Protótipo vestível para a detecção de posturas nocivas à saúde

Mauro Raibida[1], **Maria Luísa Ghizoni Gonzalez**[1]

[1]Departamento de Ciência da Computação
Universidade Estadual do Centro-Oeste (UNICENTRO)
R. Simeão Varela de Sá, 03 - Vila Carli – Guarapuava – PR – Brasil

mauroraibida@gmail.com, marialuisaghizoni@gmail.com

Abstract. *In order to improve the ergonomics of the work environment, since human beings spend considerable amounts of their time in said environment, in this article a prototype of a wearable device is developed to help a worker detect if their posture can be harmful to their health, which may cause injuries and deviations in the spine. By reading accelerometers and gyroscopes in harmful postures, it was possible to observe the behavior of the read data and to provide a base of hardware and embedded software for the device.*

Resumo. *Com o objetivo de melhorar a ergonomia do ambiente de trabalho, visto que o ser humano passa um tempo considerável do seu dia nele, neste artigo é desenvolvido um protótipo de um dispositivo vestível para auxiliar o trabalhador a detectar se sua postura pode prejudicar sua saúde, podendo lhe causar lesões e desvios na coluna vertebral. Realizando a leitura de acelerômetros e giroscópios em posturas prejudiciais foi possível observar o comportamento dos dados lidos e prover uma base de hardware e software embarcado para o dispositivo.*

1. Introdução

O ser humano passa grande parte do seu tempo no ambiente de trabalho, realizando as mais diversas atividades. Muitas vezes elas envolvem grande esforço físico, movimentos repetitivos, ficar em uma posição por muito tempo, etc. É de crucial importância garantir a saúde e bem estar dos trabalhadores, assim como evitar acidentes, reduzir nível de estresse e aumentar a produtividade. Para isso, a ergonomia no local de trabalho se faz necessária [Batiz et al. 2009].

Sendo a postura um dos principais fatores que influenciam na ergonomia, neste trabalho foi desenvolvido um protótipo de um dispositivo vestível utilizando acelerômetros e giroscópios MPU6050 e a plataforma NodeMCU focando no desenvolvimento do *hardware* e do *software* embarcado. Com o objetivo de auxiliar trabalhadores a manter suas posturas no ambiente de trabalho.

Este artigo é estruturado com as seguintes seções: Fundamentação teórica, apresentando a teoria utilizada no desenvolvimento; Materiais e métodos, descrevendo o que foi utilizado e como; Desenvolvimento, descrevendo detalhadamente como o protótipo foi feito e testado; Resultados e discussões, apresentando o que foi observado nos testes; Considerações finais, concluindo o que foi visto no artigo; Referências, mostrando artigos, livros, *sites*, manuais que foram utilizados na escrita e desenvolvimento.

2. Fundamentação Teórica

Nesta seção são apresentados os conceitos teóricos utilizados para o desenvolvimento do trabalho. Nas seguintes subseções tem-se:

- 2.1: Conceitos importantes da área da fisioterapia que motivam o desenvolvimento do trabalho
- 2.2 a 2.4: Conceitos de áreas computacionais que o projeto se encontra, como: sistemas embarcados, Internet das Coisas, computação vestível, interação humano-computador e *Body area network*.
- 2.5: Trabalhos que utilizam os mesmos materiais ou tem a mesma motivação.

2.1. Ergonomia e Postura

"A ergonomia se constitui como uma ciência aplicada capaz de adaptar as condições de trabalho às capacidades psicofisiológicas do trabalhador. É definida como um conjunto de conhecimentos científicos relativos ao homem e necessários à concepção de instrumentos, dispositivos, materiais que possam ser utilizados com o máximo de conforto, segurança e eficácia pelas pessoas." [Batiz et al. 2009]

Para garantir uma boa ergonomia no espaço de trabalho é necessária a manutenção da postura corporal correta, que consiste em: cabeça ereta, alinhada com ombros e coluna; ombros relaxados; coxas na horizontal a noventa graus do quadril; pés apoiados completamente no chão.

Posturas incorretas podem ocasionar mau alinhamento dos ossos, aumento da tensão em juntas e ligamentos, causando dor e cansaço. A longo prazo pode causar dores crônicas e desvios na coluna vertebral. Com objetivo de entender os desvios que podem ser causados, deve-se compreender melhor a divisão das regiões da coluna e sua curvatura natural, como tem-se na Figura 1 [Santos 2005].

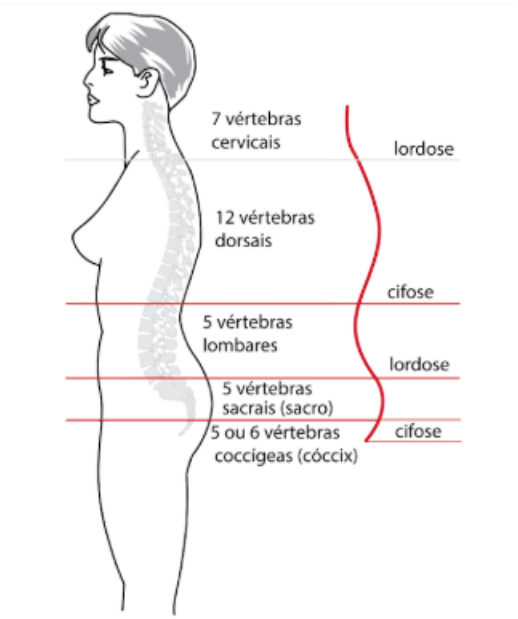


Figura 1. Regiões e curvatura natural da coluna vertebral

Os desvios são acentuações nas curvaturas naturais, os mais comuns são:

- Cifose: curvatura acentuada na região dorsal.
- Lordose: curvatura acentuada na região lombar.
- Escoliose: curvatura lateral na região dorsal.

2.2. Sistemas Embarcados

Um sistema embarcado é um sistema desenvolvido para desempenhar uma função específica de forma independente e repetida. Possuem os mesmos componentes de um computador (processador, memória, etc.), porém com capacidades limitadas, tendo em vista que tendem a ser de tamanho reduzido e com restrições de energia.

Sua comunicação com o ambiente externo é realizada por meio de sensores, que captam fenômenos e os transformam em corrente elétrica que então é processada pelo sistema. Através de atuadores (motores, relés, etc.) agem no ambiente conforme o sistema foi projetado [Heath 2002].

2.3. Internet das Coisas

É uma extensão da Internet atual, que permite que objetos troquem informações, acessem e provejam serviços. Eles podem ser definidos como um sistema embarcado que possui uma inteligência artificial que seja capaz de interpretar seu contexto, por meio dos sensores e atuar no ambiente de forma autônoma.

Pode ser vista também como uma junção de diferentes conceitos da computação: a embarcada; a ubíqua, que consiste na onipresença e na invisibilidade dos dispositivos; inteligência artificial, que simula uma inteligência similar à humana em *software*; Big Data a fim de processar grandes volumes de dados [Santos et al. 2016].

2.4. Computação vestível

Um computador vestível é um sistema ativo que age como uma extensão do usuário, aprimorando sua comunicação com o ambiente e assistindo-o em tarefas e situações diárias. Possuem grandes diferenças em relação a um sistema computacional convencional, dentre elas: consciência de contexto; habilidade de agir por si próprio ao invés de esperar comandos do usuário; transformar informações complexas em algo tangível para o usuário. Ou seja, pode ser visto como um sistema de Internet das Coisas; além de possuir uma interface que permita a mobilidade [Anliker et al. 2004].

Para se projetar um sistema vestível, devem ser levados em conta os seguintes aspectos:

- Perfis de usuário: especificações requeridas por diferentes grupos de usuários.
- Fluxo de informação: diferentes sensores e atuadores posicionados em diferentes partes do corpo.
- Restrições físicas: peso e localização do dispositivo são importantes na questão da vestibilidade.
- Recursos de hardware: Altos custos do hardware podem inviabilizar o projeto.

2.4.1. Interação Humano-Computador - IHC e *Humanistic Intelligence* - HI

IHC é o estudo da interação do ser humano com um dispositivo computacional, com objetivo de torná-la o mais natural possível. No fluxo de informações o usuário insere comandos por meio de um dispositivo de entrada que gera sinais para o sistema computacional, que por sua vez processa a requisição e mostra o resultado por meio de um dispositivo de saída, onde o usuário o interpreta [Witt 2007].

HI é o conceito que surge quando o ser humano se torna parte crucial do fluxo de informações nesse sistema computacional. Seu objetivo é fazer com que humano trabalhe muito próximo do dispositivo, usando uma interface de hardware que se encontra fisicamente próxima do usuário e constantemente acessível [Mann 2001].

A incorporação de HI em um sistema tem três modos operacionais fundamentais:

- *Constancy*: Um sistema HI deve estar em estado operacional e interativo constante, ou seja, deve estar sempre funcionando e recebendo dados.
- *Augmentation*: Deve assumir que a tarefa primária do usuário não é interagir com o sistema, mas sim realizar suas obrigações diárias enquanto o sistema coleta dados de sensores e os processa.
- *Mediation*: O dispositivo não é visto separado do usuário, como em um *smartphone*, por exemplo, mas sim como parte do dele, independente do grau de encapsulamento.

2.4.2. *Body Area Network*

É a rede de sensores sem fio de um dispositivo de computação vestível, podem ser implantados no corpo, posicionados na superfície corporal (sobre a pele ou roupas), ou em dispositivos externos ao corpo como bolsas e carteiras, por exemplo. Isso é possível devido ao processo de miniaturização de sensores, redução do consumo de energia, e evolução de redes sem fio.

Seu objetivo é possibilitar o monitoramento de parâmetros fisiológicos do usuário de forma contínua e em tempo real, permitindo acompanhar tratamentos e detectar doenças antecipadamente. Para isso, pode contar com sensores de pressão arterial, batimento cardíaco, temperatura corporal, movimento, etc. E deve possuir um processador, transmissor *wireless* e bateria [Yang and Yang 2006].

2.5. Trabalhos correlatos

Nos trabalhos correlatos apresentados abaixo, as motivações são similares a proposta deste trabalho: ajudar o usuário a manter a postura correta. Isso é feito realizando a leitura dos sensores e fazendo a análise desses dados, comparando-os a fim de fazer a detecção de posturas incorretas.

Em [Bramhapurikar et al. 2018] é usado um sensor *flex*, que varia a resistência dependendo da sua curvatura, que é preso à região lombar da coluna vertebral, onde, dependendo da resistência lida, é constatado o nível de corretude da postura. Os resultados são mostrados em uma tela LCD, por meio de um aplicativo que se comunica por *blueto-oth* com o dispositivo e por um motor de vibração, que alerta o usuário a necessidade de corrigir a postura.

Já em [Yamato 2017] são usados um *smartphone* e uma camiseta desenvolvida pela empresa Hitoe que possui um sensor preso ao peito que pode gerar um eletrocardiograma e que possui ainda um acelerômetro de três eixos. O objetivo é estimar a postura de motoristas e perceber quando ele se movimenta para pegar algo, tirando assim sua atenção do trânsito. Para isso é feita a análise dos valores dos eixos do acelerômetro da camiseta e para desconsiderar a aceleração do veículo é utilizado o acelerômetro do *smartphone*.

Em [Nath et al. 2017] dois smartphones são presos ao corpo de funcionários da construção civil, um ao pulso, outro à cintura visando analisar as posturas de trabalho. Dados são coletados dos três eixos dos acelerômetros e giroscópios dos *smartphones* em diferentes posições do usuário (flexões do tronco, cotovelo e ombro) e depois analisados e comparados com os resultados esperados.

3. Materiais e Métodos

Nesta seção são descritos os materiais e métodos utilizados para o desenvolvimento do projeto.

3.1. Arduino

É uma plataforma com *hardware* e *software* livres para prototipagem de dispositivos eletrônicos, com suporte para entradas e saídas analógicas ou digitais. Sensores e atuadores podem ser controlados por uma única placa, que possui um microcontrolador, e uma porta serial ou USB para a comunicação com o computador.

Por meio da Arduino IDE é possível desenvolver e carregar o código, desenvolvido em uma linguagem muito semelhante ao C e C++, para o dispositivo. São necessárias as funções de *setup*, que define o que deve ser feito ao ligar o Arduino, como definição de portas de entrada ou saída e inicialização da porta serial; e a função de *loop*, onde é feita a leitura de sensores, processamento e acionamento de atuadores, de forma repetida [McRoberts 2015].

É um produto popular no mercado e cada vez mais ganhando espaço, consequentemente tem uma grande quantidade de pessoas desenvolvendo projetos e com isso muito conhecimento é gerado para a plataforma, principalmente através de fóruns na Internet.

3.1.1. NodeMcu

É uma plataforma baseada na ideia do Arduino, porém é focada em aplicações de Internet das Coisas. Construído sobre o microcontrolador ESP8266EX, que possui *Wi-Fi* integrado, baixo consumo de energia e processador 32bit RISC com *clock* de 160 MHz. Sua linguagem de programação é a Lua, porém é possível utilizar a Arduino IDE com opções de compilação específicas para a placa [NodeMCU 2014].

Optou-se usar a plataforma do NodeMCU, mostrada na Figura 2, no projeto porque além de agregar todas as funcionalidades do Arduino, possui conexão*Wi-Fi* integrada, não necessitando o uso de *shields*, o que facilita o desenvolvimento da conectividade em trabalhos futuros.

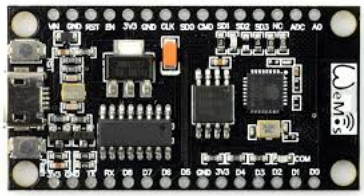


Figura 2. NodeMCU

3.2. MPU6050

O MPU6050 é um circuito integrado base para o desenvolvimento deste trabalho. Ele foi desenvolvido pela empresa *TDK Invensense*, faz parte da linha *MotionTracking*, focada em baixo consumo de energia, baixo custo e alta performance. Combina dois sensores:

- Acelerômetro: Sensor usado para medir a aceleração própria do sistema, que é a aceleração em relação com outro sistema em queda livre.
- Giroscópio: Sensor com um mecanismo de disco que mede a tendência de rotação do sistema quando forças externas agem sobre ele.

Ambos possuem três eixos (X, Y, Z), e utilizam a tecnologia MEMS (sistemas microeletromecânicos), que são fabricados usando técnicas de microeletrônica, permitindo estruturas mecânicas microscópicas. Possui um processador de movimento chamado DMP (*Digital Motion Processor*), que possibilita o processamento dos dados dos eixos diretamente no sensor [TDK 2019].

Optou-se por utilizar o MPU6050, mostrado na Figura 3, pelo seu baixo custo (cerca de 15 reais cada)[1], precisão, possibilidade de fazer o processamento dos dados diretamente no sensor, grande quantidade de material disponível e biblioteca de desenvolvimento consolidada.

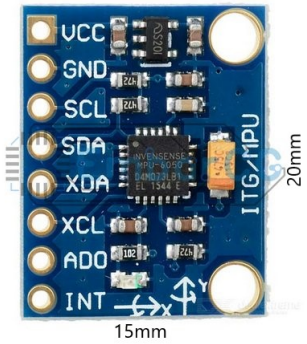


Figura 3. MPU6050 integrado a placa GY-521

3.2.1. I²C: *Inter-Integrated Circuit*

I²C é o padrão de comunicação do MPU6050, consiste em um barramento serial multimestre usado para conectar dispositivos de baixa velocidade a um sistema embarcado.

[1]Preço em fevereiro de 2019 no site mercadolivre.com.br

Conectam-se pelos pinos de *clock* (SCL) e de dados (SDA) como é mostrado a Figura 4. Estes dispositivos trabalham no modelo mestre e escravo, onde ambos podem enviar e receber informações, porém é o mestre que gera o *clock* e inicia a comunicação e o escravo recebe o sinal de *clock* e possui um endereço para referência [NXP 2014].

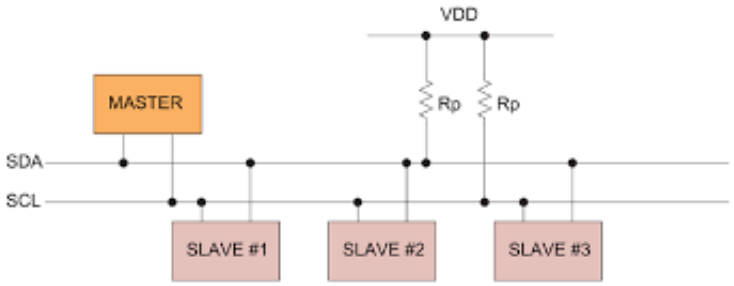


Figura 4. Diagrama do barramento I^2C

3.3. Metodologia de Desenvolvimento

Segundo [Zurita 2011], metodologias para desenvolvimento de projetos de sistemas embarcados podem ser divididas em três categorias principais:

- *Bottom-up*: Inicia-se descrevendo detalhadamente cada componente do sistema, que são conectados para formar subsistemas cada vez maiores até o sistema completo. Permite separar com clareza cada nível de abstração em uma biblioteca de componentes, facilitando o gerenciamento do projeto. Porém devem-se levar em consideração todos os componentes e parâmetros, o que é algo difícil de se prever em abstrações de baixo nível.
- *Top-down*: Inicia-se com uma formulação das características finais do sistema, de forma abstrata, sem detalhes de implementação. Com o decorrer do projeto, o sistema é dividido consecutivamente em subsistemas. É uma metodologia intuitiva e permite um bom grau de automatização, porém uma vez definido um subsistema que não atende os requisitos é necessário voltar um nível de abstração e refazer o trabalho.
- *Meet-in-the-middle*: Combina as metodologias *bottom-up* e *top-down* com objetivo de aproveitar suas vantagens e evitar suas desvantagens. Inicia-se com uma descrição de alto nível, refinando-a até um nível intermediário, após isso são feitas as descrições de componentes.

Para o desenvolvimento do projeto optou-se pela metodologia *meet-in-the-middle*, por oferecer uma boa visão geral do sistema, combinado com uma descrição detalhada de componentes já na fase intermediária, não havendo necessidade de retrabalho caso algum componente não atenda os requisitos. Na Figura 5 tem-se um fluxo de projeto compatível com metodologias *meet-in-the-middle* proposto por [Berger 2002].

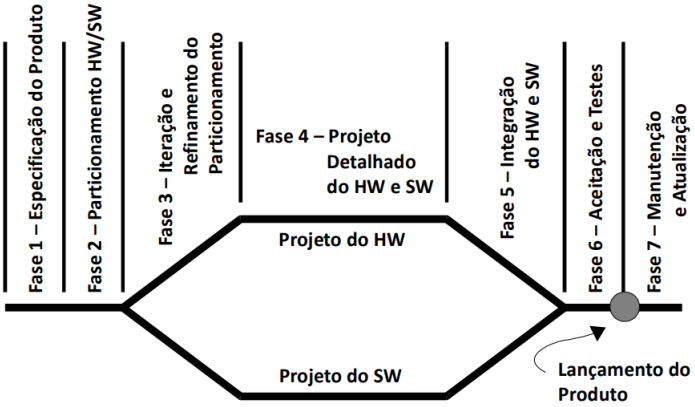


Figura 5. Fluxo de projeto de sistemas embarcados

- Fase 1 - Especificação do produto: Inicia-se com uma descrição informal do sistema, depois parte-se para a especificação dos requisitos, que são a base do projeto e estabelecem critérios para a tomada de decisão em fases futuras.
- Fase 2 - Particionamento *hardware* e *software*: Deve-se decidir quais partes serão implementadas em *hardware* ou *software*, dependendo do projeto, é preferível utilizar componentes e bibliotecas já existentes.
- Fase 3 - Iteração e refinamento do particionamento: São feitos testes em partes do *hardware* e *software*, em busca de resultados que satisfaçam os requisitos.
- Fase 4 - Projeto detalhado do *hardware* e *software*: Deve-se implementar componentes de hardware e software, corrigir erros que surgem ao longo do projeto.
- Fase 5 - Integração do *hardware* e *software*: Componentes devem ser reunidos e integrados de forma a compor o sistema embarcado completo.
- Fase 6 - Aceitação e testes: O sistema deve ser testado em condições similares às que ele será submetido após o lançamento.
- Fase 7 - Manutenção e atualização: Deve-se otimizar as características do produto lançado, correção de problemas, bem como o enriquecimento da documentação.

4. Desenvolvimento

Nesta seção são descritas as fases do desenvolvimento do projeto, baseado no fluxo proposto por [Berger 2002].

4.1. Especificação do produto

O dispositivo deve detectar posturas nocivas à saúde, por meio de sensores (acelerômetros e giroscópios) presos a um colete que o usuário possa vestir em seu local de trabalho, visando seu bem-estar e prevenindo possíveis desvios na coluna vertebral.

Para isso o dispositivo deve:

- Possuir sensores posicionados de forma que coincidam com a coluna vertebral do usuário, ou seja, na vertical e no centro da parte de trás do colete.
- Realizar a leitura dos sensores, informando valores de posição de cada um.
- Guardar os valores dos sensores com o usuário na postura correta para referência.
- Realizar leituras periodicamente e compará-las com os valores de referência.
- Desconsiderar interferências como respiração, posição sentada ou em pé, fala, etc.
- Alertar o usuário se sua postura está incorreta por mais tempo que o especificado.

4.2. Particionamento do *hardware* / *software*

O *hardware* deve conter:

- Sensores: MPU6050 para realizar a leitura da posição de diferentes pontos da coluna vertebral
- Controlador: NodeMCU para receber os dados dos sensores, realizar o processamento, e envio das informações.
- Colete: Principal meio de interação com o usuário, deve conter os sensores e o controlador presos a ele.

O *software* deve conter:

- Bibliotecas: Conjunto de funções que auxiliam na leitura dos dados dos sensores, processamento de dados, envio de informações, etc.
- Parâmetros: variáveis para controlar a periodicidade de leitura dos sensores e desconsiderar ruídos dos dados.
- Lógica: passos a serem seguidos para alcançar o objetivo. Para isso deve ser realizada a leitura dos sensores, definição da postura correta e verificação da postura segundo os parâmetros.

4.3. Iteração e refinamento do particionamento

Para que o dispositivo cumpra os requisitos, as leituras dos sensores devem ser consistentes, caso contrário o software pode interpretar de maneira errônea a postura do usuário

Os testes desenvolvidos nessa fase foram utilizados códigos *open source* disponíveis nas bibliotecas utilizadas e na Internet, com o objetivo de testar as funcionalidades do MPU6050:

Teste 1: Conexão do NodeMCU com o MPU6050 via I^2C. Foi utilizado a biblioteca Wire.h disponível na Arduino IDE. No código da Figura 6 é iniciada a conexão informando os pinos SDA e SCL, dados e *clock* respectivamente, e passado o endereço do sensor.

```
void setup(){
  Serial.begin(9600);
  Wire.begin(sda_pin, scl_pin);// D5, D6
  Wire.beginTransmission(MPU);
  Wire.write(0x6B);
  Wire.write(0);
  Wire.endTransmission(true);
}
```

Figura 6. Função *setup* iniciando a conexão com o sensor

Teste 2: Leitura de dados brutos do MPU6050, através dos registradores do sensor. No código da Figura 7 o registrador inicial é informado para ser utilizado como referência na leitura dos dados, após isso é feito um *shift* de 8 *bits* para realizar a leitura do próximo registrador.

```
Wire.beginTransmission(MPU);
Wire.write(0x3B);
Wire.endTransmission(false);
Wire.requestFrom(MPU, 14, true);

AcX = Wire.read() << 8 | Wire.read();
AcY = Wire.read() << 8 | Wire.read();
AcZ = Wire.read() << 8 | Wire.read();
GyX = Wire.read() << 8 | Wire.read();
GyY = Wire.read() << 8 | Wire.read();
GyZ = Wire.read() << 8 | Wire.read();
```

Figura 7. Atribuição dos valores dos registradores nas variáveis correspondentes na função loop

Teste 3: Leitura de dados processados pelo DMP. Foi utilizado a biblioteca MPU60506AxisMotionApps20.h[2]. Os resultados do Teste 2 mostraram valores muito grandes e com pouca variação em relação ao movimento do sensor. A solução encontrada foi utilizar o DMP, que aliado a biblioteca retorna valores mais consistentes e em diferentes formatos. O escolhido foi o sistema de ângulos de Euler, denotados por: (*roll* - α, *pitch* - β, *yaw* - γ) que descrevem a orientação de um corpo no espaço euclidiano (x,y,z).

```
void loop() {
  if (!dmpReady) return;
  while (!mpuInterrupt && fifoCount < packetSize) {
    if (mpuInterrupt && fifoCount < packetSize) {
      fifoCount = mpu.getFIFOCount();
    }
  }
  mpuInterrupt = false;
  mpuIntStatus = mpu.getIntStatus();
  fifoCount = mpu.getFIFOCount();
  if ((mpuIntStatus & _BV(MPU6050_INTERRUPT_FIFO_OFLOW_BIT)) || fifoCount >= 1024) {
    mpu.resetFIFO();
    fifoCount = mpu.getFIFOCount();
  } else if (mpuIntStatus & _BV(MPU6050_INTERRUPT_DMP_INT_BIT)) {
    while (fifoCount < packetSize) fifoCount = mpu.getFIFOCount();
    mpu.getFIFOBytes(fifoBuffer, packetSize);
    fifoCount -= packetSize;
    mpu.dmpGetQuaternion(&q, fifoBuffer);
    mpu.dmpGetEuler(euler, &q);
    Serial.print(euler[0] * 180 / M_PI);
    Serial.print(", ");
    Serial.print(euler[1] * 180 / M_PI);
    Serial.print(", ");
    Serial.println(euler[2] * 180 / M_PI);
    delay(1000);
  }
}
```

Figura 8. Leitura dos dados processados pelo DMP

Para isso é necessário realizar a configuração do sensor estabelecendo a conexão pelo barramento I^{2}C, inicializando o DMP, definindo os *offsets* do sensor (compensações

[2]https://github.com/jrowberg/i2cdevlib/tree/master/Arduino/MPU6050

dos eixos necessárias devido a falhas de construção, que são lidos por uma rotina presente na biblioteca utilizada) e definindo as interrupções (necessárias para fazer a leitura da FIFO e uso do DMP).

No código da Figura 8 os valores são lidos do *buffer* da fila do sensor, passados como parâmetro para as funções da biblioteca que fazem o processamento e mostrados no formato (α, β, γ).

Teste 4: Leitura de dados processados pelo DMP de dois MPU6050. O sensor conta com dois endereços para realizar a conexão, controlados pelo pino AD0, que quando atribuído o valor lógico LOW (0 volts) configura o endereço 0x68 e quando atribuído o valor HIGH (3.3 volts) configura o endereço para 0x69. Para realizar a leitura de ambos sensores é repetido o código do Teste 3, porém para as duas variáveis representadas na Figura 9.

```
MPU6050 mpu1;
MPU6050 mpu2(0x69);
```

Figura 9. Declaração das variáveis dos sensores com endereços diferentes

4.4. Projeto detalhado e integração do *hardware* e *software*

Com os teste tendo resultados satisfatórios, parte-se para o desenvolvimento do *hardware*. Primeiramente definiu-se todas as conexões que os sensores MPU6050 devem ter com o NodeMCU, mostradas na Figura 10[3].

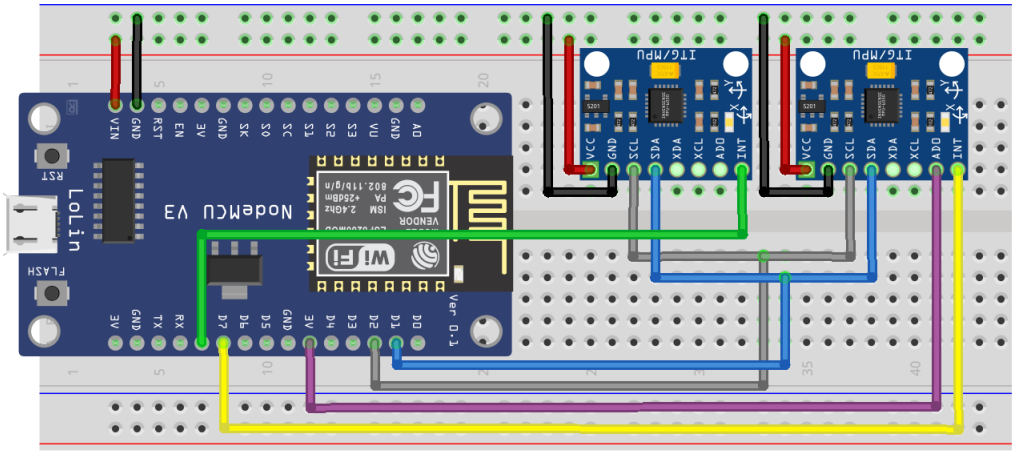


Figura 10. Diagrama esquemático das conexões

No MPU6050 tem-se os seguintes pinos:

- VCC: Alimentação positiva do sensor, recomendado utilizar 5 volts para evitar erros de leitura.
- GND: Alimentação negativa do sensor.
- SCL: Sinal de clock do barramento.

[3] fritzing.org

- SDA: Sinal de dados do barramento.
- XDA: Sinal de dados auxiliar, utilizado para conexão com outros sensores (não utilizado).
- XCL:Sinal de clock auxiliar, utilizado para conexão com outros sensores (não utilizado).
- AD0: Alteração do endereço do sensor.
- INT: Interrupção do sensor.

Foram utilizados os seguintes pinos do NodeMCU:

- VIN: Alimentação positiva vindo diretamente do USB (5V), utilizado para alimentar os pinos VCC dos sensores.
- GND: Alimentação negativa (0V), utilizado para alimentar os pinos GND do sensor.
- 3V: Alimentação positiva de 3.3V, utilizado no pino AD0 para mudar o endereço do segundo sensor.
- D1 e D2: Pinos digitais utilizados para a comunicação I^2C.
- D7 e D8: Pinos digitais utilizados para a interrupção do sensor.

Definidas as conexões, os sensores e fios devem ser presos ao colete, para isso eles foram costurados a uma das metades do velcro com uma distância de 33 centímetros, para que um dos sensores ficasse localizado na parte superior da região dorsal da coluna e o outro na região lombar. A outra metade do velcro foi costurada ao colete, na vertical e no meio da parte de trás, para que fique alinhado à coluna, como mostrado na Figura 11.

Figura 11. Parte traseira do colete

Tendo o protótipo do *hardware* desenvolvido, parte-se para a definição de alto nível do *software* embarcado. A Figura 12 representa o fluxograma, ou seja, os passos que se devem seguir para cumprir o objetivo.

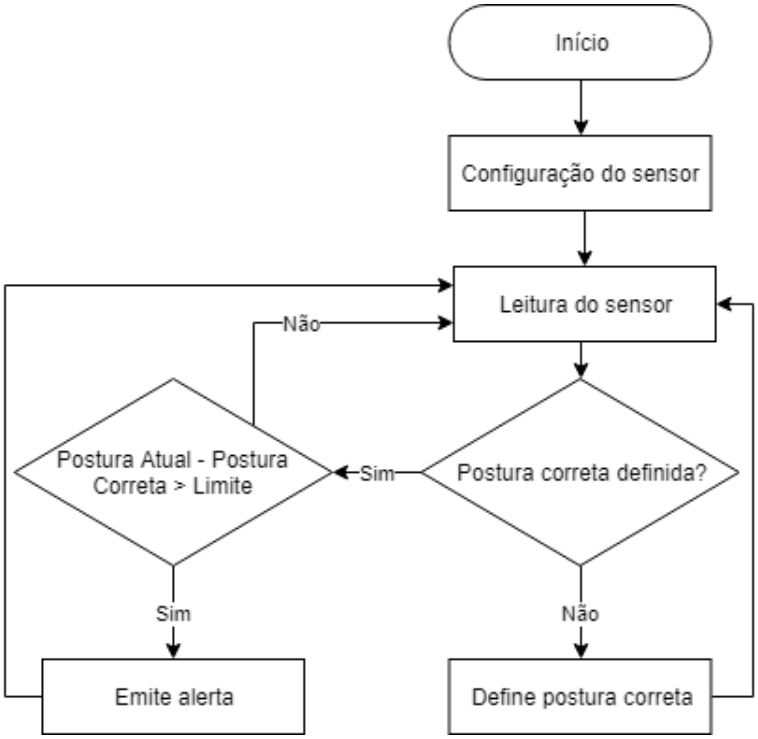


Figura 12. Fluxograma do software embarcado

Como nos testes 3 e 4 da fase de iteração e refinamento do particionamento foram bem sucedidos, os mesmos códigos podem ser usados nas fases de configuração e leitura dos sensores.

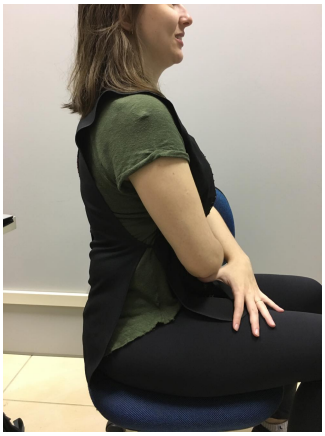
Para a definição da postura correta, o usuário deve permanecer parado até que a leitura dos sensores seja estável, então os valores deverão ser usados para comparação posteriormente.

Inicialmente deve-se realizar a leitura dos sensores em diferentes posturas incorretas, com o objetivo de encontrar parâmetros para comparar com os valores encontrados na definição da postura correta

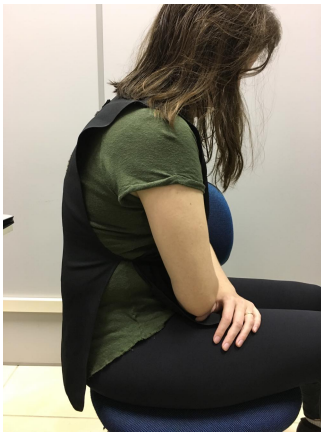
4.5. Aceitação e testes

Os testes foram feitos a fim de observar o comportamento dos ângulos processados pelo sensor quando o usuário está com a postura correta e muda para uma postura que prejudique sua saúde. Para isso foram consideradas as posturas que quando mantidas por um longo prazo levam a desvios permanentes na coluna.

Foram realizados dois testes, sendo que em cada um se aferiu o valor dos ângulos a cada segundo para os três desvios posturais, sempre partindo da postura correta mostrada na Figura 13(a) movendo-se para a incorreta mostrada nas Figuras 13(b) e 13(c) e voltando à correta novamente, sempre permanecendo dez segundos em cada. Somente para o teste da escoliose segui-se a sequência Figuras 13(a), 13(d), 13(e) e 13(a).



(a) Correta



(b) Cifose

(c) Lordose

(d) Escoliose para direita

(e) Escoliose para esquerda

Figura 13. Posturas que podem levar a desvios na coluna

Os dados foram exportados no formato csv(*comma separated values*) e utilizando o LibreOffice Calc[4], foram plotados em gráficos (um para cada sensor, sendo o MPU1 o primeiro e MPU2 o segundo) do tipo linha e ponto para uma análise manual. O eixo x representando a leitura feita ao decorrer do tempo e o eixo y o valor do ângulo.

5. Resultados e Discussões

Nesta seção são apresentadas as tabelas dos testes, descritos na seção 4.5, correspondentes a cada desvio postural.

Na Figura 14 é possível observar que os ângulos do MPU1 tem uma variação considerável quando a postura mudou, porém apenas o MPU2 seguiu um padrão na leitura, variando os ângulos *roll* e *pitch* em valores semelhantes e mantendo *yaw* constante

[4]https://pt-br.libreoffice.org/

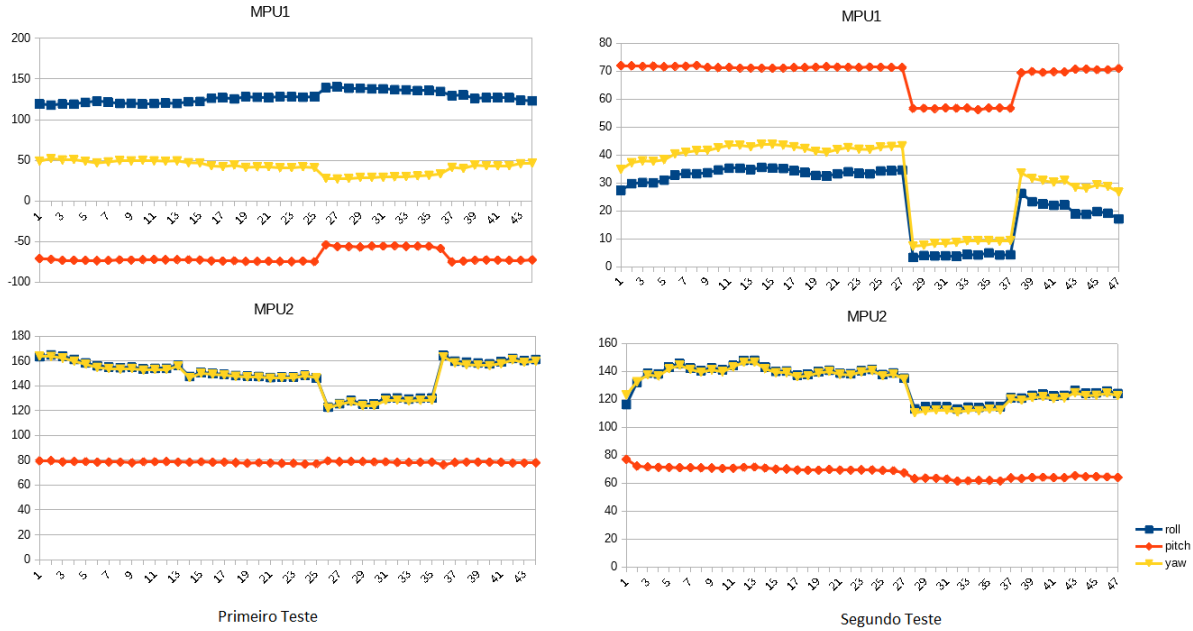


Figura 14. Ângulos lidos no teste de postura referente à cifose

Na Figura 15 pode-se perceber o mesmo padrão nos testes e sensores, no MPU1 todos os ângulos tiveram seus valores diminuídos enquanto o MPU2 teve *roll* e *yaw* diminuídos e *pitch* com um leve acréscimo quando mudou-se da postura correta para a incorreta

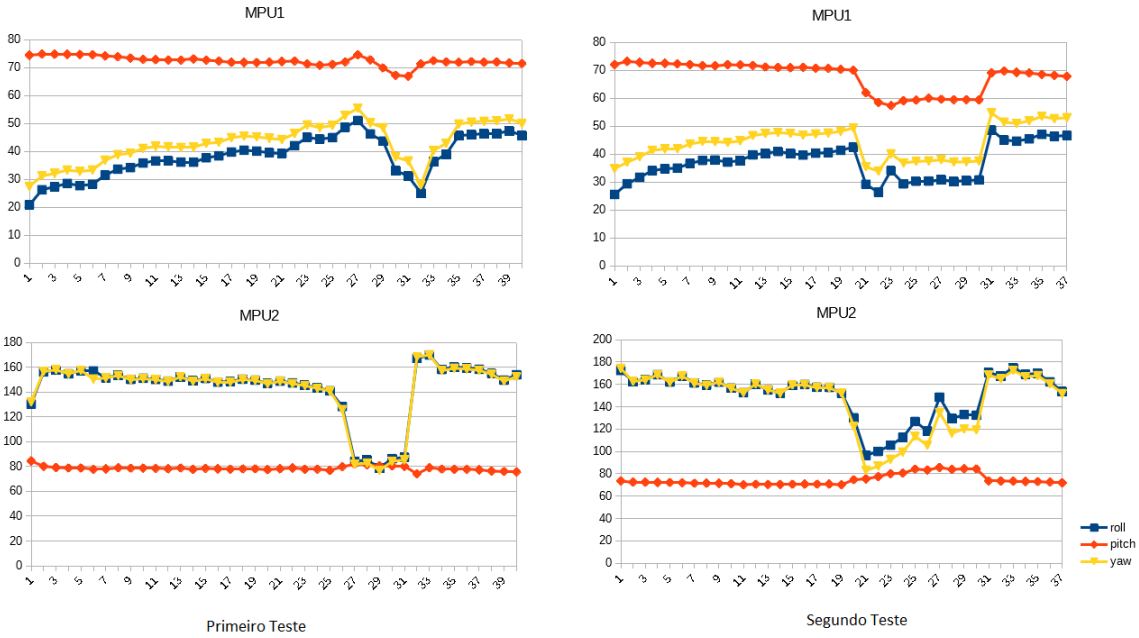


Figura 15. Ângulos lidos no teste de postura referente à lordose

Na Figura 16 os ângulos *roll* e *yaw* se comportaram de maneira diferente nos dois testes, somente o *pitch* seguiu um padrão, tendo duas variações para baixo no MPU1 e uma no MPU2.

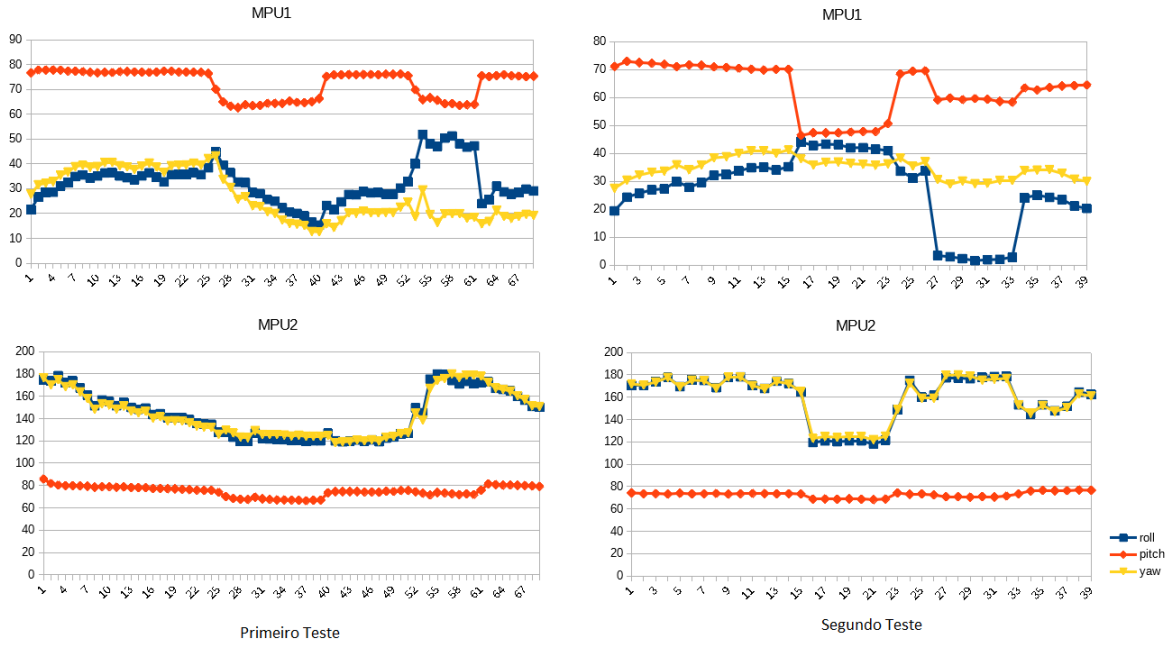


Figura 16. Ângulos lidos no teste de postura referente à escoliose

Considerando o que foi observado nos gráficos pode-se dizer que é possível detectar se o usuário passa de uma postura correta para a incorreta se algum ângulo tem uma mudança de valor significativa em um curto espaço de tempo, fenômeno que pôde ser observado em todos os testes (não podendo especificar a que desvio ele estará sujeito) mas somente se ele permanecer na mesma posição, pois nos testes não foram considerados se o usuário sentou, levantou ou girou no próprio eixo. Esses movimentos alterariam os valores dos ângulos causando uma detecção errônea.

Para resolver este problema, estes testes devem ser realizados com diferentes pessoas, em diferentes posições, e com a ajuda de um algoritmo de inteligência artificial estabelecer parâmetros iniciais para a detecção. Com isso definido deve-se usar um algoritmo de aprendizagem de máquina para a detecção das posturas de cada usuário em específico, utilizando seu *feedback* por meio de um aplicativo de *smartphone* por exemplo.

6. Considerações finais

Este artigo apresentou o desenvolvimento de um protótipo vestível para a detecção de posturas nocivas à saúde, apresentando sua especificação, desenvolvimento do *hardware* e do software embarcado assim como testes iniciais.

Verificou-se que os sensores e a plataforma de desenvolvimento tiveram um bom desempenho na leitura e processamento dos dados brutos, porém os testes iniciais sem a implementação de uma inteligência artificial para a classificação dos dados obtidos, não tiveram o resultado esperado, provando-se necessária essa área no desenvolvimento de sistemas computacionais vestíveis.

Para trabalhos futuros pretende-se: i) Elaborar de testes mais abrangentes. ii) Fazer uso de inteligência artificial e aprendizado de máquina para a detecção das posturas. iii) Desenvolver uma comunicação do dispositivo com a Internet.

Com este trabalho foi possível dar um passo inicial para que pessoas tenham uma jornada de trabalho com menos danos físicos causados pela má postura, para que assim possam desempenhar suas atividades com mais vigor e saúde.

Referências

Anliker, U., Beutel, J., Dyer, M., Enzler, R., Lukowicz, P., Thiele, L., and Troster, G. (2004). A systematic approach to the design of distributed wearable systems. *IEEE Transactions on Computers*, 53(8):1017–1033.

Batiz, E. C., dos Santos, A. F., and Licea, O. E. A. (2009). A postura no trabalho dos operadores de checkout de supermercados: uma necessidade constante de análises. *Production*, 19(1):190–201.

Berger, A. S. (2002). *Embedded Systems Design – An Introduction to Process, Tools, & Techniques*. CMP Books.

Bramhapurikar, K., Prabhune, A., Chavan, S., Ghivela, G. C., and Sengupta, J. (2018). A wearable posture corrector device. In *2018 9th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, pages 1–5. IEEE.

Heath, S. (2002). *Embedded systems design*. Elsevier.

Mann, S. (2001). Wearable computing: Toward humanistic intelligence. *IEEE Intelligent Systems*, 16(3):10–15.

McRoberts, M. (2015). Arduino básico. *Novatec Editora, 2ªed*.

Nath, N. D., Akhavian, R., and Behzadan, A. H. (2017). Ergonomic analysis of construction worker's body postures using wearable mobile sensors. *Applied ergonomics*, 62:107–117.

NodeMCU (2014). Nodemcu connect things easy. Disponível em: https://www.nodemcu.com/index_en.html Acesso em 20 de abril de 2019.

NXP (2014). *I2C-bus specification and user manual*.

Santos, A. (2005). *Postura corporal: um guia para todos*. Summus editorial.

Santos, B. P., Silva, L., Celes, C., Borges, J. B., Neto, B. S. P., Vieira, M. A. M., Vieira, L. F. M., Goussevskaia, O. N., and Loureiro, A. (2016). Internet das coisas: da teoria à prática. *Minicursos SBRC-Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuidos*.

TDK (2019). Mpu-6050. Disponível em: https://www.invensense.com/products/motion-tracking/6-axis/mpu-6050/ Acesso em 20 de abril de 2019.

Witt, H. (2007). Human-computer interfaces for wearable computers. *PhD thesis*.

Yamato, Y. (2017). Experiments of posture estimation on vehicles using wearable acceleration sensors. In *ieee 3rd international conference on big data security on cloud*

(bigdatasecurity), ieee international conference on high performance and smart computing (hpsc), and ieee international conference on intelligent data and security (ids), pages 14–17. IEEE.

Yang, G.-Z. and Yang, G. (2006). *Body sensor networks*, volume 1. Springer.

Zurita, M. E. (2011). Projeto de sistemas embarcados. *Universidade Federal do Piauí, Curso de Engenharia Elétrica, Campus Universitário Ministro Petrônio Portela*.