechado (*indoor*), disponível numa aplicação móvel, o usuário possa arrastar o ícone para uma parte específica da região do mapa, e quando houver a soltura do ícone na tela, a cadeira de rodas dirija-se até aquela posição final.

O restante deste artigo está organizado conforme segue. As seções 2, 3 e 4, respectivamente, descrevem o desenvolvimento do software de controle, os cálculos necessários ao deslocamento da cadeira e o processo de construção do hardware. A Seção 5 trata da programação do microcontrolador enquanto a Seção 6 analisa os erros verificados nos testes. Por fim, a Seção 7 extrapola o protótipo implementado considerando uma cadeira de rodas real e a Seção 8 realiza o fechamento do trabalho e propõe intenções futuras.

2 Desenvolvimento do Aplicativo

2.1 Lógica de Programação do Aplicativo

A programação, feita em blocos, é composta por: inicialização, *touchdown* (situação de pressionar a

[1] www.handtalk.me
[2] www.wheelmap.org

cadeira de rodas), *touch up* (situação de soltar a cadeira de rodas) e comunicação Bluetooth.

A Figura 1 ilustra um deslocamento onde as variáveis destacadas estão descritas na Tabela 1 e a Figura 2 traz o fluxograma referente à lógica que se aplicou no desenvolvimento do aplicativo.

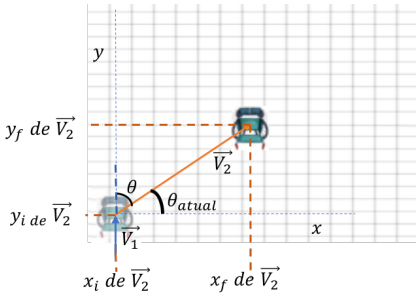

Figura 1. Exemplo de deslocamento.

Tabela 1. Variáveis e descrições

Variável	Descrição
$\lvert\overrightarrow{V_1}\rvert$	Módulo do vetor $\overrightarrow{V_1}$
$\lvert\overrightarrow{V_2}\rvert$	Módulo do vetor $\overrightarrow{V_2}$
x_i de $\overrightarrow{V_2}$	Coordenada x inicial de $\overrightarrow{V_2}$
y_i de $\overrightarrow{V_2}$	Coordenada y inicial de $\overrightarrow{V_2}$
x_f de $\overrightarrow{V_2}$	Coordenada x final de $\overrightarrow{V_2}$
y_f de $\overrightarrow{V_2}$	Coordenada y final de $\overrightarrow{V_2}$
$(x_f - x_i)$ de $\overrightarrow{V_1}$	Diferença x de $\overrightarrow{V_1}$
$(y_f - y_i)$ de $\overrightarrow{V_1}$	Diferença y de $\overrightarrow{V_1}$
$(x_f - x_i)$ de $\overrightarrow{V_2}$	Diferença x de $\overrightarrow{V_2}$
$(y_f - y_i)$ de $\overrightarrow{V_2}$	Diferença y de $\overrightarrow{V_2}$
θ	Ângulo de rotação
θ_{atual}	Ângulo entre $\overrightarrow{V_2}$ e o eixo x
z	Sentido de rotação

No subconjunto de blocos da inicialização encontra-se a criação de uma lista com 13 itens, que são variáveis essenciais para os cálculos responsáveis por gerar a trajetória e que estão descritas na Tabela 1. Na inicialização do aplicativo, a plataforma está posicionada como na Figura 1, com direção e sentido de 90°, então $\overrightarrow{V_1}$ é considerado como um vetor unitário descrito em coordenada polar por (1, 90°) e por meio dessa atribuição, os cálculos referentes ao primeiro deslocamento se tornam possíveis.

O *touchdown* é o momento em que o usuário pressiona o ícone da cadeira de rodas e deve posteriormente arrastá-la até a posição final desejada. Ou seja, o *touchdown* significa que haverá um futuro deslocamento, e a partir desse momento, os dados da lista referentes ao deslocamento atual ($\overrightarrow{V_2}$) serão realocados para os dados da lista referentes ao deslocamento passado ($\overrightarrow{V_1}$), conforme ilustrado no fluxograma da Figura 2.

Após arrastar o ícone da cadeira de rodas no mapa, o usuário solta o ícone na posição desejada.

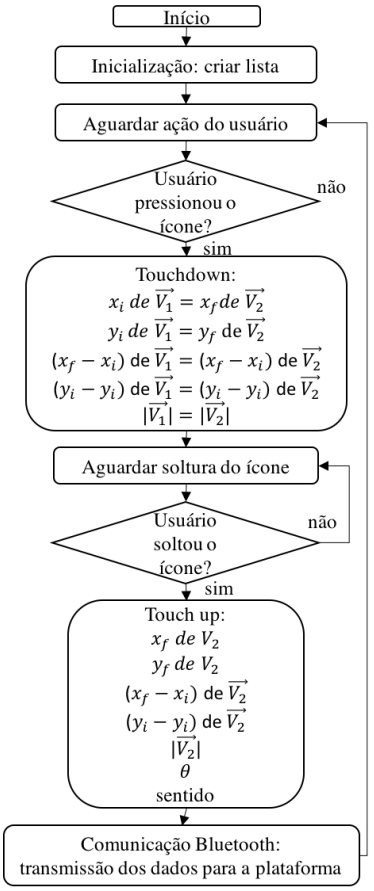

Figura 2. Fluxograma da lógica do aplicativo.

Nesse momento, denominado *touch up*, as coordenadas x e y da posição final da cadeira são salvas e são feitos todos os cálculos e condicionais apresentados na Seção 3. Além dos cálculos, é também no *touch up* realizado o envio de 3 dados da lista: o módulo do vetor $\overrightarrow{V_2}$, ângulo de rotação (θ) e o sentido de rotação. Vale salientar que, quando o sentido de rotação se dá para a direita, o valor enviado é 1, e quando se dá para a esquerda, o valor enviado é 2.

O módulo de comunicação escolhido foi o Bluetooth. O subconjunto de blocos relacionados ao Bluetooth é padrão: listagem de dispositivos disponíveis para emparelhamento, associado a uma possível notificação caso o aparelho celular esteja com o Bluetooth desligado.

2.2 Layout

O aplicativo foi desenvolvido através do software App Inventor[3], uma aplicação de código aberto mantida pelo *Massachusetts Institute Of Technology* (MIT). O software permite que pessoas criem aplicativos para o sistema operacional Android de maneira simples e sem aprofundamentos em linguagens de programação por linhas de código, uma vez que a programação é feita em blocos.

[3]www.appinventor.mit.edu

A partir da aba "*Drawing and Animation*" do software foi possível trazer duas ferramentas fundamentais: o mapa 2D, e o ícone da cadeira de rodas a ser inserido dentro do mapa. As caixas de textos que imprimem determinados valores podem ser encontradas ainda na aba *Designer*, através de "*User Interface*" e posteriormente "*Label*".

Como forma de depuração, os valores mostrados na tela são: a diferença x e y de $\overrightarrow{V_1}$, a diferença x e y de $\overrightarrow{V_2}$, ângulo atual (θ_{atual}), ângulo de rotação (θ), sentido de rotação (horário ou anti-horário), o módulo de $|\overrightarrow{V_1}|$ e o módulo de $|\overrightarrow{V_2}|$.

O aplicativo conta com dois botões: um para realizar a conexão Bluetooth ("Conecte-se"), e outro responsável por deixar o aplicativo nas mesmas condições de inicialização ("Reiniciar").

O resultado final quanto à disposição dos itens está ilustrado na Figura 3.

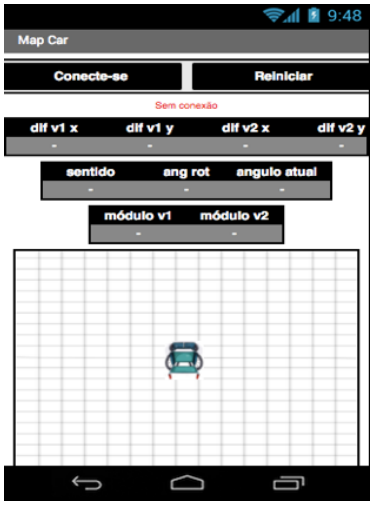

Figura 3. Layout do aplicativo Android.

3 Cálculos da Trajetória

O mapa 2D funciona como um plano cartesiano, em que os dados são as coordenadas do eixo horizontal e do eixo vertical, ou seja, x e y, respectivamente. Através desses valores é possível calcular os dados de deslocamento.

Optou-se por fazer uso de coordenadas polares (raio e ângulo) ao invés de cartesianas por apresentar uma solução mais otimizada no sentido de percurso, tempo, e consequentemente consumo de bateria, além de se assemelhar com a forma com que caminhamos.

O raio, ou módulo do deslocamento, é facilmente calculado pelo teorema de Pitágoras (1), onde os valores de x e y representam os *pixels* da tela do celular nos limites do mapa 2D.

$$\sqrt{(x_f - x_i)^2 + (y_f - y_i)^2} \tag{1}$$

Sabe-se que existem dois momentos cruciais: o deslocamento passado $\overrightarrow{V_1}$ e o deslocamento atual $\overrightarrow{V_2}$. A partir dos dados salvos na lista de cada situação, fica viável o cálculo do ângulo entre esses dois vetores por (2). Dessa forma, se obtém a angulação de rotação (θ), que faz com que a plataforma gire da direção de $\overrightarrow{V_1}$ até a direção de $\overrightarrow{V_2}$.

$$\theta = \cos^{-1}\left(\frac{\overrightarrow{V_1}.\overrightarrow{V_2}}{|\overrightarrow{V_1}|.|\overrightarrow{V_2}|}\right) \tag{2}$$

Após o cálculo do ângulo de rotação, é necessário saber o sentido de rotação que a cadeira de rodas deve tomar, isto é, sentido horário ou anti-horário. Na Figura 1, por exemplo, fica evidente que a rotação deve se dar no sentido horário. Porém, é preciso representar essa lógica em termos matemáticos.

A variável θ_{atual}, ilustrada na Figura 1, representa o ângulo que o deslocamento atual $\overrightarrow{V_2}$ faz com o eixo horizontal. Esse ângulo é o que determina o sentido de rotação através da condição mostrada em 3 .

$$\theta_{atual} < 180^\circ \begin{cases} \text{Se } x_f > x_i, \text{ sent. horário} \\ \text{Se } x_f < x_i, \text{ sent. anti-horário} \end{cases} \tag{3}$$

O valor de θ_{atual} deve ser atualizado a cada deslocamento no momento de *touch up* de acordo com 4.

$$\theta_{atual} > 180^\circ \begin{cases} \text{Se } x_f > x_i, \text{ sent. anti-horário} \\ \text{Se } x_f < x_i, \text{ sent. horário} \end{cases} \tag{4}$$

É importante observar que o θ_{atual} pode vir a assumir valor maior do que 360^o ou menor do que 0^o. A partir desse fato, antes de salvar uma atualização, é sempre feita uma verificação quanto ao θ_{atual} de forma a garantir que ele seja um número positivo e menor do que 360^o. Caso isso não seja verdade, se torna necessário trazer o ângulo para a primeira determinação positiva de um arco, representada em (5).

$$\begin{gathered} \text{Se rotação no sentido horário:} \\ \theta_{atual} = \theta_{atual} - \theta \\ \text{Se rotação no sentido anti-horário:} \\ \theta_{atual} = \theta_{atual} + \theta \end{gathered} \tag{5}$$

Finalmente, é possível chegar em três dados fundamentais: o módulo, calculado pela Equação (1), o ângulo de rotação, definido pela Equação (2), e o sentido de rotação, determinado pelas Equações (3), (4) e (5).

4 Desenvolvimento do Hardware

4.1 Materiais

Frente ao alto custo para realizar a simulação numa cadeira de rodas, foi escolhido um objeto para simular o acionamento e o controle. O objeto é uma plataforma, com motores DC de torque

de 800gf.cm, duas rodas de borracha de 7cm de diâmetro e uma roda traseira giratória, sem tração, para apoio, ilustrada na Figura 4.

Todo o processamento é realizado pelo Arduino UNO, uma plataforma de prototipagem eletrônica de hardware livre, projetada com um microcontrolador Atmel ATmega328, com suporte de entrada/saída embutido (ATmel Corporation, 2016). Possui uma linguagem de programação padrão, que é essencialmente C/C++.

O dispositivo móvel utilizado já vem munido de rádio Bluetooth, enquanto o Arduino não. É preciso associar a ele um módulo Bluetooth (nesse projeto foi adotado o HC-06) para que haja troca de dados entre plataforma e *smartphone*. Esse tipo de comunicação sem fio é caracterizado por ser de baixo consumo de energia (3.3 ou 5 Volts), baixo custo, e alcance de até 10 metros, que supre a necessidade do projeto.

A alimentação desse sistema foi realizada por um carregador portátil de celular com duas saídas de 5 Volts cada. Ao contrário de uma outra opção como a bateria lítio, o carregador portátil não tem sua tensão reduzida à medida que o dispositivo está descarregando (EEMB Company, 2007), pois ele consegue manter a tensão de saída num valor constante, o que garante maior confiabilidade no controle e dispensa o uso de um regulador de tensão externo, por exemplo. Este fato ocorre porque o *power bank* faz uso de bateria lítio-íon de tensão nominal de 3.7 Volts associada a um circuito amplificador de tensão de 3 Volts para 5 Volts e, dessa forma, mantém a tensão de saída constante até o momento em que a bateria chega na sua tensão de *cutoff*, caracterizando o descarregamento total do carregador portátil.

O acionamento e controle do motor é intermediado pela ponte H, um dispositivo eletrônico capaz de alterar a polaridade no motor, e consequentemente o sentido de rotação da roda (horário ou anti-horário). Através da ponte H, também é possível controlar a velocidade de rotação do motor usando um sinal PWM (*Pulse Width Modulation*) como entrada de tensão (Texas Instruments, 2016).

Dentre as diversas classificações de encoder, para esse trabalho, é utilizado o encoder óptico, um transdutor formado por um disco perfurado e um emissor e receptor de infravermelho. Nos furos não haverá reflexo do infravermelho, logo enviará sinal de nível lógico baixo (0 Volts), enquanto na região não furada o infravermelho será refletido e o sinal enviado ao Arduino será de nível lógico alto (5 Volts). A partir disso, fica perceptível que a troca do nível de tensão indica o eixo em movimento, e pela contagem desses pulsos num intervalo de tempo é possível calcular a velocidade ou distância percorrida do objeto (Albuquerque, 2011). A contagem é percebida pelo Arduino através do conceito de interrupção, definido na Seção 5.

Na Figura 4, é possível ver os materiais utilizados no projeto físico da plataforma:

A. Arduino;

B. Carregador portátil;

C. Módulo Bluetooth;

D. *Encoder*;

E. CI da ponte H;

F. Parte estrutural (chassi).

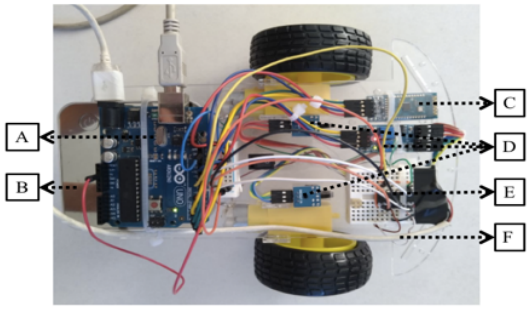

Figura 4. Montagem física da representação figurativa da cadeira.

Após a montagem do chassi, parafusagem dos motores, encaixe das rodas, posicionamento dos móulos eletrônicos, e conexão dos componentes (discutidos na seção 4.2), o peso total do sistema ficou em 560 gramas.

4.2 Esquema Elétrico

O esquema da ligação elétrica que foi implementado é ilustrado na Figura 5.

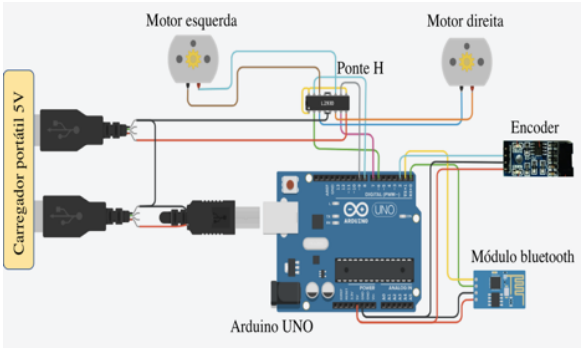

Figura 5. Esquema elétrico do projeto.

Com relação à alimentação externa, realizada pelo carregador portátil de celular, uma das duas saídas ficou responsável por alimentar o Arduino Uno e a outra saída funcionou como alimentação externa para o motor. Além disso, é importante a conexão dos fios terra (*ground*) para que haja a mesma referência de tensão em todo o sistema.

Com relação à ponte H utilizada (CI L293D), os pinos de controle de velocidade foram curto-circuitados com a alimentação externa, e isso resultou no fato de que a alimentação dos motores permaneceu sempre na tensão máxima fornecida pelo *power bank* (5 Volts), menor tensão necessária capaz de fazer a plataforma se deslocar. Consequentemente, com os recursos físicos em questão, se torna inviável o controle de velocidade. No entanto, entende-se como fundamental a implementação de acionamentos e paradas suaves caso fosse possível o desenvolvimento real de uma cadeira de rodas e, portanto, o presente trabalho aborda essa realização mesmo que de forma teórica.

Ainda sobre a ponte H, tem-se que os pinos do Arduino responsáveis pela troca de polaridade (e sentido de rotação) do motor do lado direito da plataforma são os pinos 6 e 7, enquanto no motor do lado esquerdo são os pinos 8 e 9. Já se tratando do CI, os pinos 3 e 6 são responsáveis pela alimentação do motor direito, enquanto os pinos 11 e 14 alimentam o motor esquerdo. O CI vem internamente com os pinos terra conectados.

É possível ver que o módulo Bluetooth foi conectado pelos pinos digitais 0 e 1 da placa Arduino, em que o pino 0 é o receptor de dados (RX) e o 1 é o transmissor de dados (TX), configurando, dessa forma, uma comunicação serial, com velocidade definida em 9600 baud.

O encoder possui os pinos de alimentação e dois pinos de saída, uma analógica e outra digital. Fez-se uso apenas da digital, conectada no pino 2 do Arduino, próprio para interpretação de uma interrupção.

Caso houvesse disponível para o projeto um *power bank* com saída de 9 Volts, por exemplo, o controle de velocidade se tornaria viável. Para realizar esse controle, os pinos PWM da placa Arduino seriam utilizados. O esquema elétrico mudaria somente quanto a pinos específicos, no caso, os pinos *Enable* 1 e *Enable* 9 do CI iriam passar a se conectar aos pinos 10 e 11 do Arduino, respectivamente, que são PWM. As mudanças necessárias na programação para realizar o controle de velocidade são abordadas na Seção 5

5 Programação do Arduino

Como o encoder é o principal responsável por regular a movimentação, é importante, antes de tudo, abordar o conceito de interrupção no microcontrolador. A interrupção utilizada nada mais é do que um indicativo externo de que o programa deve parar o processamento atual e executar determinada função imediatamente. O indicativo de interrupção para esse projeto é a mudança de tensão de 5 Volts para 0 Volts, isto é, de *HIGH* para *LOW* (*FALLING*) e a função chamada neste momento é o incremento de uma variável.

A declaração da interrupção na IDE do Arduino, assim como a função chamada no momento em que ocorre uma interrupção é mostrada no Algoritmo 1.

```
//declarando interrup.
attachInterrupt(0, counter, FALLING);

void counter(){
        pulses++;
}
```

Algoritmo 1. Trecho referente à interrupção.

Após a definição da interrupção, o fluxograma da lógica da programação no Arduino, ilustrado na Figura 6, pode ser visto com mais clareza.

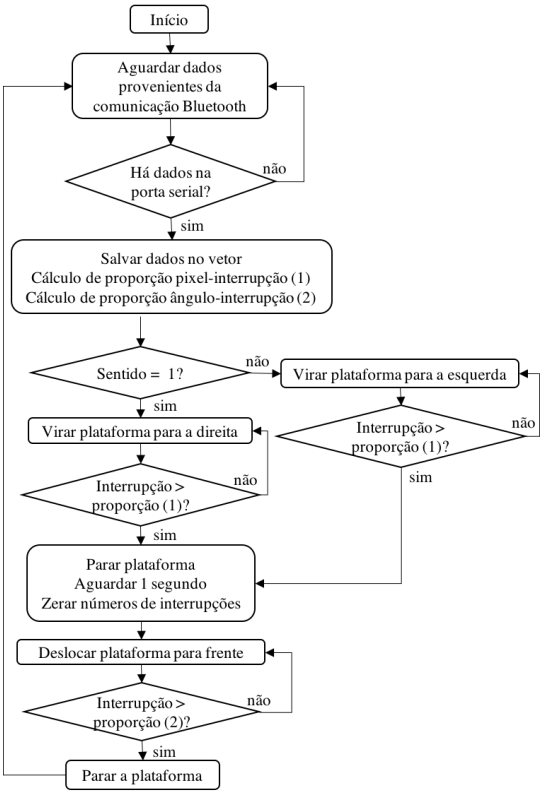

Figura 6. Fluxograma referente à lógica de programação no Arduino.

Os dados são calculados no aplicativo e posteriormente transmitidos ao Arduino via Bluetooth por comunicação serial. Três dados chegam na plataforma quando o usuário deseja se deslocar: módulo do deslocamento (em pixel), ângulo de rotação (em graus) e sentido de rotação, e a partir disso a plataforma é movimentada.

As 3 informações calculadas (discutidas na Seção 3) chegam em formato de *bytes*, 1 para cada valor. Quando chegam 3 *bytes* na porta serial RX o Arduino entende que está recebendo um comando de deslocamento, logo aceita os valores e os salva num vetor de 3 elementos. Após isso, é feito um cálculo referente à proporção de pixel-interrupção e ângulo-interrupção.

É possível perceber pela Figura 7 que 90°é equivalente a 5 interrupções, portanto, com uma

regra de proporção simples, chegou-se à Equação 6 em relação ao ângulo.

$$interrupção = rotação/18 \quad (6)$$

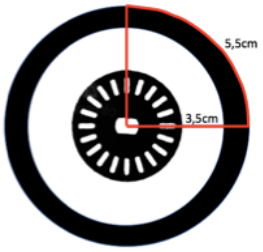

Figura 7. Relação do disco do encoder com a roda.

Após a rotação, a plataforma deve se deslocar para frente. Na Figura 7, também é visível que a cada 5 interrupções, a plataforma se desloca 5,5cm. Entende-se que a proporção pixel-interrupção deve ser ajustada de acordo com o espaço de deslocamento disponível ao cadeirante.

O trecho do código relacionado ao recebimento e alocação dos dados no vetor, bem como os cálculos de proporção, são mostrados no Algoritmo 2.

```
if (Serial.available()==3){
    for (int i=0; i <=2 ; i++){ //salvando
    array[i] = Serial.read();
    module=array[0]/11;//pixel-interrup.
    rotation=array[1]/18;} //ang-interrup.
}
```

Algoritmo 2. Dados da comunicação Bluetooth.

A partir desse momento, existem dois blocos: um quando o sentido de rotação é horária, e o outro quando é anti-horária. No Algoritmo 3, é mostrado apenas um dos casos, uma vez que o outro é similar.

```
if(array[2]==1){ //Sentido horario
    while(pulses<rotacao) Motor.direita();
    Motor.parado();
    delay(1000);
    pulses=0;
    while(pulses<modulo) Motor.frente();
    Motor.parado();
}
```

Algoritmo 3. Trecho referente ao deslocamento.

Têm-se no Algoritmo 3 que, ao entrar no bloco do tipo *if*, a plataforma permanece virando para a direita até o momento em que o número de interrupções chegue num valor maior do que o valor da proporção ângulo-interrupção definida anteriormente. Logo em seguida, o motor dá uma pausa de 1 segundo, zera a variável que conta o número de interrupções, e é inserido em um novo laço do tipo *while* que movimenta a plataforma para frente até que o número de interrupções chegue num valor maior do que o da proporção módulo-interrupção definida.

Os comandos de movimentos utilizados, como Motor.frente(), não fazem parte do acervo de palavras chaves do Arduino, portanto, houve a criação de uma biblioteca, em linguagem C++, chamada Mortordc.h, responsável por guardar os comandos de movimento da plataforma, isto é, para frente, para trás, sentido horário, anti-horário e parado.

Como discutido na seção 4, através da ponte H, o sentido de rotação da roda é alterado pela inversão de polaridade. As variáveis IN1 e IN2 são referentes ao motor da direita, e IN3 e IN4 são referentes ao motor da esquerda. Com a polaridade configurada da forma mostrada no Algoritmo 4, o movimento é para frente, no caso de ir para trás, por exemplo, basta inverter cada declaração *HIGH* por *LOW* e vice-versa.

```
void Motor::frente(){
  digitalWrite(_IN1, HIGH);//IN1 ligado
  digitalWrite(_IN2, LOW);//IN2 deslig.
  digitalWrite(_IN3, HIGH);//IN3 ligado
  digitalWrite(_IN4, LOW);//IN4 deslig.
}
```

Algoritmo 4. Deslocamento para frente.

Como dito na seção 4.2, caso houvesse disponível para o projeto um power bank com saída de 9V, por exemplo, o controle de velocidade se tornaria materialmente viável. A alteração quanto a ligação elétrica foi discutida na seção 4.2. Aqui é abordado as mudanças quanto a programação.

Quando a plataforma começar o movimento, a velocidade deve aumentar gradativamente até chegar na velocidade máxima e depois, quando estiver alcançando o ponto desejado, diminuir a velocidade gradativamente até parar. Essas situações poderiam ser percebidas através da contagem de interrupções.

A alteração na velocidade se daria através dos pinos 10 e 11 do Arduino, que são portas PWM, e um artifício a ser utilizado na IDE é o *for*, usado de maneira a aumentar ou diminuir determinado número gradativamente. O trecho do código referente ao início do movimento com controle de velocidade é mostrado no Algoritmo 5, onde a variável *fade*, que também determina a velocidade representada em valores de 0 (0 Volts) a 255 (5 Volts), é incrementada. No caso de término do movimento, a alteração se daria apenas quanto a essa variável, ou seja, ao invés de incrementar, iria decrementar. O teste foi realizado em um LED.

```
for(int fade=0; fade<=255; fade++){
    analogWrite(10, fade);
    analogWrite(11, fade);
    delay(2);
}
```

Algoritmo 5. Controle de velocidade.

6 Análise de Erro da Plataforma

A plataforma foi submetida a testes a fim de levantar uma taxa de erro do sistema. Os testes foram realizados confrontando a angulação e distância esperadas com as efetivamente realizadas pela plataforma.

O aplicativo Android enviou à plataforma, paulatinamente, 7 valores aleatórios de ângulos de rotação que foram reproduzidos pelo protótipo com um erro médio de 9,1%.

Também foi enviado à plataforma, 7 valores aleatórios do deslocamento linear que foram realizados pelo protótipo com um erro médio de 8,5%.

Os erros são causados principalmente pela má interpretação na leitura do encoder devido à sua sensibilidade e pela defasagem entre os motores.

7 Dimensionamento

Para esse projeto, toda a ideia foi desenvolvida em uma plataforma figurativa, pois haveria um alto custo financeiro ao utilizar uma cadeira de rodas real. No entanto, é importante realizar o dimensionamento.

Existem no mercado kits de motorização como o da *AAT Max-E* que inclui: uma bateria, dois motores, um *joystick* e duas rodas com engrenagens na parte interna para contato com o eixo do motor (ilustrado na Figura 8), que é o mesmo modelo de transmissão adotado para os dimensionamentos desse trabalho.

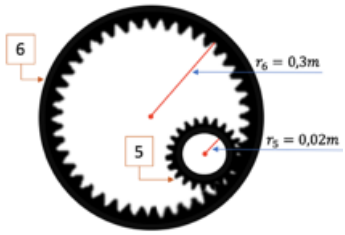

Figura 8. Acoplamento de engrenagens.

O desenvolvimento físico envolve: a estrutura da cadeira de rodas, os suportes mecânicos para receber os motores do tipo DC, os motores, o sistema de transmissão e o sistema de controle.

Adotou-se para os cálculos uma aceleração (a) de $0.4m/s^2$, aceleração da gravidade (g) de $9,81m/s^2$, massa do sistema (m) de $100kg$, velocidade máxima alcançada (v_6) de $1,6m/s$ (velocidade do caminhar humano), raio da engrenagem na roda (r_6) de $0.3m$ e raio da engrenagem 5 (r_5) de $0,02m$. Tomou-se uma média do pior caso (cimento) como coeficiente de atrito (μ) de 0.7 (El-Sherbiny et al., 2012).

A partir das atribuições, é possível calcular, pela Segunda Lei de Newton (7), a força (F) necessária para deslocar o sistema, onde, F_{at} é a força de atrito. Logo, tem-se que $F = 726,7N$.

$$\begin{aligned} F &= m.a + F_{at} \\ &= m.a + \mu.m.g \end{aligned} \tag{7}$$

O torque (T) necessário na engrenagem 5 é calculado por (8), e se conclui que $T = 14,5N.m$.

$$T = F.r_5 \tag{8}$$

No entanto, para o mercado, este valor de torque em motor DC não é comum, pois é muito alto. A fim de solucionar este dilema, se adotaria um sistema de transmissão conhecido por trem de engrenagens composto (Norton, 2013), representado na Figura 9. Nesse sistema, a engrenagem 5 estaria em contato com a engrenagem 6 da roda da cadeira, ilustrada na Figura 8.

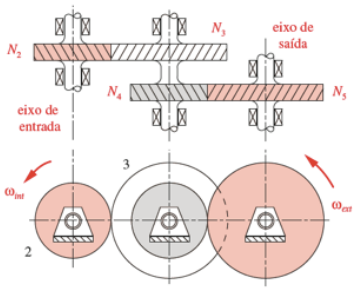

Figura 9. Engrenagens de transmissão (Norton, 2013).

Considere um motor de torque (T_{motor}) $1,5N.m$. Para que se tenha na engrenagem 5 um torque de $15N.m$, a proporção deve ser $1:10$, e a razão de engrenamento de cada par de engrenagem pode ser obtida através da raiz quadrada da razão de trem desejada, isto é, $\sqrt{10} = 3,2$. Assim, dois pares de engrenagens com essa razão seriam suficientes.

Por exemplo, ao considerar as engrenagens 2 e 4 com 15 dentes, as engrenagens 3 e 4, seguindo a proporção de $1:3,2$ de cada par, deverão ter 48 dentes. A partir dos números de dentes (N) em cada engrenagem, é possível calcular, pela Equação (9), o ganho, que é de 10,24, pouco mais que o desejado.

$$ganho = \left(\frac{N_2}{N_3}\right)\left(\frac{N_4}{N_5}\right) \tag{9}$$

Como mencionado, deseja-se que v_6 seja igual a $1,6m/s$. Para isso, é preciso considerar a relação da velocidade linear (v_6) com a velocidade angular (w_5), descrita em (10) e, dessa maneira, tem-se que $w_5 = 80rad/s$.

$$w_5.r_5 = v_6 \tag{10}$$

Tem-se que $r_3 = r_5 = 0,02m$ e que, pela proporção de cada par, $r_4 = r_2 = 0,006m$. Aplicando a relação (11), onde $n = 4$ e $m = 5$, tem-se que $w_4 = 250rad/s$.

$$w_n.r_n = w_m.w_m \quad (11)$$

Sabe-se que $w_4 = w_3$, pois estão acopladas no mesmo eixo. Logo, aplicando (11), onde $n = 2$ e $m = 3$, é possível concluir que $w_2 = 781rad/s$. Ou seja, esta é a velocidade angular que a engrenagem 2 deve girar para que a saída do sistema (engrenagem 5) gire a $1,6m/s$ como desejado.

De posse da velocidade angular da engrenagem acoplada ao eixo do motor (w_2) e do torque máximo (T_{motor}), é possível calcular a potência (P) do sistema por (12) e logo se tem que $P = 1,1kW$.

$$P = T.w_2 \quad (12)$$

Dessa forma, cada motor deve apresentar potência (P_{motor}) de, no mínimo, $550W$. Para realizar o cálculo da corrente (I_{motor}), de acordo com a Equação (13), admitiu-se uma tensão de alimentação (V) de 48 Volts. Logo, $I_{motor} = 11,4A$.

$$I_{motor} = P_{motor}/V \quad (13)$$

A partir dos cálculos, tem-se que motores de corrente contínua que apresentam, por exemplo, rotação reversível de 48 Volts, 12 Ampère, potência de 550 Watts e torque de 1,5N.m são boas opções para a implementação do projeto.

A bateria é um fator que determina a autonomia e a potência da cadeira de rodas motorizada. Ela precisa estar apta a alimentar todo o sistema: motores e o sistema embarcado. Seja um motor com as especificações mencionadas anteriormente, e o microcontrolador com consumo máximo de corrente de 600mA, é possível concluir que uma bateria recarregável de 48 Volts e 25A é capaz de suprir a necessidade do projeto, ou ainda, baterias com 48 Volts associadas em paralelo de forma que a soma das correntes seja 25A. Logo, uma bateria com 25Ah, por exemplo, seria capaz de oferecer uma autonomia ao sistema de 1 hora, ou, em movimento contínuo, cerca de 6.0km.

8 Conclusão

O trabalho desenvolvido envolve tecnologia assistiva, robótica, programação e cidadania. Concentra sua maior motivação em buscar trazer mais autonomia, conforto e, consequentemente, mais qualidade de vida aos cadeirantes, além de buscar trazer tópicos como a acessibilidade e assuntos relacionados ao debate e maior popularidade.

O resultado prático da aplicação pode ser acessado pelo *link*[4] disponível.

O resultado final do protótipo foi satisfatório, apesar da solução não fazer uso de uma cadeira de rodas por questões financeiras. Estima-se que a concretização do projeto custe cerca de 5 vezes mais que o valor de uma cadeira convencional que custa, por exemplo, R$ 400,00[5].

Para versões futuras, algumas considerações serão levantadas. Primeiro, seria interessante a implementação de um sistema de segurança, como uma chave automática, conectada ao cadeirante por um pequeno cabo, que desliga o sistema em casos de queda, ou ainda, a colocação de sensores para a parada da cadeira no caso de detecção de obstáculos. Pode-se citar outra medida como a implementação de criptografia na transmissão dos dados e um mecanismo de autenticação que só permitisse o acesso ao controle da cadeira para o próprio usuário.

Ainda sobre intenções futuras, o desenvolvimento teórico utilizado neste trabalho também poderia vir a servir em outros âmbitos, como um projeto de robô indoor para transporte de cargas com rotas previamente definidas.

[4]https://youtu.be/6IWfXRMrrr8

[5]https://goo.gl/NmjUWB

Referências

Albuquerque, P. U. B. (2011). *Sensores industriais: fundamentos e aplicações*, Editora Érica.

ATmel Corporation (2016). Atmega328/p. data sheet.

Bersch, R. (2008). Introdução à tecnologia assistiva, *Porto Alegre: CEDI* p. 21.

de Azevedo, A. M. C., Denadai, W. Z. and de Lima, L. E. M. (2016). Interface homem-máquina com uso de comandos de voz embarcada em plataforma android para controle de dirigibilidade de uma cadeira de rodas motorizada, *Congresso Brasileiro de Automática* .

EEMB Company (2007). Lithium polymer battery data sheet. data sheet.

El-Sherbiny, Y., Hasouna, A. and Ali, W. (2012). Friction coefficient of rubber sliding against flooring materials, *J. Eng. Appl. Sci* **7**(1): 121–126.

Norton, R. L. (2013). *Projeto de máquinas*, Editora Bookman.

Purwanto, D., Mardiyanto, R. and Arai, K. (2009). Electric wheelchair control with gaze direction and eye blinking, *Artificial Life and Robotics* **14**(3): 397.

Texas Instruments (2016). L293x quadruple half-h drivers. data sheet.

Velasco, A. F., Fernândez-Rodríguez, A. and Ron-Angevin, R. (2017). Switch mode to control a wheelchair through eeg signals, *Converging Clinical and Engineering Research on Neurorehabilitation II*, Springer, pp. 801–805.