UNIVERSIDADE FEDERAL DO ESPÍRITO SANTO
CENTRO TECNOLÓGICO
DEPARTAMENTO DE ENGENHARIA ELÉTRICA
PROJETO DE GRADUAÇÃO

SÉRGIO SILVA MUCCIACCIA

DESENVOLVIMENTO DE UM SISTEMA DE ANÁLISE DO MOVIMENTO HUMANO BASEADO EM SENSORES INERCIAIS

VITÓRIA – ES
AGOSTO/2014

SÉRGIO SILVA MUCCIACCIA

DESENVOLVIMENTO DE UM SISTEMA DE ANÁLISE DO MOVIMENTO HUMANO BASEADO EM SENSORES INERCIAIS

Parte manuscrita do Projeto de Graduação do aluno Sérgio Silva Mucciaccia, apresentado ao Departamento de Engenharia Elétrica do Centro Tecnológico da Universidade Federal do Espírito Santo, como requisito parcial para obtenção do grau de Engenheiro Eletricista.

Orientador: Prof. Dr. Anselmo Frizera Neto

VITÓRIA – ES
AGOSTO/2014

SÉRGIO SILVA MUCCIACCIA

DESENVOLVIMENTO DE UM SISTEMA DE ANÁLISE DO MOVIMENTO HUMANO BASEADO EM SENSORES INERCIAIS

Parte manuscrita do Projeto de Graduação do aluno Sérgio Silva Mucciaccia, apresentado ao Departamento de Engenharia Elétrica do Centro Tecnológico da Universidade Federal do Espírito Santo, como requisito parcial para obtenção do grau de Engenheiro Eletricista.

Aprovada em 7 de agosto de 2014.

COMISSÃO EXAMINADORA:

Prof. Dr. Anselmo Frizera Neto
Universidade Federal do Espírito Santo
Orientador

Prof. Dr. André Ferreira
Universidade Federal do Espírito Santo
Examinador

Laura Susana Vargas Valencia
Universidade Federal do Espírito Santo
Examinadora

RESUMO

A utilização de dispositivos para a análise da marcha humana começou no século XIX e desde então esses sistemas têm evoluído incorporando os avanços tecnológicos. Sensores inerciais são sensores capazes de medir parâmetros cinemáticos como acelerações, velocidades, deslocamentos e ângulos. No entanto por serem geralmente grandes e pesados não podiam ser fixados no corpo e utilizados para estudos clínicos ou na área de biomecânica. Os últimos avanços da tecnologia permitiram a fabricação de sensores inerciais baseados em sistemas micro-eletro-mecânicos. Estes são pequenos e leves e podem agora ser usados para medição de parâmetros relacionados com o movimento humano, como os ângulos das articulações. O presente Projeto de Graduação apresenta o desenvolvimento de um sistema de análise dos ângulos dos segmentos do corpo humano utilizando sensores inerciais desenvolvidos pela empresa espanhola Technaid S.L. O sistema consiste em um software que obtêm as informações dos sensores, realiza o tratamento necessário, armazena e mostra os dados obtidos para o usuário utilizando uma interface gráfica.

LISTA DE FIGURAS

Figura 1 - Posicionamento dos marcadores 11
Figura 2 - Eletrogoniômetro 12
Figura 3 - Protótipo de um eletrogoniômetro de tecido 13
Figura 4 - Perturbação induzida com uma vareta 14
Figura 5 - Sistem MVN Biomech da XSENS 16
Figura 6 – Pessoa utilizando o sistema MVN Biomech 16
Figura 7 - Sensor inercial e dispositivo com conexão wifi da MotionNode 17
Figura 8 - Sistema sendo utilizado em uma pessoa 17
Figura 9 - Sistema IGS 180 da Synertial 18
Figura 10 - Sistema de captura de movimento da Technaid 19
Figura 11 - Janela de interface com o usuário 27
Figura 12 - Renderização do cubo descrito 29
Figura 13 - Modelo humano feito no Blender 30
Figura 14 - Translação de um modelo tridimensional 32
Figura 15 - Transformação de escala aplicada em um modelo tridimensional 33
Figura 16 - Rotação de um modelo tridimensional. 34
Figura 17 - Rotação de um cubo em volta de um eixo 35
Figura 18 – Modelo humano utilizado no trabalho 41
Figura 19 - Sistema final com a interface gráfica e o modelo tridimensional 42
Figura 20 - Gráficos mostrando os ângulos de Euler de dois sensores 44
Figura 21 - Paralelepípedo que se movimenta junto com o sensor 45
Figura 22 - Sensores medindo a mesma orientação 46
Figura 23 - Sensores medindo orientações diferentes 46
Figura 24 - Sistema sendo testado com dois sensores 47
Figura 25 - Teste com a perna dobrada 48
Figura 26 - Teste com a perna esticada 48
Figura 27 - Sensores posicionados no paciente 49

LISTA DE QUADROS

Quadro 1 - Especificações dos sistemas de sensores inerciais 19
Quadro 2 - Métodos e eventos da classe cMCSHub..... 21
Quadro 3 - Métodos da classe cMCSSensor..... 22
Quadro 4 - Métodos estáticos da classe File..... 23
Quadro 5 - Métodos estáticos da classe Directory..... 24
Quadro 6 - Métodos da classe StreamReader 25
Quadro 7 - Métodos da classe StreamWriter..... 25
Quadro 8 - Janela com botão, label e checkbox descrita em XAML 27
Quadro 9 - Conteúdo de um arquivo que descreve o modelo tridimensional de um cubo..... 29
Quadro 10 - Métodos do programa principal 39

SUMÁRIO

1 INTRODUÇÃO 8
1.1 Apresentação 8
1.2 Justificativa 9
1.3 Objetivos 10
1.3.1 Objetivo Geral 10
1.3.2 Objetivos Específicos 10
2 SISTEMAS DE ANÁLISE DO MOVIMENTO HUMANO 11
2.1 Fotogrametria 11
2.2 Eletrogoniômetros 12
2.3 Eletrogoniômetros vestíveis 13
2.4 Sensores Inerciais 14
2.5 Sistemas de Análise de movimento baseados em Sensores Inerciais 15
2.5.1 Xsens MVN Biomech 15
2.5.2 MotionNode Bus 17
2.5.3 Synertial IGS 180 18
2.5.4 Tech MCS 18
2.5.5 Comparação entre os sistemas 19
3 MATERIAIS E MÉTODOS 20
3.1 Métodos de computação 20
3.1.1 O SDK da Technaid 20
3.1.1.1 A classe cMCSHub 20
3.1.1.2 A classe cMCSSensor 22
3.1.1.3 As classes de frames 22
3.1.2 Ler e escrever arquivos em C# 23
3.1.2.1 Classes File e Directory 23
3.1.2.2 Classes FileStream, StreamReader e StreamWriter 24
3.1.3 Interface Gráfica baseada em XAML 26
3.1.4 Computação gráfica 28
3.2 Formulação Matemática 30
3.2.1 Translação 31
3.2.2 Escala 32
3.2.3 Rotação 33

3.2.3.1 Rotação por um ângulo em torno de um eixo 34
3.2.3.2 Matriz de rotação 35
3.3 Desenvolvimento do sistema 38
3.3.1 Interface com o usuário 38
3.3.2 Aplicativo para aquisição das medições 39
3.3.3 Modelo tridimensional 40
4 VALIDAÇÃO EXPERIMENTAL 43
4.1 Protocolo experimental 43
4.1.1 Testes dos gráficos 43
4.1.2 Teste dos sensores parados 43
4.1.3 Teste dos sensores em movimento 44
4.1.4 Teste do sistema 45
4.2 Resultados 45
4.2.1 Testes dos gráficos 45
4.2.2 Teste dos sensores parados 45
4.2.3 Teste dos sensores em movimento 47
4.2.4 Teste do sistema 47
5 CONCLUSÕES E TRABALHOS FUTUROS 49
6 REFERÊNCIAS BIBLIOGRÁFICAS 51

1 INTRODUÇÃO

1.1 Apresentação

O trabalho científico mais antigo sobre a análise do movimento humano foi publicado em 1836 pelos irmãos Weber (ASSIS, 1991), Wilhelm Eduard Weber, um físico famoso por inventar junto à Carl Friedrich Gauss o primeiro telégrafo eletromagnético e Eduard Friedrich Weber anatomista e fisiologista. Usando ferramentas como um telescópio com escalas e um cronômetro eles descreveram e mediram componentes da marcha como comprimento do passo, cadência, a liberação do pé do solo e a excursão vertical do corpo.

Posteriormente, a evolução da cinematografia permitiu a captação e estudo de movimentos humanos e de animais de forma muito mais precisa que os métodos usados até então. Como exemplo, Muybridge, em 1877, usou uma sequência de fotografias para comprovar que durante o trote do cavalo existe um breve momento em que as quatro patas ficam sem tocar o solo, resolvendo uma antiga controvérsia (MUYBRIDGE, 1893).

Atualmente, laboratórios especializados de biomecânica realizam a análise de marcha em espaços estruturados utilizando câmeras de alta velocidade sincronizadas. Uma vez que essa tecnologia é raramente disponível no ambiente clínico típico, muitas vezes os médicos devem confiar na observação visual para avaliar as características de marcha dos pacientes.

Sensores inerciais são utilizados em uma vasta faixa de aplicações. Telefones celulares, sistemas de entretenimento, computadores portáteis, automóveis, aviões e helicópteros são alguns produtos que utilizam destes dispositivos (BARSHAN e DURRANT-WHYTE, 1995). O avanço na tecnologia destes sensores permitiu que fossem desenvolvidos modelos cada vez menores e mais leves, o que possibilitou seu uso na área médica, no campo da biomecânica.

Se colocados nos segmentos do corpo humano, os sensores inerciais podem medir os ângulos das articulações. Ao posicionar um sensor no braço e outro no antebraço, o ângulo entre os dois sensores pode ser obtido, o que equivale ao ângulo da articulação do cotovelo. A distribuição destes sensores pelo corpo possibilita a obtenção de várias informações sobre a cinemática do corpo humano. No entanto, os sensores inerciais apresentam a desvantagem de medirem, com precisão, apenas parâmetros angulares. A medição do deslocamento com estes

sensores não possui muita precisão. Assim mesmo, algumas posições dos membros podem ser obtidas de forma indireta utilizando-se dos ângulos encontrados e incorporando esta informação em um modelo biomecânico que contemple a medida dos segmentos do corpo humano.

Considerando os argumentos supracitados, algumas questões podem ser levantadas: Sistemas com sensores inerciais possuem as características de precisão e velocidade necessárias para serem utilizados na análise da atividade motora humana? Quais as vantagens e desvantagens do uso desse tipo de sistema sobre os outros? Um sistema desse tipo pode ser desenvolvido de modo a ser facilmente utilizado por médicos e fisioterapeutas em um consultório?

Este trabalho tem como objeto de pesquisa o desenvolvimento de um sistema de análise da atividade motora humana usando sensores inerciais, com isto as dificuldades encontradas no desenvolvimento de tal sistema e suas características podem ser avaliadas e comparadas com outros tipos de sistemas existentes descritos na literatura.

1.2 Justificativa

A análise da variação dos ângulos das articulações durante o movimento humano é importante em algumas áreas. Para avaliar o impacto de doenças neuromusculares na caminhada, o uso funcional das pernas precisa ser analisado durante as atividades diárias, o mesmo vale para os membros superiores (LUINGE e VELTINK, 2005). Um sistema de medição da cinemática das pernas com este objetivo deve ser portátil e não interferir com essas atividades. Sensores inerciais são adequados para esta tarefa, pois são pequenos e podem ser alimentados por bateria.

Na computação gráfica, movimentos de seres humanos precisam ser estudados com objetivo de criar animações digitais, utilizadas em filmes e jogos. Informações sobre como diferentes pessoas andam são importantes para a criação de animações mais detalhadas e realistas. Na robótica, sistemas com sensores inerciais são utilizados para melhorar a obtenção de movimento e o controle de robôs.

Os sistemas de fotogrametria 3D são os mais utilizados para medidas da cinemática do corpo humano, porém possuem algumas limitações, dentre elas o alto custo, a baixa portabilidade e

necessitam de um espaço estruturado para seu funcionamento (WHITTLE, 2007). Sendo assim, sistemas baseados em sensores inerciais são utilizados nas mais recentes pesquisas sobre a cinemática do corpo humano.

Os sensores inerciais utilizados atualmente possuem pequenas dimensões e são leves, de modo que podem ser colocados no corpo sem interferir nos movimentos do usuário. Essas características motivam o seu uso no desenvolvimento de sistemas de análise da atividade motora humana, principalmente para aplicações que necessitam de uma maior portabilidade do sistema.

1.3 Objetivos

1.3.1 Objetivo Geral

O principal objetivo deste trabalho é desenvolver um sistema que permita medir os ângulos dos segmentos do corpo humano ao executar movimentos utilizando sensores inerciais. O sistema deve ser capaz de mostrar e armazenar as informações medidas.

1.3.2 Objetivos Específicos

Os objetivos específicos que devem ser atingidos para alcançar o objetivo geral são:

- Pesquisar na literatura os sistemas existentes para medição dos ângulos das articulações e compará-los quanto à precisão, custo e praticidade;
- Desenvolver um sistema capaz de medir os ângulos das articulações das pernas com sensores inerciais;
- Criar uma interface com o usuário para o sistema;
- Representar os movimentos com um modelo 3D.

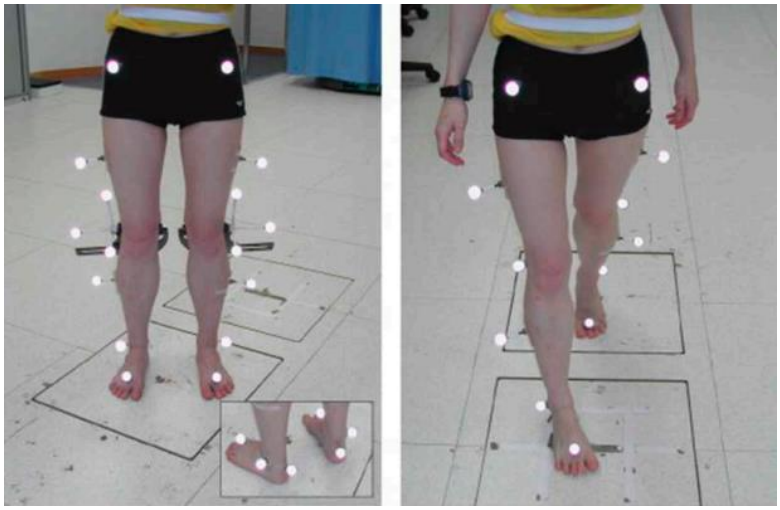
2 SISTEMAS DE ANÁLISE DO MOVIMENTO HUMANO

2.1 Fotogrametria

A fotogrametria é a prática de determinar as propriedades geométricas de um objeto a partir de imagens fotográficas (JIA, 2010). A fotogrametria 3D consiste em estimar as coordenadas de pontos tridimensionais de um objeto a partir das imagens de diferentes câmeras colocadas em ângulos distintos. Esta técnica também pode ser utilizada para se obter os ângulos das articulações da perna de uma pessoa, por exemplo. Sacco et al. (2007) apresenta um estudo no qual compara a obtenção de quatro ângulos dos membros inferiores utilizando fotogrametria e goniômetros.

Os sistemas de fotogrametria para análise do movimento humano representam os mais tradicionais e confiáveis. Normalmente são utilizadas câmeras sensíveis a frequência infravermelha e marcadores colocados nos segmentos do corpo. Com o posicionamento dos marcadores e um modelo biomecânico do corpo humano é possível criar um modelo virtual para análise dos movimentos humanos (WINTER, 2009). A Figura 1 mostra o posicionamento de marcadores em uma pessoa para a realização de estudos de análise de marcha.

Figura 1 - Posicionamento dos marcadores



Fonte: (MELIM, 2013).

As desvantagens desse tipo de sistema são a necessidade de um espaço ou de uma sala especial para fazer as medições, além do custo elevado. Algumas análises são mais

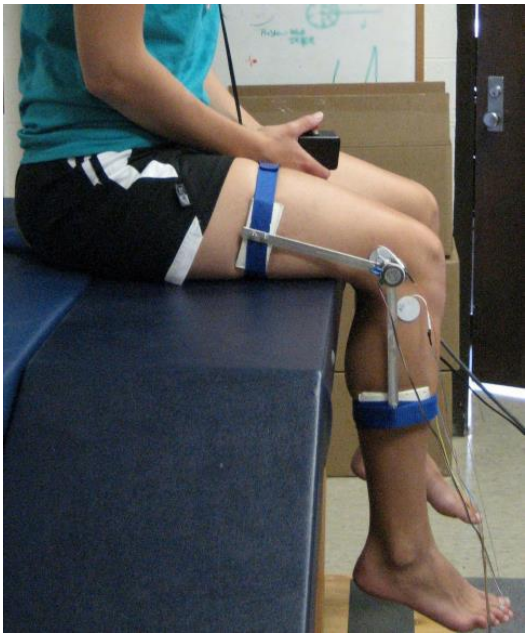
interessantes quando feitas em locais específicos e, muitas vezes, não estruturados. Além disso, objetos na linha de visada das câmeras podem atrapalhar as medições, o que inviabiliza o uso dessa técnica caso seja necessário ter outros objetos próximos a pessoa monitorada.

2.2 Eletrogoniômetros

Os eletrogoniômetro são dispositivos que utilizam transdutores que alteram suas propriedades elétricas com a mudança de ângulo. Um potenciômetro pode ser usado, de modo que seja conectado a duas hastes rígidas que podem ser colocadas nos segmentos da perna de uma pessoa fazendo com que ao dobrá-la a resistência do potenciômetro seja alterada. Um circuito pode medir a resistência do potenciômetro e inferir o ângulo entre as hastes.

Esse tipo de sistema possui baixo custo e boa precisão. As hastes podem limitar os movimentos da pessoa, mas existem eletrogoniômetros que possuem potenciômetros nos três eixos, limitando muito pouco os movimentos da pessoa. A desvantagem é que um sistema utilizando esta técnica necessita de fios para ligar os potenciômetros ao equipamento que realiza a medição, o que restringe a locomoção do indivíduo e não permite a sua utilização em qualquer local. A Figura 2 mostra um eletrogoniômetro utilizado para medir o ângulo do joelho.

Figura 2 - Eletrogoniômetro



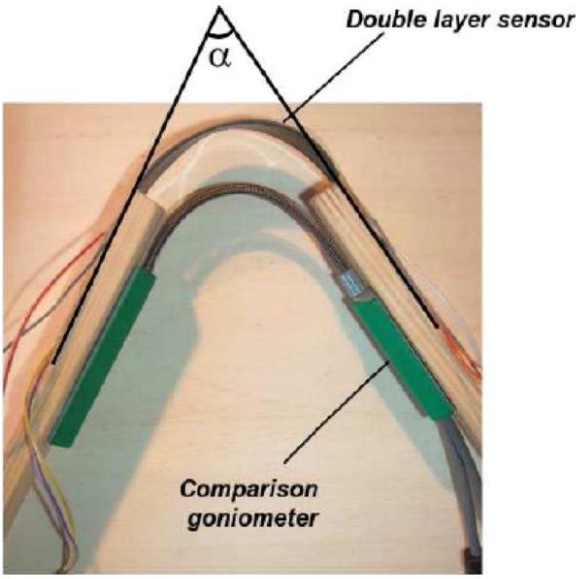
Fonte: (COLLINS, 2009).

2.3 Eletrogoniômetros vestíveis

Os sistemas clássicos de captura de movimentos podem ser bem precisos, porém, em geral, eles necessitam de um ambiente específico ou restringem os movimentos do corpo. Por isso, muitas vezes eles não são viáveis para uma monitoração a longo prazo dos movimentos humanos. O desenvolvimento de sistemas de medição vestíveis, no entanto, pode permitir que os movimentos sejam monitorados com conforto e em qualquer lugar, tornando-os aptos a executar um monitoramento de longo prazo.

Transdutores piezoelétricos feitos de fibras elásticas, apresentam diferentes características elétricas quando curvados, por isso podem ser utilizados em tais sistemas vestíveis. Um protótipo de um sensor deste tipo é descrito por Lorussi et al. (2009). A Figura 3 mostra uma tira de tecido sendo usada para medir um ângulo de curvatura enquanto a medida é feita simultaneamente por um eletrogoniômetro comum para comparar o erro.

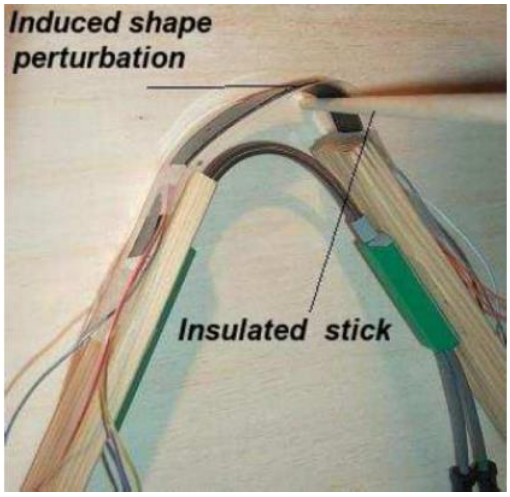
Figura 3 - Protótipo de um eletrogoniômetro de tecido



Fonte: (LORUSSI et al., 2009)

Este sensor possui um custo muito baixo, mas tem a desvantagem de que um mesmo ângulo de uma articulação pode curvar o tecido de formar diferentes de modo que medidas distintas são obtidas. A Figura 4 retrata como ocorre este tipo de perturbação.

Figura 4 - Perturbação induzida com uma vareta



Fonte: (LORUSSI et al., 2009)

O erro desse sistema varia, portanto de acordo com o desvio de curvatura percebida pelo tecido, sendo que o menor erro é obtido quando o sensor forma uma circunferência perfeita. Com o maior desvio de curvatura considerado foi obtido um erro de 10% no ângulo medido e com um erro de curvatura aceitável foi obtido um erro de 4%. Com a curvatura perfeita a medida é obtida com erro menor que 1%. Lorussi et al. (2013) demonstrou que o erro é proporcional ao cubo da curvatura máxima ao longo do tecido.

2.4 Sensores Inerciais

Os sensores inerciais são dispositivos capazes de medir sua orientação no espaço. São conhecidos internacionalmente como *Inertial Measurement Unit* (IMU). Eles são, na verdade, uma combinação de três tipos de sensores.

Três giroscópios, se combinados colocando-se um em cada eixo do espaço, conseguem obter os três ângulos de orientação no espaço. No entanto, a integração da velocidade angular dos giroscópios para obtenção de ângulos tridimensionais pode apresentar derivas importantes, comprometendo seu uso e precisão em sistemas de análise de movimento.

Três acelerômetros, se colocados um em cada eixo do espaço, conseguem identificar para onde está a gravidade, medindo a aceleração em cada sensor e combinando essas informações. Porém se estiverem em movimento acelerado, a aceleração resultante não será a da gravidade, logo eles só conseguem saber a direção da gravidade se estiverem em repouso ou em

velocidade constante. Além disso, o conhecimento da direção da gravidade permite apenas a obtenção de dois ângulos no espaço, uma vez que rotações ao redor do eixo gravitacional não são percebidas pelo sensor.

Três magnetômetros, se colocados da mesma forma, perpendiculares uns aos outros, conseguem medir o campo magnético da terra e identificar o norte magnético. Então esses sensores conseguem medir um ângulo de orientação no espaço. Celulares, antenas e outros equipamentos que emitem ondas eletromagnéticas causam interferência nas medidas desse tipo de sensor. Quando combinados com acelerômetros, o conjunto consegue identificar os três ângulos de orientação no espaço, desde que os sensores não estejam em movimento acelerado.

Um sensor inercial é usualmente uma combinação desses sensores citados (MAENAKA, 2008), pode também possuir um sensor de temperatura para corrigir os valores medidos dos outros sensores que são sensíveis à temperatura. Com esses sensores combinados, podem ser usados algoritmos de modo que o sensor resultante apresente pouca sensibilidade à deriva, à aceleração, à interferência magnética e à variação de temperatura. Todas essas interferências continuam afetando o sensor resultante, porém de maneira muito reduzida.

Considerando que os sistemas baseados em sensores inerciais são o foco deste trabalho, a seção seguinte apresenta uma breve descrição de alguns sistemas de análise de movimentos baseados nesta tecnologia.

2.5 Sistemas de Análise de movimento baseados em Sensores Inerciais

2.5.1 Xsens MVN Biomech

O sistema MVN Biomech da Xsens permite a conexão de até 17 sensores inerciais. Possui conexão sem fio de até 150 m, frequência de amostragem de até 120 Hz e erro na medição dos ângulos menor que 0.5º. A Figura 5 apresenta o sistema dentro da mala de proteção.

Figura 5 - Sistem MVN Biomech da XSENS

Fonte: Site da Xsens (www.xsens.com).

A Figura 6 apresenta os sensores inerciais posicionados nos segmentos do corpo de uma pessoa.

Figura 6 – Pessoa utilizando o sistema MVN Biomech

Fonte: Site da Xsens (www.xsens.com)

2.5.2 MotionNode Bus

O sistema da MotionNode suporta a utilização de até 20 sensores simultaneamente e possui conexão Wi-Fi. Pode gravar os dados em uma memória *flash* interna de 4 Gb. Cada sensor pesa 10 g e tem dimensões de 35x35x15 mm. A Figura 7 mostra um sensor inercial e um controlador *bus* responsável pela conexão dos sensores.

Figura 7 - Sensor inercial e dispositivo com conexão wifi da MotionNode

Fonte: Site da MotionNode (www.motionnode.com).

A Figura 8 mostra os sensores colados nos segmentos do braço de um indivíduo.

Figura 8 - Sistema sendo utilizado em uma pessoa

Fonte: Site da MotionNode (www.motionnode.com).

2.5.3 Synertial IGS 180

A Synertial possui um sistema que suporta até 18 sensores simultaneamente. Possui conexão wireless de até 50m, frequência de captura de até 500Hz e erro na medição dos ângulos menor que 1º. A bateria dura apenas 4 horas. A Figura 9 mostra o sensor a esquerda e o dispositivo que se conecta ao computador para receber o sinal sem fio à direta.

Figura 9 - Sistema IGS 180 da Synertial

Fonte: Site da Synertial (www.synertial.com).

2.5.4 Tech MCS

O dispositivo utilizado para os estudos realizados neste trabalho foi o Tech MCS (Technaid S.L., Espanha). Este sistema é apresentado na Figura 10. Recentemente foi adquirido pelo Laboratório de automação inteligente da UFES um desses sistemas e diversos Projetos de Pesquisa já fazem uso deste dispositivo para a avaliação motora e estudos relacionados com a biomecânica.

Figura 10 - Sistema de captura de movimento da Technaid

Fonte: Site da Technaid S. L. (http://www.technaid.com).

O sistema citado possui capacidade para a conexão de 16 sensores inerciais simultaneamente. O Hub (Tech HUB) pode ser conectado ao computador por um cabo USB ou por Bluetooth. Necessita de 4 baterias AA e absorve uma corrente máxima de 3 A, quando conectados 16 sensores. Possui entrada para um cartão Micro SD de 8 Gb que pode ser usado para medições sem o auxílio do computador. Cada IMU (Tech IMU) pesa 9 gramas e tem as dimensões 36x26x11 mm.

2.5.5 Comparação entre os sistemas

O quadro 1 reúne algumas especificações dos sensores citados anteriormente.

Quadro 1 - Especificações dos sistemas de sensores inerciais

Sistema	Número de sensores suportados	Precisão	Frequência máxima de captura	Alcance da conexão sem fio	Gravação em memória
Xsens MVN Biomech	17	0.5º	120 Hz	150 m	Não possui
MotionNode Bus	20	2º	100 Hz	Não possui	4 Gb internos
Synertial IGS 180	18	1º	500 Hz	50 m	Não possui
Technaid Tech MCS	16	1º	100 Hz	150 m	8 Gb (Micro SD)

Fonte: Produção do próprio autor.

3 MATERIAIS E MÉTODOS

3.1 Métodos de computação

A Technaid S.L. fornece um kit de desenvolvimento de aplicativos (SDK) para ser utilizado com o seu sistema de captura de movimentos. O SDK é incluído no projeto como uma biblioteca e com isso suas classes passam a ser acessíveis do ambiente de programação. A Technaid S.L. fornece o SDK para as linguagens C# e para Matlab. Neste trabalho serão utilizados a linguagem C#, o ambiente de programação gratuito Visual C# Express 2010 da Microsoft e a biblioteca da Technaid S.L.

As seções seguintes explicam a base teórica necessária para utilizar os sensores da Technaid S.L. em um ambiente de programação, para ler e escrever arquivos utilizando a linguagem C#, para criar um programa visual de interface com o usuário utilizando XAML e para representar modelos tridimensionais utilizando o *framework* XNA.

3.1.1 O SDK da Technaid

Esse kit de desenvolvimento provido pelo fabricante contém classes que permitem conectar, calibrar e obter os dados medidos pelos sensores inerciais, assim como obter informações sobre o sistema. A classe cMCSHub representa o dispositivo Tech hub, que conecta vários sensores e se comunica com o computador. A classe cMCSSensor representa um único sensor e possui métodos para recolher os dados capturados pelo sensor específico. As classes ORIENTATION_FRAME, QUATERNION_FRAME e PHYSICAL_FRAME, são usadas como tipos para armazenar um quadro de informações obtido em um dado momento pelo sensor.

3.1.1.1 A classe cMCSHub

Essa classe possui todos os métodos e propriedades necessários para manipular o dispositivo Tech hub e tem um vetor de objetos da classe cMCSSensor que representam todos os sensores conectados ao hub. Para que o sistema funcione é necessário seguir uma sequência de passos. Primeiro a porta USB que o hub está conectado deve ser encontrada, depois o hub deve ser conectado ao programa utilizando essa porta USB. Após conectado o comando de iniciar captura deve ser enviado ao hub, que faz com que os sensores comecem a gravar os quadros

de orientação obtidos. Quando as medições terminarem, o comando de parar captura deve ser enviado ao Hub. Finalmente o hub pode ser desconectado, ou pode ser iniciado outro processo de capturas.

Os sensores armazenam os dados medidos internamente, e só enviam para o computador quando requisitados. A classe cMCSHub possui eventos que são desencadeados quando os sensores adquirem um quadro de informações, ou quando ocorre algum erro. Portanto para que um programa leia as informações em tempo real, esses eventos podem ser conectados a métodos, de modo que quando o evento ocorre, o método é disparado e executado. Uma outra forma de fazer isso é varrer os sensores periodicamente com um certo intervalo de tempo, assim quando alguma informação nova for disponível o programa pode requisitá-la. O Quadro 2 descreve os principais métodos e eventos da classe cMCSHub, que são utilizados para executar os passos necessários para que o sistema funcione. (TECHNAID, 2008).

Quadro 2 - Métodos e eventos da classe cMCSHub

Métodos	Descrição
AutoDetectPort	Detecta automaticamente em qual porta USB está o hub
setCOMPort	Estabelece manualmente com qual porta se conectará o dispositivo
Conect	Conecta o dispositivo já especificado
Disconnect	Encerra a conexão com um dispositivo conectado
StartCapture	Inicia a captura dos dados
StopCapture	Para a captura dos dados
IsConnect	Retorna true se o dispositivo está conectado e false caso contrário
Reset	Reinicia o dispositivo
Eventos	**Descrição**
PhysicalFrameReceived	É desencadeado quando um frame contendo as medidas do sensor é recebido
OrientationFrameReceived	É desencadeado quando sensor recebe um quadro
QuaternionFrameReceived	É desencadeado quando sensor recebe um quadro

Fonte: Produção do próprio autor.

A classe possui muitos outros métodos, muitos não necessários no desenvolvimento do presente projeto, como por exemplo métodos para escrita e leitura de dados em cartão micro SD, outros que podem ser usados, mas em geral com os métodos citados no quadro anterior é possível fazer todas as funcionalidades do sistema.

3.1.1.2 A classe cMCSSensor

Essa classe representa um sensor inercial Tech IMU. Com essa classe os dados dos acelerômetros, giroscópios e magnetômetros que estão contidos nos sensores podem ser armazenados e manipulados. Também é possível obter diretamente as informações de orientação, calculadas pelo dispositivo com um algoritmo de fusão de dados de acelerômetros, giroscópios e magnetômetros baseado em Filtros de Kalman (TECHNAID, 2008). Estes filtros são implementados diretamente nos microcontroladores embarcados no sistema. O Quadro 3 descreve os métodos da classe cMCSSensor.

Quadro 3 - Métodos da classe cMCSSensor

Métodos	Descrição
getLastPhysicalFrame	Retorna o último objeto PHYSICAL_FRAME armazenado no sensor
getLastOrientationFrame	Retorna o último objeto ORIENTATION_FRAME armazenado no sensor
getLastQuaternionFrame	Retorna o último objeto QUATERNION_FRAME armazenado no sensor
getAllPhysicalFrames	Retorna uma lista com todos os objetos PHYSICAL_FRAME armazenados
getAllOrientationFrames	Retorna uma lista com todos os objetos ORIENTATION_FRAME armazenados
getAllQuaternionFrames	Retorna uma lista com todos os objetos QUATERNION_FRAME armazenados
getCountPhysicalFrames	Retorna o número de objetos PHYSICAL_FRAME armazenados
getCountOrientationFrames	Retorna o número de objetos ORIENTATION_FRAME armazenados
getCountQuaternionFrames	Retorna o número de objetos QUATERNION_FRAME armazenados
clearAllFrames	Remove todas as informações medidas armazenadas nos sensores

Fonte: Produção do próprio autor.

3.1.1.3 As classes de frames

Essas classes têm a função de armazenar as informações obtidas pelos sensores. A biblioteca da Technaid possui cinco classes que podem armazenar essas informações, as três principais são as classes PHYSICAL_FRAME, ORIENTATION_FRAME e QUATERNION_FRAME.

A primeira serve para armazenar as informações dos 3 acelerômetros, 3 giroscópios, 3 magnetômetros e também a temperatura. Cada Tech IMU possui os 10 sensores citados e para cada sensor o frame físico armazena 2 bytes que representam a medida daquele sensor. Assim cada frame físico possui 20 bytes e representa todas as informações obtidas pelo sensor em um determinado instante de tempo.

As outras duas classes servem para armazenar a orientação do sensor em um determinado instante de tempo. O Tech IMU utiliza algoritmos internos para fazer a fusão dos sensores, e disponibiliza a informação da orientação utilizando esses algoritmos. A classe ORIENTATION_FRAME representa a orientação de um IMU por meio de uma matriz de rotação 3x3 e guarda cada elemento da matriz em 2 bytes, também guarda a temperatura em 4 bytes totalizando 22 bytes. A classe QUATERNION_FRAME representa a orientação de um IMU por meio de um quatérnio, e guarda cada componente do quatérnio em 4 bytes, resultando em 16 bytes de informação por frame. (TECHNAID, 2008).

3.1.2 Ler e escrever arquivos em C#

3.1.2.1 Classes File e Directory

Para ler e armazenar informações no computador, é necessário conseguir manipular arquivos no programa. A classe File possui métodos para esta finalidade e o Quadro 4 descreve os principais métodos da classe que criam, copiam, movem e removem arquivos por exemplo.

Quadro 4 - Métodos estáticos da classe File

Métodos	Descrição
Copy	Copia o arquivo para um novo arquivo
Create	Cria um arquivo e retorna o seu FileStream
Exists	Retorna true se o arquivo existir e false caso contrário
Delete	Remove o arquivo especificado
Move	Move o arquivo especificado para o local especificado
Open	Retorna o FileStream de um arquivo especificado

Fonte: Produção do próprio autor.

Quando vários arquivos são utilizados, é interessante organizá-los em uma estrutura de diretórios. A classe Directory possui métodos para isso e eles são descritos no Quadro 5.

Quadro 5 - Métodos estáticos da classe Directory

Métodos	Descrição
CreateDirectory	Cria um diretório
Delete	Remove o diretório especificado
Exists	Retorna true se o diretório especificado existe e false se não
GetDirectories	Retorna todos os subdiretórios de um diretório
GetFiles	Retorna o nome de todos os arquivos no diretório
Move	Move o diretório especificado para o local especificado
GetCurrentDirectory	Retorna uma string contendo o diretório onde está o executável

Fonte: Produção do próprio autor.

3.1.2.2 Classes FileStream, StreamReader e StreamWriter

Na plataforma .NET todas as operações de entrada e saída envolvem o uso de *streams*. Eles são uma abstração que permitem que o mesmo código possa ser usado para operações de entrada e saída em diferentes dispositivos como o disco rígido, o teclado ou até mesmo a rede, com apenas pequenas modificações.

Um objeto da classe FileStream é utilizado para ler e escrever arquivos. Porém esta classe enxerga apenas bytes e só opera em cima deste tipo. Assim, ela pode ser usada para ler arquivos de texto, imagem, som ou qualquer tipo, mas devem ser feitas conversões para manipular esses dados. Um objeto FileStream armazena um ponteiro para o arquivo internamente que pode ser modificado com o método Seek, ou seja, tem a vantagem que o arquivo pode ser lido ou escrito em qualquer ponto apenas alterando esse ponteiro.

Lidar com vetores de bytes pode não ser muito interessante em um arquivo de textos. Neste caso, as classes StreamRead e StreamWriter podem ajudar. Essas classes são específicas para texto, lendo o arquivo e colocando diretamente em uma *string*, ou escrevendo uma dada *string* em um arquivo. Todavia ela tem a desvantagem de ler e escrever apenas sequencialmente, não é possível pular para o meio do arquivo, ou para onde quiser como em um FileStream. Essas classes utilizam internamente um objeto FileStream, e realizam apenas a conversão.

A classe FileStream pode ser instanciada com um construtor que aceita dois parâmetros. O primeiro é uma *string* contendo o caminho do arquivo especificado e o segundo um objeto

FileMode que informa se a classe deve sobrescrever (FileMode.Open), anexar a informação em arquivo um existente (FileMode.Append) ou criar um arquivo (FileMode.Create). Para leitura e escrita são utilizados os métodos ReadByte e WriteByte. Para mudar o ponteiro do arquivo é utilizado o método Seek, que aceita dois parâmetros, um número com o deslocamento e um objeto do tipo SeekOrigin. Em geral são utilizados os valores SeekOrigin.Begin e SeekOrigin.Current (NAGEL et al., 2010).

Os Quadros 6 e 7 mostram a descrição dos construtores e dos principais métodos das classes StreamReader e StreamWriter.

Quadro 6 - Métodos da classe StreamReader

Métodos	Descrição
StremReader(Stream)	Construtor - Cria um objeto StremReader a partir de um FileStream
StremReader(String)	Construtor - Cria um objeto StremReader a partir do caminho do arquivo
Read	Lê um caractere
ReadLine	Lê uma linha inteira e a retorna como uma string
ReadToEnd	Lê o arquivo até o final e retorna uma string
Close	Fecha o StreamReader
Dispose	Libera a memória dos recursos consumidos pelo StreamReader

Fonte: Produção do próprio autor.

A classe StreamWriter apresenta funcionamento parecido com a StreamReader, porém é utilizada para escrita. A comparação fica evidente no Quadro 7.

Quadro 7 - Métodos da classe StreamWriter

Métodos	Descrição
StreamWriter(Stream)	Construtor - Cria um objeto StremWriter a partir de um FileStream
StreamWriter(String)	Construtor - Cria um objeto StremWriter a partir do caminho do arquivo
Write	Escreve um objeto no arquivo que pode ser char, int, double, string e outros
WriteLine	Escreve uma string no arquivo com um terminador de linha no final
Close	Fecha o StreamWriter
Dispose	O mesmo do StreamReader, deve ser chamado quando não é mais utilizado

Fonte: Produção do próprio autor.

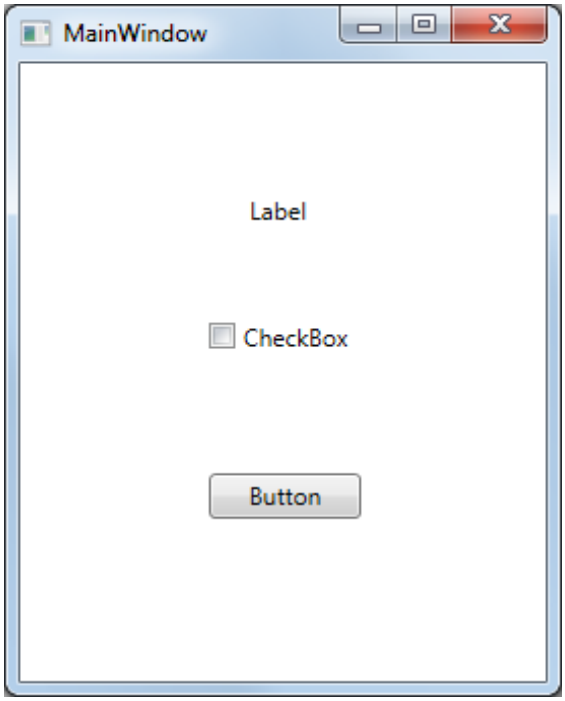
Com o conteúdo apresentado nesta seção é possível efetuar todas as operações necessárias para guardar e recuperar a informação obtida dos sensores inerciais. O usuário do sistema pode precisar guardar as informações da marcha de uma pessoa para comparar com os dados de outras pessoas no futuro. Da mesma maneira é útil que os dados possam ser recuperados para análise. Com isso além de analisar os dados em tempo real no momento em que as medidas são obtidas o usuário também pode salvar e abrir arquivos contendo sequências de movimentos.

3.1.3 Interface Gráfica baseada em XAML

Extensible Application Markup Language (XAML) é uma linguagem utilizada para desenvolver a interface com o usuário de um programa. Ela fornece a abstração de desacoplar a interface do programa, permitindo que para um dado programa, a interface possa ser modificada sem modificações no programa e para uma dada interface, o programa por trás possa ser modificado sem preocupação com a interface. Isso permite que o programador se concentre em cada objetivo de uma vez, ou que a programação seja separada da interface e cada tarefa possa ser realizada por uma pessoa diferente (CHED, 2008).

A Figura 11 demonstra uma janela criada em XAML. Um clique no botão ou na caixa de seleção da janela, desencadeia um evento, que pode ser conectado a um método em C# ou Visual Basic, além disso as propriedades dos objetos podem ser acessadas nessas linguagens.

Figura 11 - Janela de interface com o usuário



Fonte: Produção do próprio autor.

O código no Quadro 8 foi o utilizado para gerar a janela da Figura 11. Como pode ser visto, um código com elementos básicos, como botões e texto pode ser escrito facilmente e já facilita o uso do programa se comparado com uma versão da aplicação em console.

Quadro 8 - Janela com botão, label e checkbox descrita em XAML

```
<Window x:Class="WpfApplication1.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="MainWindow" Height="350" Width="275">
    <Grid Width="253">
        <Button Content="Button" Height="23" HorizontalAlignment="Left"
Margin="90,207,0,0" Name="button1" VerticalAlignment="Top" Width="75" />
        <CheckBox Content="CheckBox" Height="16" HorizontalAlignment="Left"
Margin="90,130,0,0" Name="checkBox1" VerticalAlignment="Top" Width="75" />
        <Label Content="Label" Height="28" HorizontalAlignment="Left"
Margin="105,61,0,0" Name="label1" VerticalAlignment="Top" />
    </Grid>
</Window>
```

Fonte: Produção do próprio autor.

A Biblioteca .NET utilizada em um código C#, já possui uma classe chamada Chart que possui métodos para criar diversos tipos de gráficos e atualizá-los dentro do código. Um objeto dessa classe pode ser instanciado dentro de uma janela XAML, para criar um gráfico

em meio a elementos de controle como botões e caixas de seleção. Com os controles do XAML e uma classe de gráficos já é possível criar uma interface com o usuário para servir a um programa que controle os sensores.

3.1.4 Computação gráfica

Com uma sequência de quadros obtida pelos sensores inerciais é possível criar uma animação em que um modelo humano tridimensional executa os mesmos movimentos que a pessoa executou quando monitorada pelo sistema. Então uma representação visual dos movimentos pode ser criada para um melhor entendimento dos movimentos realizados pelos sensores. Ainda, esses dados podem ser utilizados para gerar animações de movimentos complexos com facilidade, que podem ser utilizadas em filmes e jogos.

Com um modelo tridimensional, um aplicativo de computador pode gerar imagens de qualquer ângulo do modelo, com diferentes formas de iluminação, o que é chamado de renderização. Se o modelo for feito para suportar animações, um programa de computador pode aplicar transformações em partes especificas dele para modificar todo o conjunto. Quando se trata de modelos humanos, todos os ângulos do corpo podem ser controlados de modo a obter o mesmo modelo com diferentes posturas.

Um objeto tridimensional é formado por um conjunto possivelmente muito grande de vetores de posição além de um conjunto que determina como esses pontos são conectados para formar triângulos. Esses triângulos vão refletir a luz simulada no computador e formar uma imagem bidimensional do modelo que pode ser vista de qualquer ângulo (renderização). O Quadro 10 mostra as principais informações armazenadas em um arquivo de um modelo tridimensional de um cubo vermelho. Os vértices são numerados em ordem sequencial, e o número dos vértices é usado para formar triângulos, em seguida as cores de cada triângulo são representadas em um sistema de cores como o RBG.

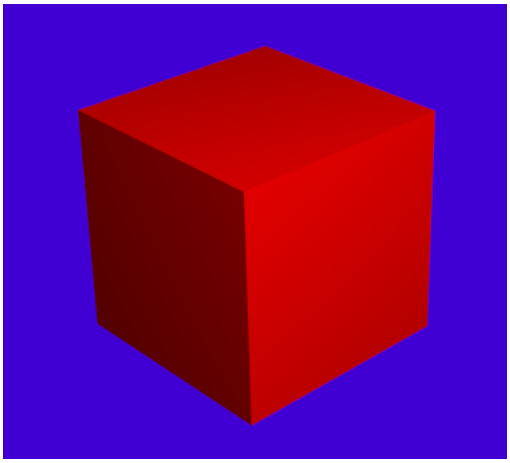
Quadro 9 - Conteúdo de um arquivo que descreve o modelo tridimensional de um cubo

Vértices = (1,1,1); (1,1,-1); (1,-1,1); (1,-1,-1); (-1,1,1); (-1,1,-1); (-1,-1,1); (-1,-1,-1) Triângulos = 0 1 3, 0 1 4, 0 3 4, 1 2 3, 1 2 5, 1 4 5, 2 3 7, 2 5 6, 2 6 7, 3 4 7, 5 6 7, 4 5 7 Cores = (255, 0, 0); (255, 0, 0); (255, 0, 0); (255, 0, 0); (255, 0, 0); (255, 0, 0); (255, 0, 0); (255, 0, 0); (255, 0, 0); (255, 0, 0); (255, 0, 0); (255, 0, 0);

Fonte: Produção do próprio autor.

Para que a renderização seja possível, a posição e orientação da câmera devem ser especificadas, assim como a posição e orientação de pelo menos uma fonte luminosa. A Figura 12 mostra a renderização do cubo descrito no quadro anterior. Como pode ser visto apesar de todos os triângulos do cubo terem a mesma cor, uma face ficou mais escura, pois a fonte luminosa se encontra a direta do cubo e assim a face esquerda recebe uma iluminação menor que as outras faces visíveis.

Figura 12 - Renderização do cubo descrito

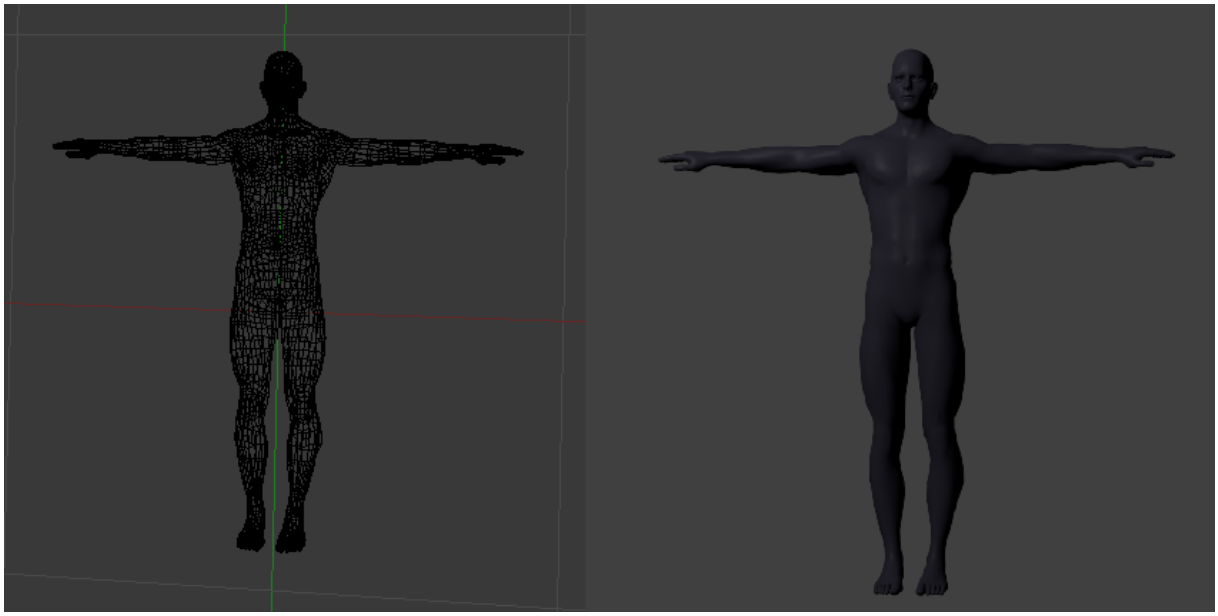


Fonte: Produção do próprio autor.

Independentemente da posição e da orientação do cubo, os triângulos formados serão os mesmos e as cores dos triângulos também. Logo, transformações matemáticas podem ser utilizadas no conjunto de vértices do cubo, de modo que ele pode ser movido, girado e escalado. Vários objetos podem ser reunidos em um arquivo, de modo que uma transformação pode ser aplicada separadamente em cada objeto.

Um modelo possui vários objetos e uma descrição da relação entre eles. A perna se junta a coxa, portanto uma rotação da coxa deve movimentar, também, a perna. Um arquivo definindo vários objetos e a relação entre eles pode ser feito por um modelador tridimensional de modo que um programador pode animar o modelo usando uma sequência de transformações específicas. A Figura 13 mostra um modelo feito no Blender, um software de modelagem gratuito. À esquerda, os triângulos que formam o modelo podem ser visualizados e, à direta, a renderização do modelo. Ele é feito de braços abertos para facilitar a animação dos braços, pois dessa forma os movimentos com o braço distorcem menos a aparência do modelo.

Figura 13 - Modelo humano feito no Blender



Fonte: Produção do próprio autor.

3.2 Formulação Matemática

Para descrever o movimento do corpo durante a caminhada, é necessário dispor de formas adequadas para representar as informações existentes neste processo. O corpo humano possui vários segmentos que podem executar movimentos como flexão, extensão, adução, abdução e rotação. Esses movimentos são representados no computador por meio de transformações, que denotam as mudanças de posição e ângulo dos segmentos entre dois períodos de tempo escolhidos.

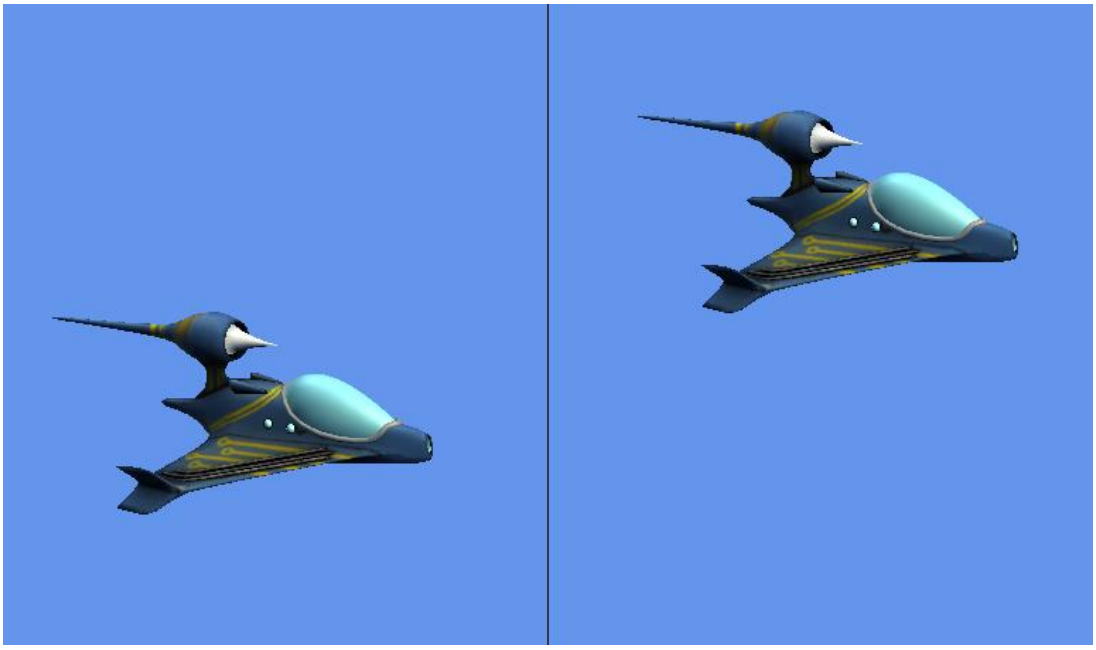
O mundo 3D é habitado por objetos chamados modelos, que são nada mais do que um conjunto de vértices unidos em forma de triângulos formando uma estrutura de rede. Um ponto dentro de um objeto, ou fora dele, é escolhido como referência para representar sua posição no espaço tridimensional, e as coordenadas desse ponto passam a denotar a posição do objeto. Uma direção também deve ser escolhida para representar a orientação do objeto no espaço.

As transformações lineares de translação, rotação e escala são utilizadas para representar a situação de um objeto no espaço.

3.2.1 Translação

A translação é adequadamente representada por um vetor, que denota o movimento do objeto nos eixos x, y e z. Os sensores inerciais medem apenas sua orientação no espaço, mas como os segmentos do corpo humano não se separam, a posição relativa do sensor pode ser obtida por cálculos indiretos, conhecendo sua posição inicial e as propriedades dos movimentos dos membros do corpo. A Figura 14 mostra um modelo tridimensional sendo transladado de uma posição para a outra. Isso foi feito por meio da soma de um vetor de translação ao vetor “modelPosition”, que representa a posição do objeto no espaço.

Figura 14 - Translação de um modelo tridimensional



Fonte: Produção do próprio autor.

3.2.2 Escala

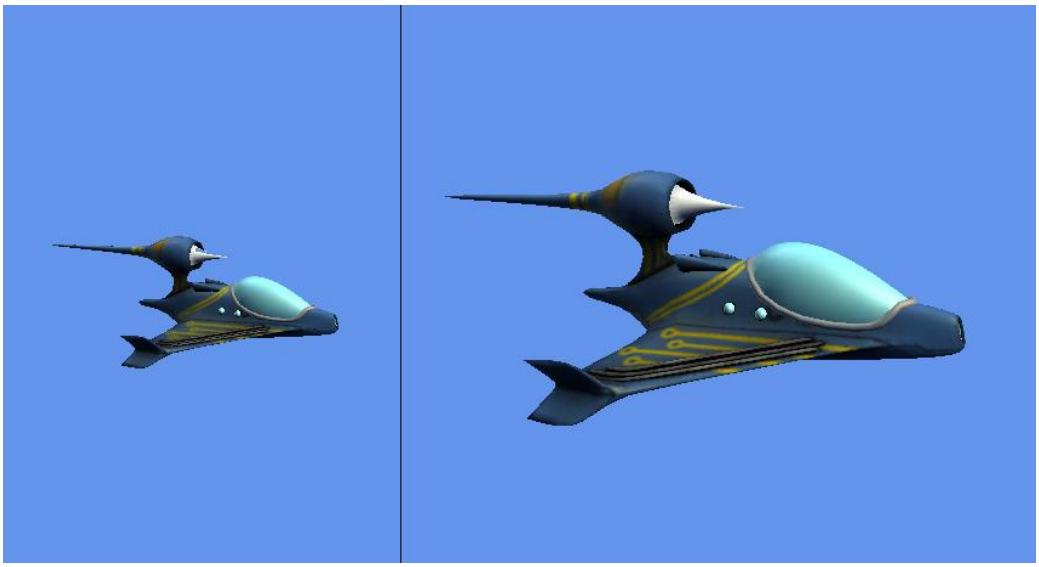
Duas pessoas com a mesma altura podem apresentar diferentes comprimentos da perna. Portanto, se um modelo tridimensional é usado para representar a caminhada de uma pessoa, pode ser interessante modificar algumas medidas do modelo para uma representação mais fiel. Ao invés de construir outro modelo do zero, aplicar algumas simples transformações de escalas em certas partes do modelo pode ser muito mais interessante.

Escalar um objeto por um fator constante, consiste em multiplicar cada vetor de posição pelo fator de escala. Isso pode ser feito multiplicando-se cada vértice do objeto por uma matriz de escala. A transformação de escala é mostrada na equação a seguir, onde uma matriz diagonal com os elementos a, b e c sendo os fatores de escala é multiplicada por um vetor de posição, gerando assim um vetor escalado. (KODICEK, 2005).

$$\begin{bmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & c \end{bmatrix} * \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} a * x \\ b * y \\ b * z \end{bmatrix} \tag{1}$$

A Figura 15 mostra uma matriz de escala aplicada em todos os pontos de um modelo. Como pode ser visto essa operação gera um aumento de todo o modelo. Nesse caso o mesmo fator de escala de 1.875 foi utilizado nos três eixos.

Figura 15 - Transformação de escala aplicada em um modelo tridimensional



Fonte: Produção do próprio autor.

3.2.3 Rotação

Das transformações citadas, a rotação é de longe a mais complexa, mas também a mais importante neste trabalho. Os sensores inercias medem apenas sua orientação no espaço, e é interessante que essas medidas sejam representadas de forma a facilitar o estudo dos movimentos monitorados por eles. A Figura 16 mostra um modelo tridimensional sofrendo quatro diferentes transformações de rotação em ângulos arbitrários nos três eixos. Esse efeito pode ser obtido de três diferentes formas que serão apresentadas a seguir, a rotação em torno de um eixo por um dado ângulo, a matriz de rotação e os quatérnios.

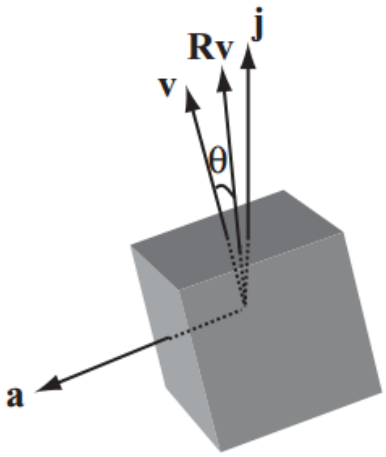
Figura 16 - Rotação de um modelo tridimensional.

Fonte: Produção do próprio autor.

3.2.3.1 Rotação por um ângulo em torno de um eixo

Uma forma muitas vezes mais conveniente de representar uma rotação é utilizando um vetor e um ângulo. Entender a orientação de um objeto no espaço por meio da informação contida em uma matriz de rotação ou em quatérnios pode ser uma tarefa bem complexa. Já a rotação em torno de um eixo é mais simples de entender quando o vetor e o ângulo são visualizados por uma pessoa tentado interpretar a orientação de um objeto no espaço (STAHLER, 2004). Por exemplo, para representar a rotação da perna em torno do joelho, pode ser escolhido um eixo perpendicular à perna e à coxa, e o ângulo de rotação teria uma relação direta com o ângulo do joelho, assim independentemente da posição da coxa, a informação de ângulo seria fácil de entender. A Figura 17 representa um cubo sofrendo uma rotação em torno do eixo a, sendo v a orientação anterior e Rv a orientação posterior à rotação do cubo e j é um vetor de referência apontando para cima.

Figura 17 - Rotação de um cubo em volta de um eixo



Fonte: (KODICEK, 2005).

3.2.3.2 Matriz de rotação

A rotação de um objeto no espaço tridimensional pode ser obtida ao girar todos os vértices que o compõe em torno de um ponto de referência. A direção de um vetor no espaço pode ser descrita por apenas dois ângulos, como é feito na representação de um vetor em coordenadas polares. Isso ocorre porque um vetor continua o mesmo ao girar em torno do próprio eixo, já os objetos podem girar em volta do próprio eixo, portanto são necessários três ângulos para representar a orientação de um objeto no espaço. Pelo mesmo fato é preciso utilizar no mínimo dois vetores para descrever a orientação de um objeto no espaço tridimensional, sendo comum a utilização de três vetores perpendiculares entre si para armazenar essa informação. O terceiro vetor pode ser obtido pelo produto vetorial dos dois primeiros.

A matriz de rotação é simplesmente uma matriz com três vetores escolhidos como referência da orientação do objeto, de modo que esses vetores são colocados nas colunas da matriz. Para formar uma matriz de rotação válida, os três vetores devem ser escolhidos de modo que sejam perpendiculares dois a dois e tenham módulo unitário (matriz ortonormal). Assim, uma matriz de rotação 3x3 é suficiente para representar qualquer possível orientação de um objeto no espaço tridimensional. Multiplicada pelo vetor de coordenadas de um ponto uma matriz de rotação é capaz de girar o ponto em volta da origem, mas em alguns casos pode ser necessário a rotação em torno de outro ponto que não seja a origem. Para isso pode-se realizar uma

translação de modo que o ponto desejado coincida com a origem, a rotação desejada e depois outra translação para colocar o ponto de referência no local desejado (LENGYEL, 2004).

Observando os valores da matriz sendo atualizados em tempo real com os movimentos do objeto é difícil perceber o que está acontecendo com o mesmo. Existe uma forma muito mais amigável para as pessoas que é a representação pelos ângulos de Euler. Com alguns cálculos os ângulos de Euler da orientação de um objeto podem ser obtidos pelos valores dos elementos da matriz de rotação. A partir dos ângulos de Euler também é possível obter a respectiva matriz de rotação.

Em duas dimensões a rotação de um ponto em torno da origem pode ser definida apenas com um ângulo θ e a matriz que gera essa rotação pode ser deduzida com alguns teoremas da trigonometria. A fórmula para obtenção da matriz 2x2 que gera uma rotação de ângulo θ no sentido anti-horário é descrita pela equação abaixo.

$$M(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \tag{2}$$

A matriz de rotação 2x2 pode ser estendida para o espaço tridimensional. Esta extensão está caracterizada nas equações mostradas a seguir, sendo a primeira uma rotação em torno do eixo x no sentido anti-horário (plano yz).

$$A(\alpha) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{bmatrix} \tag{3}$$

Rotação em torno do eixo y no sentido anti-horário (plano xz):

$$B(\beta) = \begin{bmatrix} \cos\beta & 0 & -\sin\beta \\ 0 & 1 & 0 \\ \sin\beta & 0 & \cos\beta \end{bmatrix} \tag{4}$$

Rotação em torno do eixo z no sentido anti-horário (plano xy):

$$C(\gamma) = \begin{bmatrix} \cos\gamma & -\sin\gamma & 0 \\ \sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{5}$$

As matrizes de rotação citadas anteriormente podem ser multiplicadas para gerar uma matriz contendo as rotações combinadas em torno dos três eixos. A matriz resultante é descrita a seguir e mostra como calcular a matriz de rotação por meio dos ângulos de Euler. Como a multiplicação de matrizes não é comutativa, uma ordem de multiplicação deve ser considerada. Neste trabalho, a ordem considerada é $C(\gamma) * B(\beta) * A(\alpha)$, ou seja, a rotação em torno do eixo z ocorre primeiro, depois do eixo y e posteriormente do eixo x.

$$\begin{bmatrix} \cos\beta\cos\gamma & -\sin\alpha\sin\beta\cos\gamma - \cos\alpha\sin\gamma & -\cos\alpha\sin\beta\cos\gamma + \sin\alpha\sin\gamma \\ \cos\beta\sin\gamma & -\sin\alpha\sin\beta\sin\gamma + \cos\alpha\cos\gamma & -\cos\alpha\sin\beta\sin\gamma - \sin\alpha\cos\gamma \\ \sin\beta & \sin\alpha\cos\beta & \cos\alpha\cos\beta \end{bmatrix} \tag{6}$$

Observando a matriz anterior também é possível concluir que os ângulos de Euler podem ser calculados dispondo-se dos coeficientes da matriz de rotação tridimensional. As três equações a seguir mostram como obter os ângulos de Euler por meio dos coeficientes da matriz. Os ângulos α e γ variam de -180º a 180º e o ângulo β de -90º a 90º.

$$\begin{aligned} \alpha &= \operatorname{atan}(M_{32}, M_{33}) \\ \beta &= \operatorname{asin}(M_{31}) \\ \gamma &= \operatorname{atan}(M_{21}, M_{11}) \end{aligned} \tag{7}$$

Em um sistema que utiliza sensores inerciais, o usuário pode querer conhecer a orientação de algum sensor em um determinado momento, ou inserir uma orientação específica no sistema. Com as equações apresentadas anteriormente o sistema pode manter as informações de orientação por meio das matrizes de rotação, mas efetuar uma conversão para os ângulos de Euler quando precisar mostrar essas informações ao usuário. Se desejar inserir uma informação, o usuário também pode usar os ângulos de Euler e o sistema pode convertê-los para uma matriz de rotação internamente. Como pode ser notado, manipular informações sobre a orientação de objetos é uma tarefa complexa, portanto um bom sistema deve esconder a complexidade do usuário e mostrar as informações da forma mais fácil possível. Por esta razão, é ideal que o projetista do sistema conheça todas as formas de representação e manipulação da orientação de objetos.

3.3 Desenvolvimento do sistema

O desenvolvimento do software foi dividido em três partes: a interface gráfica com o usuário, o programa propriamente dito e um programa secundário com uma animação 3D. As próximas seções desse capítulo descrevem cada uma dessas partes.

3.3.1 Interface com o usuário

A interface gráfica com o usuário foi feita com a ferramenta gratuita Visual C# 2010 Express da Microsoft, que permite o desenvolvimento em XAML. O objetivo da interface é tornar mais fácil o processo de monitoramento dos movimentos feito pelo usuário e mostrar as informações adquiridas instantaneamente após a captura e também posteriormente com arquivos gravados representando os movimentos.

O programa possui as seguintes funcionalidades:

- Procurar a porta USB, conectar, desconectar e reiniciar o Hub
- Iniciar e parar a captura dos quadros de orientação
- Salvar e abrir um arquivo com a sequência de quadros capturada
- Mostrar as informações em gráficos
- Mudar as configurações de captura

O sistema só pode ser utilizado para capturar os dados dos sensores se o Hub estiver conectado, e este só pode ser conectado se a porta USB que ele está for conhecida. Inicialmente a interface foi desenvolvida com uma *ComboBox* que permite ao usuário selecionar a porta USB do Hub e um botão com a função de conectar o Hub. Como não serão utilizados dois Hubs no mesmo computador e a função de procurar a porta USB possui a única utilidade de permitir que o Hub seja conectado, as funções de procura e conexão do sistema podem ser realizadas com um único botão. Isso agiliza a utilização do sistema e deixa o usuário com uma impressão melhor.

Para as funções conectar e desconectar existem duas opções, podem ser criados dois botões, ou apenas um botão que mostra conectar quando o Hub está desconectado e desconectar quando ele está conectado. A vantagem da primeira é que o botão pode ser configurado para ser desativado quando a função não estiver disponível, assim se o computador estiver lento o

usuário pode clicar várias vezes no botão conectar que apenas a primeira será executada. Já na segunda se o botão for clicado duas vezes, o programa vai conectar e depois desconectar o Hub. Um atraso de resposta foi colocado no botão para simular uma lentidão no computador e foi percebido que a maioria dos usuários tende a clicar várias vezes no botão se a resposta não for rápida o suficiente. Por esta razão, foi escolhida a opção de dois botões separados, um para conectar e um para desconectar. A mesma questão pode ser levantada para os botões de iniciar e parar a captura e também foi escolhida essa opção.

A configuração e exibição dos gráficos fica sob responsabilidade do programa por trás da interface, de modo que a interface em si é bem simples.

3.3.2 Aplicativo para aquisição das medições

O aplicativo principal deste trabalho tem por finalidade utilizar as classes disponíveis para obter os dados dos sensores e interagir com a interface gráfica e com o programa que mostra a animação do modelo tridimensional. Como o SDK da Technaid já disponibiliza várias classes, foram necessárias apenas mais duas. A primeira se chama ArquivoDeMarcha e possui métodos para abrir e criar arquivos com as informações adquiridas. A segunda foi nomeada GraficosDeMarcha e serve para manipular os gráficos, pois isso exige um código relativamente grande esta classe facilita o desenvolvimento do software. O Quadro 10 apresenta os métodos feitos no programa principal. Vale ressaltar que o sistema é multitarefas, de modo que cada método pode executar em paralelo e uma sinalização foi feita para que as tarefas estejam em sincronismo.

Quadro 10 - Métodos do programa principal

Métodos	Descrição
MainWindow	Inicia o programa e chama os construtores das classes utilizadas
Clique_botao_conectar	Conecta o Hub. É executada quando o usuário clica no botão conectar
Clique_botao_desconectar	Desconecta o Hub. É executada quando o usuário clica no botão desconectar
Clique_botao_iniciar	Inicia a captura. É executada quando o botão iniciar é clicado
Clique_botao_parar	Para a captura. É executada quando o botão parar é clicado
Clique_botao_reiniciar	Reinicia o Hub. É executada quando o botão reiniciar é clicado
Clique_menu	Executa operações do menu, como abrir, salvar arquivos e configurações
QuadroRecebido	Retira as informações dos sensores quando uma medição é realizada

AtualizaGrafico	Atualiza o gráfico para mostrar os novos ângulos medidos
AcessaDados	Permite que outros programas acessem os dados deste programa
IniciaAnimacao	Inicia o programa da animação dando a ele o ponteiro da função anterior

Fonte: Produção do próprio autor.

3.3.3 Modelo tridimensional

Um programa separado foi feito para mostrar o modelo tridimensional executando os movimentos monitorados pelos sensores. Isso permite que o projeto do modelo tridimensional seja modificado sem ter que alterar o projeto do sistema, podendo existir várias versões dos dois softwares que sejam utilizadas em qualquer combinação. O programa do sistema deve iniciar o programa do modelo de acordo com o comando do usuário.

A interface entre os dois programas é feita com o uso de *delegates*, um tipo do C# que funciona como os ponteiros para função existentes em C. O programa principal passa um dos seus métodos para o programa do modelo que este pode chamar para obter os dados do primeiro. Assim, se os sensores estiverem capturando os dados a uma velocidade maior que a velocidade de atualização do modelo, o programa do modelo pode retirar os quadros em uma velocidade menor, de acordo com a capacidade do computador.

O programa foi feito utilizando o XNA, que é um framework da Microsoft que permite a criação de jogos e animações em 3D e possui uma versão gratuita. É considerado uma camada de abstração de hardware (CARTER, 2008) que permite ao desenvolvedor não se preocupar com qual tipo de hardware de vídeo será utilizado. Sem essa camada o programador teria que conhecer profundamente a placa de vídeo que seria utilizada, pois sem hardware específico para vídeo a renderização se torna muito lenta.

Qualquer modelo humano feito com extensão .fbx pode ser utilizado, desde que suporte animações. O aplicativo de código aberto Makehuman cria modelos automaticamente em vários formatos, podendo ser especificadas várias características do modelo. Programas como esse podem ser utilizados para gerar modelos tridimensional humanos para o sistema. O modelo utilizado é apresentado na Figura 18 e é um dos exemplos disponibilizado pela Microsoft para XNA, utilizado para jogos.

Figura 18 – Modelo humano utilizado no trabalho

Fonte: Produção do próprio autor.

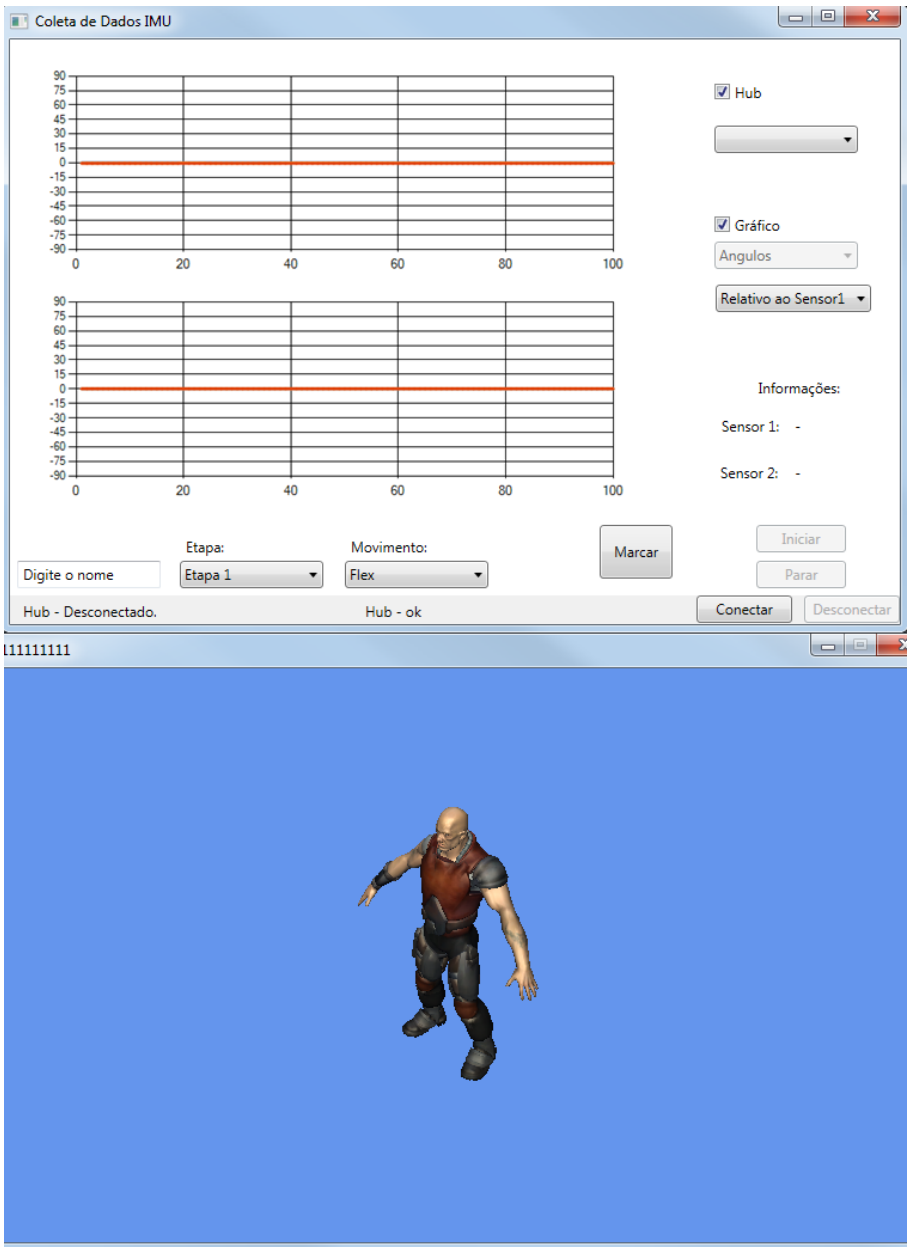
O XNA providencia uma estrutura de loop com métodos que podem ser reescritos para que a animação desejada seja produzida na tela. Essa sequência consiste nos seguintes métodos:

- Initialize (inicializar)
- LoadContent (carregar conteúdo)
- Update (atualizar)
- Draw (desenhar)
- UnloadContent (descarregar conteúdo)

Os métodos *Initialize* e *LoadContent* são executados uma única vez no início do programa, sendo que o primeiro serve para carregar os conteúdos normais e o segundo os conteúdos gráficos do programa (os objetos e posições iniciais). Os métodos *Update* e *Draw* são executados em loop, um depois o outro, até o término do programa. O primeiro serve para atualizar o conteúdo da animação, ele vai manipular as transformações feitas nos modelos e na câmera de acordo com a lógica da animação. Já o método *Draw* vai renderizar o cenário atual com as configurações especificadas, como cor e iluminação. Quando a animação terminar ou o usuário fechar o programa, o método *UnloadContent* é chamado para liberar os recursos consumidos pelo programa durante a execução.

Assim, no sistema aqui considerado, o programa principal dos sensores vai inicializar a animação e executá-la. O método Update vai ler o mouse para atualizar a posição da câmera, chamar a função passada por um *delegate* para obter os dados dos sensores e atualizar as matrizes de orientação dos modelos. Em seguida o método Draw vai renderizar e mostrar a animação na tela. Por fim, o programa termina e *UnloadContent* é chamado. Vale ressaltar que tudo isso ocorre em paralelo à aquisição dos dados dos sensores, pois os dois programas são executados em processos separados. O sistema final é apresentado na Figura 19.

Figura 19 - Sistema final com a interface gráfica e o modelo tridimensional



Fonte: Produção do próprio autor.

4 VALIDAÇÃO EXPERIMENTAL

4.1 Protocolo experimental

A divisão dos programas em módulos permitiu que os testes fossem feitos em um de cada vez. Já que alguns módulos não precisam dos sensores para serem testados, isso permite a economia de tempo.

4.1.1 Testes dos gráficos

Como o código do software pode conter erros o primeiro teste a ser realizado é verificar se recebendo as matrizes de orientação o programa imprime os ângulos corretamente nos gráficos. Um método que utiliza a equação (6) para gerar matrizes de rotação com ângulos específicos foi criado. Esse experimento consiste em variar os ângulos com diversas formas de onda como senoidal, quadrada e triangular e assim calcular as respectivas sequências de matrizes de rotação enviando-as para o módulo dos gráficos e verificando se os ângulos são mostrados corretamente. Assim, os movimentos detectados pelos sensores podem ser simulados com intuito de testar o software.

Esse experimento é fundamental pois verifica se as equações utilizadas pelo software estão corretas. Em três dimensões é fácil errar algum parâmetro de modo que algum ângulo pode ser mostrado ao contrário ou um ângulo pode ser mostrado no lugar do outro.

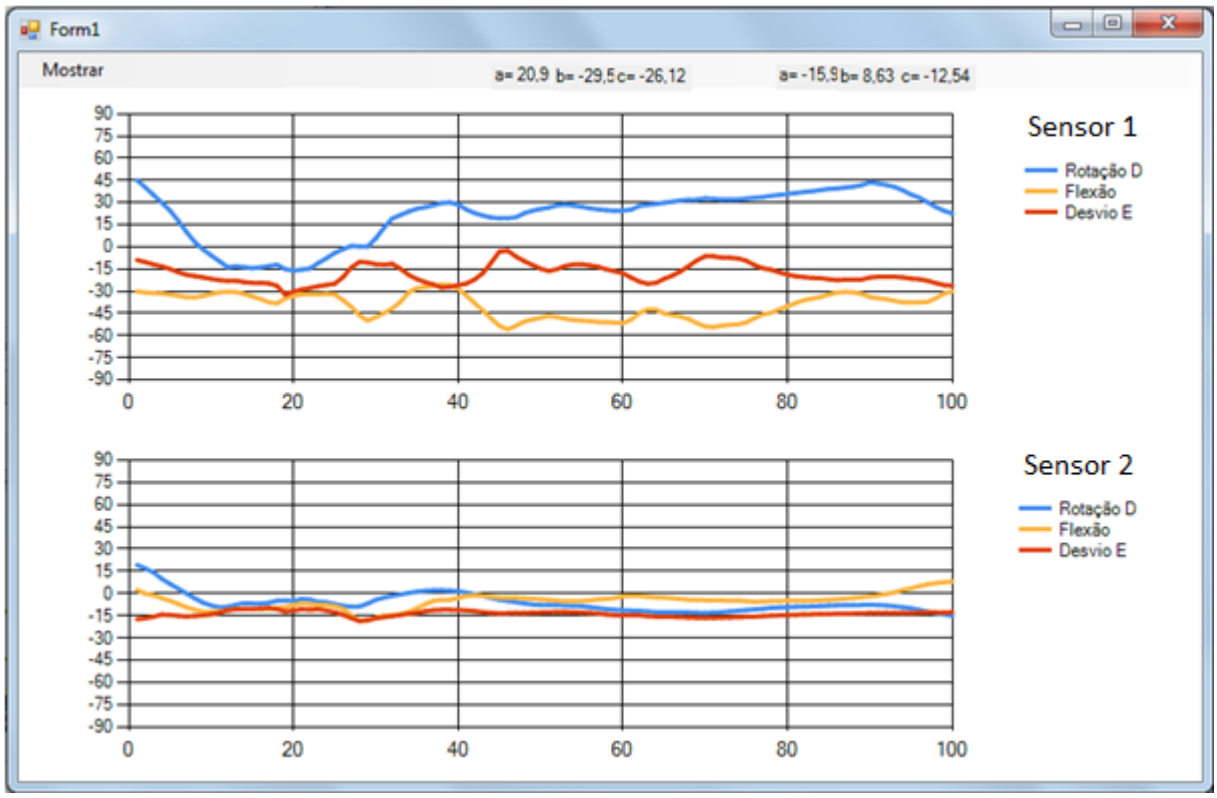
4.1.2 Teste dos sensores parados

Este teste consiste em utilizar um transferidor e uma régua para medir os três ângulos do sensor em uma determinada posição e verificar se as informações corretas são disponibilizadas na tela. Deste modo é possível testar a precisão do sensor quando este está parado.

Colando um lado do sensor em uma régua é possível medir facilmente o ângulo do sensor relativo a esse lado verificando o ângulo que a régua se encontra no transferidor. O ângulo formado do sensor com cada um dos três eixos do espaço tridimensional pode ser obtido desta forma com uma precisão de 1º. Os valores mostrados pelos gráficos podem ser comparados

com as medições para completar o teste. A Figura 20 mostra os gráficos gerados por dois sensores sendo movimentados.

Figura 20 - Gráficos mostrando os ângulos de Euler de dois sensores



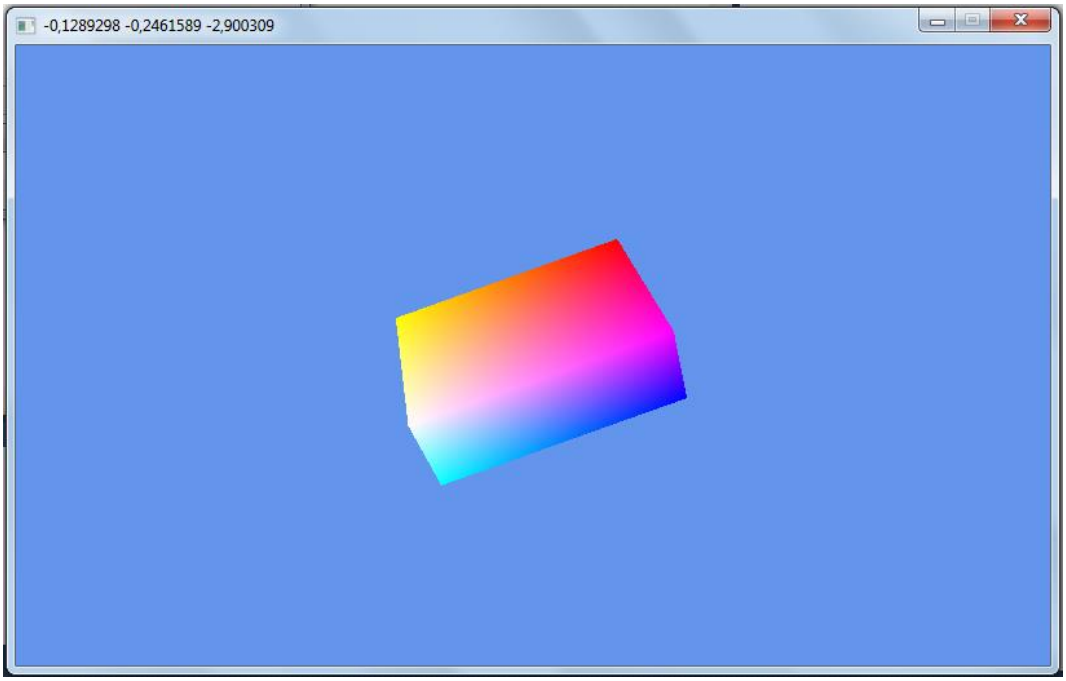
Fonte: Produção do próprio autor.

4.1.3 Teste dos sensores em movimento

Esse teste tem a finalidade de verificar a precisão dos sensores para obter ângulos durante o movimento. Devido à falta de outro sistema de medição dos ângulos do movimento humano, esse teste avaliou apenas qualitativamente a precisão do sensor. Uma câmera de vídeo foi utilizada para filmar movimentos com o sensor e verificar se os ângulos mostrados acompanham o movimento corretamente.

Alternativamente um programa foi desenvolvido para mostrar um paralelepípedo com as mesmas dimensões do sensor executando os movimentos do mesmo. Realizando o movimento com os sensores e verificando os movimentos do paralelepípedo tem-se uma ideia da precisão do sensor. A Figura 21 mostra o modelo seguindo o ângulo do sensor. As diversas cores são para facilitar a percepção do ângulo do paralelepípedo.

Figura 21 - Paralelepípedo que se movimenta junto com o sensor



Fonte: Produção do próprio autor.

4.1.4 Teste do sistema

Como o software é multitarefas, vários métodos podem ser executados em paralelo o que pode gerar bugs. Esse experimento consistiu em executar o programa vinte vezes com opções diferentes e vários movimentos sendo executados e verificar se algum erro ocorre.

4.2 Resultados

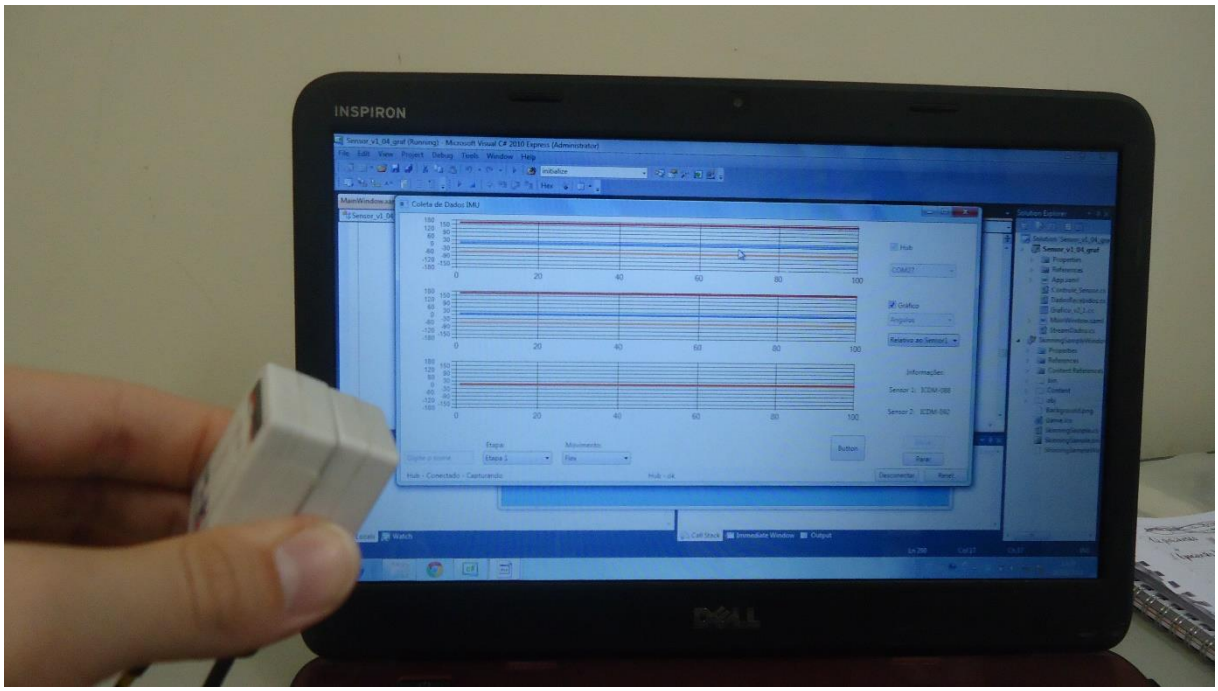
4.2.1 Testes dos gráficos

As curvas dos gráficos foram mostradas corretamente para todas as formas de onda citadas e os ângulos não estão trocados nem invertidos.

4.2.2 Teste dos sensores parados

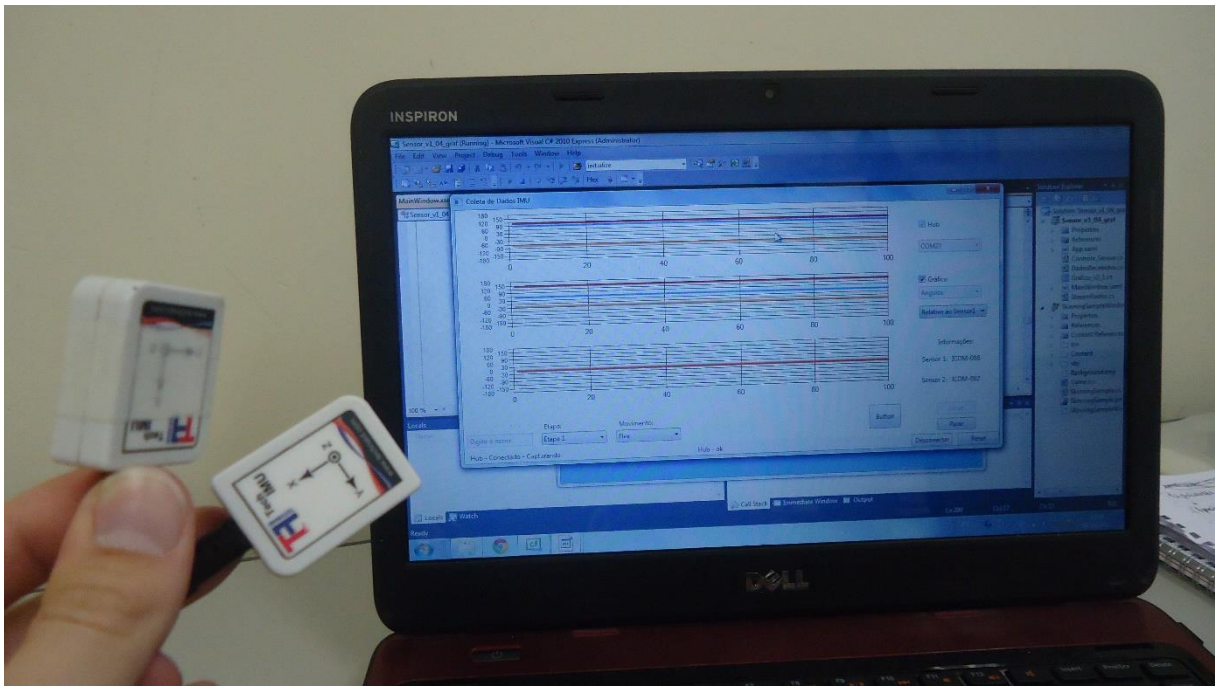
Os sensores medem os ângulos com os três eixos corretamente. Todas as medidas efetuadas apresentaram os ângulos corretos. O erro dos sensores na medição de ângulos quando não estão em movimentos é menor que 1º. As Figuras 22 e 23 mostram dois sensores medindo o mesmo ângulo e dois sensores medindo ângulos diferentes, respectivamente.

Figura 22 - Sensores medindo a mesma orientação



Fonte: Produção do próprio autor.

Figura 23 - Sensores medindo orientações diferentes



Fonte: Produção do próprio autor.

4.2.3 Teste dos sensores em movimento

Depois de uma sequência de movimentos bruscos realizados com os sensores, foi percebido que estes passam a mostrar ângulos incorretos por um tempo curto e rapidamente corrigem os ângulos. Com velocidades próximas as dos movimentos humanos naturais os sensores acompanharam rapidamente os ângulos. Isso ocorre devido aos ajustes realizados nos filtros de Kalman dos sensores feitos pelo fabricante. O paralelepípedo segue os movimentos feitos com o sensor precisamente e rapidamente.

4.2.4 Teste do sistema

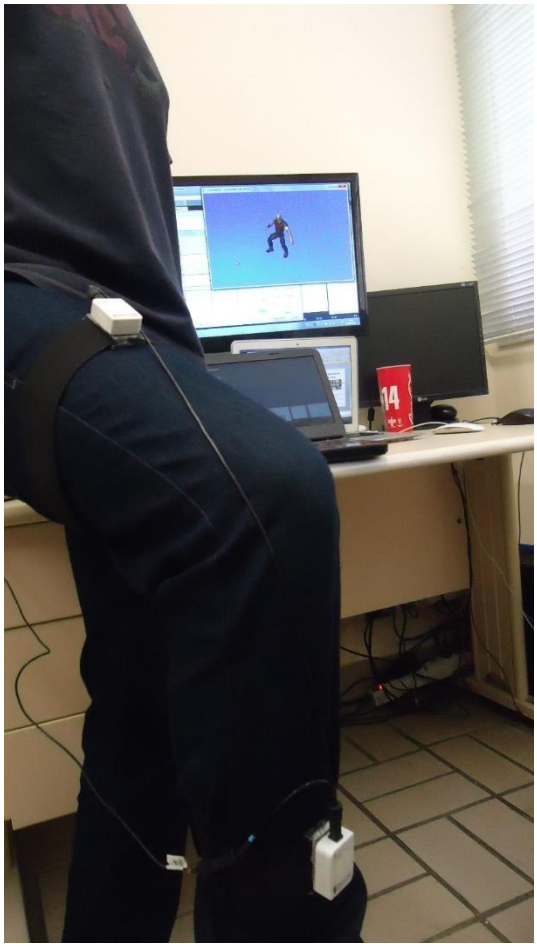
Alguns erros foram detectados por este teste. Os erros detectados foram do tipo em que o Windows diz “Este programa não está respondendo” e é necessário fechar o programa. O primeiro erro foi uma inconsistência de tarefas onde o programa tentava escrever no arquivo um quadro ainda não capturado. Depois foi detectado um erro em que se ocorresse uma falha na conexão e o usuário tentasse iniciar a captura, o programa falhava. Posteriormente foi identificado um erro em que algumas sequências de quadros eram escritas duas vezes no arquivo. Os erros ocorreram devido a exceções que não foram tratadas no programa. O código foi refeito e testado novamente de modo que a quarta versão do programa não apresentou erros. A Figura 24 mostra o sistema sendo testado com dois sensores.

Figura 24 - Sistema sendo testado com dois sensores

Fonte: Produção do próprio autor.

As Figuras 25 e 26 mostram a perna direita do modelo seguindo duas posições.

Figura 25 - Teste com a perna dobrada



Fonte: Produção do próprio autor.

Figura 26 - Teste com a perna esticada

Fonte: Produção do próprio autor.

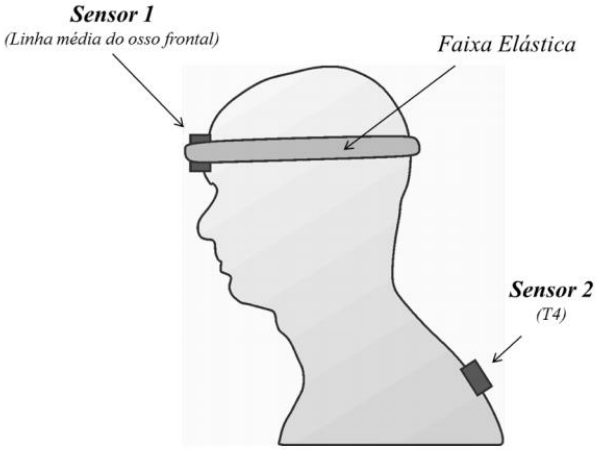
5 CONCLUSÕES E TRABALHOS FUTUROS

O sistema final foi desenvolvido e é capaz de obter os dados dos sensores inerciais, mostrar as informações durante a captura e armazená-las para análise posterior, como proposto no objetivo geral. A interface gráfica permite que usuários leigos em sistemas computacionais utilizem o sistema com apenas uma pequena explicação sobre as funcionalidades. Os gráficos e o modelo tridimensional mostram os movimentos sendo executados durante a captura. Isso permite identificar erros de calibração ou posicionamento dos sensores antes do término dos experimentos com pessoas.

O sistema com sensores inerciais possui a vantagem sobre a fotogrametria de não precisar de um espaço estruturado para seu funcionamento. Os sensores são pequenos e leves e podem ser levados facilmente para qualquer local. A vantagem sobre os eletrogoniômetros é que esses sistemas muitas vezes necessitam de fios conectados a um dispositivo fixo enquanto o Hub dos sensores inerciais pode ser ligado ao computador via *bluetooth*.

O sistema pode ser utilizado para análise dos ângulos das articulações durante o movimento humano. Uma avaliação da amplitude de movimento da coluna cervical foi realizada utilizando o mesmo sistema da Technaid e uma versão anterior deste software (BERMUDES, et al., 2013). Os sensores foram posicionados como mostrado na Figura 27 e vários movimentos foram analisados de diversas pessoas com dor cervical crônica.

Figura 27 - Sensores posicionados no paciente



Fonte: (BERMUDES et al., 2013).

Três artigos científicos foram produzidos utilizando o sistema deste trabalho (BERMUDES, et al., 2013) (ELIAS et al., 2013) (ELIAS et al., 2014). Todos eles consistiram em uma análise dos movimentos do pescoço de vários pacientes com dores crônicas no pescoço após um procedimento manipulativo.

No futuro, outros experimentos relacionados a atividade motora humana podem ser realizados. O sistema pode ser utilizado para a criação de animações tridimensionais. Podem ser feitos aprimoramentos para que o sistema seja utilizado em consultórios de fisioterapia como auxílio para o diagnóstico de certas doenças.

Com o avanço da tecnologia de fabricação dos sensores inerciais, são desenvolvidos dispositivos cada vez menores, mais baratos e mais precisos. Os sensores atuais ainda são grandes para diversos tipos de aplicação. Com sua miniaturização várias possibilidades de sistemas passam a ser possíveis, como por exemplo, o controle do computador com sensores colocados na mão ao invés de um mouse.

Também com o avanço da tecnologia novos tipos de sensores são desenvolvidos e os existentes aprimorados. Vale salientar que os sensores de fibra ótica, uma tecnologia ainda mais atual que os sensores inerciais, são capazes de medir a curvatura de uma fibra ótica por meio de um dispositivo chamado interrogador. A evolução destes sensores pode tornar possível o desenvolvimento de sistemas de análise do movimento baseado em sensores de fibra ótica talvez ainda melhores que os sistemas atuais.

6 REFERÊNCIAS BIBLIOGRÁFICAS

ASSIS, A. K. T. **Wilhelm Eduard Weber (1804-1891), sua vida e obra**. Revista da SBHC, Campinas, SP, 1991. v. 5, p. 54.

BARSHAN, B.; DURRANT-WHYTE, H. F. **Inertial navigation systems for mobile robots**. IEEE Transactions on Robotics and Automation, New York, jun. 1995. Vol. 11, n. 3.

BERMUDES, EMERSON; ELIAS, ARLINDO; MELIM, ROBERTO; RIBEIRO, DANIEL; MUCCIACCIA, SÉRGIO; FRIZERA, ANSELMO; BASTOS, TEODIANO. **Avaliação da amplitude de movimento da coluna cervical utilizando sensores inerciais: estudo piloto.** Revista Brasileira de reabilitação e atividade física, Vitória, 2013. v. 2, n. 1, p. 45-51.

CARTER, CHED. **Microsoft XNA Unleashed.** Indiana, Pearson, 2008. cap. 1.

COLLINS, AMBER; BLACKBURN, TROY; OLCOTT, CHRIS; DIRSCHL, DOUGLAS; WEINHOLD, PAUL. **The effects of stochastic resonance electrical stimulation and neoprene sleeve on knee proprioception**. Journal of Orthopaedic Surgery and Research, Chapel Hill, 2009

ELIAS, ARLINDO; BERMUDES, EMERSON; MELIM, ROBERTO; MUCCIACCIA, SÉRGIO; FRIZERA, ANSELMO; BASTOS, TEODIANO. **Avaliação da amplitude de movimento de rotação da coluna cervical com sensores inerciais após procedimento manipulativo.** In: 4o. Encontro Nacional de Engenharia Biomecânica, Vitória. Anais do 4o. Encontro Nacional de Engenharia Biomecânica, 2013. v. 1, p. 39-40.

ELIAS, ARLINDO; BERMUDES, EMERSON; MELIM, ROBERTO; MUCCIACCIA, SÉRGIO; FRIZERA, ANSELMO; BASTOS, TEODIANO. **Assessment of cervical spine range of rotation with inertial sensors after a vertebral manipulation procedure.** Journal of Medical Imaging and Health Informatics, 2014. v. 4, p. 131-136.

JIA, GUIMIN; WANG, XIANGJUN; WANG, HONG BIN; ZHANG, ZHAOCAI. **Accuracy Analysis of Space Three-line-array Photogrammetry Based on Forward Intersection**.

Tianjin, China, International Conference on Advanced Computer Theory and Engineering, 3, 2010.

KODICEK, DANNY. **Mathematics and Physics for Programmers**. Massachusetts, Charles River Media, 2005. cap. 18.

LENGYEL, ERIC. **Mathematics for 3D Game Programming and Computer Graphics**. 2. ed. Massachusetts, Charles River Media, 2004. cap. 3.

LORUSSI, FEDERICO; GALATOLO, STEFANO; BARTALESI, RAPHAEL; ROSSI, DANILO. **Modeling and Characterization of Extensible Wearable Textile-Based Electrogoniometers.** IEEE SENSORS JOURNAL, 2013. vol. 13, no. 1.

LORUSSI, FEDERICO; GALATOLO, STEFANO; ROSSI, DANILO. **Textile-Based Electrogoniometers for Wearable Posture and Gesture Capture Systems.** IEEE SENSORS JOURNAL, 2009. vol. 9, no. 9.

LUINGE, H. J.; VELTINK, P. H. **Measuring orientation of human body segments using miniature gyroscopes and accelerometers**. Medical & Biological engineering & computing, Enschede, 2005. Vol. 43.

MAENAKA, K. **MEMS Inertial Sensors and Their Applications**. ln: Networked sensing systems international conference, 5., Kanazawa, 2008. p. 1-3.

MELIM, ROBERTO. **Desenvolvimento de Sistema de Análise da Marcha Humana Utilizando Sensores Inerciais e Câmera de Profundidade**. Dissertação de mestrado, Programa de pós graduação em Engenharia Elétrica da Universidade Federal do Espírito Santo, Vitória, 2013.

MUYBRIDGE, EADWEARD. **Descriptive zoopraxography, or the science of animal locomotion made popular**. Library of the American museum of natural history, Philadelphia, 1893.

NAGEL, CHRISTIAN; EVJEN, BILL; GLYNN, JAY; WATSON, KARLI; SKINNER MORGAN. **Professional C# 4 and .NET 4**. Indiana, Wiley Publishing, 2010. cap. 29.

SACCO, I. C. N.; ALIBERT, S.; QUEIROZ, B. W. C.; PRIPAS, D.; KIELING, I.; KIMURA, A. A.; SELLMER, A. E.; MALVESTIO, R. A.; SERA, MT. **Confiabilidade da fotogrametria em relação a goniometria para avaliação postural de membros inferiores**. São Carlos, Revista Brasileira de Fisioterapia, 2007. vol. 11, no. 5, p. 411-417.

STAHLER, WENDY. **Beginning Math and Physics for Game Programmers**. Florida, Pearson Education, 2004. cap. 6.

TECHNAID S. L. **Documentación SDK MCS Studio.** Madrid, 2008.

WHITTLE, D. A. **Gait analysis an Introduction**. 4. ed. Waltham:Elsevier, 2007.

WINTER, DAVID. **Biomechanics and motor control of human movement**. 4. ed. Ontario, John Wiley & Sons Inc, 2009.