3. MATERIAIS E MÉTODOS

3.1. MODELAGEM 3D DA BASE MECÂNICA DA PLATAFORMA

A proposta para concepção deste equipamento, parte da necessidade de se construir uma estrutura compacta que permita sua mobilidade com facilidade, ao mesmo tempo rígida suficiente para suportar o peso de uma pessoa sem sofrer deformações que venham a interferir na integridade das medidas coletadas.

Inicialmente foi feita a modelagem da plataforma em um software de Projeto Auxiliado por Computador (*Computer Aided Design* CAD) 3D, especificamente SolidWorks®, com objetivo de acelerar o projeto mecânico e permitir uma visualização prévia da forma física do equipamento. Além disso, o modelo em CAD permite estimar propriedades físicas e mecânicas como peso e volume, a imagem da Figura 27 foi obtida a partir de um modelo CAD do equipamento, o desenho técnico da plataforma encontra-se no Apêndice 1.

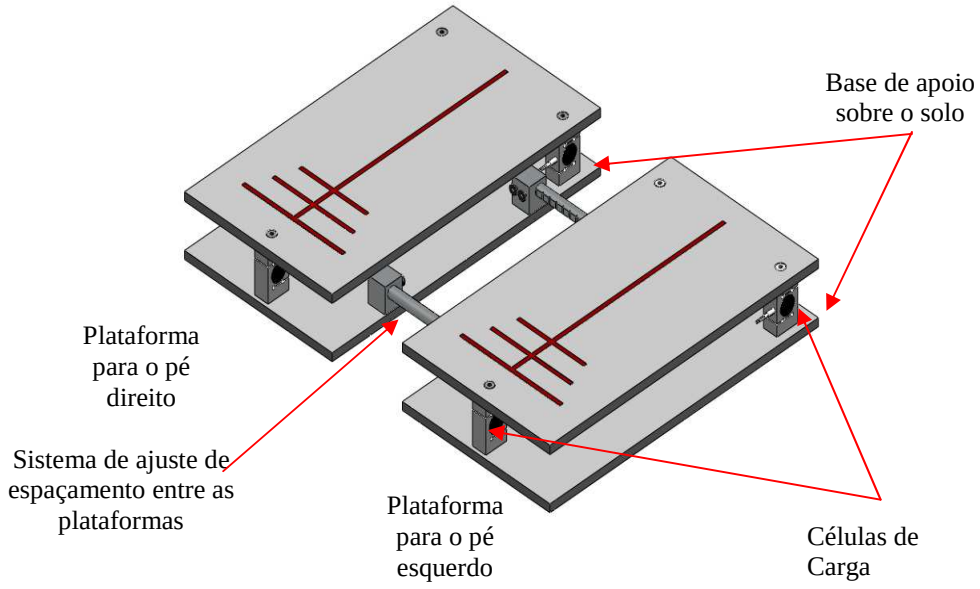


Figura 27: Desenho CAD 3D da plataforma de força.

A Base mecânica de sustentação do paciente é constituída por duas chapas retangulares que sustentam cada uma, um dos pés do paciente de forma independente, estas plataformas estão apoiadas cada uma sobre 3 células de carga, estas por sua vez estão fixas sobre outra chapa retangular apoiada sobre o solo Figura 27, diferente do arranjo comum de plataformas que contem 1 única superfície apoiada em 4 sensores, esta particularidade permite a este equipamento mensurar

as mesmas medias estabilométricas que o arranjo comum com 4 sensores, e vai além permitindo realizar a medida da diferença entre as forças resultantes aplicadas a cada pé de forma independente, tendo-se assim um Centro de Pressão (CP) e a intensidade da força aplicada sobre este para cada pé.

A força de cada pé do paciente é aplicada de forma distribuída sobre a plataforma de forma que esta força é aplica parcialmente a cada uma das 3 células de carga abaixo da plataforma, o somatório das forças aplicadas a cada célula é igual a força aplicada sobre a plataforma.

À medida que o corpo do paciente sobre a plataforma oscila durante o equilíbrio estático a intensidade da força aplicada a cada célula de carga é alterada, porém o somatório das forças permanece praticamente constante.

Dado um sistema de coordenadas onde a origem está localizada no baricentro do retângulo formado pela plataforma, o diagrama de corpo livre do sistema fica como mostrado na Figura 28, onde, *Fr* é a força resultante do peso aplicado por um dos pés do paciente sobre a plataforma, as forças *Fa*, *Fb* e *Fc* são uma parcela da força resultante aplicada a cada uma das células de carga que podem ser entendidas como a reação em cada um dos apoios da plataforma. As posições de *Fa, Fb* e *Fc* em relação a origem do sistema de coordenadas são conhecidas, mas a posição onde esta a força *Fr* é desconhecida e variante no tempo, mas sabendo a intensidade das forças *Fa*, *Fb* e *Fc* é possível determinar a posição de *Fr*.

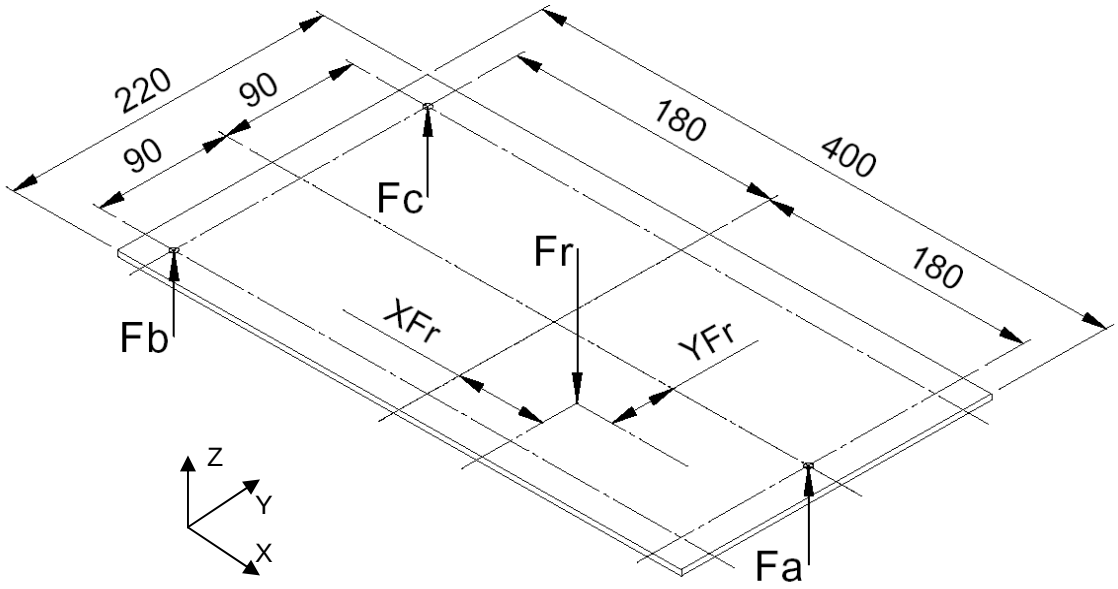


Figura 28: Diagrama de corpo livre da plataforma de apenas um dos pés do paciente.

Dado o diagrama de corpo livre da Figura 28 tem-se:

$$\sum F=0 \rightarrow Fa + Fb + Fc - Fr = 0 \rightarrow Fr = Fa + Fb + Fc \quad (33)$$

Momentos em torno do eixo Y tem-se*:*

$$\sum My=0 \rightarrow 180Fb + 180Fc + X_{Fr}Fr - 180Fa = 0$$

$$X_{Fr}Fr = 180Fa - 180Fb - 180Fc \quad (34)$$

Isolando X_{Fr} e substituindo 33 em 34 vem*:*

$$X_{Fr} = \frac{180(Fa - Fb - Fc)}{(Fa + Fb + Fc)} \quad (35)$$

A equação 35 mostra que obtendo-se os valores das forças *Fa*, *Fb* e *Fc,* encontramos a posição de (*Fr*) no eixo X ou antero-posterior do CP de um dos pés.

Agora aplicando os momentos em torno do eixo X temos:

$$\sum Mx=0 \rightarrow -90Fb + 90\,Fc + Y_{Fr}Fr = 0 \rightarrow Y_{Fr}Fr = 90Fb - 90Fc \quad (36)$$

Nesta situação a força *Fa* não promove momento no torno do eixo x. Isolando Y_{Fr} e substituindo 33 em 36 temos:

$$Y_{Fr} = \frac{90(Fb - Fc)}{(Fa + Fb + Fc)} \quad (37)$$

A equação 37 mostra que obtendo os valores das forças *Fa*, *Fb* e *Fc* é possível encontrar a posição de (*Fr*) no eixo Y ou médio-lateral do CP de um dos pés.

A mesma situação se aplica a ambos os pés, sendo assim, em posse da posição de *Fr* em coordenadas cartesianas (X_{Fr} ,Y_{Fr}) do CP do pé esquerdo (*Fre*) e direito (*Frd*) transladado-os para o sistema cartesiano do corpo e tomando a média de ambas é possível encontrar a localização do centro de pressão (CP) do corpo do paciente, assim-se para o eixo Y:

$$Y_{FrC} = \frac{(Y_{Fre} - 110) + (Y_{Frd} + 110)}{2} \quad (38)$$

E para o eixo X do CP do corpo tem-se:

$$X_{Frc} = \frac{X_{Frd} + X_{Fre}}{2} \quad (39)$$

Em posse das coordenadas do CP é possível gerar a impressão do gráfico de estabilograma que é a variação do CP do corpo nas direções X (antero-posterior) e

Y (médio-lateral) ou o estatocinesigrama, que é a variação de X e Y em função do tempo, como foi discutido nos itens 2.1.2 e 2.2.

3.2. FERRAMENTAS DE AUXILIO AO PROJETO MECÂNICO

A partir do modelo CAD pode-se realizar uma análise através do Método de Elementos Finitos (MEF) ou *Finite Element Analysis* (FEA). Esta técnica tem como objetivo auxiliar a visualização dos esforços atuando no sistema, além de permitir simular condições de cargas extremas. É também importante para visualizar em um ambiente de simulação como estão distribuídos os esforços na superfície da plataforma e também entender como acontecem as deformações das células de carga.

O MEF é uma técnica numérica que consiste em transformar elementos contínuos (peças mecânicas, estruturas metálicas, e afins) em objetos discretos, definido entre pontos (nós). Quanto maior o número de nós em um determinado elemento, mais detalhado será a sua analise. É necessário que o usuário do MEF saiba definir corretamente os nós necessários ao elemento, pois, caso contrário, poderá haver esforços subentendidos, tornando assim a análise equivocada. (ALVES FILHO, 2007)

A análise inicia com a inserção do desenho estrutural do equipamento em forma de linhas, assemelhando-se a um desenho CAD, em outras palavras, a definição de geometria. Nessa fase há a identificação do problema físico real com suas peculiaridades e detalhamento, ocorrendo assim a definição dos pontos importantes da análise. (ALVES FILHO, 2007)

A principio foi realizada uma simulação de uma primeira proposta para a configuração física da plataforma, para verificar o comportamento da superfície de suporte em relação a disposição das células de carga e a força aplicada sobre a mesma.

Utilizando o modelo CAD como referência este foi exportado para o Software FEA Algor® para realizar análise. Em um primeiro passo o software importa a geometria do arquivo CAD Figura 29 (A) e sobre este é gerada a malha a partir de alguns parâmetros como o tamanho médio de cada elemento da malha e se a mesma é uma malha de elementos de barras, placa ou sólidos. Para esta simulação foi adotada uma análise de elemento sólido com dimensão média de 4mm por

elemento que é a espessura da chapa superior, após o processamento destas informações o *software* informa a quantidade de elementos para este modelo Figura 29 (B), em seqüência é inserida as informações sobre propriedades tais quais: sistemas de unidades dos dados, material dos elementos e principalmente as condições de apoio e os carregamentos aplicados sobre o modelo Figura 29 (C).

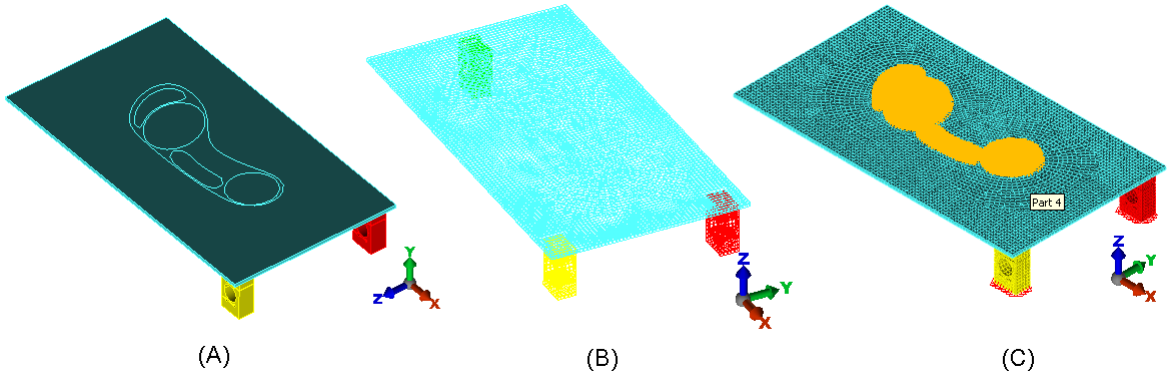


Figura 29: Etapas de geração de um modelo MEF (A) importação do CAD, (B) geração da malha e (C) inserção de parâmetros e carregamentos.

Para as análises aqui realizadas foi adotado o sistema internacional de unidades com milímetro como unidade de comprimento e Newton como unidade de força. As demais unidades não são relevantes e foram mantidas as mesmas que o *software* adota como padrão. O material aplicado foi o Alumínio 6061-T6 cujas propriedades físico-químicas já estão incluídas na biblioteca do *software*.

Foi definido como condição de apoio do modelo a fixação dos deslocamentos nos 3 eixos na região inferior da células de carga, região esta que em condições reais fica aparafusada na chapa inferior da plataforma Figura 30.

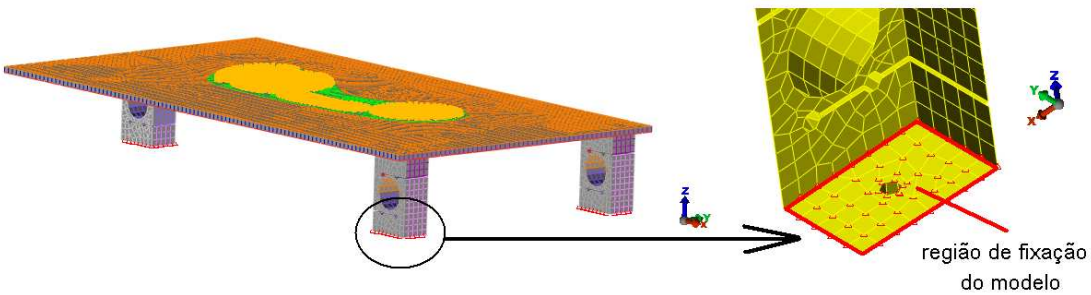


Figura 30: Região de fixação do modelo.

Nesta configuração foi aplicado ao modelo FEA um carregamento similar ao que um pé humano realizaria sobre a plataforma, produzindo uma força peso relativa a 50kg de massa ou 500N aproximadamente, neste caso a força não é aplicada unicamente em um ponto específico e sim em uma área que representa a planta do

pé, para isso é preciso entender como são distribuídas as pressões na planta dos pés.

Estudos recentes de pressão plantar de Cavanaugh *et al* (1987) sobre sujeitos descalços em bipedestação tem determinado que a distribuição da carga no pé é como segue: calcâneo 60%, mediopé 8% antepé 28%, e dedos 4% Figura 31. As pressões de pico sob o calcanhar foram 2,6 vezes maiores do que as pressões no antepé. [NORDIN e FRANKEL 2004].

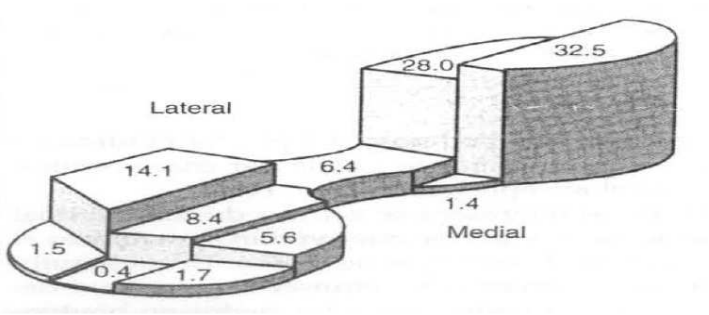


Figura 31: Distribuição média regional de peso expresso como uma porcentagem da carga total suportada pelo pé em bipedestação e pé descalço.
Fonte: NORDIN e FRANKEL, 2004 p. 246.

A Tabela 2 mostra como fica a intensidade da carga em forma de pressão plantar por unidade de área em função da área do pé do percentual de carga segundo Cavanaugh para uma carga total de 500N a Figura 32 mostra cada área na região plantar do suposto pé sobre a plataforma.

Tabela 2: Distribuição da pressão plantar.

região do pé	area mm² **	% carga*	carga N	pressão N/mm²
dedos	1478,13	4	20	0,01353
antepé	4077,42	28	140	0,03434
médiopé	1558,3	8	40	0,02567
retropé	2565,44	60	300	0,11694
total	9679,29	100	500	0,19047

** segundo estudos de Cavanaugh et al (1987).*
*** Os valores das áreas foram extraídos do modelo CAD*

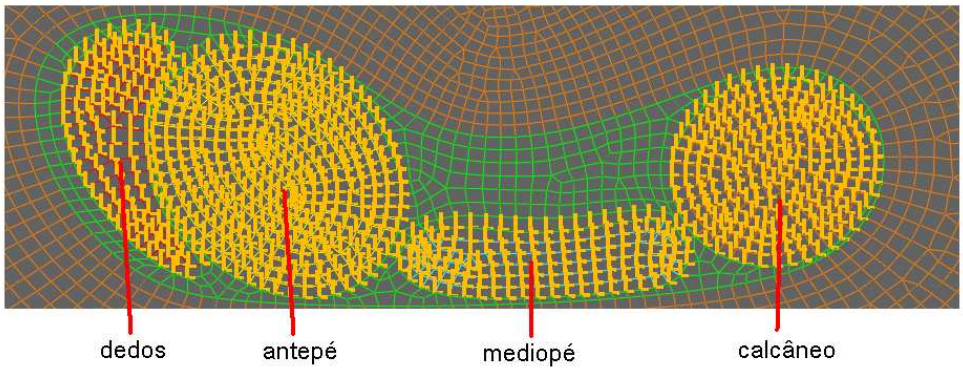


Figura 32: Aplicação das pressões nas diferentes regiões plantares, cada uma com uma intensidade diferente segundo estudos de Cavanaugh.

Depois de aplicadas as propriedades e carregamentos foi compilada a análise, o resultado da análise FEA apresenta vários parâmetros de resultado como, por exemplo: Tensões máximas e mínimas principais ou tração e compressão respectivamente; Tensões pelos critérios de *Tresca* e *Von Mises.* Estas tensões não são significativas, pois as forças atuantes são pequenas comparadas aos limites de resistência do material.

Também é possível visualizar as deformações ao longo dos eixos coordenados X, Y e Z do modelo, o resultado pode ser visto na Figura 33, que é interessante nessas análises, já que as medidas dos extensômetros dependem de deformações (deslocamentos), no entanto a magnitude das deformações nos eixo X Figura 33 (A) e eixo Y Figura 33 (B) são desprezíveis, pois não afetam a região que realiza as medidas de deformação nas células de carga, no entanto a deformação vertical no eixo Z Figura 33 (C), esta sim é considerável e causa deformação na região sensível da célula de carga. Na Figura 34, é possível ver como a plataforma deformou na direção vertical (eixo Z) sobre esta condição de carregamento e também mostra a vista lateral com a deformação aumentada em 1000 vezes. Como esperado, a região central da plataforma sofreu a maior deformação (região mais azul e lilás).

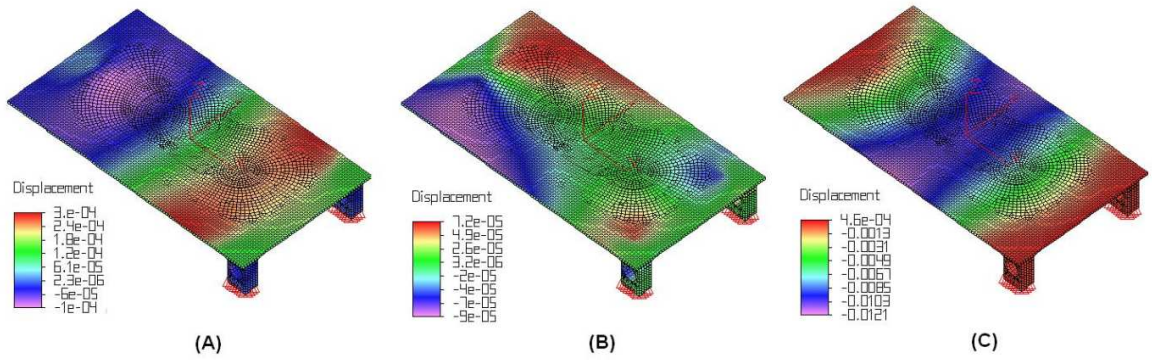


Figura 33: Deformação da plataforma nos eixos X (A), Y (B) e Z (C).

Baseando-se no resultado desta análise percebe-se que ocorreu um abaulamento da superfície de suporte, o que provoca uma torção da célula de carga. A princípio, acreditava-se que isso poderia causar uma leitura equivocada das deformações da célula, porém este efeito é pouco significativo e sua influência é desprezível em função da arquitetura do corpo da célula de carga, que anula a deformação por torção graças a dois cortes horizontais ao longo de parte da célula. Estes cortes fazem com que apenas os esforços compressivos deformem o círculo central do corpo das células, região onde estão acoplados os extensômetros.

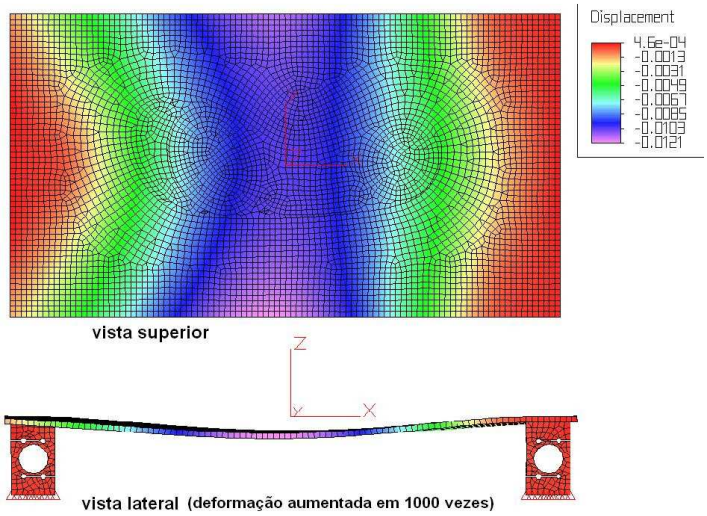


Figura 34: Deformação no eixo Z em vista superior e em vista lateral com as deformações aumentadas em 1000 vezes.

De qualquer forma, optou-se por uma superfície de suporte mais rígida, aumentando a espessura da chapa para 9,5mm. Uma nova análise foi realizada para essa nova condição, utilizando os mesmos parâmetros da análise anterior e aumentando somente a espessura da chapa da superfície para 9,5mm. O resultado pode ser visto na Figura 35 (A).

Se forem comparados os resultados da primeira análise (Figuras 33 e 34) com os resultados desta análise Figura 35 (A) e (B) é visto que o perfil da deformação pouco varia, porém a magnitude das deformações no eixo Z diminui significativamente, isso fica evidente quando comparadas as vistas laterais com deformação aumentada em 1000 vezes para ambas as análises (Figuras 34 e Figura 35 (B)), o abaulamento da superfície da plataforma é menos significativo na segunda análise.

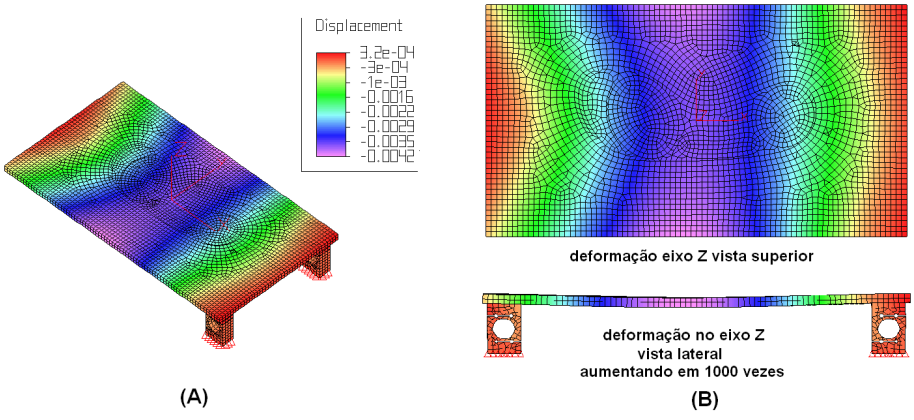


Figura 35: Resultado da análise para deformações no eixo Z para uma chapa de 9,5mm de espessura (A) e vistas superior e lateral com deformações aumentada em Z em 1000 vezes (B).

Esse aumento da rigidez da superfície de suporte confere à plataforma de força a capacidade de suportar o peso de pacientes de até 150Kg sem sofrer deformações significativas. Este limite é definido não apenas pela plataforma, mas pelo limite de carga especificado para as células de carga.

As células de carga que compõem o equipamento têm o formato em S fabricadas pela empresa Líder Balanças (*data sheet* anexo 1) esta pode ser vista na Figura 36, optou-se por células comercias, pois estas já atendem a estreitos padrões de qualidade, facilitando a montagem e eliminando erros dimensionais que podem ocorrer durante o processo de usinagem do bloco metálico que irá constituir o corpo da célula, minimizando erros de medidas devido a desigualdades entre as células de carga.

Figura 36: Célula de Carga modelo SC50 fabricada pela empresa Líder Balanças
Fonte : <http://www.liderbalancas.com.br/C%C3%A9lulas%20de%20carga.htm>

Na Figura 36 o circulo azul ao centro da célula, trata-se de um orifício passante no corpo da célula recoberto por material elástico (aparentemente silicone). É neste orifício que se localizam os sensores de deformação, *strain gauges,* colados na parede da circunferência do furo e arranjados eletricamente em forma de ponte *wheatstone.*

A partir do modelo em CAD da célula de carga, foi realizada outra análise FEA onde foi simulada uma condição de deformação da célula para um carregamento de 500N (*Newtons*) aplicado na parte superior da célula, simulando a força peso aplicada sobre esta conforme Figura 37.

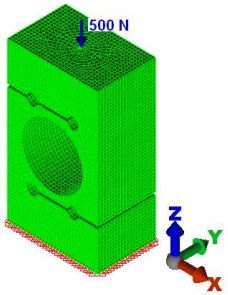


Figura 37: Condições de contorno e carregamento do modelo da célula de carga

Após a compilação do modelo, o resultado é mostrado na figura 38 para tensões máximas e mínimas principais, ou tração e compressão respectivamente. Com base neste resultado, é possível ver os pontos de maior tensão (tração) figura (A) e menor tensão (compressão) figura (B) no circulo interior da célula. São nestes pontos que estão colados os extensômetros.

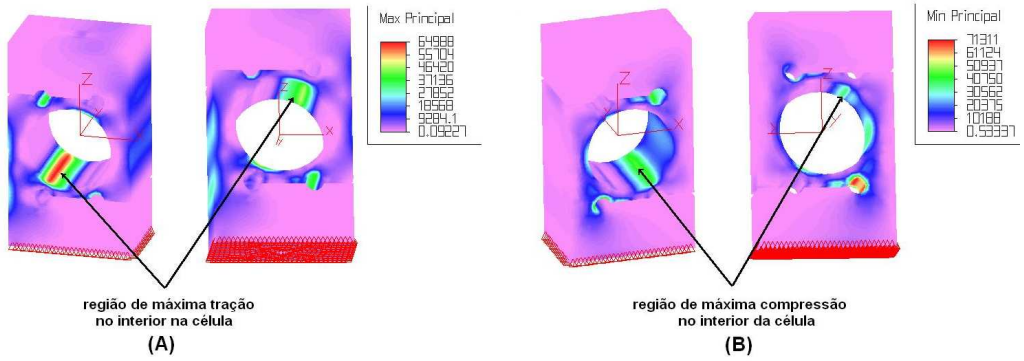


Figura 38: Ponto de maiores tensões principais máximas (A) e mínimas (B).

Outro resultado interessante de se avaliar é a deformação no eixo Z (vertical) sentido em que a força de 500N é aplicada Figura 39 (A), na Figura 39 (B) é mostrado a vista lateral de célula de carga onde é possível visualizar que a parte inferior da célula praticamente não se deforma (região lilás), no entanto a parte superior deforma-se de forma considerável (região do vermelho ao verde), a região central compreendida entre os quatro furos nos rasgos funciona como duas vigas engastadas na parte inferior da célula, e sofrem flexão quando uma força é aplicada na parte superior da mesma, o efeito fica mais evidente quando se vê a deformação da célula ampliada em 2000 vezes Figura 39 (C), é como se o círculo central da célula sofresse uma ovalização no sentido da força, isso faz com que parte do circulo esteja sujeita a compressão e parte a tração, ideal para a fixação dos extensômetros nestes pontos em forma de ponte de *wheatstone*.

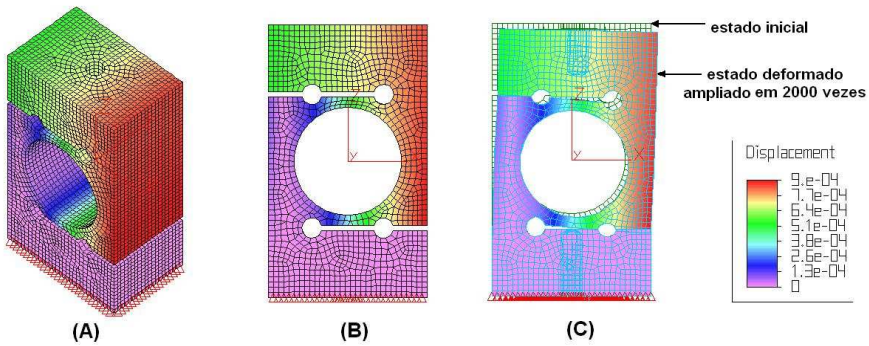


Figura 39: Deformação da célula na direção do eixo Z para o carregamento de 500 N.

O intuito desta simulação é entender como ocorrem as deformações na estrutura da célula sujeita a um carregamento, e como as informações extraídas desta análise ajudam a compreender como estão dispostos os sensores extensômetros no interior desse sensor.

A estrutura mecânica é composta basicamente pela superfície de suporte, células de carga e base de apoio sobre solo que constituem apenas a parte do equipamento que interage com o paciente, mas existe outra parte de cunho eletrônico não menos importante que é responsável por transmitir os dados lidos nas células de carga para o computador que fará a interação com o profissional de saúde, e será discutido adiante.

3.3. INTERFACE ELETRÔNICA

A eletrônica necessária ao funcionamento do equipamento esta dividida em duas partes distintas: circuito de condicionamento/pré-amplificação e circuito micro-processado de conversão analógica/digital e comunicação.

3.3.1. Circuito de condicionamento e pré-amplificação

O arranjo em ponte *wheatstone* dos extensômetros no interior da célula de carga por si só funciona como um circuito pré amplificador como discutido no item 2.3.3, porém o sinal de saída que é variação de tensão (V) da ponte que é proporcional ao deslocamneto da superficie onde está acoplado é muito baixa e requer que seja amplificado ainda mais a níveis de tensões compatíveis com o circuito de conversão analógico digital utilizado.

Para a realização da aplificação do sinal é utilizadpo um amplificador operacional modelo AD623 (ANALOG DEVICE, 2008), dispositvo indicado para instrumentação de precisão que requer grandes níveis de ganho, este dispositivo possui ganho de saída entre 100 a 1000 vezes a intesidade do sinal de entrada e configurado através de resistor de ganho (R_G) Figura 40, onde o dispositivo é ligado diretamente aos terminais de medida da ponte de *wheatstone,* se R_G for configurado para se obter um ganho de 200, por exemplo, para a tensão de 10mV da entrada será medido na saída 2V.

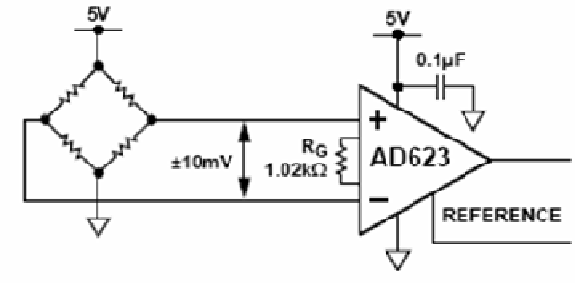


Figura 40:Esquema ligação da ponte wheatstone ao OpAmp AD623.
Fonte: ANALOG DEVICES. OpAmp AD623 Datasheet.

O diagrama esquemático simplificado do AD623 é mostrado na Figura 41, onde é possivel verificar que este é composto por um arranjo de três amplificaodres operacionais internos.

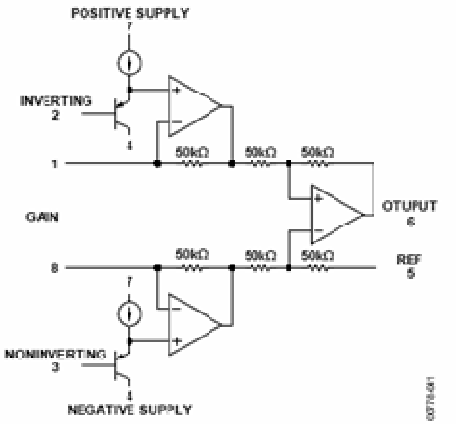


Figura 41:Diagrama lógico do circuito integrado do AD623.
Fonte: Analog Devices. OpAmp AD623 Datasheet.

Dada a baixa amplitude do sinal de entrada torna-se necessário realizar uma filtragem do sinal, pois células de carga são muito sensíveis a vibração e perturbações causadas por agentes externos bem como ruídos gerados na fonte do sistema elétrico. Para realizar a filtragem, é utilizado um circuito chamado “filtro passa baixa”. Esse filtro, cosntituído por um circuito RC (Resistor Capacitor) de primeira ordem, é utilizado para cortar sinais harmônicos elevados e ruídos de alta frequência. Esses filtros devem ser analógicos e calculados conforme a característica do sinal de interesse a ser digitalizado (JOHNSON, 1993 *apud* CAJUHI, 2010).

A freqüência de aquisição do sinal de oscilação do Centro de Pressão (CP) é dependente da tarefa que é investigada. Para a postura ereta quieta em indivíduos normais, as componentes de freqüência do CP estão abaixo de 10Hz (WINTER, 1995 *apud* FREITAS, 2006), segundo o teorema de Nyquist, uma freqüência de aquisição de 20Hz seria suficiente. Devido às freqüências do ruído presente no sinal,

são utilizadas, na prática, freqüências mais altas, tipicamente 100Hz (CARON 1997, *apud* FREITAS, 2006), sendo assim um filtro passa baixa com frequência de corte de 100Hz é suficiente.

A Figura 42 mostra o circuito de pré-amplificação e filtragem do sinal. Este circuito é replicado para cada uma das células de carga.

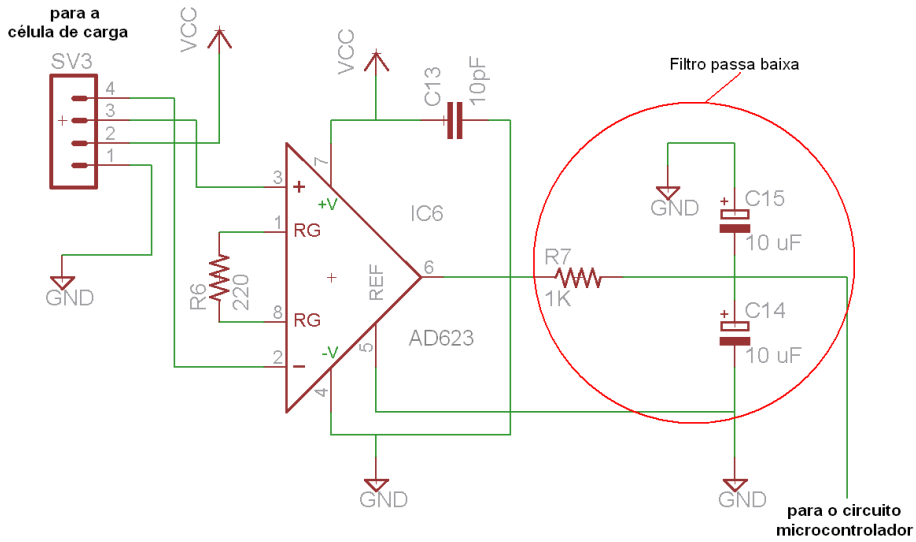


Figura 42: Circuito de pré-amplificação e filtragem. Fonte: Eagle software.

Uma vez amplificados e filtrados os sinais de saída das células de carga, esses sinais devem ser digitalizados, o que nada mais é que converter uma grandeza elétrica em uma sequência de números binários que possam ser lidos pelo computador. Essa etapa do processo chamada de conversão analógica/digital ou (A/D), ocorre em um circuito integrado chamado de microcontolador.

3.3.2. Circuito Microcontrolado

É responsável pela leitura do sinal analógico de cada célula de forma independente e realiza em nível de software embarcado a conversão destes em sinais digitais.

Para a digitalização dos sinais das 6 células de carga, é utilizado um microcontrolador modelo PIC16F877A (MICROCHIP, 2003). Este dispositivo conta como um de seus recursos um conversor A/D de 10bits em sua arquitetura interna, tornando o circuito eletrônico mais simples, e deixando a nível de software as etapas subsequentes a digitalização como a “buferização” (aramazenamento prévio e temporal dos dados) e preparação para comunicação com o computador através de

protocolo RS232 (serial), já que este dipositivo também dispõe desse meio de comunicação como um de seus recursos internos.

O conversor A/D interno deste dispositivo possui 10bits de resolução e possibilita que este acesse até 8 portas do microcontrolador, sendo uma por vez, o que é suficiente, já que é preciso 6 portas para conversão A/D, uma para cada célula de carga. A Figura 43 mostra o circuito microcontrolador, o circuito de comunicação e o circuito de alimentação.

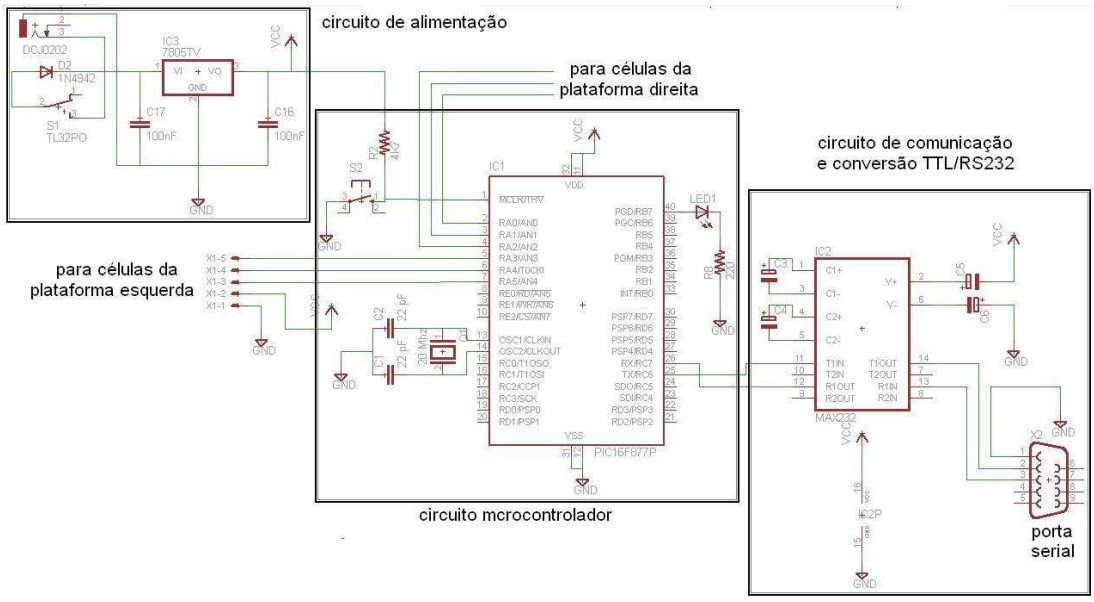


Figura 43: Circuito microcontrolador e comunicação. Fonte: Eagle software

3.3.3. Circuito de interface de comunicação serial.

Todos os aspectos referentes a protocolos de comunicação são estabelecidos no código embarcado no microcontrolador. No entanto é necessário o uso de um dispositivo eletrônico que converta o sinal elétrico de saída típico do microcontrolador (lógica TTL) em um sinal compatível com sinal elétrico padrão da comunicação RS232 (serial) que é utilizado nos computadores pessoais. Este dispositivo trata-se de um circuito integrado denominado Max232 (Texas Instruments, 2004), conforme Figura 43.

3.4. A LÓGICA EMBARCADA

Para o microcontrolador executar as operações de leitura e conversão A/D das entradas bem como executar operações de comunicação com um computador é necessário que haja uma rotina em código de máquina (hexadecimal) previamente

escrita e salva em uma região dento do microcontrolador denominada de memória de programa. Quando ligado, o dispositivo executa essa rotina e realiza as operações nela estabelecidas, Essa rotina ou seqüência de operações é chamada de lógica embarcada.

Durante a execução do projeto foi escrito o código de máquina que é executado pelo microcontrolador. O código fonte do programa, que funciona dentro do micro controlador, encontra-se Apêndice 02, este foi escrito em linguagem C adaptada para microcontroladores, utilizando o compilador CCS.

O diagrama lógico das operações a serem executadas pelo microcontrolador pode ser visto na Figura 44.

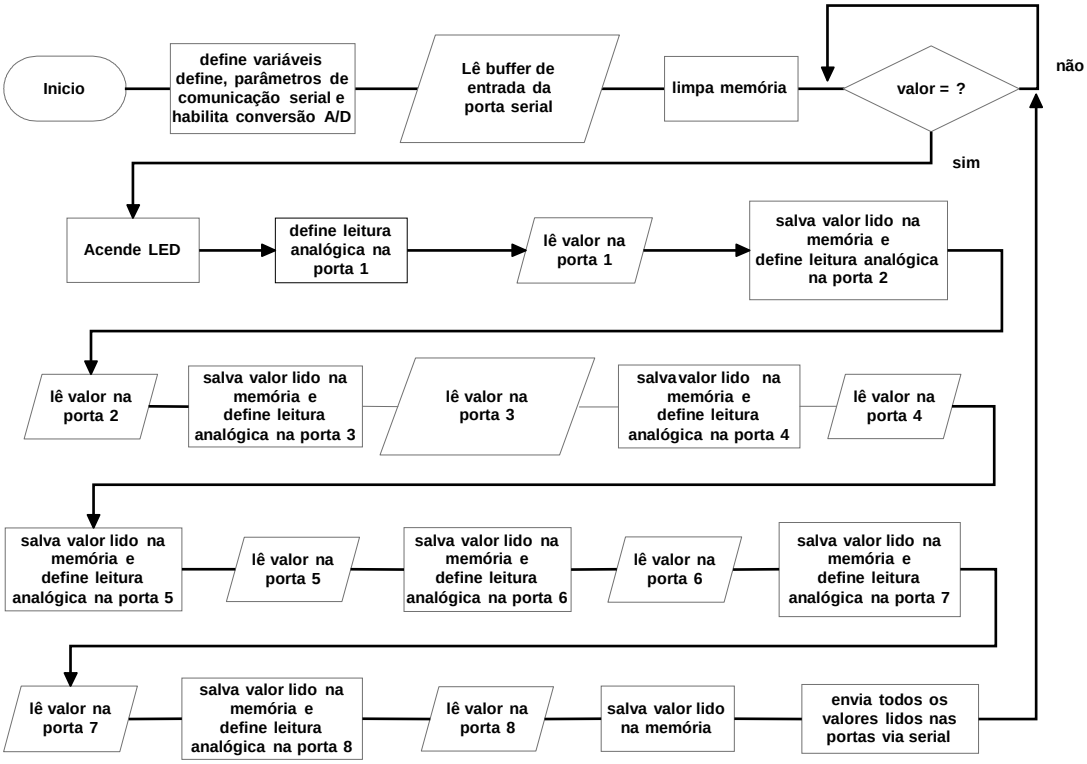


Figura 44: Digrama lógico do programa embarcado.

A rotina do código que funciona no microcontrolador executa a operação de varredura dos sinais de entrada das células de carga já pré-amplificados um a um e vai salvando cada um em uma variável pré-definida. Após realizar a leitura de todos os canais, ele aguarda uma solicitação de envio do pacote de dados contendo os valores de todos os sinais medidos em cada um dos canais, enviando-os via serial para o programa instalado no computador.

Depois de enviados os dados, o programa entra em uma rotina para acender e apagar um *Led* ligado a uma das saídas digitais do microcontrolador. O intuito desta

rotina é mostrar ao usuário do equipamento que o mesmo está funcionado e que as leituras e envio dos dados estão ocorrendo normalmente.

3.5. O SOFTWARE E A INTERFACE COM USUÁRIO

O Funcionamento da plataforma exige que se tenha um software dedicado previamente instalado no computador onde deseja realizar a aquisição de dados. esse software tem a função de solicitar e receber os dados enviados via serial pelo microcontrolador e processá-los a ponto desses dados estarem disponíveis para uso do estudo desejado pelo profissional de saúde. O software foi desenvolvido em linguagem C++, utilizando o framework QT4 O código fonte pode ser visto no Apêndice 03.

A informação básica de saída para o usuário são os gráficos de estabilograma e estatocinesigrama do paciente, discutidos anteriormente, que são tomados a partir dos valores da intensidade de força aplicada a cada célula de carga que foi convertido em um valor digital de 10bits. O software disponibiliza algumas opções de configuração como, por exemplo, em qual porta serial dentre as disponíveis está conectado o cabo de comunicação da plataforma, ou qual a taxa de amostragem das medidas, ou seja o tempo entre uma medida e outra. Alem destes recursos também disponibiliza algumas opções de visualização dos gráficos como, por exemplo, o estabilograma independente dos pés direito e esquerdo ou o do corpo paciente que é a media entre direito e esquerdo, conforme Figura 45.

Antes realizar as medidas, é possível calibrar o equipamento, já que no momento inicial as células de carga podem conter um valor inicial diferente para cada uma. O software lê estes valores e os toma como uma referência inicial, a partir daí, todas as medidas terá esse valor decrescido do valor lido.

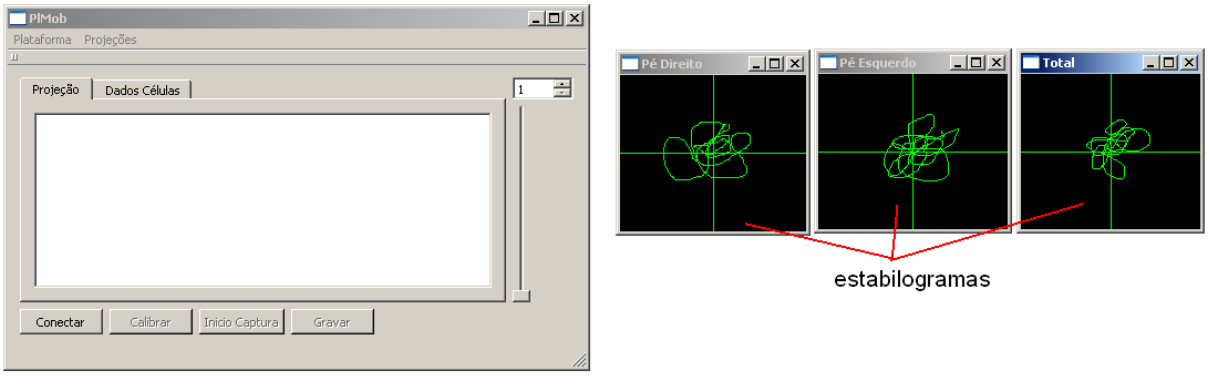


Figura 45: Interface gráfica do software.

3.6. LISTAS DE MATERIAIS UTILIZADOS

Os materiais utilizados na confecção da plataforma estão divididos em materiais mecânicos e eletrônicos conforme Tabela 3.

Tabela 3: Lista de materiais utilizados

Quantidade	Descrição do Componente	Onde se Aplica	R$
4	Chapas de alumínio 220x400x9,5mm	Plataforma mecânica	200,00
2	Hastes cilíndricas alumínio 300x12,7mm		2,00
6	Células de carga CS50		1500,00
20	Parafusos alem escareados M6x20		10,00
4	Blocos de alumínio de 1”x1”x1,5”		5,00
1	Placa para confecção de circuito impresso	Eletrônica	20,00
1	Caixa Patola PB-220/70 70x225x180mm		40,00
1	Solda estanho		1,00
1	Rabo de porco de 1,5 metro		5,00
1	Microcontrolador PIC16F877A	Eletrônica lógica	20,00
1	Cristal oscilador 20Mhz		1,00
2	Capacitores cerâmicos 22nF		0,40
1	Chave táctil		0,30
1	LED Amarelo 5mm		0,20
6	CI´s OpAmp AD623	Eletrônica pré-amplificação	90,00
6	Capacitores cerâmicos 10pF		0,60
12	Capacitores eletrolíticos 10uF 50v		1,80
6	Resistores 1KΩ		0,60
6	Resistores 220Ω		0,60
1	CI MAX 232 módulo conversor TTLxRS232	Eletrônica comunicação	3,00
1	Cabo conversor serialxUSB		35,00
1	Conector DB9 para placa 90°		3,00
4	Capacitores eletrolíticos 1uF 50v		0,80
1	Regulador de tensão 5 V	Eletrônica Alimentação	2,00
1	Chave duas posições		1,00
1	Diodo N4004		0,15
1	Conector coaxial 2 vias fêmea p/ placa		1,00
1	LED vermelho 5mm		0,20
1	Fonte alimentação 9 volts		5,00
1	Conector coaxial 2 vias macho p/ cabo		0,50
2	Capacitores cerâmicos 100nF		0,40
		Custo total R$	1953,55

3.7. CALIBRAÇÃO E TESTES

Durante a fase de teste foi realizado um ensaio para calibrar o equipamento onde foram aplicadas forças determinadas por pesos padrão em pontos específicos da superfície da plataforma. Para composição destes pesos foram utilizadas anilhas de academia sobre um anteparo que concentra a força peso em uma pequena área sobre a superfície, a Figura 46 mostra como estes foram aplicados sobre a plataforma.

Figura 46: Aplicação de pesos em pontos específicos sobre a plataforma.

A tabela 5 mostra a composição dos pesos totais (nominal e real medido em balança) a partir das anilhas disponíveis. Em posse dessas é possível obter qualquer quantidade de peso em números múltiplos 5 entre 5 e 35 kg, para a calibração foram aplicadas as seguintes quantidades de peso: 5 kg, 10 kg, 15, kg, 20 kg, 25 kg, 30 kg e 35 kg totalizando 7 pontos, porem para a composição dos gráficos foram utilizados apenas 5 pontos.

Tabela 4: Composição dos pesos com as anilhas disponíveis

Quantidade de anilhas	*5 (Kg) **5,280	*10 (Kg) **9,215	*10 (Kg) **9,365	*10 (Kg) **9,890	Peso total real (Kg)
1	x				5,280
1		x			9,215
2	x	x			14,495
2		x	x		18,580
3	x	x	x		23,860
3		x	x	x	28,470
4	x	x	x	x	33,750

** Peso nominal de cada anilha.*
*** Peso real medido em balança.*

O objetivo do ensaio é verificar as características dos sinais digitalizados das células de carga em função das cargas aplicadas em cada um dos pontos onde estas foram aplicadas. Na Figura 47 os pontos P1 a P10 representam os pontos onde foram aplicados os carregamentos, sedo que os pontos P1, P2, P3, P6, P7 e P8 representam a posição exata da localização das células de carga abaixo da plataforma. Os pontos P4 e P9 representam os baricentros dos triângulos com vértices posicionados em cada uma das 3 células de carga de cada lado da plataforma.

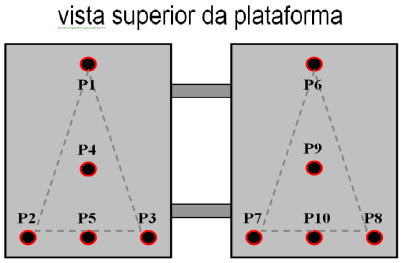


Figura 47: Localização dos pontos de aplicação de carga sobre a plataforma de força

O objetivo de se aplicar a carga aos pontos P1, P2, P3, P6, P7 e P8 é verificar que toda ou praticamente toda força peso deve estar aplicada unicamente na célula de carga localizada abaixo do ponto aplicado, nesta situação espera-se que nos sinais de leitura apenas a célula de carga sob este ponto sofra variação significativa, por exemplo, o gráfico da Figura 48 mostra a variação dos sinais das células de carga para a aplicação de carga no ponto **P1** e sua respectiva curva de ajuste linear, neste caso foram considerados apenas os sinais das células da plataforma esquerda.

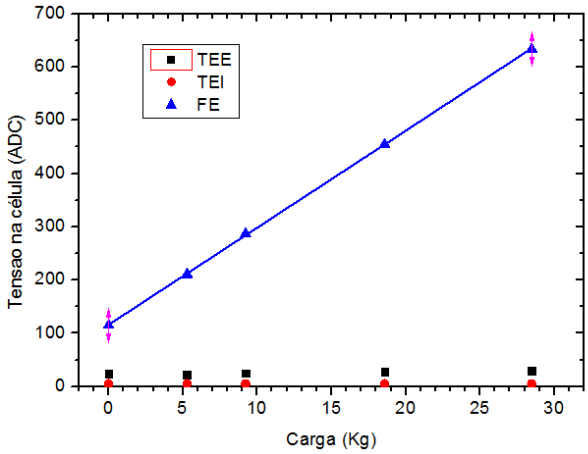


Figura 48: Sinal medido nas células de carga em função do peso aplicado no ponto P1.

No gráfico da Figura 48 a função da reta para a curva da célula frontal esquerda (FE) fica:

$$y = 116\pm0,5 + (18,23\pm0,03)x \quad (40)$$

Neste caso as demais curvas não são consideradas por não terem sofrido variação significativa.

O mesmo pode ser observado no gráfico da Figura 49 quando é aplicado carga no ponto **P6**, porém neste há variação significativa apenas no sinal da célula frontal direita (FD) da plataforma, exatamente abaixo do ponto **P6**.

No gráfico da Figura 49 a curva de ajuste linear da função da reta da célula frontal direita (FD) fica:

$$y=185,18\pm0,38 + (17,036\pm0,024)x \quad (41)$$

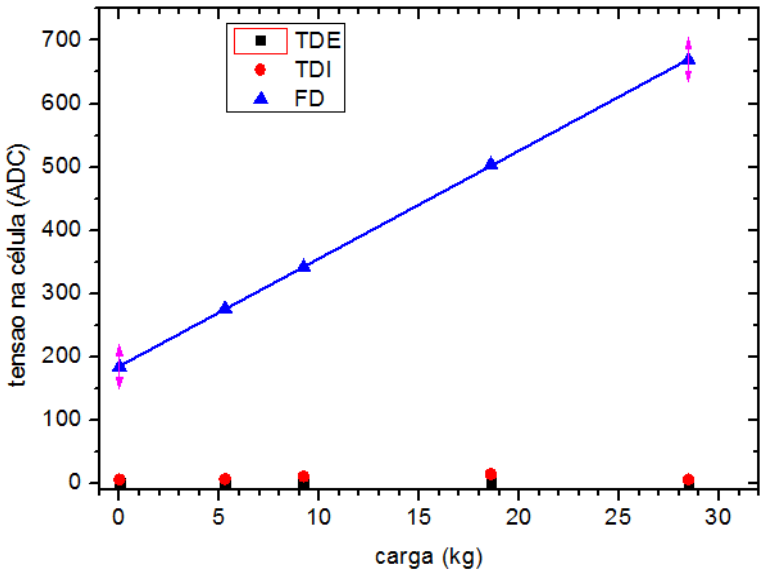


Figura 49: Sinal medido nas células de carga em função do peso aplicado no ponto P6.

Outros pontos a serem observado são o ponto **P4** Figura 50 e ponto **P9** Figura 51 que estão localizados no baricentro do triangulo com vértice no centro das células para o pé esquerdo e direito respectivamente, nestes pontos é observado que todas as células da plataforma devem sofrer variações parecidas.

No gráfico da Figura 50 a função da reta para a curva de cada célula da plataforma esquerda fica:

traseira esquerda externa (TEE) $y = 26,2\pm1,1 + (4,56\pm0,06)x$ (42)
traseira esquerda interna (TEI) $y = 3,7\pm1,0 + (4,13\pm0,05)x$ (43)
frontal esquerda (FE) $y = 110,0\pm1,16 + (8,82\pm0,06)x$ (44)

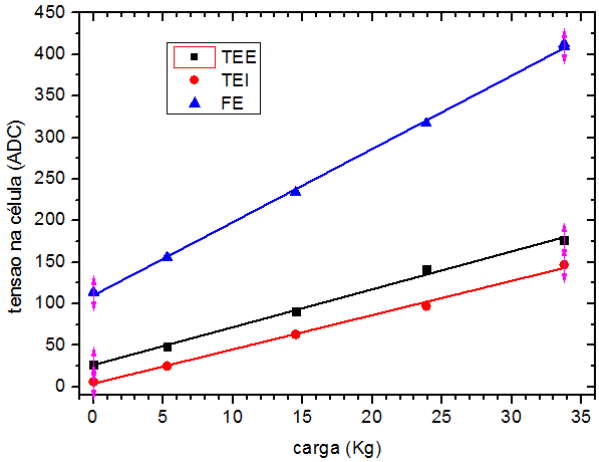


Figura 50: Sinal medido nas células de carga em função do peso aplicado no ponto P4.

No gráfico da Figura 51 a função da reta para a curva de cada célula da plataforma direita fica:

traseira direita externa (TDE) $y = -1{,}65 \pm 0{,}7 + (4{,}06 \pm 0{,}04)x$ (45)
traseira direita interna (TDI) $y = 8{,}0 \pm 0{,}42 + (4{,}62 \pm 0{,}03)x$ (46)
frontal direita (FD) $y = 183{,}7 \pm 0{,}97 + (7{,}65 \pm 0{,}06)x$ (47)

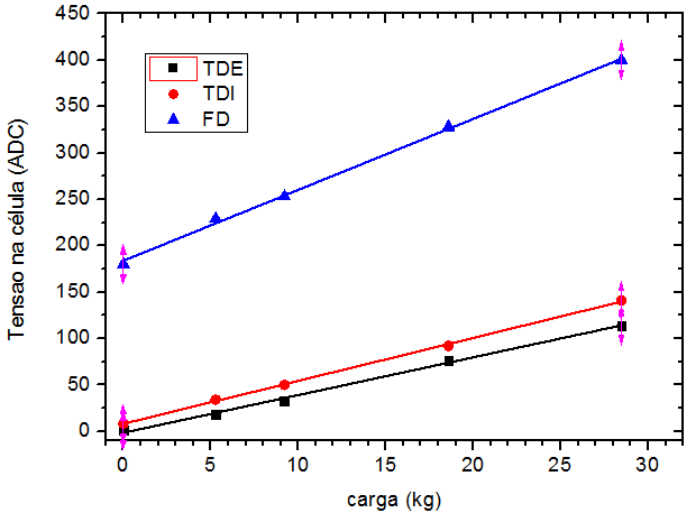


Figura 51: Sinal medido nas células de carga em função do peso aplicado no ponto P9.

Observando a inclinação das curvas de ambas às plataformas para os pontos P4 e P9 é possível perceber que possuem a mesma inclinação, confirmando que as células de ambas as plataformas respondem da mesma forma para um mesmo carregamento aplicado em cada uma delas de forma independente. A Figura 52 mostra os sinais de ambos os pontos P4 e P9, onde é possível verificar a similaridade das respostas dos sensores para uma mesma condição de carregamento em ambas plataformas.

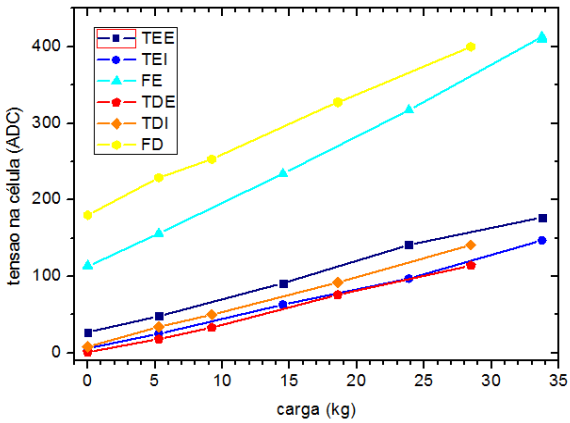


Figura 52: Aplicação das cargas nos pontos P4 e P9, plataforma esquerda e direita respectivamente.

Observando agora os pontos P2, P3, P7 e P8, pontos onde a carga é aplicada em apenas uma célula da parte traseira das plataformas, obtem-se os gráficos das Figuras 53 (A) e Figura 53 (B) para o ponto P2 e P3 respectivamente, e os gráficos da Figura 54(A) para P7 e Figura 54(B) para P8. Também é mostrada a curva de ajuste linear dos gráficos apenas das células que sofreram variação significativa.

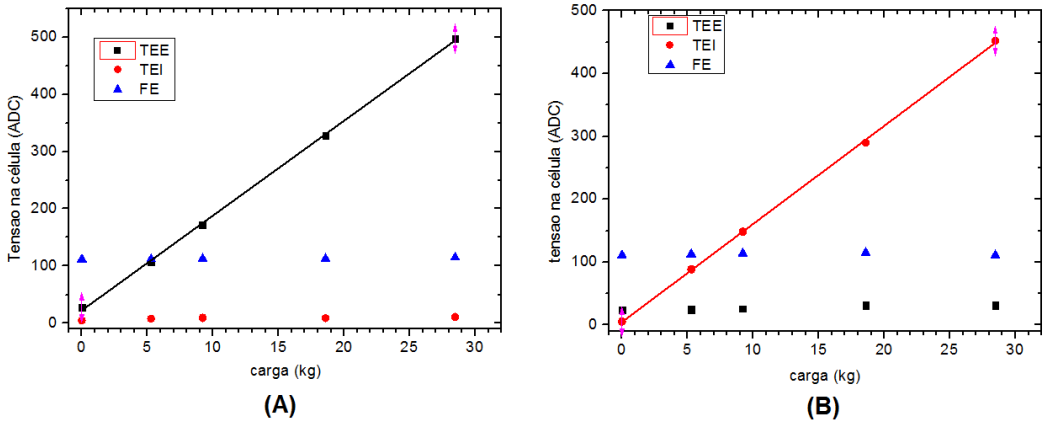


Figura 53: Respostas das células da plataforma esquerda para cargas nos pontos P2 (A) e P3 (B) para as cargas aplicadas.

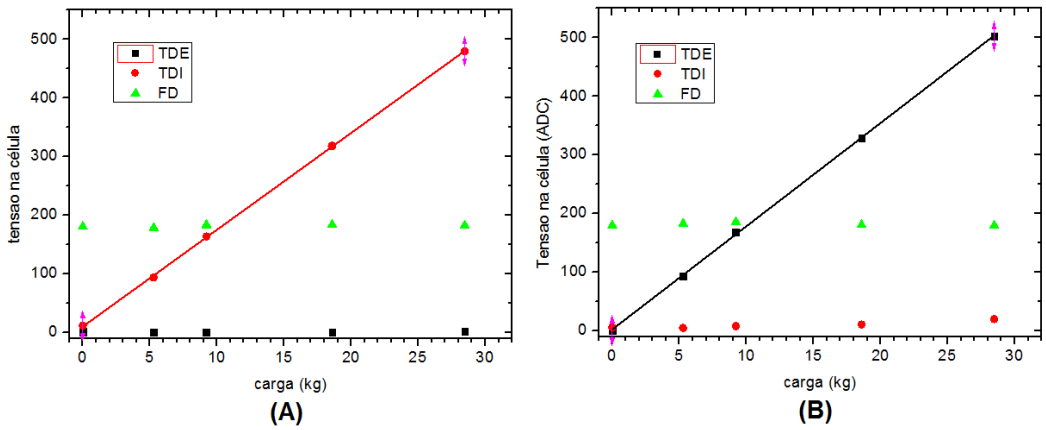


Figura 54: Respostas das células da plataforma esquerda nos pontos P7 (A) e P8 (B) para as cargas aplicadas.

Fazendo o ajuste linear das retas dos gráficos das Figuras 53 e Figura 54 obtemos:

Traseira esquerda externa (TEE) $22{,}6\pm1{,}2+(16{,}55\pm0{,}07)x$ (48)

Traseira esquerda interna (TEI) $4{,}75\pm0{,}85+(15{,}60\pm0{,}05)x$ (49)

Traseira direita externa (TDE) $2{,}5\pm0{,}7+(17{,}60\pm0{,}05)x$ (50)

Traseira direita interna (TDI) $10{,}4\pm0{,}64+(16{,}51\pm0{,}04)x$ (51)

Para os pontos P5 e P10 onde a cargas são aplicadas em um ponto eqüidistante entre as células traseiras obtem-se os gráficos da Figura 55 (A) para o ponto P5 e Figura 55 (B) para o ponto P10, em ambas as plataformas o sinal da célula frontal praticamente não sofre alteração.

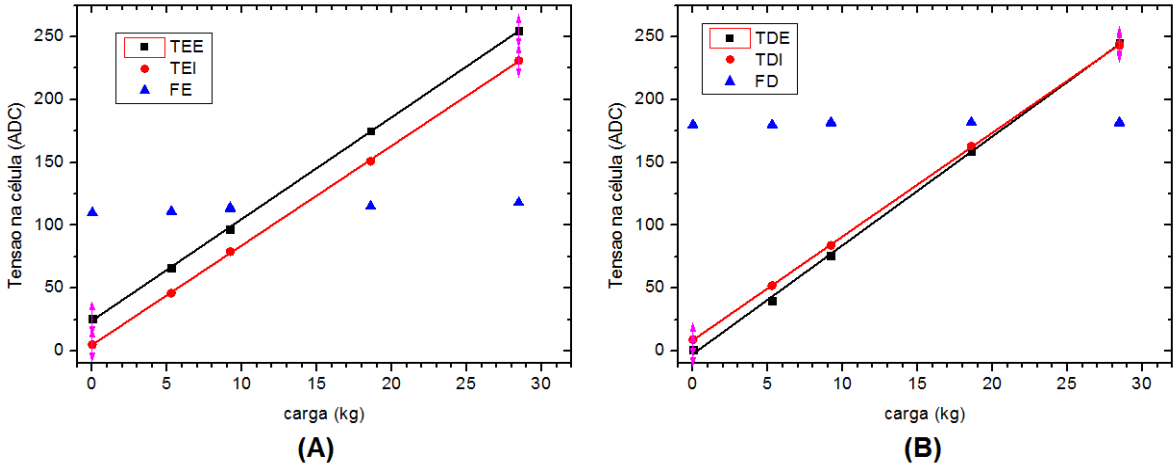


Figura 55: Respostas das células as cargas nos pontos P5 (A) e P10 (B).

O ajuste linear dos sinais fica da seguinte forma:

Traseira esquerda externa (TEE) $24{,}2\pm0{,}4+(8{,}1\pm0{,}02)x$ (52)

Traseira esquerda interna (TEI) $4{,}82\pm0{,}26+(7{,}93\pm0{,}02)x$ (53)

Traseira direita externa (TDE) $-2{,}5\pm0{,}74+(8{,}66\pm0{,}05)x$ (54)

Traseira direita interna (TDI) $8{,}64\pm0{,}2+(8{,}25\pm0{,}01)x$ (55)

RESULTADOS FINAIS

3.8. RESULTADOS PRELIMINARES

Um ensaio preliminar foi realizado com intuito de avaliar a validade dos dados medidos pelo equipamento. Neste estudo foi efetuada a captura dos dados a uma taxa de aquisição de dados de aproximadamente 30 Hz de uma seqüencia de tarefas realizadas por um paciente sobre a plataforma da seguinte forma: Inicialmente a plataforma permanece em repouso por 5 segundos (sem carga); em seguida, o paciente apóia primeiro o pé direito sobre a plataforma e depois o esquerdo e permanece na posição de equilíbrio estático durante 5 segundos de olhos abertos, em seguida o paciente fecha os olhos e permanece por mais 5 segundos, ao final desse ultimo período o paciente abre os olhos e desce da plataforma, primeiramente com o pé esquerdo e em seguida o direito.

Os dados coletados pelo sistema de aquisição foram transferidos para o *software Origin®,* onde foram representados em forma de gráficos. Estes representam as forças de reação na superfície da plataforma e a situação de equilíbrio do paciente durante o ensaio.

Na Figura 56, podem ser vistos os sinais elétricos relativos ao carregamento de cada célula de carga para o pé direito Figura 56 (A) e pé esquerdo Figura 56 (B).

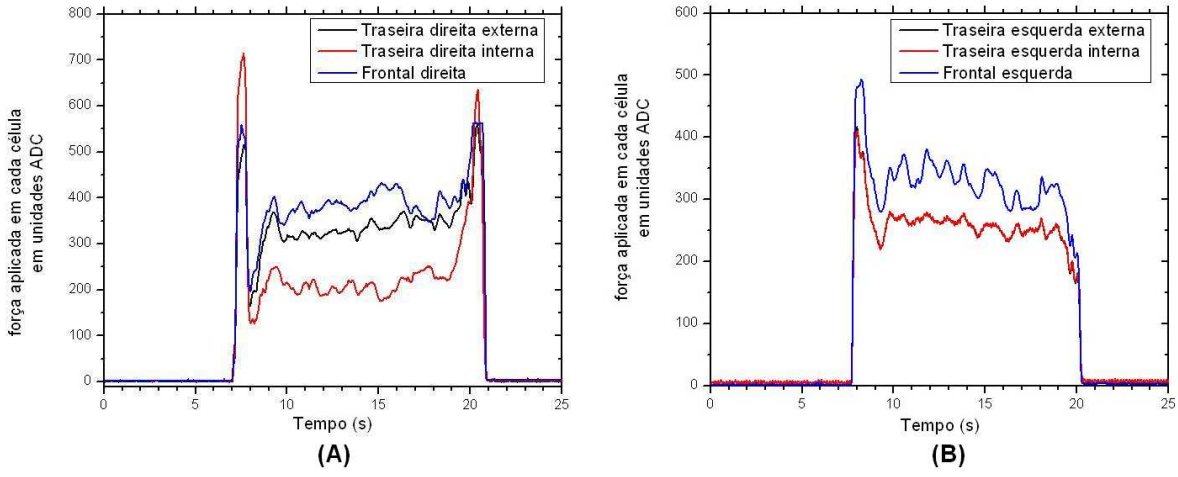


Figura 56: Dados lidos em cada uma das células de carga, (A) pé direito e (B) pé esquerdo.

Nos gráficos da Figura 56 o eixo X representa o tempo de execução dos movimentos pelo paciente. O eixo Y representa a intensidade da carga em unidade

digital referente às tensões (V) medidas nas células de carga. Cada curva mostra a intensidade de cada uma das 3 células de carga para cada uma das plataformas.

Aplicando a raiz quadrada da soma dos quadrados dos sinais lidos por cada célula para cada pé temos o gráfico da Figura 57, onde a curva RSQE é a raiz da soma dos quadrados das células do pé esquerdo e a curva RSQD a raiz da soma dos quadrados do pé direito.

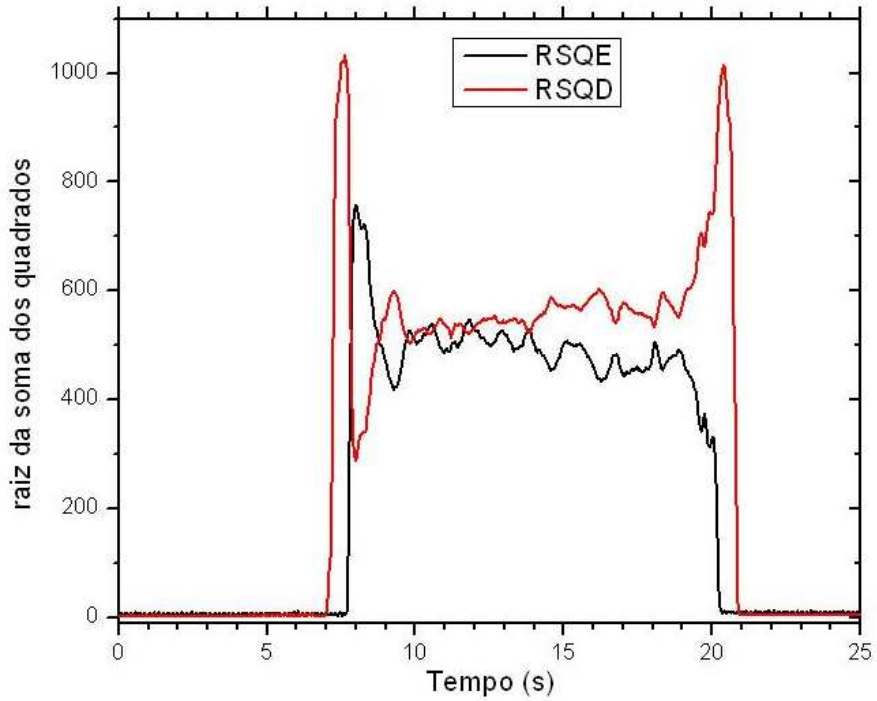


Figura 57: Raiz quadrada da soma dos quadrados dos sinais de cada uma das células de cada um dos pés (vermelho pé direito e preto pé esquerdo).

É possível observar que no instante em que o pé direito toca a plataforma toda a força peso do paciente é aplicada sobre a plataforma direita (RSQD) até que o pé esquerdo toque a plataforma (RSQE) causando uma diminuição expressiva da carga do pé direito até o corpo entrar em equilíbrio estático. Apesar do suposto equilíbrio, o corpo continua oscilando com pequena amplitude. Quando o paciente fecha os olhos, observa-se um aumento da amplitude de oscilação. Isto é verificado até que o pé esquerdo seja retirado voltando a carga a ser totalmente aplicada ao pé direito. O sistema somente volta ao repouso inicial quando o ultimo pé (direito) deixa a plataforma. Neste momento a intensidade do sinal das células volta a ser próximo de zero. Mas o importante a ser observado, de fato, é o instante entre o 9° e 19° segundo, período em que o corpo encontra-se em equilíbrio *quasi-estático*.

A partir desses mesmos dados foi feita a soma e a diferença da raiz quadrada da soma dos quadrados dos dados de ambas as plataformas Figura 58, neste gráfico a soma das raízes é a resultante das forças aplicadas sobre a plataforma.

Já a diferença das raízes quadradas pode representar a tendência de oscilação médio-lateral (ML) do corpo que é a oscilação lateral do corpo em relação a um eixo mediano imaginário que divide o corpo humano de forma simétrica. O quadrante superior ou positivo corresponde ao lado direito do paciente, enquanto o quadrante inferior ou negativo corresponde ao lado esquerdo do mesmo.

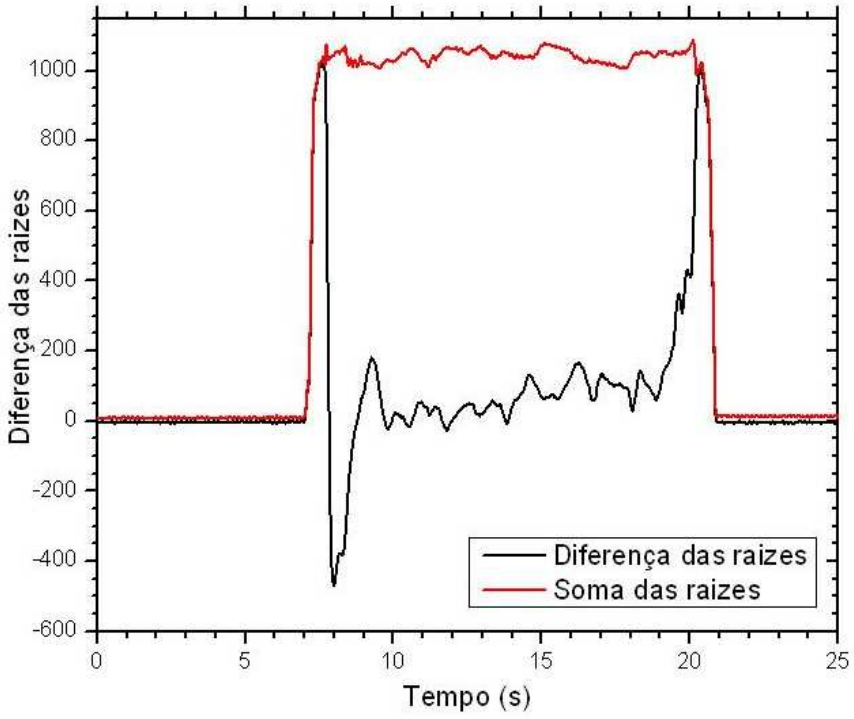


Figura 58: Soma e diferença da raiz da soma dos quadrados dos sinais (vermelho soma e preto diferença).

Outro dado importante tomado deste experimento é a raiz da soma dos quadrados dos sinais das células de carga frontais e traseiras de forma independente, que é mostrado no gráfico da Figura 59 (A), onde a curva RSQF é a raiz da soma dos quadrados das células da frente e a curva RSQT a raiz da soma dos quadrados das células de traz. No gráfico da Figura 59 (B) é mostrada a diferença das raízes (*RSQT-RSQF*), que pode ser entendido como a oscilação Antero-posterior do corpo do paciente, durante o período compreendido entre o 9° e 19° segundo de análise, que é o tempo em que o paci ente permanece em equilíbrio *quasi-estático* sobre a plataforma.

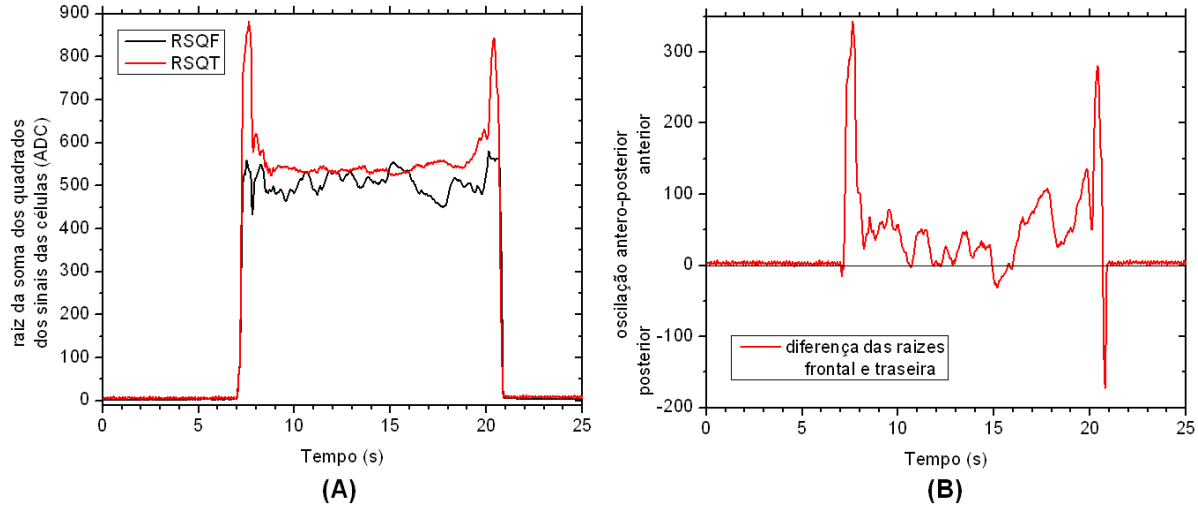


Figura 59: Raiz da soma dos quadrados das células da frente e de traz (A) e diferença das raizes entre frente e traz (B).

Aplicando a equação 35 e equação 37 para os dados coletados nesta análise, para o intervalo entre o 9º e o 19º segundo, é possível obter as coordenadas em X e Y para o centro de pressão (CP) dos pés esquerdo e direito, e aplicando o resultado destas nas equações 38 e 39 é possível obter as coordenadas do CP para o corpo do paciente, e em posse destas gerar o gráfico do estabilograma Figura 60 que é a oscilação do CP sobre a plataforma e também o estatocinesigrama onde a Figura 61 (A) representa as oscilações nas direções médio-lateral e a Figura 61 (B) representa as oscilações na direção Antero-posterior, para o pé esquerdo, pé direito e corpo do paciente durante o intervalo de tempo.

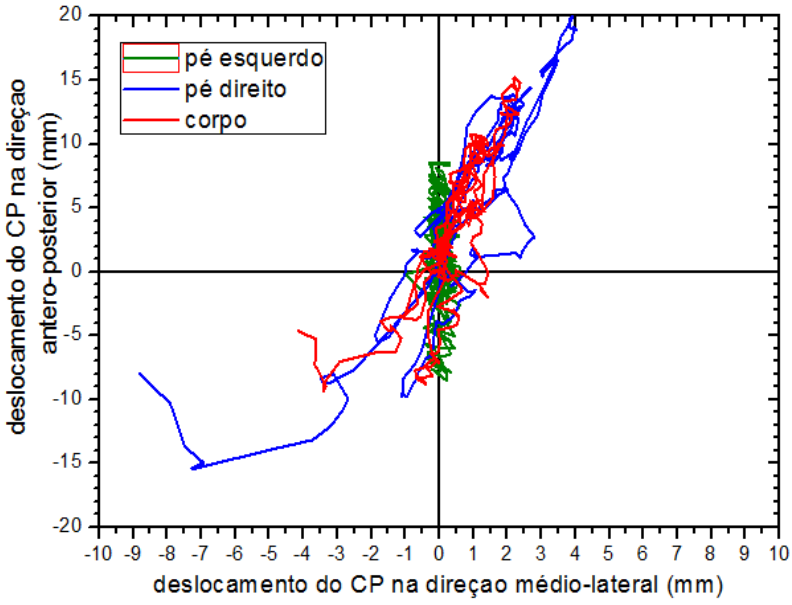


Figura 60: Estabilograma para a análise estudada.

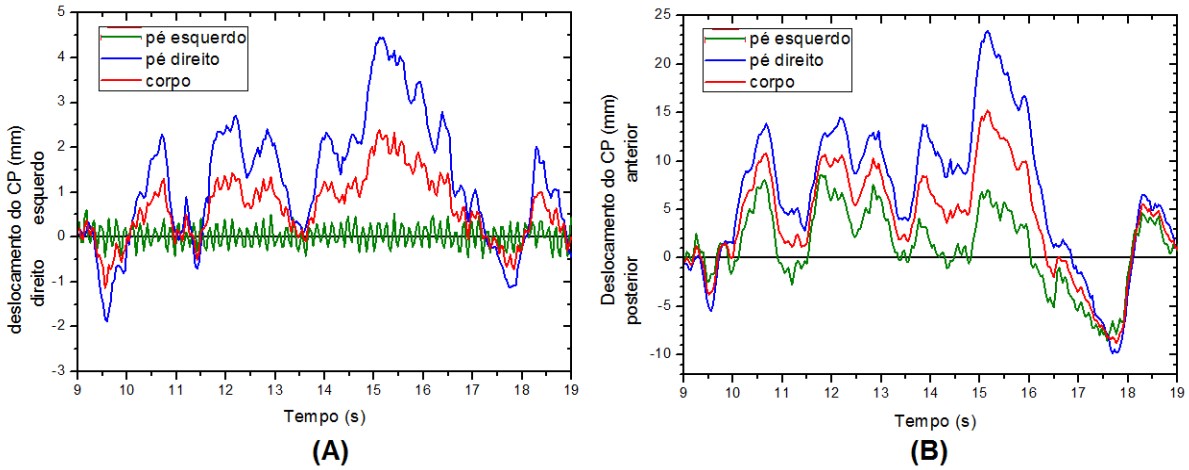


Figura 61: Estatocinesigrama para a análise estudada, plano medio-lateral (A) e plano antero-posterior (B).

Observando o estabilograma Figura 60 e o estatocinesigrama Figura 61 (A), é verificado que o pé esquerdo do paciente não sofreu oscilações significativas no plano médio lateral. Isso é possível por conta da posição em que o pé esquerdo permaneceu sobre a plataforma em uma posição desfavorável a medidas neste plano. Esse fato se dá por conta do tipo de movimentos executados. O ato de subir e descer da plataforma implica, às vezes, em um mau posicionamento dos pés sobre a mesma, este tipo de ocorrência pode ser evitada adicionando marcas na superfície da plataforma que indiquem a melhor posição dos pés sobre a plataforma.

Outra análise foi realizada com um paciente com idade de 18 anos 172 cm de altura e 64 kg de peso.

Este paciente permaneceu durante 30 segundos sobre a plataforma sendo os 15 primeiros de olhos abertos em seguida 10 segundo de olhos fechados e mais 5 segundos de olhos abertos, na posição ereto-estática olhando para um ponto fixo na parede a 2 metros de distancia.

A Figura 62 mostra o estabilograma para o paciente durante o período analisado, a Figura 63 mostra o estatocinesigrama na direção Antero-posteior, e a Figura 64 mostra o estatocinesigrama na direção Médio-lateral para o mesmo período. Analisando o estatocinesigrama da Figura 63, percebe-se, pelo aumento da amplitude das oscilações, que o corpo oscila com maior intensidade na direção Antero-posteior quando o paciente encontra-se de olhos fechados, no entanto as oscilações na direção Médio-lateral (Figura 64) durante o mesmo período, não são

significativas, mas é observada uma mudança brusca na posição do CP neste sentido, momentos antes do paciente abrir os olhos.

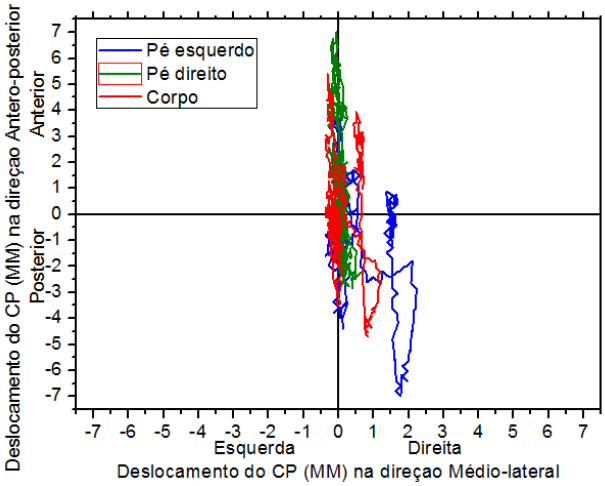


Figura 62: Estabilograma durante período de 30 segundos.

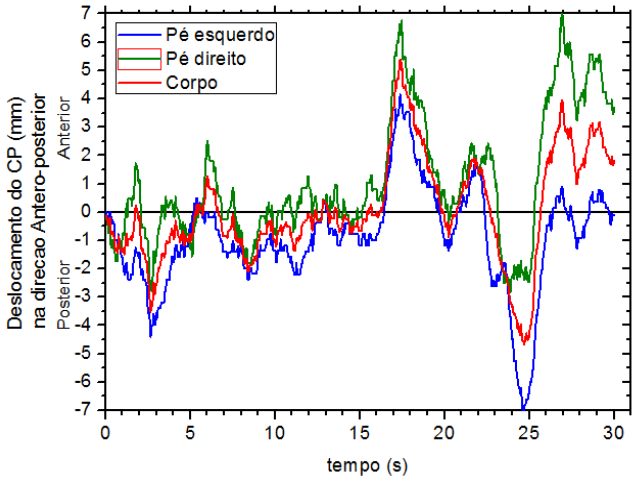


Figura 63: Estatocinesigrama na direção Antero-posterior durante período de 30 segundos.

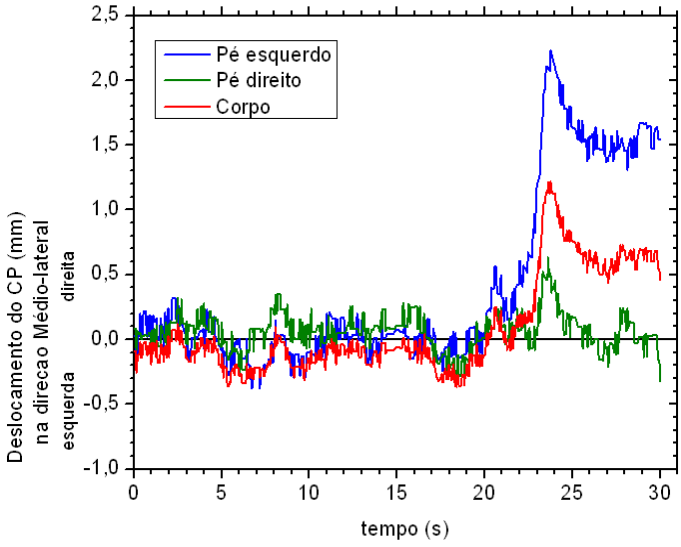


Figura 64: Estatocinesigrama na direção médio lateral durante período de 30 segundos.

Avaliando as amplitudes de oscilação do segundo paciente, observadas nas Figuras 62,63 e 64 e comparando-as com as amplitudes de oscilaçõao do 1º paciente, (Figuras 60 e 61) o segundo pacinte apresenta uma condição de equilíbriio mais estável sobre a plataforma que o primeiro paciente, porém com base nesta análise comparitiva não é possível determinar se existe algum probelma no sistema de controle postural. Para isso seria necessária uma análise mais aprofundada acompanhada por um profissional de saúde e seria necessário um grupo de pacientes com as mesmas características para a apartir daí compor um cenário de diasgnóstico.

Ambos os casos analisados não são conclusivos quanto ao comportamento do equilíbrio postural dos indivíduos, porem estes mostram que o equipamento responde as mudanças da condição de equilíbrio o que prova que este equipamento é funcional e que pode ser utilizado para análise do equilíbrio do corpo humano em regime de equilíbrio estático, bastando apenas que os estudos sejam realisados sob a orientação de um profissional de saúde, com procedimentos bem definidos e em grupos de pacientes específicos.

CONCLUSÃO

Neste trabalho foi projetada, desenvolvida e montada uma plataforma de força, baseada em seis células de carga extensométricas comerciais. A geometria de disposição desses sensores é uma proposta inovadora em relação aos equipamentos de mercado.

Ainda neste trabalho, foi desenvolvida a instrumentação analógica e digital dos sensores de força. Neste esforço, foram projetados, desenvolvidos e montados os pré-amplificadores, filtros e amplificadores, além de um sistema embarcado micro-controlado baseado no PIC 16F877A, que tem um conversor analógico digital interno de 10 bits.

Para o desenvolvimento da eletrônica de condicionamento de sinais analógicos, foi necessário um estudo sobre a configuração, o comportamento e as características dos sinais elétricos de saída das células de carga, quando submetidas ao esforço de pés humanos sobre a geometria da plataforma de força em diferentes situações de carregamento.

Para o funcionamento do microcontrolador e a aquisição de dados foram escritos os códigos de programação do *firmware* embarcado, o protocolo de comunicação para a porta RS232 e um *software* de interface com o usuário. Esse *software* tem finalidade de apresentar as informações úteis ao usuário sob forma gráfica ou por indicadores da posição do centro de pressão do paciente estudado.

O objetivo deste trabalho de construir uma plataforma de força instrumentada com a finalidade de auxiliar no estudo de padrões biomecânicos e postural de pacientes com problemas locomotores foi alcançado.

Uma vez a plataforma montada, foi feita a calibração dos sensores e testes de aquisição de dados para a validação do equipamento foram realizados. Esses ensaios não tiveram a finalidade de identificar distúrbios de equilíbrio dos indivíduos estudados, mas tão somente, avaliar a sensibilidade, a estabilidade e a repetibilidade do funcionamento da plataforma desenvolvida para aplicações de diagnóstico clínico. O mercado alvo para este equipamento são centros de saúde que trabalham com reabilitação de pacientes, tratamento e diagnostico de problemas do sistema motor, como por exemplo: o Centro de Prevenção e Reabilitação do Portador de Deficiência (CEPRED) - BAHIA. Esta instituição já manifestou interesse em dispor e explorar os recursos deste equipamento em seus estudos e tratamentos a doenças como mal de Alzheimer e mal de Parkinson.

SUGESTÃO DE TRABALHOS FUTUROS

Tendo o equipamento sido montado é possível explorar uma gama muito grande experimentos e estudos posturográficos. Como exemplo, acompanhar um grupo de pacientes com obesidade em fase preparatória para realização de cirurgia bariátrica, acompnhando-os também no pós-operatório, com intuito de avaliar como a perda de peso influencia na estabilidade e equilíbrio do paciente.

A plataforma de força pode passar por melhoramentos, no intuito investigar um número maior de parâmetros do equilíbrio, atualmente, ela mede apenas esforços verticais sobre a superfície de suporte, outra possibilidade, seria a montagem de uma plataforma que investigasse tambem os momentos torçores produzidos pelo corpo durante a postura ereta, ou montar uma plataforma dita de seis componentes que capta os esforços nos 3 eixos, além dos momentos torçores nos 3 eixos.

Outra possibilidade, seria um estudo dos sinais da plataforma montada com 6 sensores e compará-los aos sinais de uma plaraforma com 4 sensores, para um mesmo paciente, com a finalidade de avaliar as diferenças existentes nos sinais ou parâmetros motores investigados.

Desenvolver um sistema com uma plataforma de força abaixo de uma esteira ergométrica voltada para análises dinâmicas de estabilidade, avaliando condições de esforços em caminhadas e corridas.

REFERÊNCIAS

ALVES FILHO, Avelino. **Elementos Finitos**: a base da tecnologia CAE. São Paulo: Editora Érica, 2007.

ANALOG DEVICES **Single-Supply, Rail-to-Rail, Low Cost Instrumentation Amplifier AD623**. data sheet. Norwood, MA, 2008, 24 p.
Dispnível em: http://www.analog.com/static/imported-files/data_sheets/AD623.pdf

ANDOLFATO, Rodrigo Piernas; CAMACHO, Jefferson Sidney; BRITO, Gilberto Antônio de. **Extensometria Básica .**NUPAE – UNESP**,** Ilha Solteira, SP, 2004.

CAJUHI, Aguinaldo José. **Desenvolvimento de um sistema para sensoriamento de cargas em veículos automotores**. Dissertação de mestrado (mestrado em mecatrônica) Universidade Federal da Bahia. Salvador, 2010.

DUARTE, Marcos. FREITAS, Sandra M. S. F. , **Revisão sobre posturografia baseada em plataforma de força para avaliação do equilíbrio.** Revista Brasileira de Fisioterapia. São Carlos: v. 14, n. 3, p. 183-92, maio/jun, 2010.

FERREIRA, Flávia Porto Melo. **Produção do Journal of Biomechanics entre os anos de 2000 e 2001 relacionada ao tema equilíbrio**. Rio de Janeiro, 2003.

FRADEN, Jacob. **Handbook of modern sensors**: physics, designs, and applications**.** 3ª ed. AIP Press, 2003.

FREITAS, João Paulo. **Influência da manipulação osteopática sacroilíaca sobre a pressão plantar e oscilação corporal através do sistema de baropodometria e estabilometria**. Dissertação de mestrado (mestrado em engenharia biomédica). Universidade do vale do Paraíba. São José dos Campos, 2010.

FREITAS, Sandra M. S. F.; DUARTE, Marcos. **Métodos de análise do controle Postural.** Laboratório de Biofísica, Escola de Educação Física e Esporte, Universidade de São Paulo, 2006.
Disponível em: <http://demotu.org/pubs/Estabilografia.pdf>

GAZZOLA, Juliana Maria ; DONÁ, Flávia *et al.* **Realidade virtual na avaliação e reabilitação dos distúrbios vestibulares.** ACTA ORL/Técnicas em Otorrino-laringologia - Vol. 27 (1: 22-7, 2009).

GURALNIK J. M.; LACROIX A. Z.. **Assessing physical function in older populations**. In Wallace R. B. Woolson R. F., editors. The epidemiologic study of the elderly. New York: Oxford University Press; 199.p.159-81, 1996.

LÍDER BALANÇAS, **Célula de carga modelo CS50**.
Disponível em: <http://www.liderbalancas.com.br>

MICROCHIP. **PIC16F87XA Data Sheet, 28/40/44 - Pin Enhanced Flash Microcontrollers.** Microchip Technology Inc., 2003, 232 pg.
Disponível em: http://ww1.microchip.com/downloads/en/devicedoc/39582b.pdf

MOCHIZUKI, Luis; AMADIO, Alberto C. **Aspectos biomecânicos da postura reta: a relação entre o centro de massa e o centro de pressão**. Revista Portuguesa de Ciências do Desporto. 2003, vol. 3, nº 3 [77–83]
Disponível em:<http://www.fade.up.pt/rpcd/_arquivo/artigos_soltos/vol.3_nr.3/Mochizuki.pdf> Acesso em: 15/05/11 ás 12:00.

MONTEIRO, Wagner. *et al.* **Análise do equilíbrio dinâmico em idosas praticantes de dança de salão.** Revista Fisioterapia em Movimento, Curitiba, v. 20, n. 4, p. 125-136, out./dez, 2007.

NORDIN, Margareta; FRANKEL, Vitor H. **Biomecanica basica del sistema muscoesqueletico**. Madrid : McGraw-hill Interamericana, 2004. 485 p.

OMS, Organização Mundial de la Salud. **Aplicaciones de la epidemiología al estudio de los ancianos**: informe. Ginebra;1984. (OMS-serie de informes técnicos, 706).

PEIXOTO L.N.R.M.; VIVAS, Miranda J. G. *et al*. **Validação de ferramenta computacional livre e de código aberto, para análise acurada do movimento humano**. XVII EFNNE – Encontro de Físicos do Norte-Nordeste. Belém do Pará, 2009**.**

PORTO, Flávia; GURGEL,Jonas. *et al*. **Replicabilidade da EMGs do exercício rosca bíceps após período de 10 minutos.** 11º Congresso Brasileiro de Biomecânica, Sociedade Brasileira de Biomecânica UFPB. João Pessoa PB, 2005.

RAMOS, R. L. **Fatores determinantes do envelhecimento saudável em idosos residentes em centro urbano**: Projeto Epidoso, São Paulo. Cad. Saúde Pública v.19 n.3 Rio de Janeiro, 2003.

SBRITT, D. W.; BYLES J. E.; REGAM C. **Factors associated with decline in physical functional health in a cohort of older health in a cohort of older women**. Age and Aging. 36(4):382-388. 2007.

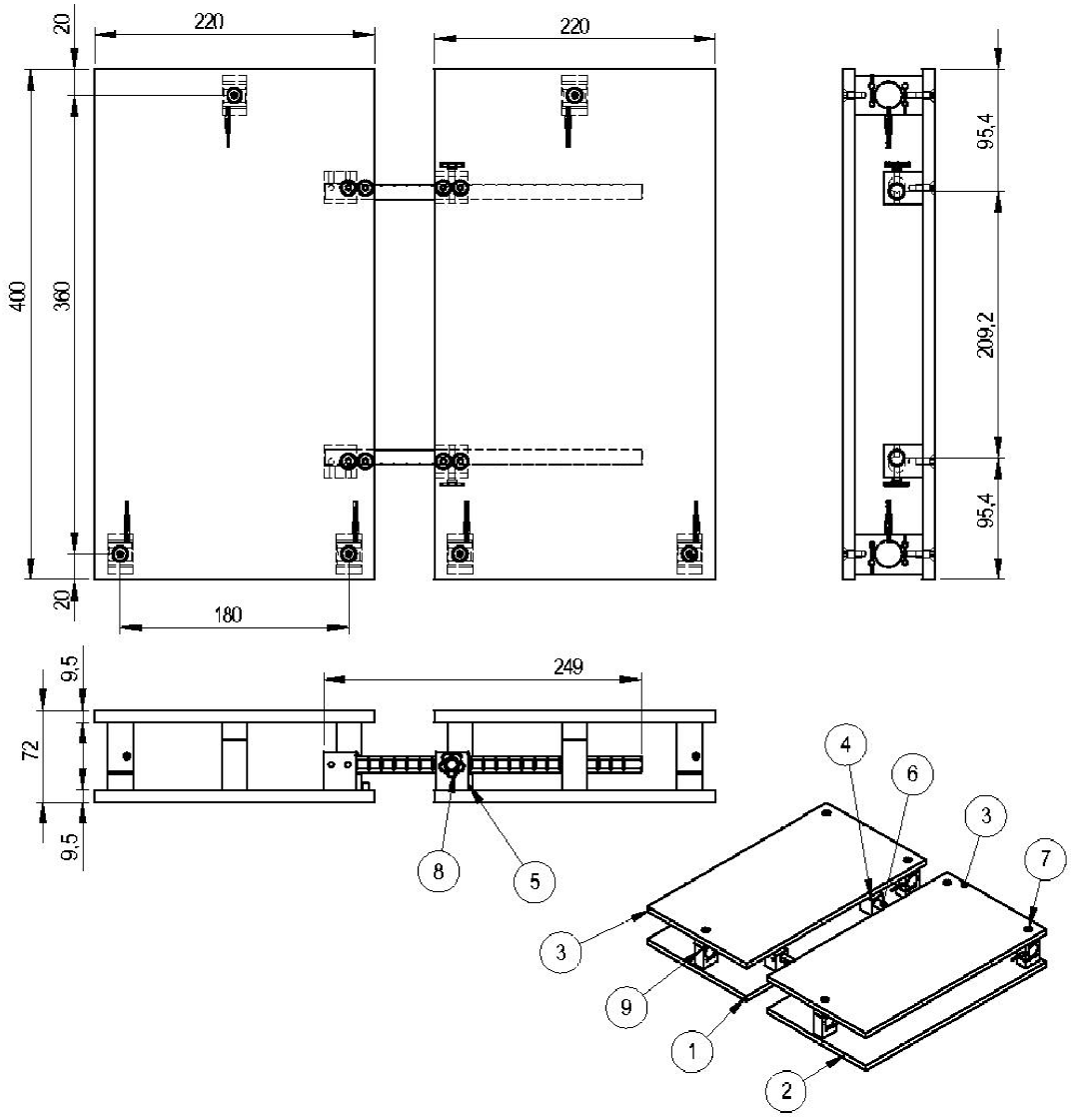
TEXAS INSTRUMENTS. **Max232 Dual EIA-232 drivers/Receivers Data Sheet**. Texas Instruments Inc.18 p. Dallas, Texas, 2004.
Disponível em: http://www.ti.com/lit/ds/symlink/max232.pdf

TSUTIYA, Nelson Yukishigue. **Análise comparativa do equilíbrio postural em idosos parknisonianos e não parkinsonianos através de parâmetros estabilométricos.** Universidade do Vale do Paraíba Instituto de Pesquisa e Desenvolvimento. São José dos Campos – SP: 2006.

VINHAS, I.; CUNHA, M.V.; VIVAS, M. J. G. ; PEÑA, N.. **Validação de uma proposta de análise cinesiologica dos movimentos humanos utilizando visão computacional**. Publicado nos anais do IV Congresso Brasileiro de Engenharia Clínica, Salvador-BA, 2008.

Apêndice 01 – Desenho técnico da estrutura da plataforma.

LaPO	PROJETO: Plataforma de Força Instrumentada	FORMATO.	A4
	TÍTULO: desenho plataforma de força	ESCALA:	1:5
		REVISÃO:	
	DES. Nº	FOLHA:	1 DE 1



Nº DO ITEM	Nº DA PEÇA	DESCRIÇÃO	QDT.
1	CH#01	Chapa aluminio 400x220x9,5	1
2	CH#02	Chapa aluminio 400x220x9,5	1
3	CH#03	Chapa aluminio 400x220x9,5	2
4	BL#01	bloco aluminio de 25x25x25	2
5	BL#02	bloco aluminio 25x25x25	2
6	HC#01	BARRA CIL 1/2 " X 250	2
7	PAE-M5#01	Parafuso allen escareado M5x25	20
8	PB#01	Parafuso borboleta M5	2
9	CC-CS50	Célula de carga mod CS50	6

Apêndice 02 – Código fonte do programa de aquisição de dados e comunicação embarcado no microcontrolador.

```
#include <16F877A.h>
#device ADC=10
#FUSES NOWDT          //No Watch Dog Timer
#fuses PUT            //No Power Up Timer
#FUSES NOPROTECT      //Code not protected from reading
#FUSES NOBROWNOUT     //Reset when brownout detected
#FUSES NOLVP          //Low Voltage Programming on B3(PIC16) or B5(PIC18)
#FUSES NOCPD          //No EE protection
#FUSES NODEBUG        //DEBUG DISABLE
#fuses HS             //Clock high speed

#use delay(clock=20000000)
#use rs232(baud=19200,xmit=PIN_C6,rcv=PIN_C7,parity=N,bits=8,stop=1,stream=pc)
#include "ForceEngine.h"
                              /*DEFINIÇÕES DE USO GERAL */
        #define  BUFFER_SIZE 50
                                    /*PINOS DE SAIDA */
        #define  LED PIN_B7 // PORTA B0
                               /*    Variaveis Globais     */
        char  buffer[BUFFER_SIZE];
        char  dados_lidos[BUFFER_SIZE];
                                  // FUNCOES ADICIONAIS
        #int_rda

void main()
{
 //Configuração para Conversor Analógico/Digital
  setup_adc_ports(ALL_ANALOG);
  setup_adc(ADC_CLOCK_INTERNAL);

// Configuração de Interrupções
   enable_interrupts(INT_RDA);
   enable_interrupts(GLOBAL);
// aguarda interrupção
    while(TRUE);
 }
void le_adc(void)          //rotina de leitura dos canail adc de cada sensor
{
long s1,s2,s3,s4,s5,s6,s7,s8; //define variáveis para cada canal
   fgets(buffer,pc);
  if (buffer[0]=='?') // testa se valor do buffer é igual a ?

    {
                                 // LÊ DADOS DOS SENSORES
      set_adc_channel(0);        //seta canais de leitura do conversor A/D
      delay_us(10);
      s1 = read_adc();          //atribui a variavel sx
     set_adc_channel(1);       // o valor lido no canal de A/D setado anteriormente
      delay_us(10);
      s2 = read_adc();
      set_adc_channel(2);
      delay_us(10);
      s3 = read_adc();
      set_adc_channel(3);
```

```
      delay_us(10);
      s4 = read_adc();
      set_adc_channel(4);
      delay_us(10);
      s5 = read_adc();
      set_adc_channel(5);
      delay_us(10);
      s6 = read_adc();
      set_adc_channel(6);
      delay_us(10);
      s7 = read_adc();
     set_adc_channel(7);
      delay_us(10);
      s8 = read_adc();
       fprintf (pc,",%Lu,%Lu,%Lu,%Lu,%Lu,%Lu,%Lu,\r\n",s1, s2, s3, s4, s5, s6, s7 s8);
           // envia daddos via serial
    }
else
   fprintf(pc,"Erro\r\n"); // envia mensagem de erro

    output_toggle(LED); // muda status do LED se aceso apaga se apagado acende
}
```

Apêndice 03 – Código fonte do programa de aquisição de dados e visualização dos dados capturados.

```
#include <QtGui/QApplication>
#include "interface.h"
int main(int argc, char *argv[])
{
    QApplication a(argc, argv);
    interface w;
    w.show();
    return a.exec();
}

                            //-------------------------------------------------------------------//
#include "boxplot.h"
#include <QtGui>
#include <QWidget>

boxPlot::boxPlot(QWidget *parent) :
    QWidget(parent)
{
}
void boxPlot::setData(vector <double> cells)
{
    this->data.clear();
    for(int i=0;i<cells.size();i++)
        this->data.push_back(cells[i]);
}

void boxPlot::paintEvent(QPaintEvent *event)
{
    if(this->data.size()==0) return;
    QPainter estojo;
    int lx=this->geometry().width();
    int ly=this->geometry().height();
    double fx=lx/this->data.size(); //this->plateSizeFootX;
    double fy=ly/1024.0; //this->plateSizeFootY;
    estojo.begin(this);
    //estojo.setBrush(Qt::black);
    estojo.drawRect(0,0,lx-1,ly-1);
    estojo.setBrush(Qt::blue);
    estojo.setPen(Qt::green);

    for(int i=0;i<this->data.size();i++)
        estojo.drawRect(i*fx,ly-this->data[i]*fy,fx,this->data[i]*fy);
    estojo.end();
}
                            //-------------------------------------------------------------------//
#ifndef BOXPLOT_H
#define BOXPLOT_H
#include <QWidget>
#include <vector>
using namespace std;
class boxPlot : public QWidget
{
    Q_OBJECT
public:
    explicit boxPlot(QWidget *parent = 0);
    void setData(vector <double> cells);
private:
    vector <double> data;   // vetor com os dados de cada celula

signals:
public slots:
```

```
protected:
    void paintEvent(QPaintEvent* event);

};
#endif // BOXPLOT_H
                          //-------------------------------------------------------------//
#include "device.h"
#include <cstdlib>
#include <QStringList>
#include <qextserialport.h>
#include <QtDebug>
#define BYTESWORD 200  // nunca acima de 1024
device::device()
{
    if(!this->readConfig())
    {
        this->port = 9;
        this->numbOfCells = 8;
        this->numbOfCellsForFoot=3;
        this->timeLag=1000;      // segundos
    }
    x=y=x1=y1=x2=y2=x1Calib=y1Calib=x2Calib=y2Calib=0;
    x1o=y1o=x2o=y2o=0;
}
device::~device()
{
    delete this->dvPort;
}

int device::getData()
{
    int i,j=0;
    char buff[1024];
    QString send = "?\n\r";     // Pedido de dados
    i = this->dvPort->write(send.toAscii(),send.length());
    i = this->dvPort->readLine(buff,1023);
    if(i==0)return 0;
    buff[i] = '\0';
    QString data(buff);
    QStringList dataList = data.split(",");
    int siz=dataList.size();
    int dat;
    bool ok=false;
    this->cellData.clear();
    for(int i=0;i<siz;i++)
    {
        dat=dataList[i].toInt(&ok);
        if(ok)
           this->cellData.push_back(dat);
    }
    // filtro passa baixa (calcula a média dos ultimos 5 valores de entrada
    for(int i=0;i<cellData.size();i++)
    {
        if(i>=this->cellQueue.size())
        {
            vector <int> aux;
            aux.push_back(this->cellData[i]);
            cellQueue.push_back(aux);
        }
        else
        {
            this->cellQueue[i].push_back(this->cellData[i]);
            if(this->cellQueue[i].size()>20) this-
>cellQueue[i].erase(cellQueue[i].begin());
        }
    }
```

```
    for(int i=0;i<cellQueue.size();i++)
    {
        double media=0;
        for(int j=0;j<this->cellQueue[i].size();j++)
            media+=this->cellQueue[i][j];
        this->cellData[i]=media/cellQueue[i].size();
    }
    return this->cellData.size();
}
int device::detectNumOfCells()
{
    char buff[600];
    int numCells=0;
    if(this->port==0)
    {
        file.read(buff,550);
    }
    else
    {
        int numBytes;

        numBytes = this->dvPort->bytesAvailable();
        if(numBytes >= 0)
        {
            if(numBytes > BYTESWORD) numBytes = BYTESWORD;

            int i = this->dvPort->read(buff, numBytes);
            buff[i] = '\0';
        }
    }
    QString data(buff);

    QStringList dataList = data.split(" ");
    int siz=dataList.size();
    for(int i=0;i<siz;i++)
    {
        qDebug()<< dataList[i];
        if(dataList[i]=="#")
        {
            while(dataList[i]!="$")
            {
                numCells++;
                i++;
                if(i>=siz)
                {
                    qDebug()<<i;
                    return -1;    // error in number of cells detection.
                }
                qDebug()<< dataList[i];
            }
            break;
        }
    }
    qDebug()<<numCells;
    if(numCells) {
        this->cellData.assign(numCells,0.0);
    }
    return numCells-1;

}

bool device::openPort()
{
    if( this->dvPort != NULL)
        if(this->dvPort->isOpen())
            this->dvPort->close();
```

```
    if(this->port==0)
    {
        file.open(this->fileName.c_str(),ios::in);
        if(file.fail()) return false;
        this->numbOfCells=this->detectNumOfCells();
        return true;
    }
    this->portCOM="COM"+QString::number(this->port).toStdString();
    //modify the port settings on your own
    this->dvPort = new QextSerialPort(this->portCOM.c_str());

    //this->dvPort->setPortName();
    this->dvPort->setBaudRate(BAUD19200);
    this->dvPort->setFlowControl(FLOW_OFF);
    this->dvPort->setParity(PAR_NONE);
    this->dvPort->setDataBits(DATA_8);
    this->dvPort->setStopBits(STOP_1);
    this->dvPort->setTimeout(500,500);
    this->dvPort->open(QIODevice::ReadWrite);
    if(!this->dvPort->isOpen()) return false;
    return true;
}
int device::autoSelectPort()
{
    return 9;
}
bool device::saveConfig()
{
    fstream fpConfig;
    fpConfig.open("config.pl",ios::out);
    if(fpConfig.fail()) return false;
    fpConfig << this->port << endl;
    fpConfig << this->timeLag << endl;
    if(this->port==0)           // se a entrada for um arquivo previamente gravado
    {
        fpConfig << this->fileName << endl;
    }
    return true;
    fpConfig.close();
}

bool device::readConfig()
{
    fstream fpConfig;
    fpConfig.open("config.pl",ios::in);
    if(fpConfig.fail()) return false;
    fpConfig >> this->port;
    fpConfig >> this->timeLag;

    if(this->port==0)           // se a entrada for um arquivo previamente gravado
    {
        fpConfig >> this->fileName;
    }
    return true;
    fpConfig.close();
}

bool device::calibrate()
{
    this->x1Calib=this->x1o;
    this->y1Calib=this->y1o;
    this->x2Calib=this->x2o;
    this->y2Calib=this->y2o;
    return true;
}
```

```
void device::calcProj()
{
    double Fa1,Fb1,Fc1;     // forças sobre as celulas da plataforma esquerda
    double Fa2,Fb2,Fc2;     // forças sobre as celulas da plataforma direita
    double XFr1, YFr1;      // localizacao da forca resultante na plat esquerda
    double XFr2, YFr2;      // localizacao da forca resultante na plat Direita
    double XFr,YFr;         // loc. forca resultante total
    if(this->cellData.size()==0)return;

    Fa1=this->cellData[2];
    Fb1=this->cellData[1];
    Fc1=this->cellData[0];
    Fa2=this->cellData[6];
    Fb2=this->cellData[5];
    Fc2=this->cellData[4];

    Fa1= 1/5.0 * Fa1;
    Fb1= 1 * Fb1;
    Fc1= 1 * Fc1;

    Fa2= 1/5.0 * Fa2;
    Fb2= 1 * Fb2;
    Fc2= 1 * Fc2;

    this->x1o = (90*(Fb1-Fc1))/(Fa1+Fb1+Fc1);
    this->y1o = (180*(Fa1-Fb1-Fc1))/(Fa1+Fb1+Fc1);
    this->x2o = (90*(Fc2-Fb2))/(Fa2+Fb2+Fc2);
    this->y2o = (180*(Fa2-Fb2-Fc2))/(Fa2+Fb2+Fc2);
    // pe direito
    this->x2=x2o-this->x2Calib;
    this->y2=y2o-this->y2Calib;
    // pe esquerdo
    this->x1=x1o-this->x1Calib;
    this->y1=y1o-this->y1Calib;
    // total
    this->x=((x1-50)*(Fa1+Fb1+Fc1)+(x2+50)*(Fa2+Fb2+Fc2))/((Fa2+Fb2+Fc2)+(Fa1+Fb1+Fc1));
    this->y=(y1*(Fa1+Fb1+Fc1)+y2*(Fa2+Fb2+Fc2))/((Fa2+Fb2+Fc2)+(Fa1+Fb1+Fc1));

}
                        //-------------------------------------------------------------------//
#ifndef DEVICE_H
#define DEVICE_H
#include <vector>
#include <queue>
#include <QThread>
#include <iostream>
#include <fstream>
using namespace std;
class QextSerialPort;
class device
{
public:
    device();
    ~device();
    int getData();
    double get_cellData(int cell);      // retorna o valor medido da celula "cell"
    bool calibrate();    // varre a plataforma vazia e preenche os vetores de calibracao
    vector <double> cellData;   // vetor com os dados de cada celula
    //vector <double> cellCalib;  // vetor de calibracao dos dados.
    vector < vector <int> > cellQueue;
    double x1Calib, y1Calib, x2Calib,y2Calib; // projecoes de calibracao de cada pe
    bool openPort();
    int autoSelectPort();//busca em cada porta uma que tenha os dados no formato correto
    void set_fileName(string name){fileName = name;}
    void set_numbOfCells(int num) {numbOfCells = num;}
    void set_port(int pt){port = pt;}
```

```
    void set_timeLag(double tl) {timeLag = tl;}
    int get_port(){return port;}
    int get_timeLag(){return timeLag;}
    int get_numbOfCells(){return numbOfCells;}
    string get_fileName(){return fileName;}
    string portCOM;
    bool saveConfig();  // save variables in configuration file.
    bool readConfig();
    int detectNumOfCells();
    void calcProj();
    double x1,y1,x2,y2,x,y;    // proj. pe esquerdo, direito e total respect. Corrigido
Calibracao
    double x1o,y1o,x2o,y2o,xo,yo;     // projeçoes do pe esquerdo, direito e total
respectivamente ORIGINAIS
private:
    int port;
    int numbOfCellsForFoot;
    int numbOfCells;            // numero de celulas da plataforma
    double timeLag;             // tempo de captura da porta
    fstream file;
    string fileName;
    QextSerialPort *dvPort;

protected:
    void run();
};
#endif // DEVICE_H
                          //-------------------------------------------------------------------//
#include "interface.h"
#include "ui_interface.h"
#include <QGridLayout>
#include <QLayout>
interface::interface(QWidget *parent) :
    QMainWindow(parent),
    ui(new Ui::interface)
{
    ui->setupUi(this);
    this->projFootL.setWindowTitle("Pé Esquerdo");
    this->projFootR.setWindowTitle("Pé Direito");
    this->projFootT.setWindowTitle("Total");
    this->projFootL.setGeometry(100,100,300,300);
    this->projFootR.setGeometry(100,100,300,300);
    this->projFootT.setGeometry(100,100,300,300);
    this->projFootL.hide();
    this->projFootR.hide();
    this->projFootT.show();
    this->timer = new QTimer(this);
    connect(timer, SIGNAL(timeout()), this, SLOT(update()));
    this->isCapturing = false;
    this->ui->CaldaVerticalSlider->setRange(1,400);
}
interface::~interface()
{
    delete ui;
}
void interface::on_actionP_direito_toggled(bool enable)
{
    if(enable) this->projFootR.show();
    else       this->projFootR.hide();
}
void interface::on_actionP_esquerdo_toggled(bool enable)
{
    if(enable) this->projFootL.show();
    else       this->projFootL.hide();
}
void interface::on_actionTotal_toggled(bool enable)
```

```
{
    if(enable) this->projFootT.show();
    else       this->projFootT.hide();
}
void interface::on_pushButtonInitCap_clicked()
{
    if (this->isCapturing)
    {
        this->ui->pushButtonInitCap->setText("Inicio Captura");
        this->timer->stop();
        this->ui->pushButtonRecord->setEnabled(false);
    }
    else
    {
        this->ui->pushButtonInitCap->setText("Para Captura");
        this->timer->start(this->plate.get_timeLag());
        this->ui->pushButtonRecord->setEnabled(true);
    }
    this->isCapturing = !this->isCapturing;
}

void interface::update()
{
    while(this->plate.getData()!=8);
    this->plate.calcProj();
    this->projFootT.setProj(this->plate.x,this->plate.y);
    this->projFootL.setProj(this->plate.x1,this->plate.y1);
    this->projFootR.setProj(this->plate.x2,this->plate.y2);
    this->projFootL.set_rastro(this->ui->CaldaSpinBox->value());
    this->projFootR.set_rastro(this->ui->CaldaSpinBox->value());
    this->projFootT.set_rastro(this->ui->CaldaSpinBox->value());

    this->plot.setData(this->plate.cellData);
    this->repaint();
    this->projFootL.repaint();
    this->projFootR.repaint();
    this->projFootT.repaint();
    this->timer->start(this->plate.get_timeLag());
    QString texto;
    for(int i=0;i<this->plate.cellData.size();i++)
        if(i!=3 && i!=7)
            texto += QString::number(this->plate.cellData[i],3,0)+"\t";
    this->ui->CelulasTextEdit->append(texto);
    texto= QString::number(this->plate.x,5,2)+"\t"+
            QString::number(this->plate.y,5,2)+ "\t";
    texto+= QString::number(this->plate.x1,5,2)+"\t"+
            QString::number(this->plate.y1,5,2)+ "\t";
    texto+= QString::number(this->plate.x2,5,2)+"\t"+
            QString::number(this->plate.y2,5,2);
    this->ui->ProjecaoTextEdit->append(texto);
}
void interface::on_pushButtonRecord_clicked()
{
}
void interface::on_action_Conectar_Plataforma_triggered()
{
    int port;
    port=this->plate.get_port();
    //this->plate.autoSelectPort();
    //interface para usuario selecionara porta
    this->plate.set_port(port);
    if(this->plate.openPort())
    {
        this->ui->pushButtonInitCap->setEnabled(true);
        this->ui->calibrarPushButton->setEnabled(true);
    }
```

```
}
void interface::on_actionC_onfigurar_triggered()
{
    this->configDialog.setParameter(this->plate.get_port(),this->plate.get_timeLag(),
    this->plate.get_fileName());
    this->configDialog.exec();
    this->plate.set_fileName(this->configDialog.getFileName());
    this->plate.set_port(this->configDialog.getPort());
    this->plate.set_timeLag(this->configDialog.getSampleFreq());
    this->plate.saveConfig();
}

void interface::on_actionC_alibrar_triggered()
{
    if (this->isCapturing)
    {
        this->plate.calibrate();
    }
    else
    {
        while(this->plate.getData()!=8) for(int h=0;h<500000000;h++);
        this->plate.calibrate();
    }
    this->ui->ProjecaoTextEdit->append("Calibrado...");
    this->ui->CelulasTextEdit->append("Calibrado...");
}
void interface::closeEvent(QCloseEvent *)
{
    this->projFootL.close();
    this->projFootR.close();
    this->projFootT.close();
}
void interface::on_conectarPushButton_clicked()
{
    this->on_action_Conectar_Plataforma_triggered();
    this->on_actionC_alibrar_triggered();
}
void interface::on_calibrarPushButton_clicked()
       {
       this->on_actionC_alibrar_triggered();
       }
void interface::on_CaldaVerticalSlider_valueChanged(int value)
{
}
                          //-------------------------------------------------------------//
#ifndef INTERFACE_H
#define INTERFACE_H
#include <QMainWindow>
#include <QGraphicsScene>
#include "device.h"
#include "projectview.h"
#include "portselect.h"
#include "boxplot.h"
#include <QTimer>
namespace Ui {
    class interface;
}
class interface : public QMainWindow
{
    Q_OBJECT
public:
    explicit interface(QWidget *parent = 0);
    ~interface();
private:
    Ui::interface *ui;
    device plate;
```

```
    projectView projFootL;
    projectView projFootR;
    projectView projFootT;
    boxPlot plot;
    QTimer *timer;
    bool isCapturing;
    portSelect configDialog;
private slots:
    void on_actionC_onfigurar_triggered();
    void on_action_Conectar_Plataforma_triggered();
    void on_pushButtonRecord_clicked();
    void on_pushButtonInitCap_clicked();
    void on_actionTotal_toggled(bool );
    void on_actionP_esquerdo_toggled(bool );
    void on_actionP_direito_toggled(bool );
    void update();
    void on_actionC_alibrar_triggered();
    void on_conectarPushButton_clicked();
    void on_calibrarPushButton_clicked();
    void on_CaldaVerticalSlider_valueChanged(int value);
protected:
    void closeEvent(QCloseEvent *);
};
#endif // INTERFACE_H
                          //-------------------------------------------------------------------//
#include "portselect.h"
#include "ui_portselect.h"
#include <QFileDialog>
#include <QString>
portSelect::portSelect(QWidget *parent) :
    QDialog(parent),
    ui(new Ui::portSelect)
{
    ui->setupUi(this);
    this->flagIni=false;
}
portSelect::~portSelect()
{
    delete ui;
}
void portSelect::setParameter(int port,int sample, string fileName)
{
    ui->sampleRate->setValue(sample);
    ui->spinBoxPort->setValue(port);
    if(port==0)
    {
        this->flagIni=true;
        ui->checkBoxArquivo->setChecked(true);
        this->flagIni=false;
        ui->lineEditFileName->setEnabled(true);
        ui->spinBoxPort->setEnabled(false);
        ui->lineEditFileName->setText(QString(fileName.c_str()));
    }
}
void portSelect::on_buttonBox_accepted()
{
    this->sampleFreq = ui->sampleRate->value();
    this->port=ui->spinBoxPort->value();
    if (ui->checkBoxArquivo->isChecked())
    {
        this->port=0;
        this->fileName = ui->lineEditFileName->text().toStdString();
    }
}
void portSelect::on_checkBoxArquivo_toggled(bool checked)
{
```

```
    if (this->flagIni)
        return;
    if (checked)
    {
        QString name = QFileDialog::getOpenFileName(this,"Escolha um arquivo com os
dados",".","texto (*.dat)");
        if(!name.isEmpty())
        {
            ui->lineEditFileName->setText(name);
            ui->lineEditFileName->setEnabled(true);
        }
        else
        {
            ui->checkBoxArquivo->setChecked(false);
            ui->lineEditFileName->setEnabled(false);
        }
        ui->spinBoxPort->setEnabled(false);
    }
    else
    {
        ui->lineEditFileName->setEnabled(false);
        ui->spinBoxPort->setEnabled(true);
    }
}
                          //-------------------------------------------------------------------//
#ifndef PORTSELECT_H
#define PORTSELECT_H
#include <iostream>
#include <fstream>
using namespace std;
#include <QDialog>

namespace Ui {
    class portSelect;
}
class portSelect : public QDialog
{
    Q_OBJECT

public:
    explicit portSelect(QWidget *parent = 0);
    ~portSelect();
    int getPort() { return port;}
    int getSampleFreq() { return sampleFreq;}
    string getFileName() { return fileName;}
    void setParameter(int port,int sample,string fileName);

private:
    Ui::portSelect *ui;
    int port;
    string fileName;
    int sampleFreq;
    bool flagIni;       // Flag que indica a inicializacao da janela.

private slots:
    void on_checkBoxArquivo_toggled(bool checked);
    void on_buttonBox_accepted();
};
#endif // PORTSELECT_H
                          //-------------------------------------------------------------------//
#include "projectview.h"
#include <QDebug>
#include <QtGui>
projectView::projectView()
{
    this->plateSizeFootX=10;
```

```
    this->plateSizeFootY=10;
}
void projectView::paintEvent(QPaintEvent *event)
{
    QPainter estojo;
    QPen penCalda;
    int lx=this->geometry().width();
    int ly=this->geometry().height();
    double fx=lx/90.0; //this->plateSizeFootX;
    double fy=ly/180.0; //this->plateSizeFootY;
    estojo.begin(this);
    estojo.setBrush(Qt::black);
    estojo.drawRect(0,0,lx,ly);
    estojo.setBrush(Qt::blue);
    estojo.setPen(Qt::green);
    estojo.drawLine(lx/2,0,lx/2,ly);
    estojo.drawLine(0,ly/2,lx,ly/2);
    int tam=this->vX.size();
    if (tam==0) return;
    estojo.drawEllipse((this->vX[tam-1])*fx+lx/2-2,ly/2-((this->vY[tam-1])*fy+2),5,5);
    penCalda.setWidth(2);
    penCalda.setCapStyle(Qt::RoundCap);
    penCalda.setJoinStyle(Qt::RoundJoin);
    for(int i=0;i<(int)tam-2;i++)
    {
        penCalda.setBrush(QColor(0,i*255.0/tam,0));
        estojo.setPen(penCalda);
        estojo.drawLine((this->vX[i])*fx+lx/2,ly/2-(this->vY[i])*fy,
                        (this->vX[i+1])*fx+lx/2,ly/2-(this->vY[i+1])*fy);
    }
    estojo.setPen(Qt::green);
    estojo.drawLine(lx/2,0,lx/2,ly);
    estojo.drawLine(0,ly/2,lx,ly/2);
    estojo.end();
}
void projectView::setProj(double x,double y)
{

    this->vX.push_back(x);
    this->vY.push_back(y);
    while(this->vX.size()>this->rastro)
    {
        this->vX.erase(this->vX.begin());
        this->vY.erase(this->vY.begin());
    }
}
                            //-------------------------------------------------------------//
#ifndef PROJECTVIEW_H
#define PROJECTVIEW_H
#include <QWidget>
#include <vector>
using namespace std;

class projectView : public QWidget
{
    Q_OBJECT
public:
    projectView();
    void setProj(double x,double y);
    int getPrjX(){return this->vX[0];}
    int getPrjY(){return this->vY[0];}
    void set_rastro(int r){ this->rastro = r;}
private:
    vector <double> vX;
    vector <double> vY;
    double plateSizeFootX;      // largura de um pe da plataforma
```

```
    double plateSizeFootY;
    int rastro;
protected:
    void paintEvent(QPaintEvent* event);
};

#endif // PROJECTVIEW_H
                                //--------------------------------------------------------------//
/********************************************************************************
** Form generated from reading UI file 'interface.ui'
** Created: Fri 30. Sep 13:37:12 2011
**      by: Qt User Interface Compiler version 4.7.3
** WARNING! All changes made in this file will be lost when recompiling UI file!
********************************************************************************/
#ifndef UI_INTERFACE_H
#define UI_INTERFACE_H
#include <QtCore/QVariant>
#include <QtGui/QAction>
#include <QtGui/QApplication>
#include <QtGui/QButtonGroup>
#include <QtGui/QHBoxLayout>
#include <QtGui/QHeaderView>
#include <QtGui/QMainWindow>
#include <QtGui/QMenu>
#include <QtGui/QMenuBar>
#include <QtGui/QPushButton>
#include <QtGui/QSlider>
#include <QtGui/QSpacerItem>
#include <QtGui/QSpinBox>
#include <QtGui/QStatusBar>
#include <QtGui/QTabWidget>
#include <QtGui/QTextEdit>
#include <QtGui/QToolBar>
#include <QtGui/QVBoxLayout>
#include <QtGui/QWidget>
QT_BEGIN_NAMESPACE

class Ui_interface
{
public:
    QAction *actionP_direito;
    QAction *actionP_esquerdo;
    QAction *actionTotal;
    QAction *action_Conectar_Plataforma;
    QAction *actionC_alibrar;
    QAction *actionC_onfigurar;
    QWidget *centralWidget;
    QVBoxLayout *verticalLayout;
    QHBoxLayout *horizontalLayoutGraphs;
    QTabWidget *SaidaTabWidget;
    QWidget *tab;
    QVBoxLayout *verticalLayout_3;
    QTextEdit *ProjecaoTextEdit;
    QWidget *tab_2;
    QVBoxLayout *verticalLayout_4;
    QTextEdit *CelulasTextEdit;
    QVBoxLayout *verticalLayout_2;
    QSpinBox *CaldaSpinBox;
    QSlider *CaldaVerticalSlider;
    QHBoxLayout *horizontalLayout_2;
    QPushButton *conectarPushButton;
    QPushButton *calibrarPushButton;
    QPushButton *pushButtonInitCap;
    QPushButton *pushButtonRecord;
    QSpacerItem *horizontalSpacer;
    QMenuBar *menuBar;
```

```
    QMenu *menuIniciar;
    QMenu *menuVer;
    QToolBar *mainToolBar;
    QStatusBar *statusBar;

    void setupUi(QMainWindow *interface)
    {
        if (interface->objectName().isEmpty())
            interface->setObjectName(QString::fromUtf8("interface"));
        interface->resize(522, 435);
        actionP_direito = new QAction(interface);
        actionP_direito->setObjectName(QString::fromUtf8("actionP_direito"));
        actionP_direito->setCheckable(true);
        actionP_esquerdo = new QAction(interface);
        actionP_esquerdo->setObjectName(QString::fromUtf8("actionP_esquerdo"));
        actionP_esquerdo->setCheckable(true);
        actionTotal = new QAction(interface);
        actionTotal->setObjectName(QString::fromUtf8("actionTotal"));
        actionTotal->setCheckable(true);
        actionTotal->setChecked(true);
        action_Conectar_Plataforma = new QAction(interface);
        action_Conectar_Plataforma-
>setObjectName(QString::fromUtf8("action_Conectar_Plataforma"));
        actionC_alibrar = new QAction(interface);
        actionC_alibrar->setObjectName(QString::fromUtf8("actionC_alibrar"));
        actionC_onfigurar = new QAction(interface);
        actionC_onfigurar->setObjectName(QString::fromUtf8("actionC_onfigurar"));
        centralWidget = new QWidget(interface);
        centralWidget->setObjectName(QString::fromUtf8("centralWidget"));
        verticalLayout = new QVBoxLayout(centralWidget);
        verticalLayout->setSpacing(6);
        verticalLayout->setContentsMargins(11, 11, 11, 11);
        verticalLayout->setObjectName(QString::fromUtf8("verticalLayout"));
        horizontalLayoutGraphs = new QHBoxLayout();
        horizontalLayoutGraphs->setSpacing(6);
        horizontalLayoutGraphs->setObjectName(QString::fromUtf8("horizontalLayoutGraphs"));
        SaidaTabWidget = new QTabWidget(centralWidget);
        SaidaTabWidget->setObjectName(QString::fromUtf8("SaidaTabWidget"));
        tab = new QWidget();
        tab->setObjectName(QString::fromUtf8("tab"));
        verticalLayout_3 = new QVBoxLayout(tab);
        verticalLayout_3->setSpacing(6);
        verticalLayout_3->setContentsMargins(11, 11, 11, 11);
        verticalLayout_3->setObjectName(QString::fromUtf8("verticalLayout_3"));
        ProjecaoTextEdit = new QTextEdit(tab);
        ProjecaoTextEdit->setObjectName(QString::fromUtf8("ProjecaoTextEdit"));
        verticalLayout_3->addWidget(ProjecaoTextEdit);
        SaidaTabWidget->addTab(tab, QString());
        tab_2 = new QWidget();
        tab_2->setObjectName(QString::fromUtf8("tab_2"));
        verticalLayout_4 = new QVBoxLayout(tab_2);
        verticalLayout_4->setSpacing(6);
        verticalLayout_4->setContentsMargins(11, 11, 11, 11);
        verticalLayout_4->setObjectName(QString::fromUtf8("verticalLayout_4"));
        CelulasTextEdit = new QTextEdit(tab_2);
        CelulasTextEdit->setObjectName(QString::fromUtf8("CelulasTextEdit"));
        verticalLayout_4->addWidget(CelulasTextEdit);
        SaidaTabWidget->addTab(tab_2, QString());
        horizontalLayoutGraphs->addWidget(SaidaTabWidget);
        verticalLayout_2 = new QVBoxLayout();
        verticalLayout_2->setSpacing(6);
        verticalLayout_2->setObjectName(QString::fromUtf8("verticalLayout_2"));
        CaldaSpinBox = new QSpinBox(centralWidget);
        CaldaSpinBox->setObjectName(QString::fromUtf8("CaldaSpinBox"));
        CaldaSpinBox->setMinimum(1);
        CaldaSpinBox->setMaximum(99999);
```

```
        verticalLayout_2->addWidget(CaldaSpinBox);
        CaldaVerticalSlider = new QSlider(centralWidget);
        CaldaVerticalSlider->setObjectName(QString::fromUtf8("CaldaVerticalSlider"));
        CaldaVerticalSlider->setMinimum(1);
        CaldaVerticalSlider->setOrientation(Qt::Vertical);
        verticalLayout_2->addWidget(CaldaVerticalSlider);
        horizontalLayoutGraphs->addLayout(verticalLayout_2);
        verticalLayout->addLayout(horizontalLayoutGraphs);
        horizontalLayout_2 = new QHBoxLayout();
        horizontalLayout_2->setSpacing(6);
        horizontalLayout_2->setObjectName(QString::fromUtf8("horizontalLayout_2"));
        conectarPushButton = new QPushButton(centralWidget);
        conectarPushButton->setObjectName(QString::fromUtf8("conectarPushButton"));
        horizontalLayout_2->addWidget(conectarPushButton);
        calibrarPushButton = new QPushButton(centralWidget);
        calibrarPushButton->setObjectName(QString::fromUtf8("calibrarPushButton"));
        calibrarPushButton->setEnabled(false);
        horizontalLayout_2->addWidget(calibrarPushButton);
        pushButtonInitCap = new QPushButton(centralWidget);
        pushButtonInitCap->setObjectName(QString::fromUtf8("pushButtonInitCap"));
        pushButtonInitCap->setEnabled(false);
        horizontalLayout_2->addWidget(pushButtonInitCap);
        pushButtonRecord = new QPushButton(centralWidget);
        pushButtonRecord->setObjectName(QString::fromUtf8("pushButtonRecord"));
        pushButtonRecord->setEnabled(false);

        horizontalLayout_2->addWidget(pushButtonRecord);
        horizontalSpacer = new QSpacerItem(40, 20, QSizePolicy::Expanding,
QSizePolicy::Minimum);
        horizontalLayout_2->addItem(horizontalSpacer);
        verticalLayout->addLayout(horizontalLayout_2);

        interface->setCentralWidget(centralWidget);
        menuBar = new QMenuBar(interface);
        menuBar->setObjectName(QString::fromUtf8("menuBar"));
        menuBar->setGeometry(QRect(0, 0, 522, 21));
        menuIniciar = new QMenu(menuBar);
        menuIniciar->setObjectName(QString::fromUtf8("menuIniciar"));
        menuVer = new QMenu(menuBar);
        menuVer->setObjectName(QString::fromUtf8("menuVer"));
        interface->setMenuBar(menuBar);
        mainToolBar = new QToolBar(interface);
        mainToolBar->setObjectName(QString::fromUtf8("mainToolBar"));
        interface->addToolBar(Qt::TopToolBarArea, mainToolBar);
        statusBar = new QStatusBar(interface);
        statusBar->setObjectName(QString::fromUtf8("statusBar"));
        interface->setStatusBar(statusBar);
        menuBar->addAction(menuIniciar->menuAction());
        menuBar->addAction(menuVer->menuAction());
        menuIniciar->addAction(action_Conectar_Plataforma);
        menuIniciar->addAction(actionC_alibrar);
        menuIniciar->addAction(actionC_onfigurar);
        menuVer->addAction(actionP_direito);
        menuVer->addAction(actionP_esquerdo);
        menuVer->addAction(actionTotal);

        retranslateUi(interface);
        QObject::connect(CaldaVerticalSlider, SIGNAL(sliderMoved(int)), CaldaSpinBox,
SLOT(setValue(int)));
        QObject::connect(CaldaSpinBox, SIGNAL(valueChanged(int)), CaldaVerticalSlider,
SLOT(setValue(int)));

        SaidaTabWidget->setCurrentIndex(0);

        QMetaObject::connectSlotsByName(interface);
    } // setupUi
```

```
    void retranslateUi(QMainWindow *interface)
    {
        interface->setWindowTitle(QApplication::translate("interface", "PlMob", 0,
QApplication::UnicodeUTF8));
        actionP_direito->setText(QApplication::translate("interface", "P\303\251 &direito",
0, QApplication::UnicodeUTF8));
        actionP_esquerdo->setText(QApplication::translate("interface", "P\303\251
&esquerdo", 0, QApplication::UnicodeUTF8));
        actionTotal->setText(QApplication::translate("interface", "Total", 0,
QApplication::UnicodeUTF8));
        action_Conectar_Plataforma->setText(QApplication::translate("interface",
"&Conectar", 0, QApplication::UnicodeUTF8));
        actionC_alibrar->setText(QApplication::translate("interface", "C&alibrar", 0,
QApplication::UnicodeUTF8));
        actionC_onfigurar->setText(QApplication::translate("interface", "C&onfigurar", 0,
QApplication::UnicodeUTF8));
        SaidaTabWidget->setTabText(SaidaTabWidget->indexOf(tab),
QApplication::translate("interface", "Proje\303\247\303\243o", 0,
QApplication::UnicodeUTF8));
        SaidaTabWidget->setTabText(SaidaTabWidget->indexOf(tab_2),
QApplication::translate("interface", "Dados C\303\251lulas", 0,
QApplication::UnicodeUTF8));
        conectarPushButton->setText(QApplication::translate("interface", "Conectar", 0,
QApplication::UnicodeUTF8));
        calibrarPushButton->setText(QApplication::translate("interface", "Calibrar", 0,
QApplication::UnicodeUTF8));
        pushButtonInitCap->setText(QApplication::translate("interface", "Inicio Captura",
0, QApplication::UnicodeUTF8));
        pushButtonRecord->setText(QApplication::translate("interface", "Gravar", 0,
QApplication::UnicodeUTF8));
        menuIniciar->setTitle(QApplication::translate("interface", "&Plataforma", 0,
QApplication::UnicodeUTF8));
        menuVer->setTitle(QApplication::translate("interface", "Proje\303\247\303\265es",
0, QApplication::UnicodeUTF8));
    } // retranslateUi
};
namespace Ui {
    class interface: public Ui_interface {};
} // namespace Ui
QT_END_NAMESPACE
#endif // UI_INTERFACE_H

/********************************************************************************
** Form generated from reading UI file 'portselect.ui'
** Created: Thu 29. Sep 10:32:56 2011
**      by: Qt User Interface Compiler version 4.7.3
** WARNING! All changes made in this file will be lost when recompiling UI file!
********************************************************************************/
#ifndef UI_PORTSELECT_H
#define UI_PORTSELECT_H
#include <QtCore/QVariant>
#include <QtGui/QAction>
#include <QtGui/QApplication>
#include <QtGui/QButtonGroup>
#include <QtGui/QCheckBox>
#include <QtGui/QDialog>
#include <QtGui/QDialogButtonBox>
#include <QtGui/QGroupBox>
#include <QtGui/QHBoxLayout>
#include <QtGui/QHeaderView>
#include <QtGui/QLabel>
#include <QtGui/QLineEdit>
#include <QtGui/QSpacerItem>
#include <QtGui/QSpinBox>
#include <QtGui/QTabWidget>
```

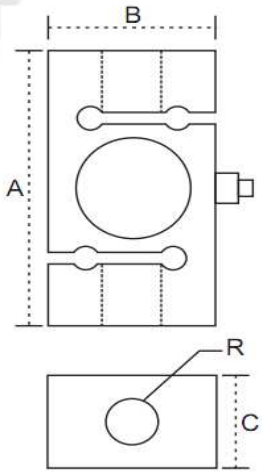
```
#include <QtGui/QVBoxLayout>
#include <QtGui/QWidget>
QT_BEGIN_NAMESPACE
class Ui_portSelect
{
public:
    QVBoxLayout *verticalLayout_2;
    QTabWidget *tabConfiguration;
    QWidget *port;
    QVBoxLayout *verticalLayout_3;
    QGroupBox *groupBox;
    QVBoxLayout *verticalLayout_4;
    QHBoxLayout *horizontalLayout_4;
    QLabel *label_3;
    QSpinBox *spinBoxPort;
    QSpacerItem *horizontalSpacer_3;
    QHBoxLayout *horizontalLayout_3;
    QCheckBox *checkBoxArquivo;
    QLineEdit *lineEditFileName;
    QSpacerItem *horizontalSpacer_2;
    QWidget *sample;
    QVBoxLayout *verticalLayout;
    QHBoxLayout *horizontalLayout_2;
    QLabel *label_2;
    QSpinBox *sampleRate;
    QDialogButtonBox *buttonBox;
    void setupUi(QDialog *portSelect)
    {
        if (portSelect->objectName().isEmpty())
            portSelect->setObjectName(QString::fromUtf8("portSelect"));
        portSelect->resize(258, 224);
        verticalLayout_2 = new QVBoxLayout(portSelect);
        verticalLayout_2->setObjectName(QString::fromUtf8("verticalLayout_2"));
        tabConfiguration = new QTabWidget(portSelect);
        tabConfiguration->setObjectName(QString::fromUtf8("tabConfiguration"));
        port = new QWidget();
        port->setObjectName(QString::fromUtf8("port"));
        verticalLayout_3 = new QVBoxLayout(port);
        verticalLayout_3->setObjectName(QString::fromUtf8("verticalLayout_3"));
        groupBox = new QGroupBox(port);
        groupBox->setObjectName(QString::fromUtf8("groupBox"));
        verticalLayout_4 = new QVBoxLayout(groupBox);
        verticalLayout_4->setObjectName(QString::fromUtf8("verticalLayout_4"));
        horizontalLayout_4 = new QHBoxLayout();
        horizontalLayout_4->setObjectName(QString::fromUtf8("horizontalLayout_4"));
        label_3 = new QLabel(groupBox);
        label_3->setObjectName(QString::fromUtf8("label_3"));
        label_3->setAlignment(Qt::AlignRight|Qt::AlignTrailing|Qt::AlignVCenter);
        horizontalLayout_4->addWidget(label_3);
        spinBoxPort = new QSpinBox(groupBox);
        spinBoxPort->setObjectName(QString::fromUtf8("spinBoxPort"));
        spinBoxPort->setMinimum(1);
        horizontalLayout_4->addWidget(spinBoxPort);

        horizontalSpacer_3 = new QSpacerItem(40, 20, QSizePolicy::Expanding,
QSizePolicy::Minimum);
        horizontalLayout_4->addItem(horizontalSpacer_3);
        verticalLayout_4->addLayout(horizontalLayout_4);
        horizontalLayout_3 = new QHBoxLayout();
        horizontalLayout_3->setObjectName(QString::fromUtf8("horizontalLayout_3"));
        checkBoxArquivo = new QCheckBox(groupBox);
        checkBoxArquivo->setObjectName(QString::fromUtf8("checkBoxArquivo"));
        horizontalLayout_3->addWidget(checkBoxArquivo);
        lineEditFileName = new QLineEdit(groupBox);
        lineEditFileName->setObjectName(QString::fromUtf8("lineEditFileName"));
        lineEditFileName->setEnabled(false);
```

```
        horizontalLayout_3->addWidget(lineEditFileName);
        horizontalSpacer_2 = new QSpacerItem(40, 20, QSizePolicy::Expanding,
QSizePolicy::Minimum);
        horizontalLayout_3->addItem(horizontalSpacer_2);
        verticalLayout_4->addLayout(horizontalLayout_3);
        verticalLayout_3->addWidget(groupBox);
        tabConfiguration->addTab(port, QString());
        sample = new QWidget();
        sample->setObjectName(QString::fromUtf8("sample"));
        verticalLayout = new QVBoxLayout(sample);
        verticalLayout->setObjectName(QString::fromUtf8("verticalLayout"));
        horizontalLayout_2 = new QHBoxLayout();
        horizontalLayout_2->setObjectName(QString::fromUtf8("horizontalLayout_2"));
        label_2 = new QLabel(sample);
        label_2->setObjectName(QString::fromUtf8("label_2"));
        horizontalLayout_2->addWidget(label_2);
        sampleRate = new QSpinBox(sample);
        sampleRate->setObjectName(QString::fromUtf8("sampleRate"));
        sampleRate->setMaximum(999999);
        horizontalLayout_2->addWidget(sampleRate);
        verticalLayout->addLayout(horizontalLayout_2);
        tabConfiguration->addTab(sample, QString());
        verticalLayout_2->addWidget(tabConfiguration);
        buttonBox = new QDialogButtonBox(portSelect);
        buttonBox->setObjectName(QString::fromUtf8("buttonBox"));
        buttonBox->setOrientation(Qt::Horizontal);
        buttonBox->setStandardButtons(QDialogButtonBox::Cancel|QDialogButtonBox::Ok);
        verticalLayout_2->addWidget(buttonBox);
        retranslateUi(portSelect);
        QObject::connect(buttonBox, SIGNAL(accepted()), portSelect, SLOT(accept()));
        QObject::connect(buttonBox, SIGNAL(rejected()), portSelect, SLOT(reject()));
        tabConfiguration->setCurrentIndex(0);
        QMetaObject::connectSlotsByName(portSelect);
    } // setupUi
    void retranslateUi(QDialog *portSelect)
    {
        portSelect->setWindowTitle(QApplication::translate("portSelect", "Dialog", 0,
QApplication::UnicodeUTF8));
        groupBox->setTitle(QApplication::translate("portSelect", "Entrada de dados", 0,
QApplication::UnicodeUTF8));
        label_3->setText(QApplication::translate("portSelect", "Porta COM", 0,
QApplication::UnicodeUTF8));
        checkBoxArquivo->setText(QApplication::translate("portSelect", "Arquivo", 0,
QApplication::UnicodeUTF8));
        tabConfiguration->setTabText(tabConfiguration->indexOf(port),
QApplication::translate("portSelect", "Porta", 0, QApplication::UnicodeUTF8));
        label_2->setText(QApplication::translate("portSelect", "Intervalo de Amostagem
(ms)", 0, QApplication::UnicodeUTF8));
        tabConfiguration->setTabText(tabConfiguration->indexOf(sample),
QApplication::translate("portSelect", "Amostragem", 0, QApplication::UnicodeUTF8));
    } // retranslateUi
};

namespace Ui {
    class portSelect: public Ui_portSelect {};
} // namespace Ui

QT_END_NAMESPACE

#endif // UI_PORTSELECT_H
```

Anexo 1 – Datasheet Célula de carga CS50

CS 50/100/200 kg

Capacidades Nominais: 50, 100 e 200 kg
Sensibilidade: 2mV/V
Material: Alumínio
Cabo: 5 metros

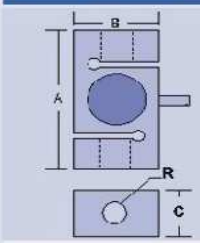


APLICAÇÃO

- Conversão de balanças mecânicas p/ eletrônicas.
- Pesagem à tração e compressão.
- Máquinas de teste e ensaios de materiais.
- Testes de potência e ensaios de motores.
- Uso geral onde requer medida de esforços de tração e compressão.

CARACTERÍSTICAS TÉCNICAS

CÉLULAS DE CARGA MOD. CS



MODELO CS	Med.	A	B	C	R	CABO
50Kg	mm	60	32	20	M 6x1	3.0 m
100Kg	mm	60	32	24	M 12x1.75	3.0 m
200Kg	mm	60	32	32	M 12x1.75	3.0 m

Capacidade nominal	50,100,200 kg
Sensibilidade da célula de carga em MV/V	2mV/V +/- 0,1 %
Classe C3	3000 divisões
Creep à carga total aplicada	20 minutos : <0,03 6hs:<0,05
Zero inicial saída nominal	+/- 1%
Temperatura de teste 0C	-5 a + 50 graus
Temperatura compensada 0C	0 a + 50
Efeito da temperatura	compensada
Sobrecarga sem danos, capacidade nominal	150
Erro combinado % saída nominal	<0,03
Sobrecarga de ruptura	300 % da capacidade
Deflexão máxima mm a capacidade nominal	<1,0mm
Tensão de Excitação VCC ou VCA	máxima:15 recomendado:10
Impedância de saída em ohms	350 +/- 1
Impedância de entrada em ohms	400 +/-15
Resistência de isolação	>5000 mega ohms
Material da célula de carga	Alumínio anodizado
Grau de proteção	IP67

Ivanoé João Rodowanski

ivanoeh@gmail.com