



CENTRO UNIVERSITÁRIO LUTERANO DE PALMAS

COMUNIDADE EVANGÉLICA LUTERANA "SÃO PAULO"
Recredenciado pela Portaria Ministerial nº 3.607 - D.O.U. nº 202 de 20/10/2005

Marlon David Domingos

AVALIAÇÃO POSTURAL COMPUTADORIZADA UTILIZANDO O KINECT

Palmas - TO

2014

Marlon David Domingos

**AVALIAÇÃO POSTURAL COMPUTADORIZADA UTILIZANDO O
KINECT**

Trabalho de Conclusão de Curso (TCC) elaborado e apresentado como requisito parcial para obtenção do título de bacharel em Sistemas de Informação pelo Centro Universitário Luterano de Palmas (CEULP/ULBRA).

Orientador: Prof. M.Sc. Fernando Luiz de Oliveira

Palmas – TO

2014

Marlon David Domingos

AVALIAÇÃO POSTURAL COMPUTADORIZADA UTILIZANDO O KINECT

Trabalho de Conclusão de Curso (TCC) elaborado e apresentado como requisito parcial para obtenção do título de bacharel em Sistemas de Informação pelo Centro Universitário Luterano de Palmas (CEULP/ULBRA).

Orientador: Prof. M.Sc. Fernando Luiz de Oliveira

Aprovada em ____/____/____ de 2014.

BANCA EXAMINADORA

Prof. M.Sc. Fernando Luiz de Oliveira
Centro Universitário Luterano de Palmas - CEULP

Prof. M.Sc. Fabiano Fagundes
Centro Universitário Luterano de Palmas - CEULP

Prof. M.Sc. Pierre Soares Brandão
Centro Universitário Luterano de Palmas - CEULP

Palmas – TO

2014

Dedico este trabalho aos meus avós paternos e maternos: Arlindo Domingos, Júlia Venâncio, Maria Aparecida Salviano Guilherme e Orizon David Guilherme; por todos os ensinamentos e valores ensinados, foram exemplos que me seguiram por toda a minha vida.

AGRADECIMENTOS

Agradeço primeiramente a Deus por me conceder o dom da vida. Agradeço a toda a minha família, minha mãe Áurea David Salviano que tanto me ajudou nas horas difíceis, agradeço meu irmão Alexandre David Domingos que sempre me inspirou nos estudos e nunca se ateu em me ajudar quando precisei, até para tirar dúvidas até tarde da noite sempre esteve disposto, a minha irmã Vanessa David Domingos por tudo quanto me ajudou. Agradeço a minha namorada Blenda Costa pelo incentivo e compreensão aos momentos de ausência devido a dedicação a este trabalho. Agradeço também ao meu pai Otacílio Domingos, que sempre me apoiou em várias escolhas da minha vida. Agradeço ao Pr. Wagner Machado pelos grandes ensinamentos e palavras de conforto em momentos difíceis. Agradeço ao Professorº Fernando Luiz de Oliveira, por me orientar neste trabalhado e ter contribuído de forma significativa. Agradeço também ao Professorº Pierre Soares Brandão pelas orientações na área da fisioterapia, dando dicas e conceitos.

RESUMO.

A escoliose é uma deformidade que provoca uma tortuosidade na coluna vertebral. Esta deformidade, assim como qualquer outro tipo de doença, necessita ser diagnosticado o mais breve possível para que se obtenha sucesso no tratamento. Os métodos mais precisos para diagnosticar a escoliose são os que se utilizam de raios-X, porém seu uso demanda um custo maior e impõe limitações ao tempo de exposição à radiação. Utilizando imagens e medidas externas do corpo humano ainda não é possível mensurar com a mesma precisão obtida pelos métodos radiológicos. Sendo assim, este trabalho teve por objetivo desenvolver uma ferramenta que aproxime os resultados obtidos pelos métodos mais precisos, com a finalidade de executar triagem e o acompanhamento durante o tratamento expondo a evolução do paciente. Assim esta ferramenta implementa duas formas de avaliar a escoliose, sendo pela imagem de profundidade buscando o relevo das costas ou por dados inferidos pela SDK Kinect de pontos de articulação do corpo humano.

PALAVRAS-CHAVE: Escoliose, *Kinect*, Avaliação.

LISTA DE FIGURAS

Figura 1 - Ângulo de Cobb.....	10
Figura 2 - Escoliose e gibosidade	11
Figura 3 - Visão lateral da coluna vertebral	11
Figura 4 - Classificação de KING.....	12
Figura 5 - Postura padrão proposta por Kendall.....	15
Figura 6 - Transferidor para medir o ângulo de Coob	16
Figura 7 - Régua de Raimondi.....	18
Figura 8 - Régua fabricada por Ferreira et al para medir gibosidade	19
Figura 9 - Teste de Adams para mensurar gibosidade.....	19
Figura 10 - Protocolo de marcação do sistema SAPO.....	21
Figura 11 - Avaliação no sistema SAPO.....	22
Figura 12 - Exemplo de franja de moiré.	23
Figura 13 - Curvas de níveis topográficos.	24
Figura 14 - Técnica moiré de sombra.	25
Figura 15 - Moiré de sombra em reconhecimento facial.	26
Figura 16 - Topografia de moiré para mensurar escoliose.....	26
Figura 17 - Processo para gerar o mapa de apontamento.....	29
Figura 18 – Processo para gerar mapa de apontamento recebido.....	30
Figura 19 – Processo para gerar o mapa de fluxo acumulado	30
Figura 20 - Imagem de profundidade do Kinect.....	31
Figura 21 - Visão frontal do sensor Kinect sem capa	33
Figura 22 - Arquitetura física do Kinect	34
Figura 23 - Arquitetura SDK Kinect.....	36
Figura 24 - Habilitar fluxo de imagem colorida	37
Figura 25 - Método para capturar um quadro de imagem.....	38

Figura 26 - Manipulador de evento para fluxo de imagem colorida	38
Figura 27 - Diagrama de classe do fluxo de imagem colorida.	39
Figura 28 - Habilitando <i>SkeletonStream</i>	40
Figura 29 - Esqueleto proposto pela SDK Kinect	41
Figura 30 - Classes que representam o fluxo do esqueleto.	42
Figura 31 - Eixo 3D do esqueleto.	43
Figura 32 - Manipulador de evento para o fluxo do esqueleto.....	43
Figura 33 - Área de alcance do Kinect.....	45
Figura 34 - Habilitar e capturar fluxo de imagem de profundidade.	46
Figura 35 - Classes que representam o fluxo de imagem de profundidade.....	47
Figura 36 - Ângulos de inclinação dos ombros e cabeça.	50
Figura 37 – Arquitetura da ferramenta desenvolvida.	51
Figura 38 – Tela da ferramenta apresentando a avaliação por esqueleto.....	53
Figura 39 – Diagrama de classes.	54
Figura 40 – Método para converter coordenadas das articulações.	54
Figura 41 – Método para obter o esqueleto.....	55
Figura 42 – Método para representar o esqueleto graficamente.....	55
Figura 43 – Método para calcular ângulo.....	56
Figura 44 - Método para converter índice.....	57
Figura 45 – Método <i>SearchColuna</i>	57
Figura 46 - Método de mapa de direção.....	58
Figura 47 – Método para o mapa de apontamento	59
Figura 48 – Método <i>FluxoAcumulado</i>	60
Figura 49 - Sistema gerando fluxo acumulado	61
Figura 50 - Pixels do usuário em escala de cinza	61

LISTA DE TABELAS

Tabela 1 - Diagrama da classificação de Lenke.	13
Tabela 2 - Determinação da rotação vertebral pelo método de NASH & MOE.	17

LISTA DE ABREVIATURAS

CEULP	Centro Universitário Luterano de Palmas
CMOS	Semicondutor metal-óxido complementar
DLL	Biblioteca de vínculo dinâmico
SAPO	Sistema de Avaliação Postural
SDK	Kit de Desenvolvimento de Software
ULBRA	Universidade Luterana do Brasil

SUMÁRIO

1	INTRODUÇÃO	7
2	REFERENCIAL TEÓRICO	9
2.1	Escoliose.....	9
2.1.1	Diagnóstico da escoliose.....	10
2.1.2	Classificação.....	11
2.2	Avaliação Postural	14
2.2.1	Métodos por análise de imagem radiográfica	16
2.2.2	Métodos por análise de imagens digitais.....	18
2.3	Visão Computacional	27
2.3.1	Sentido do fluxo	29
2.4	Imagem de profundidade do Kinect	31
3	MATERIAIS E MÉTODOS.....	32
3.1	Local e Período	32
3.2	Hardware e Softwares.....	32
3.2.1	Kinect	32
3.2.2	Arquitetura do Kinect	33
3.2.3	SDK Kinect	34
3.3	Metodologia.....	47
4	RESULTADOS E DISCUSSÃO	49
4.1	Soluções propostas	49
4.2	Arquitetura da ferramenta	51
4.3	Diagrama de classes	53
5	CONSIDERAÇÕES FINAIS	63
6	REFERÊNCIAS BIBLIOGRÁFICAS.....	64

1 INTRODUÇÃO

A escoliose é uma deformidade que quando diagnosticada precocemente aumenta significativamente as chances de recuperação. Os métodos de avaliações mais confiáveis apontado pelos especialistas são os métodos por imagem radiológica. Porém, tais métodos não podem ser aplicados em curto espaço de tempo por causarem efeitos colaterais oriundos da radiação (FERREIRA et al, 2009).

Levando em consideração que a maior parte das incidências ocorre em adolescentes e que durante o tratamento é necessário efetuar constantes avaliações, muitos pesquisadores têm se empenhado em desenvolver métodos que possam ser utilizados de forma mais frequente. Os métodos por imagem digital não provocam nenhum efeito colateral e possuem baixos custos de aplicação. No entanto, a maior parte destes métodos ainda não possuem precisão e eficácia para substituir os métodos invasivos (DÖHNERT e TOMASI, 2008).

Dentro deste contexto, este projeto tem por objetivo desenvolver um sistema capaz de auxiliar fisioterapeutas na avaliação postural da escoliose utilizando o Microsoft Kinect. Este é um dispositivo dotado de câmeras e microfones que possibilita o usuário interagir com computadores e consoles de vídeo game a partir dos próprios movimentos do corpo. Devido a essa capacidade de reconhecer membros do corpo humano e gestos, ele possui recursos que podem propiciar a mensuração do corpo humano.

O Kinect possui uma SDK – *Software Development Kit*, ou seja, kit de desenvolvimento de software. Esta por sua vez possui bibliotecas de classes e métodos para desenvolvimento utilizando o Kinect. A SDK apresenta três fluxos de dados obtidos do usuário. Estes fluxos disponibilizam a aplicação dados de imagens coloridas, imagens de profundidade e um modelo de esqueleto proposto pela SDK.

Assim o uso deste equipamento abre diversas possibilidades de pesquisa no âmbito da fisioterapia. Estas informações disponibilizadas pelo sensor podem ser utilizadas para mensurar diversos aspectos do corpo humano, sendo possível assim mensurar e avaliar deformidades que acometem pessoas de várias idades, como é o caso da escoliose.

Desta forma este trabalho surge com este objetivo de buscar uma ferramenta que possa mensurar a deformidade causada pela escoliose. Assim possibilitando avaliações mais objetivas e com menor custo de aplicação. Com avaliações de menor custo e de rápida aplicação esta ferramenta poderá contribuir para que novas pesquisas no âmbito da fisioterapia possam ser realizadas em um grande numero de indivíduos de uma determinada população, para fins estatísticos. Além disto, esta ferramenta poderá realizar triagem em diversos pacientes encaminhando-os a exames mais minuciosos como a radiografia.

2 REFERENCIAL TEÓRICO

Esta seção apresenta os conceitos necessários para o desenvolvimento deste trabalho. Inicialmente serão abordados os conceitos e informações inerentes a deformidade denominada escoliose. Seguindo ao próximo tópico desta revisão será abordado como se dá a avaliação postural, ou seja, padrões de postura estabelecidos, bem como os métodos de avaliação sugeridos na literatura.

2.1 Escoliose

A escoliose trata-se de um desvio lateral no plano frontal e rotatório das vértebras da coluna no plano horizontal, que são dois dos três planos básicos de referências existentes, derivados das dimensões no espaço, sendo perpendiculares entre si. Ocorre devido a um movimento de torção generalizado de toda a coluna. Esse movimento é resultante de uma perturbação localizada que ocasiona uma ruptura do equilíbrio raquidiano (PEDRIOLLE, 1979).

Com o passar do tempo o conceito da escoliose mudou em alguns aspectos. Recentes pesquisas definem a escoliose como uma deformação na coluna vertebral em três dimensões: sendo no plano frontal favorecendo uma curvatura lateral, no transverso provocando uma rotação vertebral e no plano sagital ocasionando uma hiperlordose (FERREIRA et al, 2009).

A escoliose pode ter suas causas em doenças ou por problemas de ordem genética como má-formação. Quando possui causas específicas são chamadas de escolioses congênitas. Estas podem ocorrer devido a alterações do sistema nervoso, doenças dos músculos entre outras. A escoliose idiopática não possui causa conhecida e ocorre em crianças e adolescentes. Um fator que pode influenciar neste tipo de escoliose seria a fase de crescimento, podendo ocorrer de forma desordenada. A escoliose idiopática ainda pode ser subdividida em infantil, juvenil e do adolescente de acordo com a idade (KNOPLICH, 1989); (REEDER, 1981).

A escoliose idiopática infantil tem maior ocorrência no sexo masculino, este tipo de escoliose é detectada aos seis meses de idade. Possui baixa ocorrência, nos Estados Unidos foi estimada em 0,5% de todos os casos de escoliose idiopática. Normalmente desaparecem até os 3 anos de idade, sem a necessidade de tratamento. A escoliose idiopática juvenil representa cerca de 10 a 15% das escolioses idiopáticas. Já a escoliose idiopática do adolescente possui o maior número de ocorrências podendo chegar a mais de 80% dos casos diagnosticados. No Brasil a ocorrência da escoliose idiopática em adolescentes estima-se que esteja em torno 4,3% de todos os adolescentes (WICK et al, 2009); (SOUZA, 2013).

2.1.1 Diagnóstico da escoliose

O diagnóstico da escoliose se torna complexo devido à condição tridimensional da deformidade, assim a escoliose exige uma análise cuidadosa nos planos sagital e coronal. O avaliador poderá executar alguns testes e/ou exames por imagem radiológicos como: ressonância magnética, tomografia e o método de Cobb (WICK et al, 2009).

O grau de gravidade da escoliose normalmente é avaliado através do ângulo de Cobb, no plano coronal, calculado por meio de radiografias. O ângulo de Cobb é definido como o ângulo máximo entre duas linhas desenhadas paralelas às placas terminais das vértebras escolióticas no plano frontal (SALMINGO, et al, 2013). A figura 1 abaixo demonstra as linhas perpendiculares a vértebra limite superior e a vértebra limite inferior, que compreendem a curva, a vértebra ápice representa o ápice da curva da escoliose.

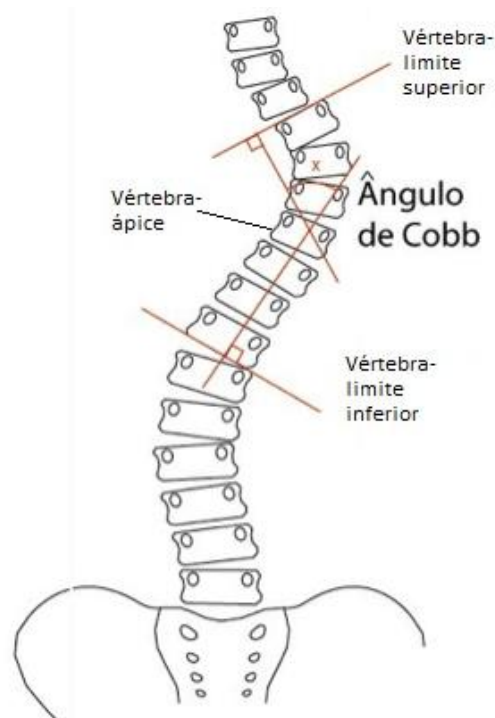


Figura 1 - Ângulo de Cobb

Fonte: Adaptada de <http://www.dryunes.com/doencas-da-coluna/escoliose>

A obtenção do ângulo de Cobb se dá ao traçar tangentes à vértebra limite superior e inferior; no cruzamento das perpendiculares das duas linhas, pode-se obter o ângulo de Cobb.

Nas escolioses estruturais ocorre a rotação das vértebras no plano transversal, esse componente rotacional é caracterizado pela presença de uma proeminência (gibosidade) no lado convexo da curva, com os corpos vertebrais rodados no sentido da convexidade. A gibosidade torácica aparece devido à rotação da caixa torácica (subjacente à rotação vertebral) e a gibosidade lombar ou a tóraco-lombar ocorre por aumento do volume e proeminência da

musculatura. Em ambos os casos, a gibosidade pode ser correlacionada com a magnitude da deformidade espinhal (FERREIRA et al, 2010). A figura 2, abaixo, ilustra a ocorrência de gibosidade na deformação da costela.



Figura 2 - Escoliose e gibosidade

Fonte: <http://www.institutocoluna.com.br/6escolioses.htm>

2.1.2 Classificação

A escoliose pode ser classificada conforme o número de curvas e sua localização. A figura 3, a seguir, ilustra as localizações da coluna vertebral que é dividida em: cervical, torácica, lombar, sacro e cóccix.

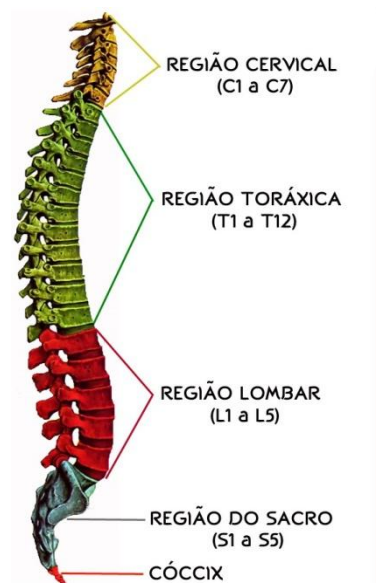


Figura 3 - Visão lateral da coluna vertebral

Fonte: Adaptada de <http://marciasymanowicz.com.br/site/o-que-as-vertebras-mal-posicionadas-podem-causar>

Os sistemas de classificação para escoliose têm como principais funções: categorizar, distinguir grupos de deformidades, auxiliar no prognóstico bem como o tratamento cirúrgico ou não cirúrgico (HEBELA e TORTONANI, 2009).

King et al (1983) propôs um sistema de classificação com cinco classificações de curvas da escoliose idiopática do adolescente, são eles: tipo I – curva em forma de “S” na qual a curva lombar é maior e menos flexível que a torácica quando se avaliam as radiografias em inclinação lateral; Tipo II – curva em forma “S” na qual a curva torácica é maior e menos flexível que a lombar; Tipo III - curva torácica isolada sem curva compensatória lombar; Tipo IV - curva torácica longa em que a quarta vértebra lombar está incluída; Tipo V - curva torácica dupla. Esta classificação é ilustrada na figura 4, a seguir.

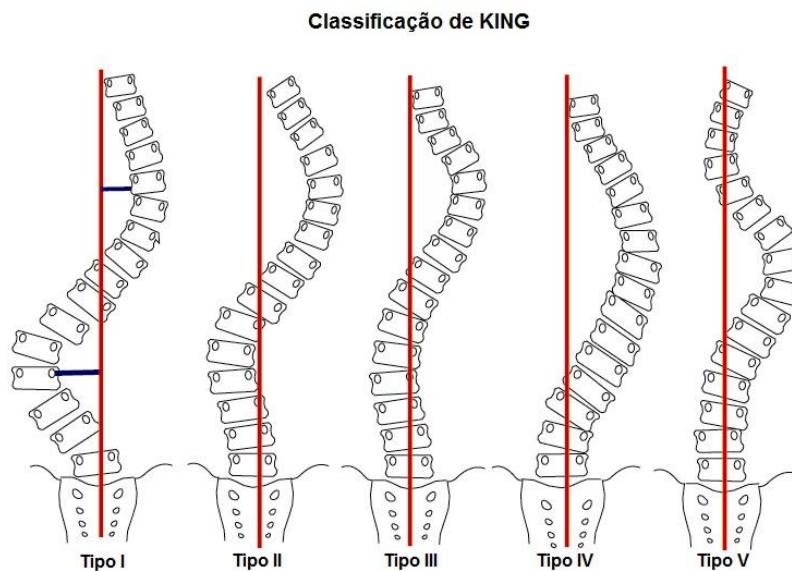


Figura 4 - Classificação de KING

Fonte: Adaptada de

http://www.drivanrocha.com.br/website/index.php?option=com_content&view=article&id=60&Itemid=86

Durante anos, a classificação de King foi o sistema amplamente utilizado para orientar o tratamento em escoliose idiopática do adolescente. Porém, a baixa confiabilidade da classificação de King indicou a necessidade de uma nova classificação mais abrangente considerando o conceito da escoliose ser uma deformação tridimensional (LENKE et al, 2001).

Assim, na classificação proposta por Lenk et al, a escoliose foi classificada de acordo com o tipo de curva (Tipos de 1 a 6), combinada com um modificador da coluna lombar (A, B, ou C) e um modificador torácica sagital (-, N ou +). A tabela 1, a seguir, exemplifica de

forma resumida a classificação de Lenke et al (2001).

Tabela 1 - Diagrama da classificação de Lenke.

Modificador Lombar	Tipo 1 (torácica principal)	Tipo 2 (dupla torácica)	Tipo 3 (dupla principal)	Tipo 4 (tripla principal)	Tipo 5 (TL/L)	Tipo 6 (TL/L-TP)
A (Nenhuma ou mínima curva)	1A*	2A*	3A*	4A*		
B (Curva moderada)	1B*	2B*	3B*	4B*		
C Curva grande	1C*	2C*	3C*	4C*	5C*	6C*
Critérios estruturais sagitais possíveis para determinação do tipo de curva	Normal	Cifose Torácica Proximal $\geq 20^\circ$	Cifose Toracolumbar $\geq 20^\circ$	Cifose TP+TL $\geq 20^\circ$	Modificador Sagital <ul style="list-style-type: none"> • (-) : $<10^\circ$ • (N) : $10-40^\circ$ • (+) : $>40^\circ$ 	

- Modificador do alinhamento sagital T5-T12: -, N, ou +

Fonte: traduzido e modificado de LENKE et al. (2001).

O tipo de curva é dado pela localização da vértebra que representa o ápice da curva e também a quantidade de curvas. Para definir o modificador lombar é traçada uma linha sacral central e localizar a vértebra estável. Portanto se a linha sacral central atravessar entre os pedículos das vértebras lombares terá modificador A, se a linha atravessar tangenciando o

lado côncavo da curva lombar terá um modificador B, caso a linha passe totalmente de forma lateral ao ápice da curva terá um modificador tipo C. E por fim para se definir o modificador sagital, deve-se aplicar o ângulo de Coob entre T5 e T12 na radiografia de perfil, e se for menor que 10° o modificador será (-), se o ângulo for entre 10-40° o modificador será (N), e caso seja maior que 40° o modificador será (+).

2.2 Avaliação Postural

Kendall et al (1995) citam como definição de postura um relato do Comitê de Postura da *American Academy of Orthopaedic Surgeons*, que define como o arranjo relativo das partes do corpo, de forma que o estado de equilíbrio muscular e esquelético que protege as estruturas de suporte do corpo contra lesão ou deformidade progressiva independentemente da ação do corpo.

A definição de postura segundo Knoplich (1998) trata-se da posição assumida pelo corpo que preenche as necessidades biomecânicas, com esforço muscular mínimo, objetivando a posição ereta do corpo.

É necessário manter a boa postura para que a estrutura e função do corpo proporcionem em sua totalidade as potencialidades do corpo. Os defeitos posturais que persistem podem dar origem a desconforto, dor e dependendo do grau de amplitude dos efeitos levarem a incapacidade mecânica corporal (KENDALL et al, 1995).

Algumas deformidades que alteram a boa postura, como a escoliose, apresenta um risco evolutivo. Assim, avaliar e mensurar torna-se indispensável, a fim de adaptar o tratamento (SOUCHARD e OLLIER, 2001).

Para avaliar a postura é necessário ter parâmetros de uma boa postura, a definição de alinhamento postural proposta por Kendall é a referência utilizada internacionalmente como padrão de postura normal e a fisioterapia considera como alteração da postura qualquer assimetria entre os segmentos corporais, e a avaliação é sistematicamente feita de modo qualitativo (FERREIRA, 2005).

A postura proposta por Kendall et al (1995), é denominada postura padrão, a qual a coluna apresenta as curvaturas normais e os ossos dos membros inferiores ficam em alinhamento ideal para sustentação do peso. A posição “neutra”, ou seja, sem inclinação, conduz ao bom alinhamento do abdômen e do tronco. O tórax e coluna superior ficam em uma posição que favorece a função ideal dos órgãos respiratórios. A cabeça fica ereta em uma posição bem equilibrada de forma que não sobrecarregue a musculatura cervical (KENDALL et al, 1995).

O teste do fio de prumo (que se trata de uma linha com um peso preso na ponta com a finalidade de determinar uma linha vertical absoluta), é usado para determinar se os pontos de referência da pessoa estão no mesmo alinhamento que os pontos correspondentes da postura padrão. A linha estabelecida na vertical coincide com as seguintes posições: levemente anterior ao maléolo lateral, levemente anterior ao eixo da articulação do joelho, levemente posterior ao eixo da articulação do quadril, corpos das vértebras lombares, articulação do ombro, corpos da maioria das vértebras cervicais, meato auditivo externo e levemente posterior ao ápice da sutura coronal, conforme demonstra a figura 5 abaixo.

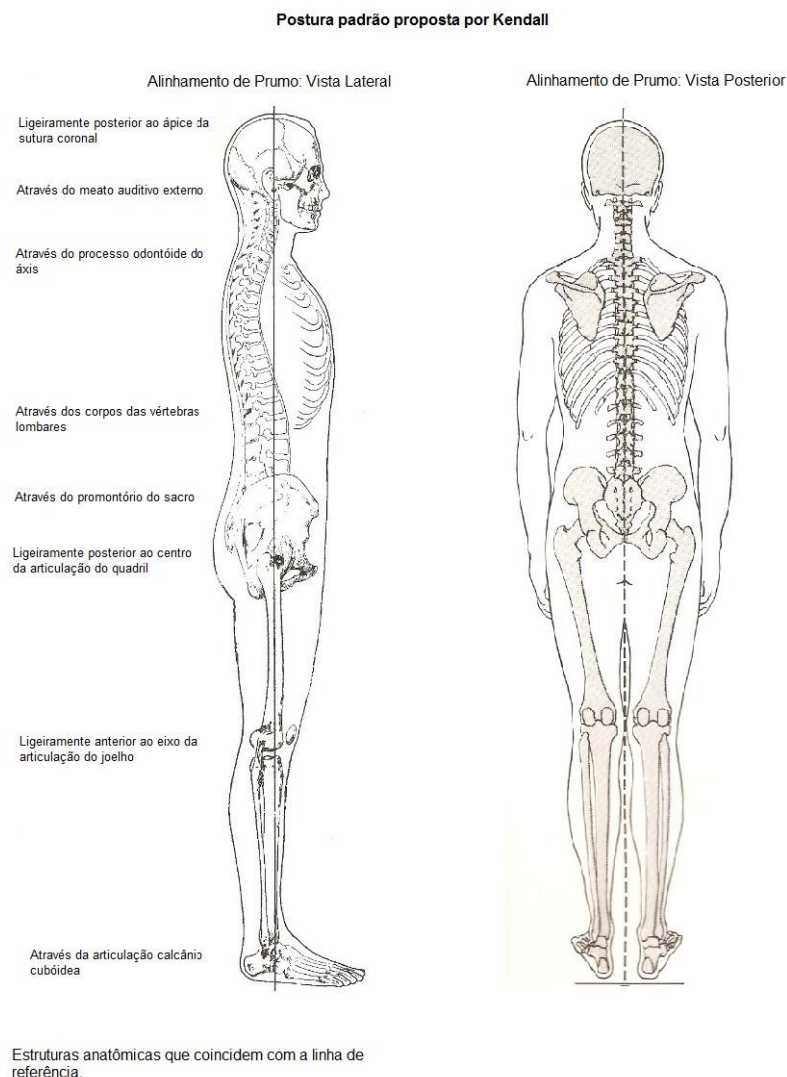


Figura 5 - Postura padrão proposta por Kendall

Fonte: adaptado de Kendall et al, (1995).

O teste de fio de prumo deverá ser comparado com o modelo de postura padrão e ser observada qual a diferença no paciente avaliado. Apesar do teste de fio de prumo ser um teste

subjetivo, pois depende totalmente da experiência do profissional que o está executando, Kendall et al (1995), já afirmava que a busca por maior objetividade nos testes é crescente e cada vez mais é necessário se ter medidas com números exatos afim de poder mensurar os melhores resultados de diversos tratamentos.

Bullock-Saxton (1993), em sua pesquisa afirma que a postura ereta assumida como confortável por uma pessoa, pode ser avaliada e representa o seu verdadeiro alinhamento. Tal postura fornece base para determinar os valores normativos para a postura. Além disso, o estudo revelou a repetição de um alinhamento postural do indivíduo em posição ereta sobre um período de dois anos.

Assim, outros métodos para a avaliação e estudo das assimetrias observadas do tronco de indivíduos com escoliose têm sido desenvolvidos. Alguns métodos por análise e de imagem digital são descritos na literatura como métodos que podem ser empregados para a detecção de mudanças posturais. No entanto, nenhuma destas técnicas foi capaz de substituir os raios-X para os seguimentos da progressão da escoliose (CHAN, 2014).

2.2.1 Métodos por análise de imagem radiográfica

Como mencionado anteriormente o exame baseado na radiografia é o mais preciso de acordo com a literatura. Ferreira et al, (1999), utilizaram um método simples de mensurar curvatura lateral pelo ângulo de Coob usando um transferidor para medir o ângulo em graus, traçando retas perpendiculares às vértebras de início e fim da curvatura conforme exemplificado no capítulo anterior. A figura 6 a seguir demonstra a ferramenta utilizada.

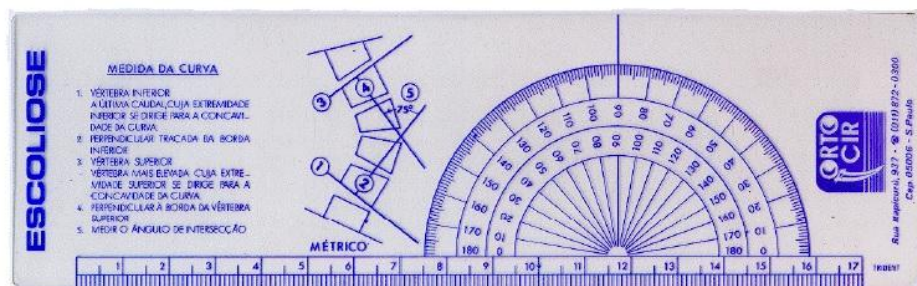




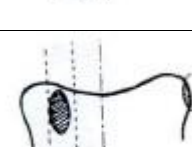
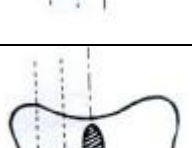
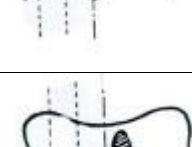
Figura 6 - Transferidor para medir o ângulo de Coob

Fonte: FERREIRA, (1999).

Para mensurar a rotação das vértebras, ou seja, gibosidade é utilizada o método de NASH & MOE, que é baseado na relação entre os pedículos vertebrais e o centro do corpo vertebral nas radiografias em antero-posterior (AP), sendo a rotação classificada em cinco

diferentes graus de acordo com o afastamento dos pedículos. Nas situações em que os pedículos estão equidistantes das margens laterais dos corpos vertebrais, não há rotação vertebral, e é considerado como grau 0. À medida que a projeção do pedículo da vértebra apical desloca-se para a linha média nas radiografias em AP, o grau de rotação progride na escala de avaliação, atingindo o maior valor (grau IV) quando ultrapassa essa linha (DELFINO e ARAUJO, 2004), conforme demonstra a tabela 2 a seguir.

Tabela 2 - Determinação da rotação vertebral pelo método de NASH & MOE.

Grau	Convexo	Pedículo	Côncavo
Neutro	Sem assimetria		Sem assimetria
Grau +	Migra dentro do 1º segmento Inicia distorção		Pode iniciar o desaparecimento Inicia a distorção
Grau ++	Migra para o 2º segmento		Gradualmente desaparece
Grau +++	Migra para o meio do segmento		Não é visível
Grau ++++	Migra passando da linha média para o lado côncavo do corpo vertebral		Não é visível

Fonte: Ferreira et al, (1999).

O método de Raimondi utiliza a projeção dos pedículos vertebrais e a largura da vértebra como referência para as medidas. O maior eixo do pedículo é demarcado e mensurado no lado da convexidade da curva, e a distância da linha longitudinal desde o pedículo até a borda da vértebra no lado convexo é mensurada. Esses dois valores são transportados para a régua, obtendo-se o valor da rotação (DELFINO e ARAUJO, 2004). A

figura 7, abaixo, demonstra o esquema de Raimondi que contém os valores em centímetros e o valor correspondente em graus.

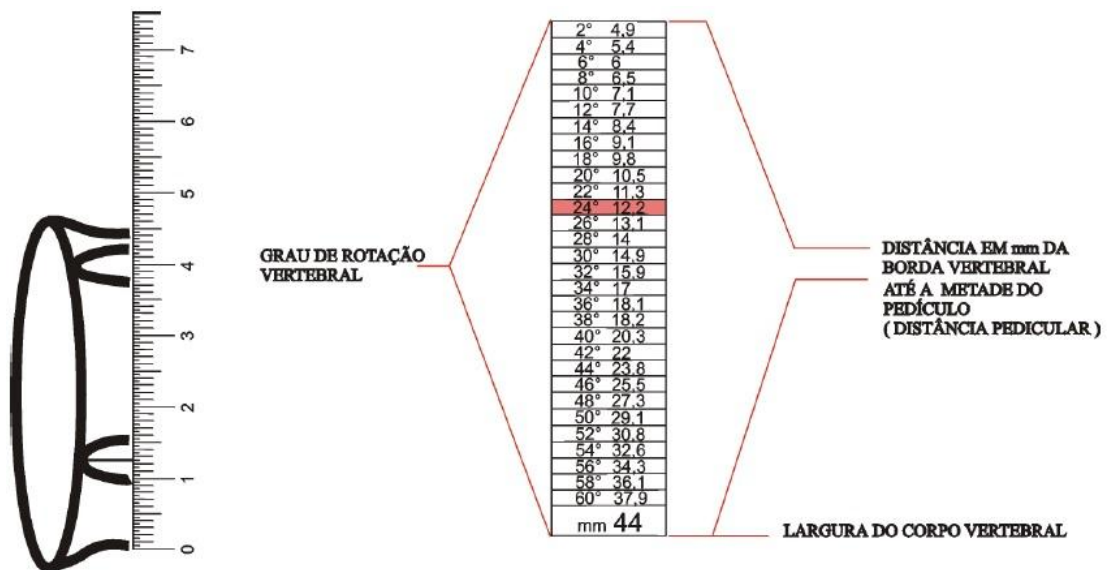


Figura 7 - Régua de Raimondi.

Fonte: Ferreira et al (1999).

Na figura 7 acima exemplifica a medida do corpo vertebral, que possui 44 mm de largura, e a distância pedicular é de 12 mm e o valor da rotação vertebral correspondente no esquema é de 24 graus.

O subtópico a seguir apresenta métodos por análise de imagem digital capazes de mensurar a postura através de imagens e contornos externos do corpo humano.

2.2.2 Métodos por análise de imagens digitais

Os métodos de avaliação postural por análise de imagens têm incentivado diversas pesquisas no sentido de aperfeiçoar tais métodos e validar sua eficácia. A necessidade surge pelo fato que os métodos que utilizam raios-X possuem um custo elevado, além de expor os pacientes à radiação, não sendo o método mais adequado para acompanhamento periódico do paciente e, principalmente, para o propósito de triagem (TEIXEIRA e CARVALHO, 2007).

Por isto, o teste de Adams tornou-se um procedimento padrão para detectar escoliose. Esse consiste na mensuração da gibosidade através da flexão anterior do tronco. Estudos clássicos já utilizavam esse teste por meio de um equipamento de madeira adaptado com nível d'água e régua para mensurar a altura da gibosidade, em rastreamento de escoliose em grandes populações escolares (FERREIRA et al, 2010). A figura 8, a seguir, demonstra o equipamento utilizado para medir a gibosidade.

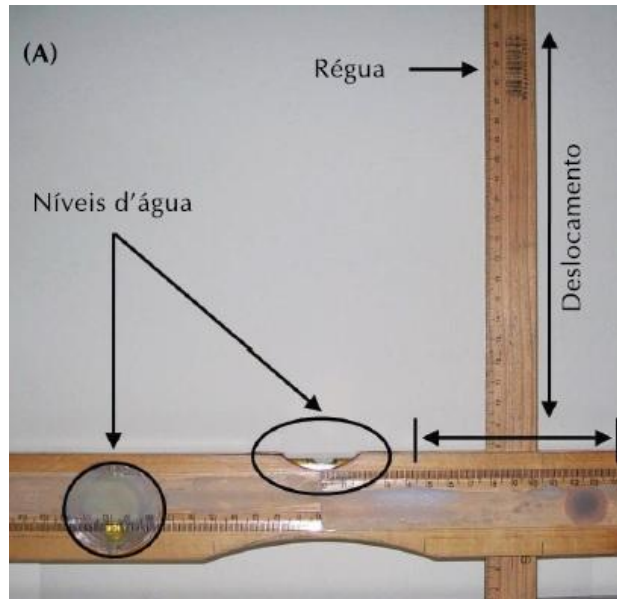


Figura 8 - Régua fabricada por Ferreira et al para medir gibosidade

Fonte: FERREIRA et al, (2010).

Ferreira et al, (1999) relatam que para medir a gibosidade com este equipamento o paciente deve se colocar na posição ortostática com os pés paralelos e afastados aproximadamente 10 cm e colocados sobre um desenho da impressão plantar com o objetivo de aumentar a base de sustentação e melhorar o equilíbrio. Solicita-se a flexão do tronco até aproximadamente 90° com os membros superiores ao longo do corpo livremente (pendurados). Assim é realizada a mensuração da gibosidade da coluna torácica ou lombar com o equipamento e registra somente a maior medida da coluna torácica ou da lombar conforme é demonstrado na figura 9 a seguir.

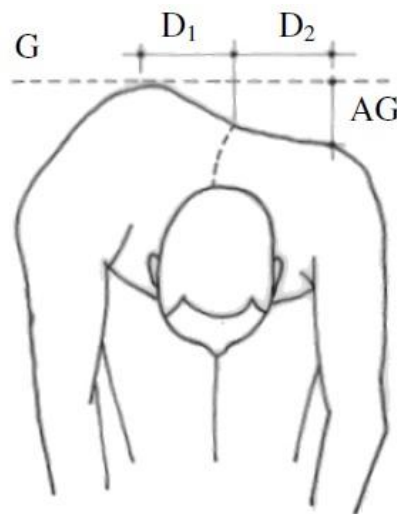


Figura 9 - Teste de Adams para mensurar gibosidade.

Fonte: FERREIRA, (1999).

No esquema apresentado na figura 9 acima, D1 corresponde à distância da coluna ao ponto mais alto da gibosidade, D2 é lançada a mesma distância registrada em D1, AG corresponde à altura da gibosidade, lembrando sempre de manter o equipamento nivelado observando o nível d'água conforme a linha pontilhada G.

2.2.2.1 Fotogrametria computadorizada

De acordo com a *American Society for Photogrammetry and Remote Sensing*, a fotogrametria é a arte, ciência e tecnologia de obtenção de informação confiável sobre objetos físicos e o meio ambiente por meio de processos de gravação, medição e interpretação de imagens fotográficas e padrões de energia eletromagnética radiante e outras fontes (SACCO, et al, 2007).

A fotogrametria possibilita o registro de mudanças sutis e da inter-relação entre partes diferentes do corpo humano difíceis de serem mensuradas ou registradas por outros meios (SACCO, et al, 2007).

A fotogrametria computadorizada é considerada um método consistente e confiável para avaliar o alinhamento postural, além de ser uma ferramenta eficaz e segura na avaliação, análise e quantificação de alterações na postura (KRAWCZKY et al, 2014).

Esse é um recurso acessível à maioria dos fisioterapeutas que já utilizam a fotografia e possuem equipamentos básicos, como uma câmera digital e um computador, permitindo realizar a avaliação postural e quantificar as alterações encontradas (IUNES, 2005).

A execução da avaliação postural por fotogrametria passa por algumas etapas. Primeiro é realizada a marcação no paciente que consiste de palpação e marcação pelo fisioterapeuta dos pontos de referência anatômicos. Depois de marcado são realizadas então a aquisição e calibração das imagens do paciente para posterior mensuração. A mensuração é realizada por software de tratamento de imagens ou sistemas desenvolvidos para os fins de avaliação postural.

De acordo com a literatura pesquisada, para efetuar a marcação dos pontos de referência não há um padrão estabelecido, cada autor adota um tipo de marcação. Ferreira et al (1999), elaborou um protocolo de marcação para o sistema SAPO, desenvolvido por um grupo interdisciplinar onde pode ser visualizado na figura 10 a seguir. Furlanetto et al (2012) adotaram o padrão de marcação proposto por Charrière e Roy apud Furlanetto et al (2012).

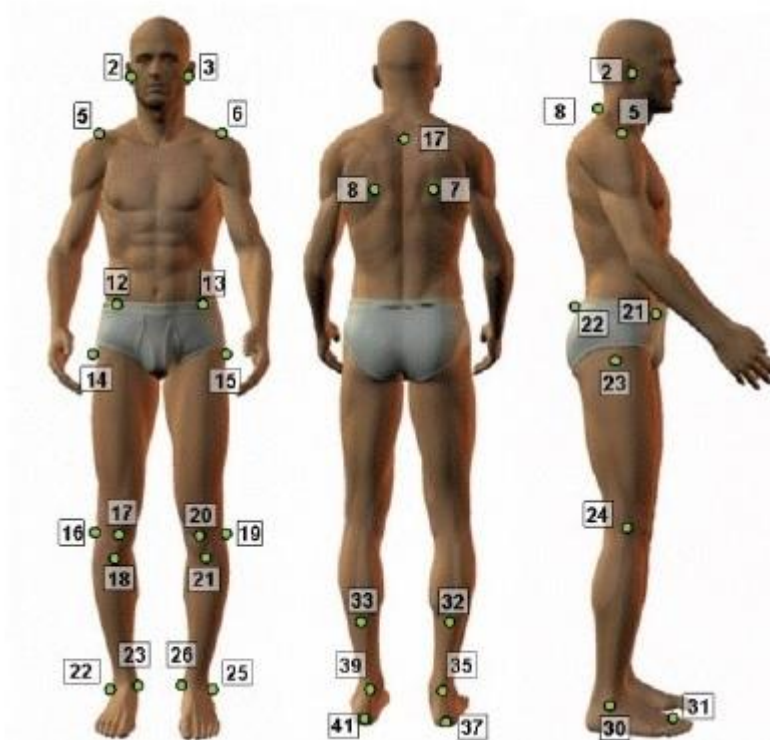


Figura 10 - Protocolo de marcação do sistema SAPO.

Fonte: Adaptada de FERREIRA, (1999).

Na obtenção das imagens que serão utilizadas é necessário tomar alguns cuidados metodológicos para padronizar as fotos e evitar efeitos de distorção para não causar erros. Na literatura é recomendado que: o fotógrafo deve ser experiente, utilizar sempre a mesma câmera e sempre à mesma distância do paciente, usar tripé e marcas no chão para posicionar o paciente, a fotografia deve ser de alta qualidade, sem distorções, ter alta nitidez, dimensão suficiente para permitir observações de contrastes, para que detalhes sejam visíveis na fotografia, evitando equívocos ao examinador. Quanto ao ambiente de avaliação deve ser livre de interferências, temperatura agradável, confortável, iluminação adequada e que ofereça privacidade, por ser necessário que o paciente esteja trajado de banho para maior exposição das partes do corpo avaliadas (IUNES et al, 2005).

Depois de gravada a foto digital a mesma é aberta pelo software e realizada a calibração por meio de medidas conhecidas na fotografia, como o fio de prumo posicionado ao lado do paciente. O avaliador irá posicionar os pontos conhecidos do protocolo com a marcação posicionada no corpo do paciente. Sendo assim o software poderá obter as coordenadas X (horizontal) e Y (vertical) de cada ponto, para que então seja calculado o ângulo dos membros avaliados (FERREIRA, 1999). A figura 11, abaixo, ilustra a avaliação no sistema SAPO.

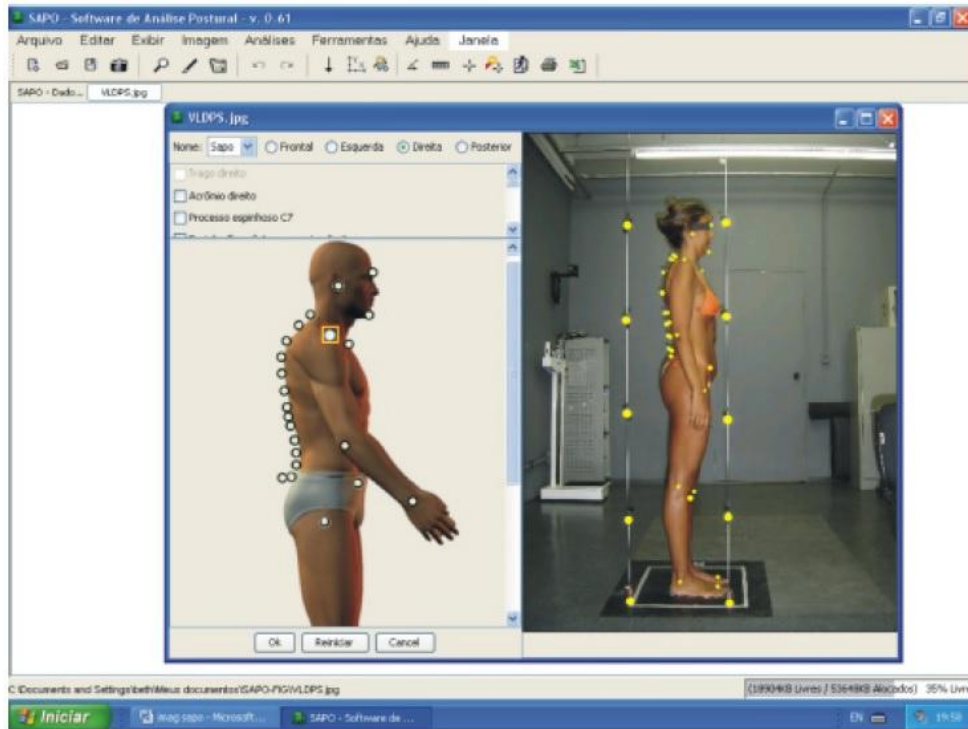


Figura 11 - Avaliação no sistema SAPO.

Fonte: FERREIRA, (1999).

Na figura 11 acima, o usuário deverá selecionar o ponto referente ao modelo, posicionado do lado esquerdo da tela, e clicar posicionar na foto, do lado direito da tela, o ponto respectivo ao modelo, para que assim o sistema possa calcular os valores da avaliação.

Logo a baixo será apresentada a topografia de moiré, que é utilizada na avaliação postural baseada nas medidas tridimensionais do relevo do corpo humano.

2.2.2.2 Topografia de Moiré

Topografia significa a arte de representar num plano as formas de uma superfície com os acidentes naturais ou artificiais que contém, ou a descrição de um lugar e de seus acidentes (Aurélio, 2000). A palavra moiré é proveniente da textura da seda, em que mudanças suaves no relevo do tecido forma franjas escuras e mais claras na luz (BARTL et al, 2001).

A topografia de moiré é um bom método clínico para avaliação e documentação da escoliose. Possui baixo custo, imagens simples, técnica não invasiva, com uma correlação bastante elevada ao ângulo de Cobb, exceto para os indivíduos com curvatura severa ou significativa obesidade (DOMAGALSKA, 2011).

O efeito moiré denota do padrão observado quando duas estruturas periódicas de frequência são relacionadas, mas com fases ligeiramente diferentes são sobrepostas. O efeito é

facilmente observado na vida cotidiana quando cortinas de nylon se movem em uma brisa ou quando dois conjuntos de trilhos semelhantes são sobrepostos (MORAN, 1994).

Quando se olha através de duas telas ou grades sobrepostas, nota-se a formação de padrões ou franjas, que são resultado da combinação das linhas dessas telas. Esse fenômeno é chamado de fenômeno ou efeito de moiré; e as franjas produzidas são chamadas de padrões ou franjas de moiré (LINO, 2002).

As franjas de moiré são mapas de contorno gerados por retículos na direção perpendicular as linhas de grade, estão tipicamente gerados pela superposição de duas grades periódicas: uma grade de referência e uma deformada pela superfície visualizada (LI et al, 2007). A figura 12, abaixo, demonstra um exemplo com interferência de luz formando ondas, chamadas franjas de moiré.

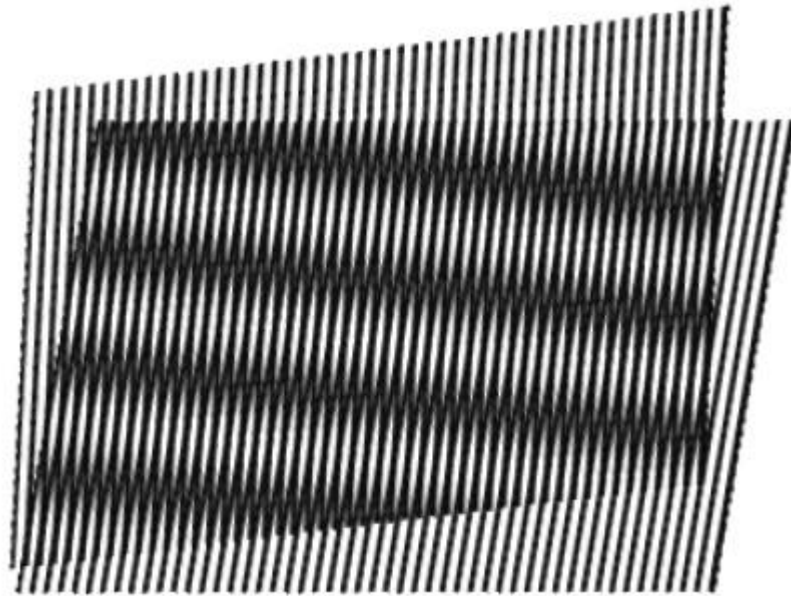


Figura 12 - Exemplo de franja de moiré.

Fonte: LINO, (2002).

Na figura acima as duas camadas atuam como dois retículos, um serve como referência denominado retículo de referência (R_r) e outro serve de modelo denominado retículo modelo (R_m). A captura e processamento destes dois retículos possibilitam a obtenção das franjas.

Sciammarella apud Lino, (2002), classifica os métodos de moiré em três métodos básicos:

- A. **Técnica de Moiré ou Técnica de Moiré Intrínseca:** provê o deslocamento dos pontos de uma superfície observada em relação a sua posição inicial.

B. **Moiré de Projeção:** também conhecido como moiré de sombra, que prove o deslocamento dos pontos de uma superfície observada em relação a uma superfície de referência.

C. **Moiré de Reflexão:** provê a inclinação dos pontos de uma superfície observada em relação a um estado de referência.

Takasaki, (1970), iniciou a utilização da topografia de moiré na medicina. Em sua pesquisa buscou obter curvas de níveis topográficos de um indivíduo. Foi utilizado um retículo, (espécie de grade), em frente ao indivíduo com espaçamento de 1 mm, construído de linhas de pesca de nylon com diâmetro de 0,45 mm. Foi projetada a iluminação com duas lâmpadas de 500 W, posicionadas lateralmente em relação à câmera fotográfica a uma distância de 500 mm da câmera fotográfica de cada lado. A câmera e as lâmpadas estão localizadas a 2000 mm do retículo. As imagens obtidas são demonstradas na Figura 13, onde as franjas de moiré possuem cotas de níveis intercaladas em 4 mm.

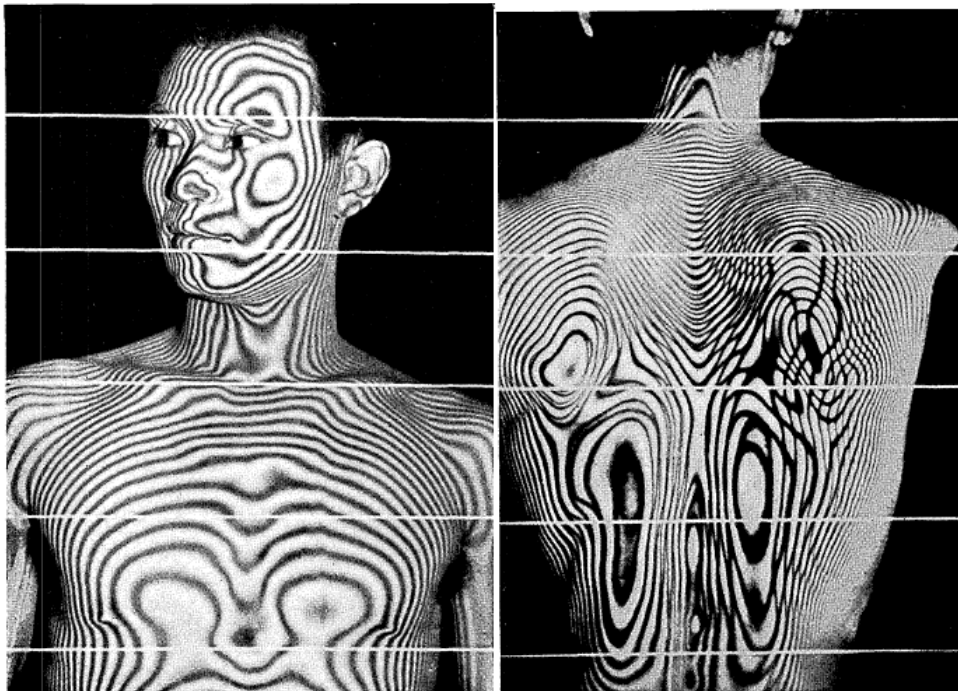


Figura 13 - Curvas de níveis topográficos.

Fonte: TAKASAKI, (1970).

Ao longo do tempo, as técnicas de moiré foram aperfeiçoadas. Atualmente são aplicadas em diversas áreas como: automação industrial, reengenharia, exames médicos, segurança, entre outras.

Lay et al (2012), em sua pesquisa aplicou a técnica de moiré de sombra com a finalidade de efetuar o reconhecimento facial de pessoas. O projeto consiste na obtenção das curvas de níveis da face de uma pessoa aplicando uma grade denominado retículo de

referência a frente da face, lateralmente a câmera uma lâmpada a ilumina gerando o efeito para a obtenção das curvas de acordo com a figura 14 a seguir.

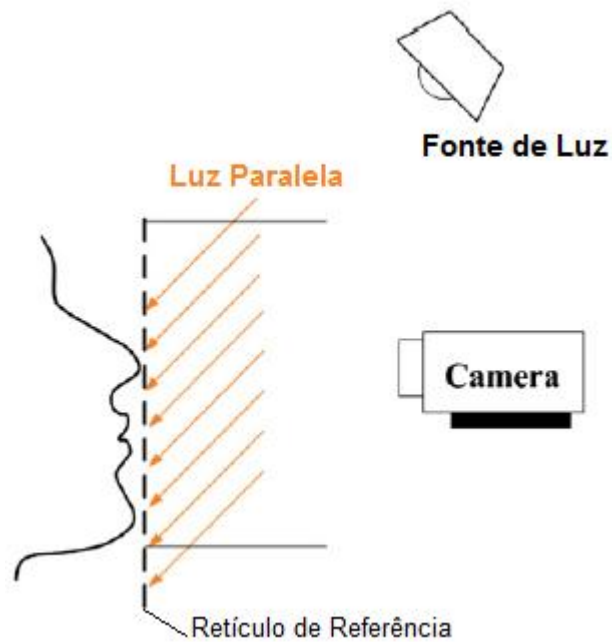


Figura 14 - Técnica moiré de sombra.

Fonte: traduzida de LAY, et al, (2012).

Lay et al (2012), afirma que para obter o contorno é aplicada a fórmula do triângulo em cada ponto da face para calcular a profundidade das superfícies. Assim ele propõe a seguinte fórmula.

$$d = \frac{Np}{\tan \theta_1 + \tan \theta_2}$$

Onde d é a distância (profundidade), Np significa o numero de linhas do retículo de referência e dividido pela soma das tangentes dos ângulos (1) e (2), onde o ângulo (1) representa o ângulo da fonte de luz e o ângulo (2) representa o ângulo da câmera que obtém as imagens. A figura 15, a seguir, demonstra o resultado da imagem obtida.



Figura 15 - Moiré de sombra em reconhecimento facial.

Fonte: LAY, et al, (2012).

Por fim, Lay et al (2012), conclui que moiré é um bom método para medir o perfil 3D de um objeto.

Batouche et al (1996), utilizou a topografia de moiré com objetivo de mensurar escoliose em pacientes. Após gerada as curvas de níveis da imagem do dorso do paciente realizou-se a segmentação da imagem com algoritmo de processamento de imagens para extrair as curvas de nível. Assim, para avaliar, procurou obter pontos específicos para mensuração como: linhas dorsais, ou seja, aquelas que aparecem acima dos centros das escápulas, linhas lombares, isto é, que são linhas verticais e se encontram na proximidade do vão da coluna, e linhas lombar dorsal, ou seja, aquelas que estão entre os centros da omoplata e oco lombar, tais linhas aparecem como curvas "horizontais", conforme a figura 16 demonstra.

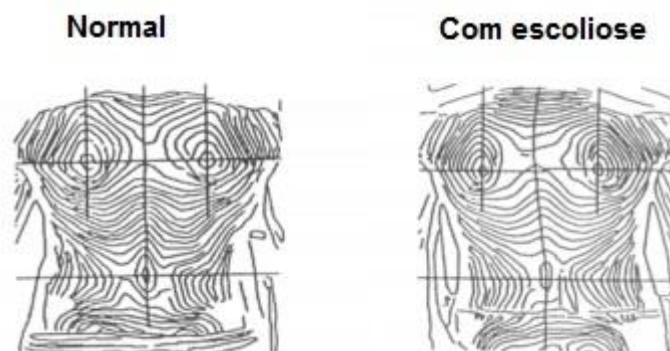


Figura 16 - Topografia de moiré para mensurar escoliose.

Fonte: adaptada de BATOUCHE, et al, (1996).

Batouche et al (1996), conclui que o sistema de apoio à decisão proposto foi testado extensivamente e os resultados são aceitáveis se comparado aos métodos por imagem radiológica, porém não sugere que possa substituí-los, as maiores partes das avaliações estão de acordo com o diagnóstico do médico. Ele ressalta que o sistema seja utilizado para fins de triagem e locais de grande volume de atendimento.

Para melhor entendimento de como se dá o processamento das imagens tridimensionais da topografia de moiré se faz necessário abordar neste trabalho aspectos da visão computacional que será abordada na seção a seguir.

2.3 Visão Computacional

A visão é um dos sentidos mais importante para a maior parte dos animais, fornece grande quantidade de informação como formas de objetos, distância, texturas. Estas informações o auxiliam na tomada de decisões em situações de perigo, identificar alimentos, fugir de predadores, capturar presas etc.

Desta forma a visão computacional procura dotar computadores deste sentido para determinadas funções, desempenhadas até então apenas pelo homem. Sendo assim cientistas e pesquisadores buscam entender como funciona o sistema visual biológico, e desenvolver técnicas semelhantes para computadores.

A visão computacional é uma disciplina que surgiu de certa forma recente. Os primeiros experimentos foram realizados no final de 1950, e muitos dos conceitos essenciais foram desenvolvidos recentemente. Com este rápido crescimento, as ideias cruciais surgiram em áreas distintas, como a inteligência artificial, psicologia, computação gráfica e processamento de imagem (BALLARD; BROWN, 1982).

Os sistemas de visão computacional não possuem uma arquitetura bem definida, pois dependem do conhecimento de outras áreas no qual será aplicado. Assim são classificados como sistemas especialistas.

De acordo com Rehem e Trindade (2009) apud MILANO et al (2010), os sistemas de visão computacional possuem algumas funcionalidades em comum como:

- **Aquisição de Imagem:** O processo de aquisição de imagem consiste em obter uma ou uma sequência de imagens digitais através de sensores geralmente contidos em câmeras digitais, como por exemplo, webcam. Dependendo do tipo de sensor o resultado da captação pode variar entre uma imagem bidimensional, em uma cena tridimensional ou em uma sequência de imagens. Os pixels indicam em cada coordenada, valores de intensidade de luz em uma cor (gerando imagem preta e

branca) ou em faixas de cores (gerando imagens coloridas). Podem também identificar valores físicos como profundidade, absorção e reflexão das ondas eletromagnéticas.

- **Pré-processamento:** O pré-processamento consiste em aplicar métodos de processamento de imagem, antes de extrair informações desta para o sistema de visão computacional. Podemos tomar como exemplo a aplicação de métodos que destacam o contorno, com a finalidade de buscar a forma arredondada da íris dos olhos para facilitar o processo de identificação de um olho em uma imagem.
- **Extração de características:** Esse processo consiste em garimpar informações de uma imagem. Uma imagem é formada por modelos matemáticos como matrizes estas contêm características que podem matematicamente ser identificadas como: textura, cantos, bordas e etc.
- **Detecção e segmentação:** Esse processo consiste em destacar uma determinada região de uma imagem e segmentá-la com a finalidade de guardar essa informação para processamento posterior. Em rastreamento é comum definir um pedaço da imagem (template) e na execução do vídeo tentar achar regiões semelhantes a esse pedaço.
- **Processamento de alto nível:** No processamento de alto nível entrada é geralmente um conjunto restrito de dados. O processo consiste na verificação da satisfação dos dados, a estimativa de parâmetros sobre a imagem e a classificação dos objetos detectados em diferentes categorias.

Para executar tais procedimentos é necessário que um sistema de visão computacional possua alguns componentes básicos. Na maior parte das aplicações de visão computacional, o sistema é composto de: sensor de aquisição, iluminação da cena, sistema de aquisição de imagem, dispositivo de processamento de imagem, software de processamento de imagens, display e armazenamento (MILANO et al, 2010).

Os processos de visão computacional, muitas vezes, necessitam de uma etapa de pré-processamento envolvendo o processamento de imagens. As imagens de onde queremos extrair alguma informação em alguns casos precisam ser convertidas para um determinado formato ou tamanho e precisam ainda ser filtradas para remover ruídos provenientes do processo de aquisição da imagem. Os ruídos podem aparecer de diversas fontes, como por exemplo, o tipo de sensor utilizado, a iluminação do ambiente, as condições climáticas no momento da aquisição da imagem, a posição relativa entre o objeto de interesse e a câmera (MARENGONI e STRINGHINI, 2009).

2.3.1 Sentido do fluxo

No corpo humano de todas as pessoas, é comum na topografia das costas seu relevo apresentar um vão na região da coluna vertebral. Sendo assim em busca de alguma forma para detectar a coluna vertebral através de uma imagem de profundidade, percebeu-se que seu relevo é semelhante a um vale, então poderia aplicar este método de análise de imagens de satélite, utilizado para demarcar rios, para então detectar o vale da região da coluna vertebral.

Uma imagem de profundidade possui em cada pixel um valor referente à distância que esta superfície se encontra do sensor. Assim temos uma cota do relevo da superfície pixel a pixel.

O método para buscar o sentido do fluxo é simples e consiste em três etapas. Inicialmente é construído um mapa de apontamento da imagem, logo após deve ser gerado um mapa de apontamentos recebidos por fim um mapa de fluxos acumulados (JENSON e DOMINGUE, 1988). Na figura 17 a seguir será demonstrado o processo para gerar o mapa de apontamento.

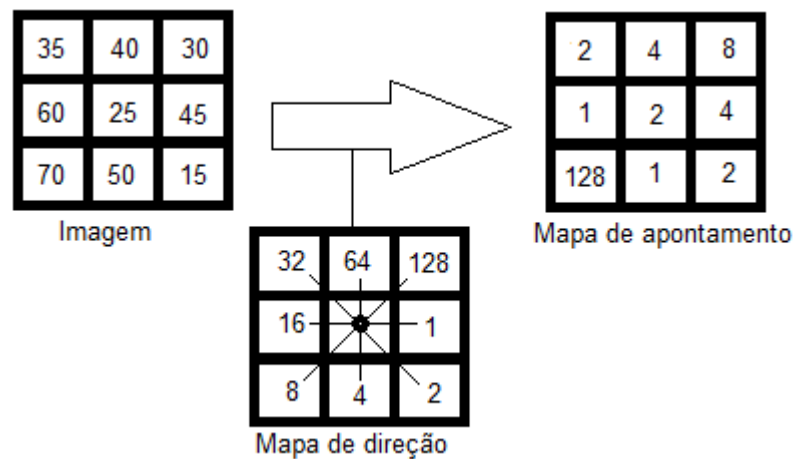


Figura 17 - Processo para gerar o mapa de apontamento.

Na figura acima se deve percorrer cada pixel da imagem e verificar toda a vizinhança qual o pixel de menor valor, ou seja, mais baixo. Assim atribua-se o valor da direção conforme o mapa de direção no mapa de apontamento. A seguir na figura 18 será demonstrado o processo para gerar o mapa de apontamentos recebidos.

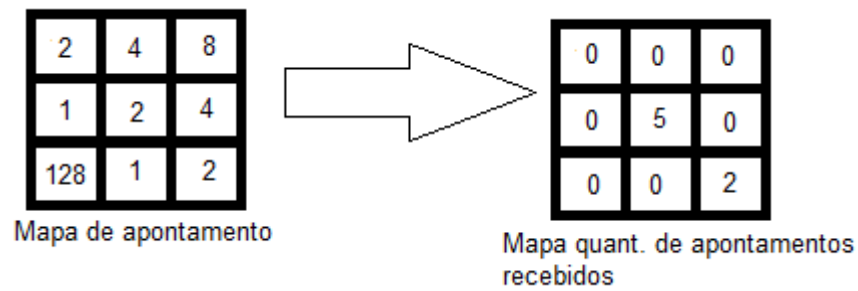


Figura 18 – Processo para gerar mapa de apontamento recebido.

Nesta etapa ao visitar cada pixel verifica-se toda a vizinhança a fim de somar quantos pixels vizinhos apontam para o pixel visitado. Obtendo o valor do somatório atribui a quantidade ao pixel correspondente no mapa de apontamentos recebidos. Na figura 19 a seguir será demonstrado o terceiro e último processo, que tem o objetivo de gerar o mapa de fluxo acumulado.

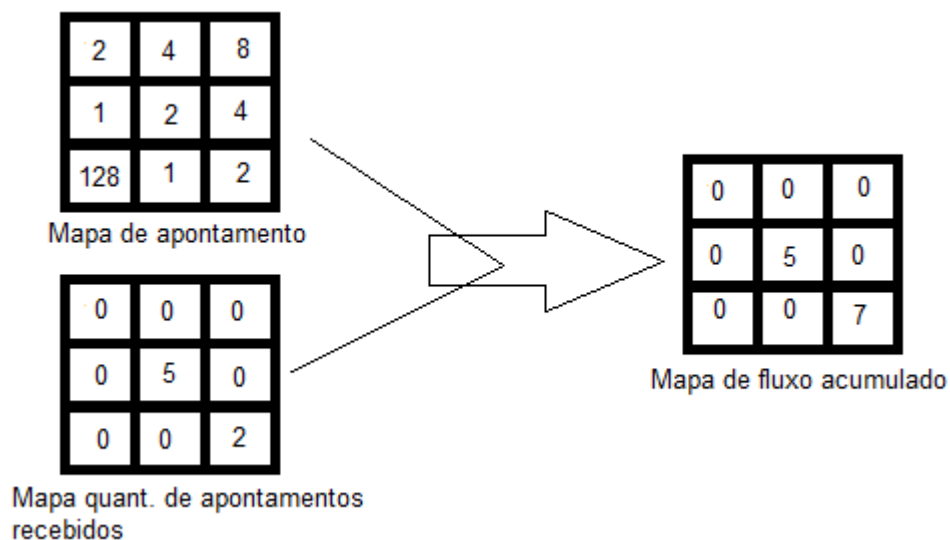


Figura 19 – Processo para gerar o mapa de fluxo acumulado

O processo demonstrado na figura acima demonstra que ao visitar cada pixel verifica-se toda a vizinhança de pixel, e cada pixel irá acumular os valores dos apontamentos recebidos somados aos valores dos pixels que apontam para ele, gerando assim o mapa de fluxo acumulado.

Com o mapa de fluxo acumulado é possível gerar uma imagem em escala de cinza na qual será perceptível os maiores valores. Estes pixels com maiores valores assumiram cores mais claras na escala de cinza, destacando a linha formada pelos fluxos.

2.4 Imagem de profundidade do Kinect

O sensor Kinect possui um emissor de laser infravermelho, uma câmera infravermelha e uma câmera RGB. O laser é emitido em um feixe único que é dividido em múltiplos feixes por uma grade, assim cria um padrão constante de pontos projetados na cena. Este padrão é capturado pela câmera infravermelha e está correlacionada com um padrão de referência. Conforme a distância entre os pixels projetados do padrão de referência aos pixels capturados pela câmera infravermelha pode-se conhecer a distância de cada pixel na superfície da cena. A figura 20 abaixo ilustra a medida de profundidade a partir da configuração de cores (KHOSHELHAM e ELBERINK, 2012).

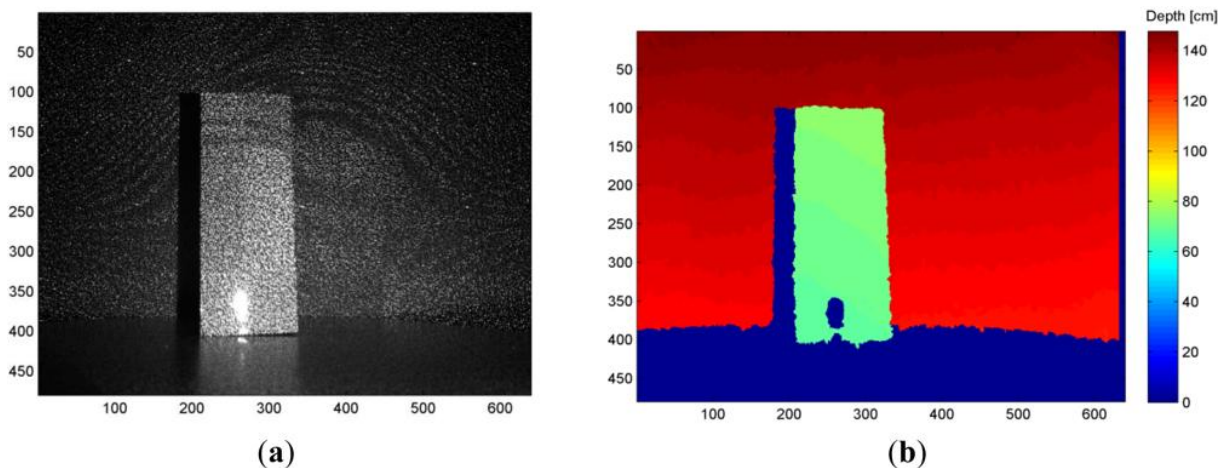


Figura 20 - Imagem de profundidade do Kinect.

Fonte: Khoshelman e Elberink (2012).

a) Imagem de uma cena com padrão de pontos projetados em uma cena. b) Imagem de profundidade resultante.

3 MATERIAIS E MÉTODOS

Neste capítulo será apresentados materiais e métodos utilizados no desenvolvimento deste projeto. Na próxima seção será apresentado o local e período de desenvolvimento do trabalho; na seção 2.2 será descrito todos os equipamentos e softwares que foram utilizados no desenvolvimento deste projeto.

3.1 Local e Período

O presente trabalho foi desenvolvido como requisito parcial para a disciplina de “Trabalho de Conclusão de Curso em Sistemas de Informação II (TCC II)” no decorrer do segundo semestre de 2014. O local utilizado para o desenvolvimento foi, basicamente, o Laboratório Banco de Dados e Engenharia de Software (LBDES) em conjunto com o Laboratório de Tecnologia em Saúde (LTS), ambos do Centro Universitário Luterano de Palmas (CEULP/ULBRA), vinculados ao grupo de Pesquisa, Tecnologia, Saúde e Qualidade de Vida.

3.2 Hardware e Softwares

Para o desenvolvimento foi utilizado um notebook com processador Intel Core 2 Duo 2GHz, 3 GB de memória RAM, H.D. 250 GB; com sistema operacional Microsoft Windows 7 64 bits e o sensor Microsoft Kinect. Como ferramenta de desenvolvimento foi instalado o Microsoft Visual Studio 2012 Ultimate em conjunto com SDK Kinect for Windows Versão 1.8, para que o projeto possa fazer uso das bibliotecas disponibilizadas. No subitem subsequente será descrito alguns aspectos inerentes ao funcionamento do sensor Kinect.

3.2.1 Kinect

O sensor Microsoft Kinect foi lançado em 04 de novembro de 2010. Apesar de ser um produto criado pela Microsoft, o Kinect foi desenvolvido em parceria com a empresa israelense PrimeSense. Essa parceria e resultou no sensor que aliado aos seus *drivers* o torna capaz de reconhecer partes do corpo humano, monitorando gestos e movimentos do corpo humano.

O Kinect inovou o conceito de interface homem-computador, permitiu a interação de usuários a jogos e aplicativos através de gestos do corpo humano e comandos de voz surgindo então o conceito de interface natural do usuário ou *natural user interface* (NUI). Desta forma, o sensor Kinect proporcionou que muitas pesquisas em diversas áreas surgissem, assim como este projeto (BORENSTEIN, 2012). A seguir no próximo subitem veremos aspectos da arquitetura do Kinect.

3.2.2 Arquitetura do Kinect

Quanto a sua arquitetura o sensor Kinect possui na sua parte frontal quatro microfones para captar comandos de voz do usuário, um projetor de infravermelho que projeta uma nuvem de pontos invisível ao olho humano, uma câmera de infravermelho que capta a cena com os pontos de infravermelho projetados e uma câmera colorida que capta a cena em cores. A figura 21 abaixo ilustra o sensor Kinect internamente, ou seja, sem sua capa de proteção.

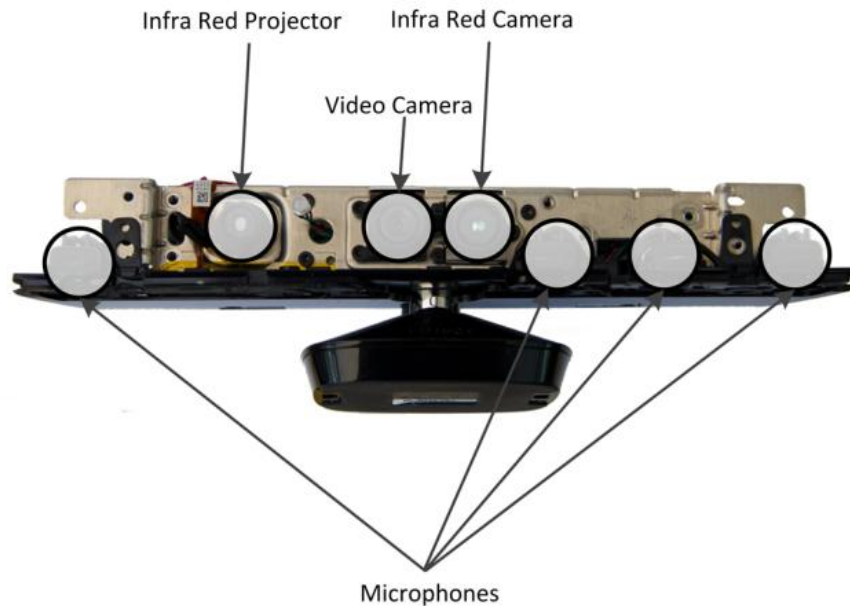


Figura 21 - Visão frontal do sensor Kinect sem capa

Fonte: Microsoft, (2014).

A camera colorida do Kinect indicada na figura acima como “*video camera*”, trata-se de um sensor CMOS (*Complementary metal-oxide-semiconductor*), semelhante ao empregado em *webcam* e cameras digitais e celulares. Este possui capacidade de captar imagens coloridas, mas quanto ao formato, fica a cargo do *driver* do Kinect definir, no qual será demonstrado em tópico posterior.

O projetor infravermelho esta denominado na figura “*Infra red projector*” e a camera de infravermelho denominada “*Infra red camera*”, assim os dois componentes de infravermelho possibilita que o *driver* do Kinect gere os dados da imagem de profundidade. Segundo Catuhe, 2012, não existe qualquer CPU no interior do sensor, e sim somente um processador de sinal digital, que é usado para processar o sinal dos microfones e cameras dispostas no aparelho. O processamento de dados das imagens coloridas e da imagem de profundidade é executada no lado do computador pelo driver do Kinect. A figura 22 abaixo exemplifica a arquitetura física do Kinect, demonstrando o fluxo dos sinais obtidos pelos

microfones e cameras e processados pelo chip do Kinect e repassados ao computador através da porta USB do computador.

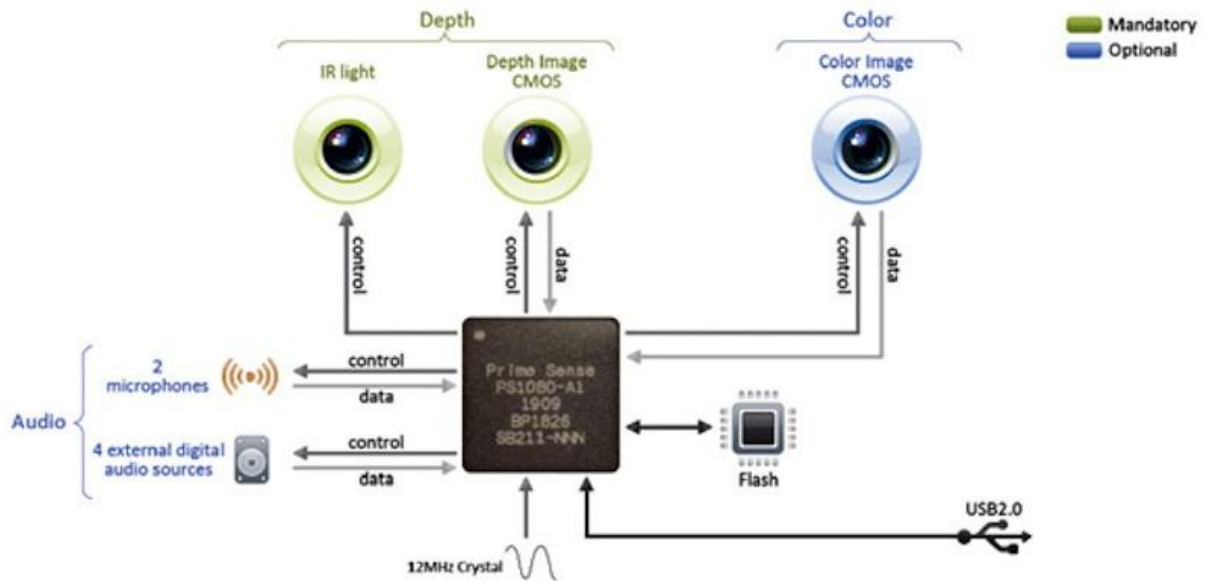


Figura 22 - Arquitetura física do Kinect

Fonte: <https://www.ifixit.com/Teardown/Microsoft+Kinect+Teardown/4066>

Como a figura acima exemplifica o Kinect é conectado ao computador pela porta USB, mas é necessário conectar um adaptador USB que possui uma fonte de alimentação, pois a energia fornecida pela porta USB não é suficiente para funcionar o sensor Kinect.

Como se utiliza de lentes ópticas nas cameras instaladas, o sensor possui algumas limitações físicas de alcance. O alcance do sensor na horizontal limita-se a um ângulo de 57° e um ângulo vertical de 43°. A distância para o melhor rastreamento do usuário é de 1,2 metros até 4 metros a partir do aparelho. O Kinect ainda possui um motor de passo para que possa ser ajustado o foco em -28° a + 28°. Na próxima seção será descrito os aspectos da SDK Kinect V. 1.8, suas bibliotecas possuem métodos imprescindíveis para o desenvolvimento de aplicações que utilizam o Kinect.

3.2.3 SDK Kinect

Foi utilizado no desenvolvimento deste trabalho a SDK (*Software Development Kit* ou kit de desenvolvimento de software) versão 1.8. A SDK é distribuída gratuitamente pela Microsoft. Durante a fase desenvolvimento verificou-se que só há restrições de bibliotecas caso estiver utilizando o Kinect para XBOX 360. Utilizando o Kinect for Windows terá acesso a bibliotecas como: Kinect Fusion e a função de rastreamento no modo *near*, ou seja, de perto. A biblioteca Kinect Fusion possui funções para renderizar imagens 3D a partir da

imagem de profundidade (*depth image*). O modo *near* é utilizado para rastrear o usuário em menores distancias.

Este trabalho foi desenvolvido utilizando o Kinect para XBOX 360, no entanto, este projeto se utilizou das funções de imagem de profundidade, imagem colorida e esqueleto. A Microsoft no site oficial do Kinect afirma que não é proibido utilizar o Kinect para XBOX 360 para fins de aprendizado e estudo, porém não se responsabiliza por falhas e incompatibilidades. Para comercializar softwares desenvolvidos utilizando a SDK do Kinect é necessário adquirir o sensor Kinect for Windows, pois este possui a licença para tal finalidade.

A SDK Kinect V 1.8 possibilita o desenvolvedor utilizar a linguagem C# ou C++ no desenvolvimento de suas aplicações. Para este projeto foi utilizado a linguagem C# devido ter maior comunidade de desenvolvedores e literatura direcionada a esta linguagem facilitando o desenvolvimento. Ao iniciar o desenvolvimento de um projeto que irá utilizar o sensor Kinect, basta que o projeto tenha como referência a DLL (*Dynamic-link library* ou biblioteca de vínculo dinâmico) chamada "*Microsoft.Kinect.dll*". Nela contém diversas classes e métodos necessários para o desenvolvimento com o Kinect. A seguir veremos em mais detalhes como esta DLL foi utilizada neste projeto.

O SDK Kinect fornece através de seus métodos os fluxos de dados de imagem colorida, imagem de profundidade, som e esqueleto. A figura 23 a seguir ilustra o funcionamento do SDK intermediando o acesso aos dados entre o sensor e a aplicação desenvolvida.

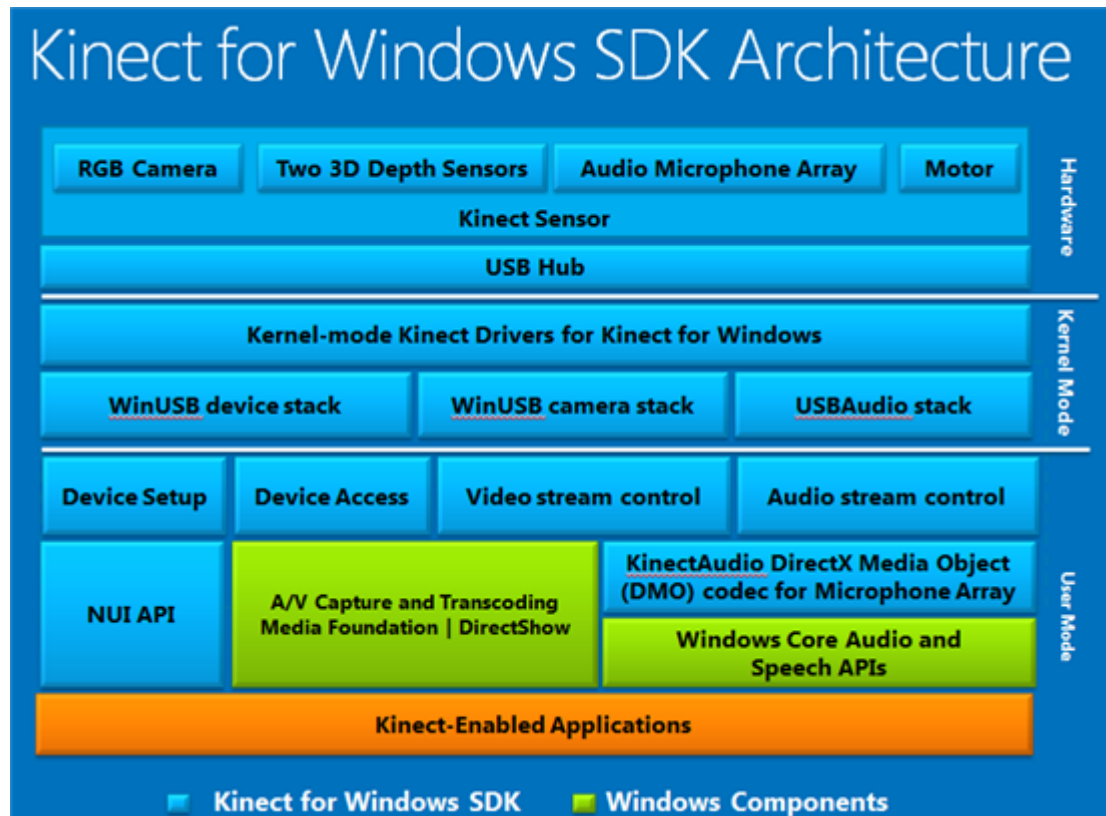


Figura 23 - Arquitetura SDK Kinect.

Fonte: <http://blogs.msdn.com/b/ampuri/archive/2012/02/09/kinect-for-windows-sdk-nui.aspx>

Na figura 23 acima, é ilustrado os componentes físicos (*hardware*) do Kinect, que consiste na câmera colorida (RGB), dois sensores de profundidade (*Depth*), microfones de áudio e o motor de passo, que conectam ao kernel pela interface USB.

No *Kernel Mode* é ilustrado as pilhas de dados armazenados que chegam do sensor que são: “*WinUSB device stack*”, “*WinUSB Camera stack*” e “*USBAudio stack*”. Assim são disponibilizados ao desenvolvedor as funcionalidades contidas na camada *User Mode* que são: “*Device Access*”, “*Video stream control*” e “*Audio stream control*”.

Na “*Device Access*” trata-se do acesso a configurações do sensor como, por exemplo, o ângulo do sensor controlado pelo motor de passo. Na “*Video stream control*” disponibiliza métodos de acesso e manipulação dos fluxos de imagem colorida e de profundidade e na “*Audio stream control*” fornece métodos para obter sons capturados pelo sensor.

Na cor verde trata-se de componentes que interagem com os dados obtidos da SDK, porém pertence ao sistema operacional Windows. O “*Windows Core Audio and Speech APIs*” disponibiliza funções para o reconhecimento de voz, muito útil para que a aplicação possa atender a comandos de voz realizados pelo usuário a aplicação.

Na cor laranja “*Kinect-Enabled Applications*” representa-se a aplicação desenvolvida que interage com os métodos disponibilizados pela SDK, processa-os e apresenta ao usuário.

Dentro desta arquitetura apresentada do SDK Kinect foram utilizados neste projeto os fluxos de imagem colorida, imagem de profundidade e o esqueleto.

3.2.3.1 Imagem colorida

O fluxo de imagem colorida é denominado *ColorStream*. Este fluxo pode ser ativado na aplicação através do método *KinectSensor.ColorStream.Enable()*. Assim este fluxo possui três opções de formato e resolução que são:

- *RgbResolution640x480Fps30*
- *RgbResolution1280x960Fps12*
- *RawYuvResolution640x480Fps15*
- *YuvResolution640x480Fps15*

Por padrão, caso o desenvolvedor não especifique o formato, é definido o primeiro da lista acima. O formato RGB (*Red, Green, Blue*), ou seja, (Vermelho, Verde, Azul), é um padrão de sobreposição de cores em cada pixel, uma matriz de pixels resulta em uma imagem. A cor de cada pixel é armazenado pela representação de 8 bits para cada cor do padrão RGB, então os valores de cada cor pode variar de 0 a 255, sendo 0 a cor preto e 255 a cor absoluta. Sendo assim podemos representar vermelho absoluto da seguinte forma (255, 0, 0), verde absoluto (0,255,0) e azul absoluto (0,0,255).

Assim o padrão RGB pode gerar mais de 16 milhões de cores com a combinação destes valores. O formato 640x480 representa que a imagem será constituída de uma matriz de pixels do tamanho de 640 colunas de pixel por 480 linhas de pixel. A sigla *Fps (Frames per second)* ou (Quadros por segundo), estabelece o numero de quadros de imagens que o sensor disponibilizará a aplicação em um segundo. Na figura 24 a seguir segue um exemplo de código para habilitar o fluxo de imagem colorida e iniciar o sensor Kinect.

```

32 |         var kinectSensor = KinectSensor.KinectSensors[0];
33 |         kinectSensor.ColorStream.Enable(ColorImageFormat.RgbResolution640x480Fps30);
34 |         kinectSensor.Start();

```

Figura 24 - Habilitar fluxo de imagem colorida

Na figura 24 acima na linha 32 é referenciado o objeto *kinectSensor* ao sensor conectado, vale ressaltar que é possível ter diversos sensores conectados a uma mesma aplicação. Na linha 33 é habilitado o fluxo através do método *Enable()*, passando por

parâmetro o formato da imagem que será disponibilizada a aplicação. Assim na linha 34 o sensor é iniciado pelo método *Start()*.

Até então o sensor já estará em funcionamento disponibilizando o fluxo de imagem colorida. Porém a aplicação ainda necessita capturar os quadros de imagem que estão sendo disponibilizados com o passar do instante de tempo. Caso necessite capturar apenas um quadro de imagem no instante de tempo, deve-se executar o método “*OpenNextFrame()*”, e como parâmetro do método deverá passar o tempo em milissegundo que ele deve esperar para capturar, se for imediato basta passar como parâmetro o valor zero, que será capturado apenas um quadro de imagem, assim como uma fotografia. A figura 25 a seguir demonstra o método realizando a captura com tempo de espera de quinhentos milissegundos, ou meio segundo de espera.

```
36 | ImageFrame quadro = kinectSensor.ColorStream.OpenNextFrame(500);
```

Figura 25 - Método para capturar um quadro de imagem.

Outra forma de capturar o fluxo de imagem colorida, de forma que a aplicação possa capturar vários quadros por segundo, se dá criando um manipulador de eventos, ou *delegate*, recurso da linguagem C# para criar eventos personalizados para a aplicação. Desta forma, a cada instante em que for disponibilizado um novo quadro pelo sensor será disparado o evento criado. Sendo assim, lembrando que no formato utilizado por este projeto que disponibiliza trinta quadros por segundo, este evento será disparado trinta vezes em um segundo. A figura 26 a seguir demonstra a implementação deste método.

```
38 | kinectSensor.ColorFrameReady += kinectSensor_ColorFrameReady;
39 |
40 | }
41 |
42 | void kinectSensor_ColorFrameReady(object sender, ColorImageFrameReadyEventArgs e)
43 | {
44 |     var quadro = e.OpenColorImageFrame();
45 | }
```

Figura 26 - Manipulador de evento para fluxo de imagem colorida

Desta forma é possível transmitir para a tela do software as imagens geradas pelo Kinect ao vivo, ou então estas poderão ser armazenadas em variáveis ou então serializadas em disco para utilização posterior. A figura 27 abaixo demonstra o diagrama de classes que envolve a classe *ColorImageFrame* que representa um quadro da classe *ColorImageStream*. Ela resulta no retorno dos métodos *OpenNextFrame()* e o *OpenColorImageFrame()*.

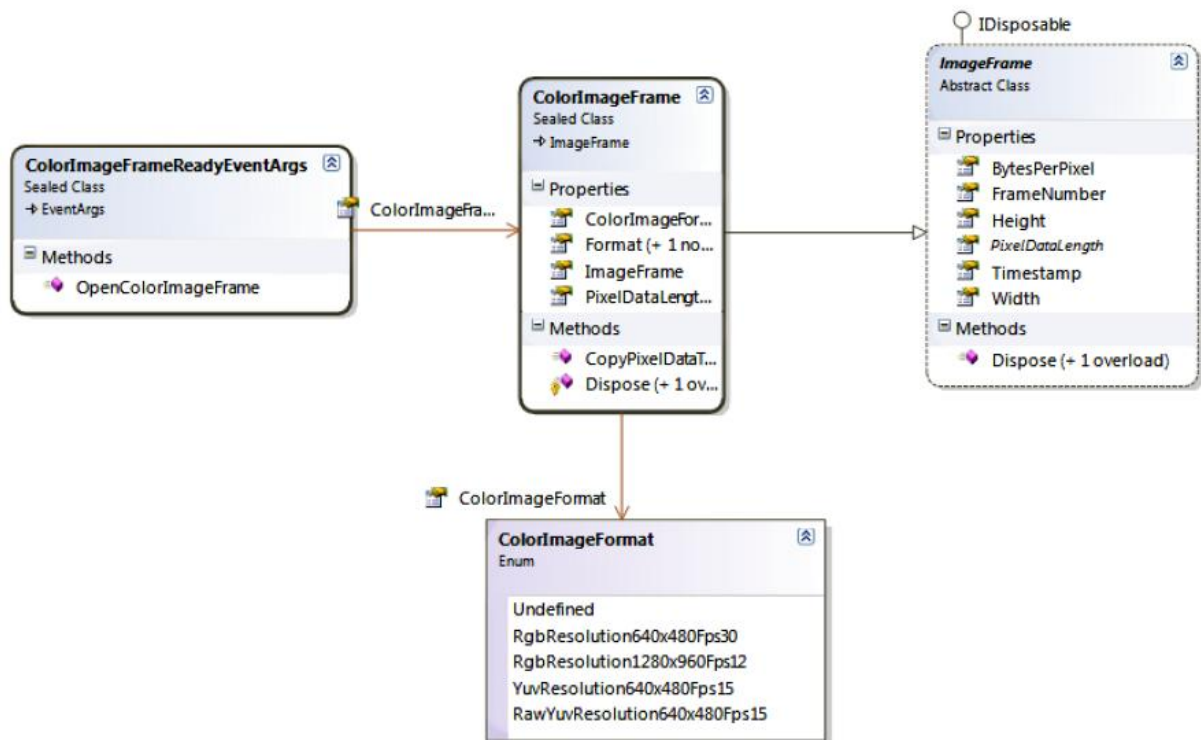


Figura 27 - Diagrama de classe do fluxo de imagem colorida.

Na figura 27 acima pode-se observar que *ColorImageFrameReadyEventArgs* possui uma agregação da classe *ColorImageFrame*. Esta por sua vez herda da classe abstrata *ImageFrame*. A classe *ColorImageFrame* possui um atributo *Format*, no qual se trata de uma enumeração denominada *ColorImageFormat*. Segundo Troelsen, (2009), *Enum* é um conjunto de nomes ou registros simbólicos que representa algum valor.

As classes possuem uma referência da linguagem C# *sealed*, que significa que não é possível criar novas classes que herdem estas classes. A seguir serão demonstrados aspectos do fluxo do esqueleto denominado “*SkeletonStream*”.

3.2.3.2 Esqueleto

Neste projeto foi utilizado o fluxo do esqueleto na avaliação postural com finalidade de buscar desvios e anormalidades na postura. O fluxo do esqueleto é o único que não é originado diretamente do sensor Kinect, ele é representado pela classe *SkeletonStream* e é processado pelo SDK Kinect. O SDK utiliza a imagem de profundidade representada pela classe *DepthStream* para interpretar as partes do corpo humano através de inteligência artificial (SHOTTON, et al, 2013).

Para habilitar o fluxo do esqueleto deve-se utilizar o método *Enable()*. Também é necessário habilitar o fluxo de imagem de profundidade, pois o SDK necessita deste para

interpretar o esqueleto do usuário. Ainda há a opção de rastrear o usuário sentado, bastando atribuir a propriedade *TrackingMode* para o modo *SkeletonTrackingMode.Seated*, a figura 28 demonstra o trecho de código para estas opções.

```
35 | kinectSensor.DepthStream.Enable();  
36 | kinectSensor.SkeletonStream.Enable();  
37 | kinectSensor.SkeletonStream.TrackingMode = SkeletonTrackingMode.Seated;  
38 | kinectSensor.Start();
```

Figura 28 - Habilitando *SkeletonStream*

No trecho de código demonstrado acima, na linha trinta e cinco o fluxo de imagem de profundidade é habilitado. Na linha trinta e seis o fluxo do esqueleto é habilitado e na linha trinta e sete é atribuído o modo sentado ao fluxo do esqueleto. Valem ressaltar que ao atribuir o modo *Seated* (sentado), será rastreado apenas as articulações referentes aos membros superiores e a cabeça.

O SDK é capaz de rastrear o corpo de duas pessoas em frente ao sensor em posição frontal. Em poses laterais terá dificuldades de rastrear os pontos referentes a partes do corpo humano (CATUHE, 2012). O fluxo de dados do esqueleto é representado pela classe *SkeletonStream*. O modelo de esqueleto proposto pelo SDK contém 20 posições, que representam as articulações do corpo humano. A figura 29 abaixo demonstra como são distribuídos os 20 pontos de articulação e seus respectivos nomes.

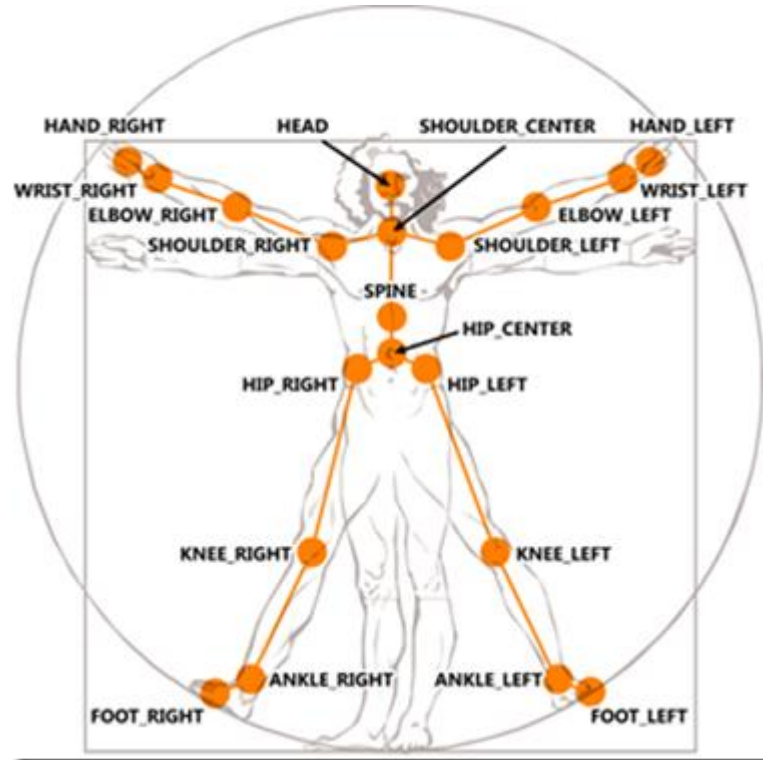


Figura 29 - Esqueleto proposto pela SDK Kinect

Fonte: <http://blogs.microsoft.co.il/shair/2011/06/17/kinect-getting-started-become-the-incredible-hulk>

Assim como demonstrado acima, cada articulação está acessível por métodos das classes que compõem o fluxo de esqueleto. A classe *SkeletonStream* representa o fluxo de dados do esqueleto gerado.

Nesta classe possui o método *OpenNextFrame()* que assim como vimos no fluxo anterior retorna um quadro de esqueleto representado pela classe *SkeletonFrame*. Os dados deste quadro podem ser copiados para um vetor da classe *Skeleton*. A cópia é realizada pelo método *CopySkeletonDataTo*. Este método está acessível através de um objeto *SkeletonFrame*, e recebe como parâmetro um vetor de *Skeleton*, o qual receberá os dados do quadro, em que geralmente irá retornar seis itens de *Skeleton* ao vetor.

A classe *Skeleton* no caso representa o esqueleto rastreado do usuário que se posicionou em frente ao sensor, nela podemos retornar as articulações do esqueleto pela propriedade *Joints* que irá retornar uma coleção de *enum* de *Joint*. Na figura 30 a seguir será mais bem visualizado com o diagrama de classes que representa o esqueleto.

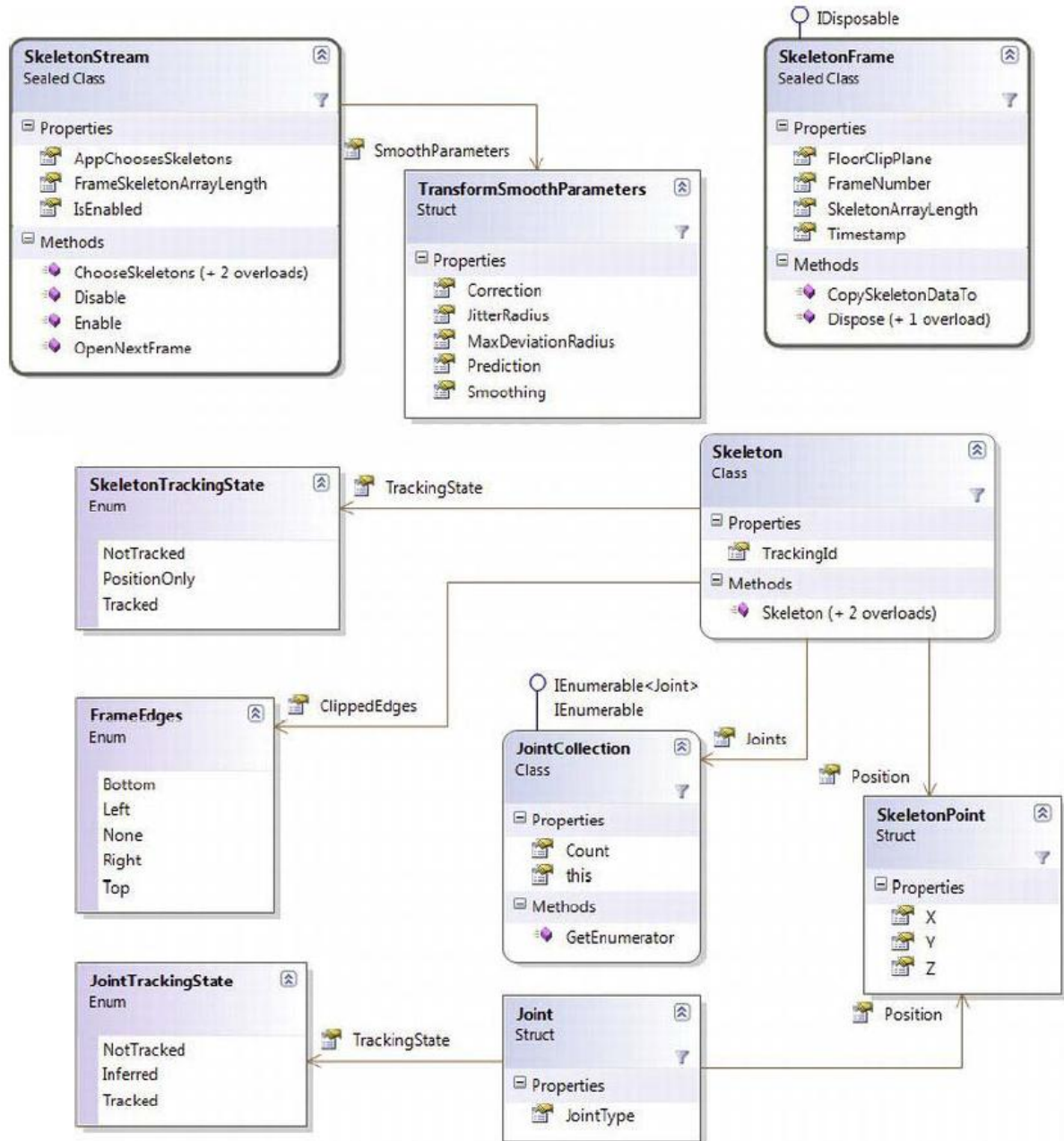


Figura 30 - Classes que representam o fluxo do esqueleto.

No diagrama demonstrado acima, pode-se observar a estrutura *SkeletonPoint* no qual representa um ponto no esqueleto, no qual pode ser o centro de uma esqueleto e também uma articulação ou *Joint*.

As propriedades *TrackingId* e *TrackinState* são bastante importantes no desenvolvimento. Durante o rastreo de duas pessoas simultâneas a propriedade *TrackingId* que irá diferenciar os usuários posicionados. Já a propriedade *TrackingState* indicará se este esqueleto está rastreado ou não, através das opções do Enum *SkeletonTrackingState* que indicará *NotTracked* para não rastreado e *Tracked* para rastreado.

A enumeração denominada *Joint* irá fornecer as coordenadas da articulação a qual está representando, as coordenadas são descritas em três dimensões, ou seja, teremos valores x, y e z. A figura 31 a seguir ilustra os eixos de dimensões no espaço que o esqueleto ocupa.

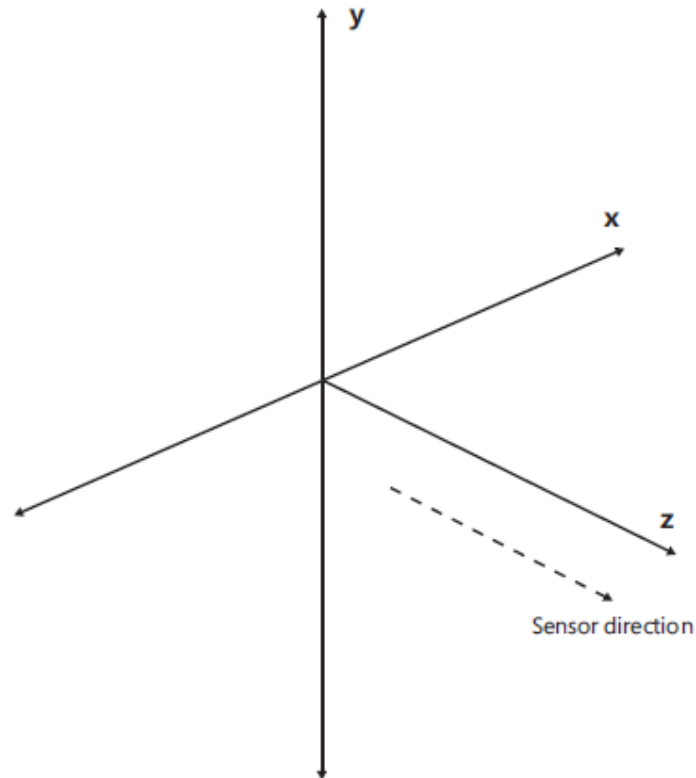


Figura 31 - Eixo 3D do esqueleto.

Fonte: Catuhe, (2012).

No plano demonstrado acima, o eixo x estende-se para a direita do usuário, o eixo y se estende para cima, e o eixo z segue na direção do sensor para o usuário. A forma de capturar os movimentos dos esqueletos em tempo real é semelhante ao fluxo de imagem colorida, utilizando-se do *delegate* para tratar o fluxo de dados que é gerado. Na figura 32 a seguir é demonstrado um trecho de código no qual é criado o evento.

```

38 |         kinectSensor.SkeletonFrameReady += kinectSensor_SkeletonFrameReady;
39 |
40 |     }
41 |
42 |     void kinectSensor_SkeletonFrameReady(object sender, SkeletonFrameReadyEventArgs e)
43 |     {
44 |         var quadroEsqueleto = e.OpenSkeletonFrame();
45 |     }

```

Figura 32 - Manipulador de evento para o fluxo do esqueleto.

Neste trecho apresentado acima o método *OpenSkeletonFrame()* retorna um objeto do tipo *SkeletonFrame* para a variável dinâmica *quadroEsqueleto*. A seguir serão demonstrados mais aspectos da imagem de profundidade, pois através deste tipo de dado poderemos reproduzir e calcular superfícies apresentadas na cena que o sensor Kinect estiver capturando.

3.2.3.3 Imagem de profundidade

O corpo humano é uma superfície que possui características anatômicas semelhantes entre as pessoas, porém a métrica de cada superfície não se repete. Com a imagem de profundidade é possível processar imagens que possa reproduzir suas características anatômicas.

O fluxo de imagem de profundidade que é representado pela classe *DepthImageStream*, é originado das imagens capturadas pela câmera de infravermelho como vimos anteriormente na subseção 3.2.2. Esta imagem é capturada já com a projeção dos pontos de infravermelho produzidos pelo projetor de infravermelho. Assim o SDK processa a distorção dos pontos infravermelhos na superfície e calcula a distância que cada pixel se encontra do sensor. Efeito este semelhante ao que foi abordado no referencial teórico pela topografia de moiré.

O alcance da câmera de infravermelho atinge um espaço de oitenta centímetros do sensor até quatro metros, com ângulo de abertura horizontal de cinquenta e três graus (53°) e de cinquenta e sete graus (57°) na vertical a partir do sensor. A figura 33 abaixo demonstra o espaço que o Kinect possui melhor alcance.

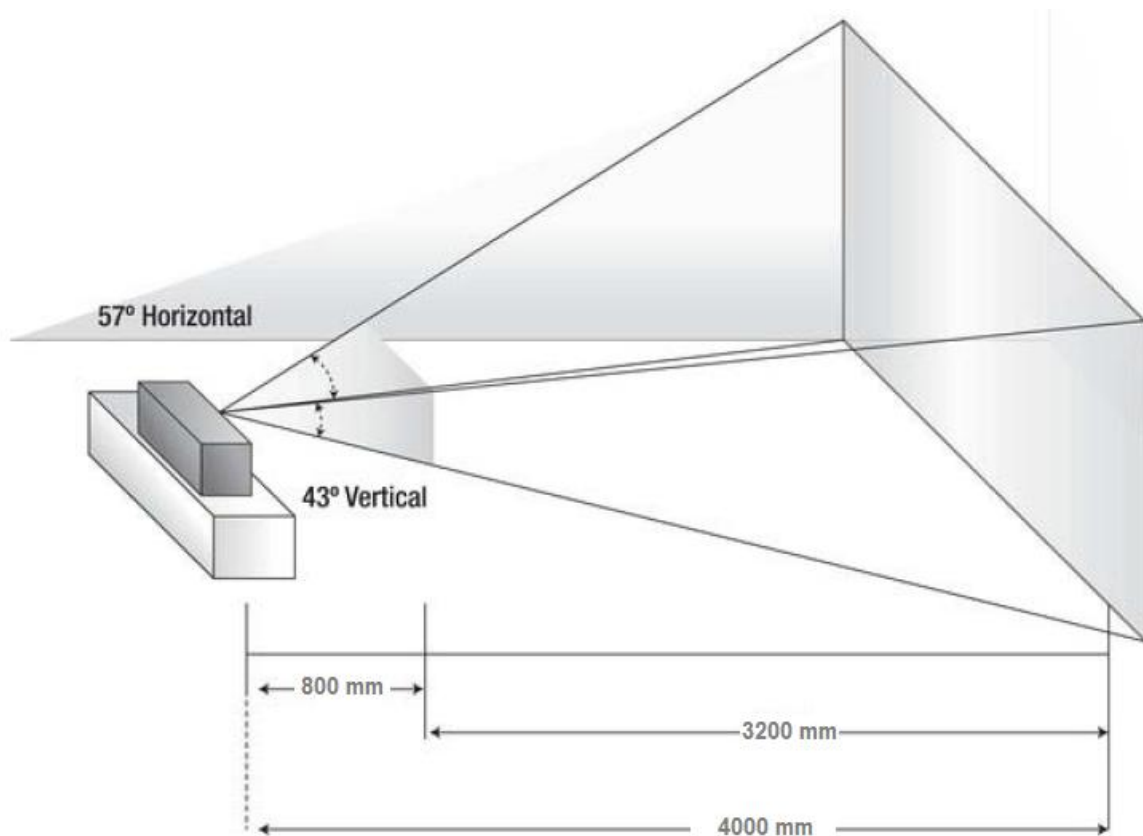


Figura 33 - Área de alcance do Kinect.

Fonte: WEBB e ASHLEY, (2012).

Na figura acima se pode visualizar que o melhor alcance do sensor Kinect está no intervalo de distância de 3200 mm.

O fluxo de imagem de profundidade é bem semelhante ao fluxo de imagem colorida. A imagem de profundidade no caso possui dezesseis bits (16 bits) em cada pixel, sendo que os três primeiros bits expressam a referência do usuário e os treze últimos a distância que este pixel se encontra do sensor.

Assim percorrendo a matriz de pixel e analisando os três primeiros bits de cada pixel da imagem de profundidade é possível distinguir os pixels que pertence ao usuário dos demais pixels da cena.

Ao contrário da imagem colorida, a imagem de profundidade não possui valores referentes às cores. Nos treze bits restantes do pixel, é expresso o valor da distância que este se encontra do sensor.

O fluxo da imagem de profundidade representado pela classe *DepthStream* suporta três tipos de resolução e números quadros por segundo. Estes tipos de resolução estão enumerados da seguinte forma:

- *Resolution640x480Fps30*
- *Resolution320x240Fps30*
- *Resolution80x60Fps30*

Assim como nas imagens coloridas a forma para habilitar o fluxo de imagem de profundidade se dá na mesma forma através do método *Enable()*.

Para capturar os quadros de imagem de profundidade que são gerados pelo sensor Kinect, é mantida a forma como se obtêm o fluxo de imagem colorida. Sendo assim o método *OpenNextFrame()* e o método *OpenDepthImageFrame()*. A figura 34 abaixo exemplifica como habilitar o fluxo de imagem de profundidade, capturar um quadro de imagem e capturando quadros em tempo real através do evento.

```

35         kinectSensor.DepthStream.Enable(DepthImageFormat.Resolution640x480Fps30);
36         kinectSensor.Start();
37         DepthImageFrame quadroDepth = kinectSensor.DepthStream.OpenNextFrame(500);
38
39         kinectSensor.DepthFrameReady += kinectSensor_DepthFrameReady;
40
41     }
42
43     void kinectSensor_DepthFrameReady(object sender, DepthImageFrameReadyEventArgs e)
44     {
45         var quadroProfundidade = e.OpenDepthImageFrame();
46     }

```

Figura 34 - Habilitar e capturar fluxo de imagem de profundidade.

Como demonstrado no trecho de código acima, observa-se que na linha trinta e cinco o fluxo de imagem de profundidade é habilitado e também é passado por parâmetro o tipo de resolução do fluxo.

Na linha trinta e seis o sensor é iniciado, e em seguida na linha trinta e sete é capturado apenas um quadro e armazenado no objeto *quadroDepth* da classe *DepthImageFrame*. Na linha trinta e nove é declarado o evento correspondente à leitura dos dados de imagem de profundidade e da linha quarenta e três até a linha quarenta e seis, compreende o evento que será disparado. Neste evento será capturado o frame do mesmo instante de tempo pelo método *OpenDepthImageFrame()*.

Assim como no fluxo de imagem colorida o fluxo de imagem de profundidade também possui um estrutura contendo classes, enumerações e classes abstratas. Na figura 35 abaixo se pode visualizar e entender melhor pelo diagrama de classes como está definida a imagem de profundidade proposto pela SDK Kinect.

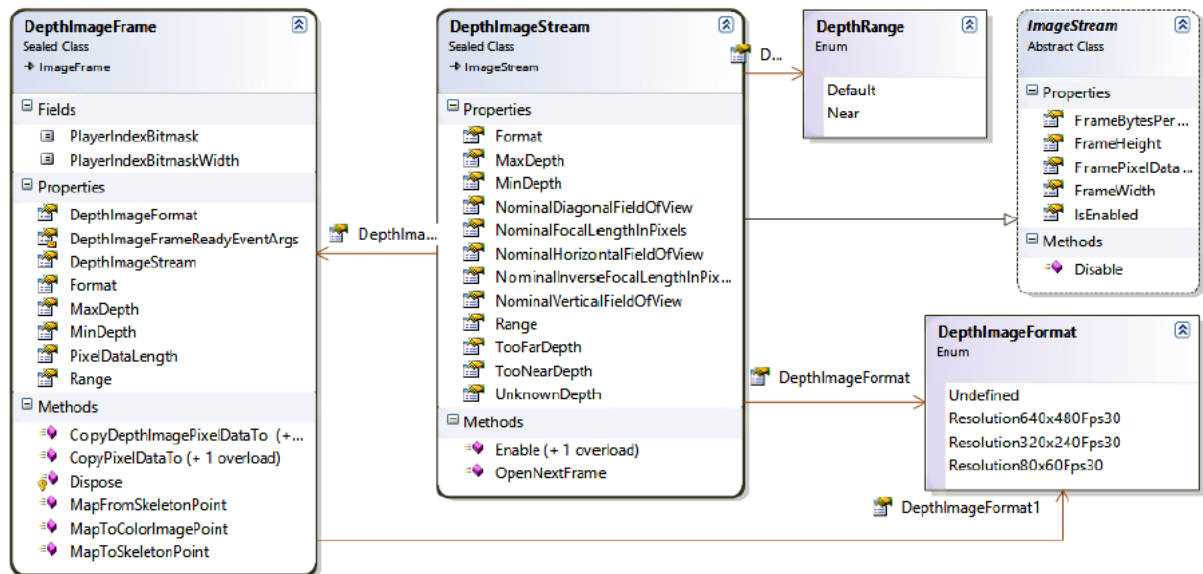


Figura 35 - Classes que representam o fluxo de imagem de profundidade.

Observando o diagrama demonstrado acima nota-se que *DepthImageStream* é classe filha de *ImageStream* assim como a classe *ColorImageStream*. A classe *DepthImageFrame* compõe a classe *DepthImageStream*. A enumeração *DepthImageFormat*, define a resolução que o sensor irá capturar a imagem. Na subsecção a seguir será descrito o método utilizado no processamento de imagens de profundidade com intuito de buscar a direção do fluxo da topografia da superfície. Na seção seguinte serão abordados os métodos utilizados para o desenvolvimento desta ferramenta.

3.3 Metodologia

Este trabalho foi desenvolvido por meio de uma pesquisa aplicada com abordagem qualitativa para obter uma ferramenta de auxílio ao fisioterapeuta na avaliação postural para os fins de detectar e monitorar pacientes acometidos de escoliose. Porém, para que este objetivo fosse obtido, algumas etapas se sucederam. Inicialmente foi realizada uma etapa de estudos acerca dos conceitos envolvidos no trabalho. Para isso, foi utilizado o procedimento metodológico que se baseia em uma pesquisa bibliográfica, com embasamento em livros, artigos, dissertações de mestrado, teses de doutorado e trabalhos de conclusão de curso. Esta etapa resultou na revisão bibliográfica do trabalho, bem como permitiu conhecer as técnicas e métodos envolvidos no problema.

Posteriormente, foi iniciada a fase de levantamento de requisitos da aplicação. Esta atividade foi realizada juntamente com o fisioterapeuta, Pierre Soares Brandão, para obter particularidades da avaliação postural na prática do profissional. Por meio destas conversas e juntamente com o orientador do trabalho, foi definido o objeto de estudo, que consiste na avaliação postural com informações antropométricas geradas pelo Microsoft Kinect. No caso, as variáveis consideradas para o estudo são: o grau de tortuosidade da coluna mensurado pelo ângulo de Coob, grau de gibosidade (medição da rotação vertebral) e grau de modificador sagital (LENKE, et al, 2001).

Por ser um método de baixo custo este projeto contribui para aumentar o quantitativo de avaliações, proporcionando maior eficiência na estimativa de ocorrências de escoliose em diversas faixas etárias da sociedade. Após a mensuração das variáveis citadas, foi dado início a modelagem e elaboração da ferramenta, na qual através dela é possível mensurar desvios posturais em pacientes escolióticos.

Após a modelagem, deu-se início a uma pesquisa exploratória com o intuito de aprofundar conhecimentos a respeito da ferramenta de desenvolvimento Microsoft Visual Studio 2012, SDK Kinect V 1.8. Assim, foram utilizados como bibliografia de apoio no desenvolvimento do software a documentação oficial do Microsoft Kinect (MICROSOFT, 2014) e Catuhe (2012). O software foi desenvolvido e testado em laboratório.

4 RESULTADOS E DISCUSSÃO

Esta seção apresenta os resultados obtidos com desenvolvimento deste projeto, bem como dificuldades enfrentadas e soluções elaboradas. Para executar a avaliação postural visando detectar a escoliose foram desenvolvidas duas funcionalidades utilizando o sensor Kinect. A avaliação com base nos dados do esqueleto proposto pelo SDK Kinect e a avaliação com base nas imagens de profundidade. Para tanto, será apresentado a arquitetura da ferramenta desenvolvida, bem como os detalhes da implementação. No caso da implementação, será abordado as duas formas de avaliação sendo: a avaliação com base nos dados do esqueleto e na imagem de profundidade. Na subseção seguinte serão abordadas algumas soluções propostas para avaliar as variáveis levantadas na análise de requisitos junto ao profissional fisioterapeuta.

4.1 Soluções propostas

Na análise de requisitos junto ao fisioterapeuta Pierre Soares Brandão, ficou constatado que indivíduos que apresentam deformidades na coluna como a escoliose, possuem assimetrias em algumas partes do corpo. Entre as assimetrias que foram mensuradas trata-se da diferença de altura dos ombros e a inclinação lateral da cabeça em relação ao centro da região pélvica.

Desta forma a diferença de altura entre os ombros e a inclinação da cabeça foi mensurada através de ângulos formados pelos pontos de articulação proposto pela SDK Kinect. Assim esta ferramenta calcula estas assimetrias expressando valores do ângulo formado em graus. Na figura 36, a seguir, são demonstrados os vértices dos ângulos formados para serem calculados.

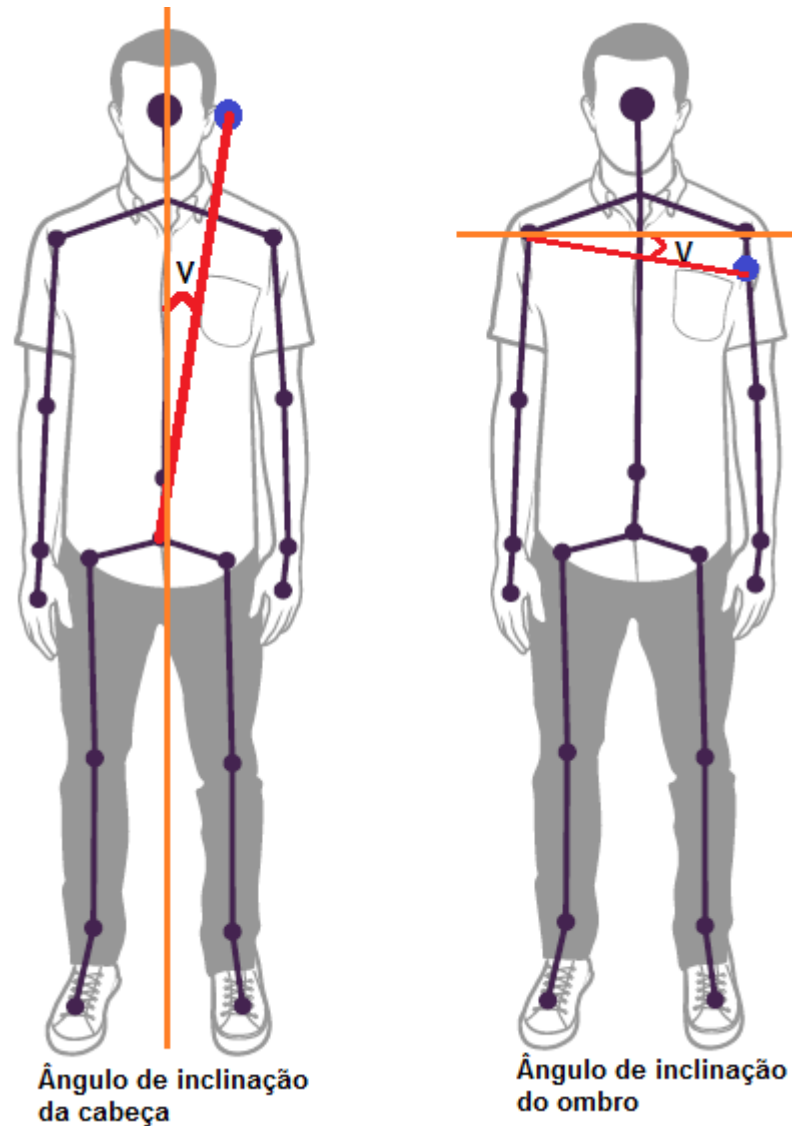


Figura 36 - Ângulos de inclinação dos ombros e cabeça.

Fonte: Adaptada de SDK Kinect.

No desenho acima é apresentado o gráfico que representa o ângulo de inclinação da cabeça. Este ângulo é calculado a partir da linha mensurada no indivíduo que está representado pela linha vermelha. Esta linha compreende o centro da região pélvica até o centro da cabeça. A linha laranjada representa a linha vertical absoluta, assim formando o ângulo “V”. Desta forma o ângulo de inclinação do ombro é formado pela linha vermelha que compreende entre o ombro direito e ombro esquerdo, e a linha laranjada que representa o horizontal absoluto, formando o ângulo “V”.

Na avaliação por imagem de profundidade, foi desenvolvido um método capaz de capturar a imagem de profundidade das costas do paciente. Depois de capturada, esta é passada por parâmetro para um método que irá filtrar os pixels que pertence ao indivíduo

posicionado na cena. Assim esta imagem será processada pelo método *searchColuna()*, no qual efetuará as três etapas mencionadas na subseção 3.3. Onde primeiro será criado o mapa de direção, após o mapa de apontamentos recebido, e em seguida o mapa de fluxo acumulado. Na subseção seguinte será demonstrada a arquitetura da aplicação bem como métodos implementados.

4.2 Arquitetura da ferramenta

A ferramenta foi implementada com interface em WPF 4.5 no padrão *code-behind*, onde os códigos da interface são elaborados em arquivo separado em linguagem XAML. Os eventos disparados pela interface estão codificados em uma classe em linguagem C# na qual representa a interface em questão. Os eventos, por sua vez, requisitam as classes intermediárias que executam as tarefas entre a interface e a SDK Kinect. Na sequência, a SDK fará a comunicação com o sensor Kinect repassando as informações solicitadas. Na figura 37, a seguir, será demonstrada a arquitetura de forma mais detalhada.

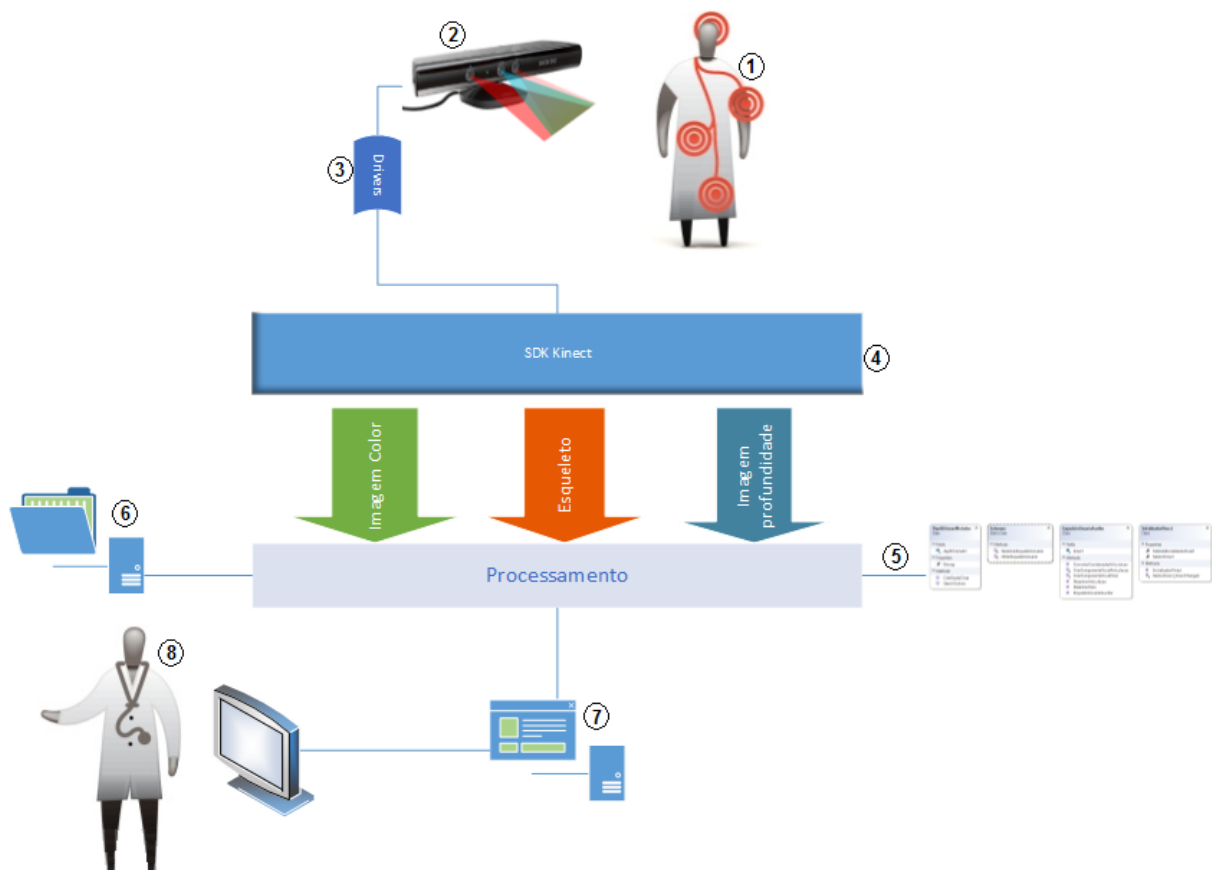


Figura 37 – Arquitetura da ferramenta desenvolvida.

Na figura 37 é demonstrada a visão geral da arquitetura da ferramenta desenvolvida. Nesta, é demonstrado os seguintes elementos na ordem apresentada: paciente, drivers, SDK

Kinect, processamento, serialização, interface, avaliador. A seguir será descrito o papel de cada elemento na arquitetura e detalhes do funcionamento.

1. **Paciente** – O paciente deve posicionar-se em frente ao sensor Kinect a uma distância de um a três metros, conforme descrito anteriormente na seção 2.1.1.3. Para a avaliação postural através dos dados do esqueleto, o paciente deve posicionar-se de frente para o sensor em pé. Para avaliação postural através da imagem de profundidade, o paciente deve estar posicionar de costa para o sensor e em pé, sendo que as costas do paciente devem estar visíveis ao sensor.
2. **Sensor Kinect** – O sensor Kinect deve estar devidamente conectado ao computador pela porta USB, com sua fonte auxiliar ligada a energia. O sensor deve estar posicionado a certa altura na qual o paciente possa estar totalmente visível na tela apresentada na ferramenta. O sensor Kinect irá enviar os dados de imagens para a SDK Kinect.
3. **Drivers** – Os drivers que controlam o hardware do Kinect são instalados automaticamente ao conecta-lo ao computador e seus responsáveis pelo acesso direto com os componentes do sensor.
4. **SDK Kinect** – A SDK do Kinect fornece diversas classes de acesso aos dados gerados ao sensor, dentre elas: imagem colorida, imagem de profundidade, esqueleto e sons. Porém esta ferramenta utilizou somente os três fluxos apresentados na arquitetura acima.
5. **Processamento** - Utilizando as classes e métodos da SDK Kinect, nos quais foram apresentados na seção 2.2.3, foi possível desenvolver classes auxiliares com métodos que podem capturar e processar esses dados e apresentar na interface.
6. **Serializar** – Método que possibilita ao avaliador salvar as avaliações executadas em disco rígido no computador.
7. **Interface** – Interface desenvolvida em WPF 4.5 no padrão code-behind.
8. **Avaliador** – Ator responsável por auxiliar o paciente a se posicionar na forma correta e executar a avaliação através da interface apresentada.

A tela da ferramenta desenvolvida possui dois componentes do tipo *Canvas*. A tela da esquerda apresentado o fluxo de imagem colorida e a tela da direita apresenta a tela de avaliações. Na tela de avaliações é apresentado tanto as avaliações por dados do esqueleto como as avaliações por imagem de profundidade. A figura 38 a seguir apresenta a tela do sistema em execução.

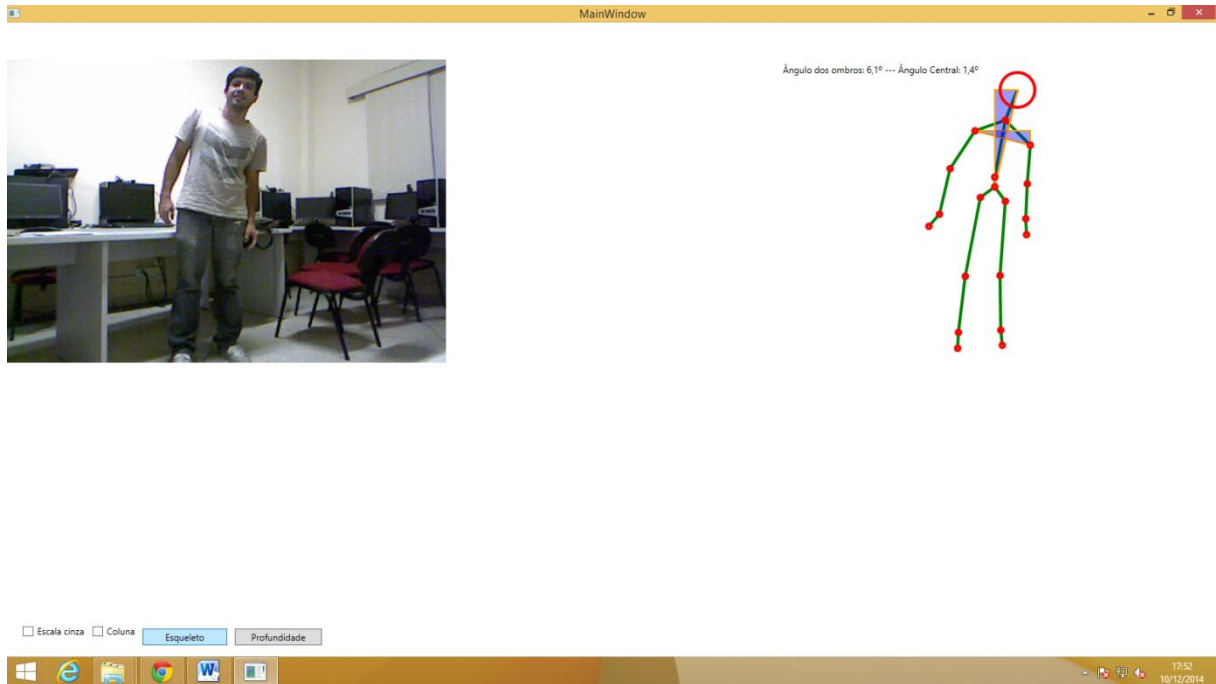


Figura 38 – Tela da ferramenta apresentando a avaliação por esqueleto.

Ao clicar no botão esqueleto este acionará as classes e métodos relacionados a avaliação do esqueleto conforme segue nos diagramas e trechos de código da próxima seção. O diagrama de classes apresentado na subseção a seguir irá demonstrar a camada de processamento da aplicação citada no item cinco da figura 37.

4.3 Diagrama de classes

O objetivo do diagrama de classes é representar as estruturas da aplicação, os tipos de classes e o relacionamento entre elas, além de demonstrar os atributos e propriedades e as operações da ferramenta desenvolvida neste projeto. O diagrama de classes da ferramenta é ilustrado pela figura 39 abaixo.

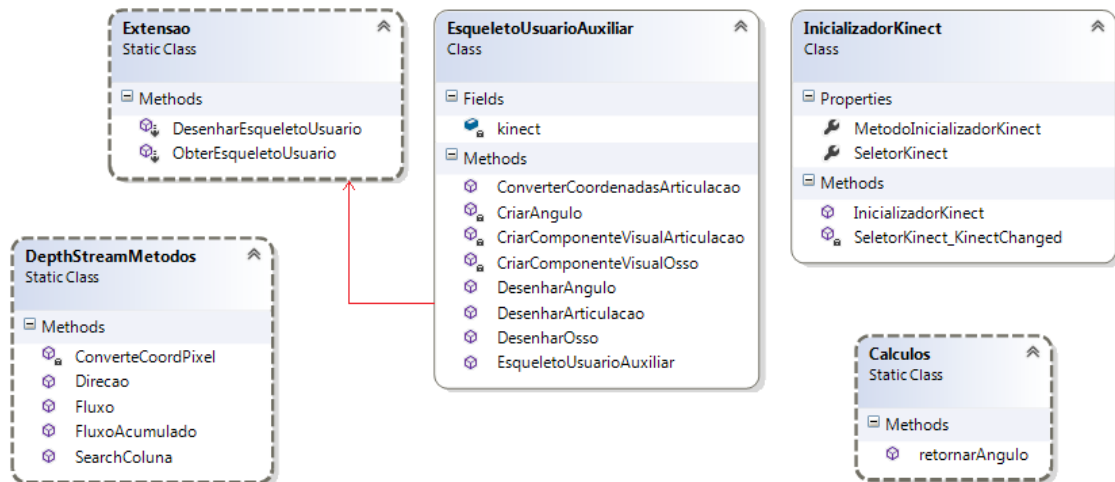


Figura 39 – Diagrama de classes.

Na classe *EsqueletoUsuarioAuxiliar* existe métodos que possibilitam criar objetos que serão compostos os gráfico do esqueleto do usuário. O método *ConverterCoordenadasArticulacao* converte as coordenadas dos dados do esqueleto para as coordenadas da imagem colorida. Assim através da função disponibilizada pela SDK Kinect *kinect.CoordinateMapper.MapSkeletonPointToColorPoint*, o desenho gráfico do esqueleto ficará com a escala adequada ao quadro de imagem colorida. Os demais métodos desta classe são responsáveis apenas por desenhar objetos de acordo com os dados do esqueleto obtidos pelo Kinect. A figura 40, abaixo, demonstra o trecho de código deste método.

```

22 public ColorImagePoint ConverterCoordenadasArticulacao(Joint articulacao, double larguraCanvas, double alturaCanvas)
23 {
24     ColorImagePoint posicaoArticulacao = kinect.CoordinateMapper.MapSkeletonPointToColorPoint(articulacao.Position,
25     kinect.ColorStream.Format);
26     posicaoArticulacao.X = (int)(posicaoArticulacao.X * larguraCanvas) / kinect.ColorStream.FrameWidth;
27     posicaoArticulacao.Y = (int)(posicaoArticulacao.Y * alturaCanvas) / kinect.ColorStream.FrameHeight;
28
29     return posicaoArticulacao;
30 }

```

Figura 40 – Método para converter coordenadas das articulações.

A classe *InicializadorKinect* possui métodos responsáveis por monitorar a conexão do sensor. Assim evita que seja disparada exceções e erros na execução da ferramenta quando não for possível encontrar o Kinect conectado ao computador.

A classe *Extensao* trata-se de uma classe estática, ou seja, seus métodos são acessados diretamente sem que seja necessário instanciar um objeto da classe. Nesta contém dois métodos que possibilitam obter e apresentar de forma gráfica a avaliação do esqueleto. O método *ObterEsqueletoUsuario* recebe como parâmetro um quadro de esqueleto e retorna o esqueleto capturado. A figura 41 abaixo demonstra o trecho de código deste método.

```

35 public static Skeleton ObterEsqueletoUsuario(this SkeletonFrame quadro)
36 {
37     Skeleton esqueletoUsuario = null;
38     Skeleton[] esqueletos = new Skeleton[quadro.SkeletonArrayLength];
39
40     quadro.CopySkeletonDataTo(esqueletos);
41
42     IEnumerable<Skeleton> esqueletosRastreados =
43     esqueletos.Where(esqueleto => esqueleto.TrackingState == SkeletonTrackingState.Tracked);
44     if (esqueletosRastreados.Count() > 0)
45         esqueletoUsuario = esqueletosRastreados.First();
46     return esqueletoUsuario;
47 }

```

Figura 41 – Método para obter o esqueleto

O método apresentado acima recebe um objeto da classe *SkeletonFrame* como parâmetro. Nas linhas trinta e oito, é declarado o objeto *esqueletoUsuario* da classe *Skeleton* que representará o esqueleto do paciente analisado. Na linha trinta e oito é declarado um vetor de esqueletos do tamanho do quadro passado por parâmetro. Após na linha quarenta a função *CopySkeletonDataTo* copia os dados contidos no quadro para o vetor de esqueletos. Assim a partir da linha quarenta e dois é criada uma coleção de objetos, e através da expressão lambda que segue é selecionado o primeiro esqueleto que for válido ou *tracked* como se lê na expressão. Assim o esqueleto é retornado na linha quarenta e seis.

Já o método *DesenharEsqueletoUsuario* é responsável por desenhar o gráfico do esqueleto do usuário no componente da interface apresentada ao avaliador. A figura 42, abaixo, apresenta o trecho de código deste método.

```

13 public static void DesenharEsqueletoUsuario(this SkeletonFrame quadro, KinectSensor kinectSensor, Canvas canvasParaDesenhar)
14 {
15     bool angulo = false;
16     if (kinectSensor == null)
17         throw new ArgumentNullException("kinectSensor");
18     if (canvasParaDesenhar == null)
19         throw new ArgumentNullException("canvasParaDesenhar");
20
21     Skeleton esqueleto = ObterEsqueletoUsuario(quadro);
22     if (esqueleto != null)
23     {
24         EsqueletoUsuarioAuxiliar esqueletoUsuarioAuxiliar = new EsqueletoUsuarioAuxiliar(kinectSensor);
25         foreach (BoneOrientation osso in esqueleto.BoneOrientations)
26         {
27             esqueletoUsuarioAuxiliar.DesenharOsso(esqueleto.Joints[osso.StartJoint], esqueleto.Joints[osso.EndJoint],
28             canvasParaDesenhar, angulo);
29             esqueletoUsuarioAuxiliar.DesenharArticulacao(esqueleto.Joints[osso.EndJoint], canvasParaDesenhar);
30         }
31     }
32 }

```

Figura 42 – Método para representar o esqueleto graficamente.

O método apresentado acima recebe três objetos como parâmetros sendo: um objeto da classe *SkeletonFrame*, um objeto da classe *KinectSensor* e um objeto da classe *Canvas*. Na linha dezesseis a vinte realiza verificações se o sensor está ativo. Na linha vinte e um armazena no objeto da classe *Skeleton* o resultado do método *ObterEsqueletoUsuário* descrita anteriormente, passando como parâmetro o quadro recebido. Na linha vinte e quatro após

verificar se o esqueleto é diferente de nulo é instanciado um objeto da classe *EsqueleUsuarioAuxiliar* a qual possui métodos para desenhar o esqueleto do usuário. Através do laço de repetição *foreach* foi utilizado o método *BoneOrientation* para retornar toda a coleção de articulações do esqueleto, sendo desnecessário declarar todas as vinte articulações do esqueleto.

A classe estática *Calculos* possui um método para calcular os ângulos entre duas articulações chamado *RetornarAngulo*. Este método recebe como parâmetro dois valores do tipo *Joint* e retorna um valor do tipo *Double* contendo o ângulo a ser apresentado na avaliação. A figura 43 abaixo demonstra a implementação deste método.

```

13 public double retornarAngulo(Joint articulacao1, Joint articulacao2)
14 {
15
16     double angulo = 0;
17     Point p1 = new Point();
18     Point p2 = new Point();
19
20     p1.X = articulacao1.Position.X;
21     p1.Y = articulacao1.Position.Y;
22     p2.X = articulacao2.Position.X;
23     p2.Y = articulacao2.Position.Y;
24
25     angulo = Math.Atan2((p2.Y - p1.Y), (p2.X - p1.X));
26     if (angulo < 0)
27         angulo = 2 * Math.PI + angulo;
28
29     return angulo;
30 }

```

Figura 43 – Método para calcular ângulo

No trecho de código acima o ângulo é calculado pelo arco tangente da diferença entre os valores do eixo y dividido pela diferença dos valores do eixo x. Caso o ângulo fique com valor negativo ele é somado por duas vezes π . Assim o método retorna o ângulo da reta entre duas articulações.

Na classe *DepthStreamMetodos* possui métodos para o processamento de imagens de profundidade. Uma imagem é representada por uma matriz, ou seja, podemos dizer que trata-se de um vetor bidimensional. Porém os dados das imagens que são recebidos pelos métodos do SDK são vetores unidimensionais. Então foi necessário criar o método *ConvertCoordPixel*. Este método recebe a posição do pixel na matriz através do objeto *Point* e também o quadro da imagem de profundidade. A figura 44 demonstra o trecho de código deste método.

```

19 private int ConverteCoordPixel(Point p, DepthImageFrame d)
20 {
21     int Index = 0;
22     int x = Convert.ToInt32(p.X);
23     int y = Convert.ToInt32(p.Y);
24
25     Index = x + (y * d.Width);
26     return Index;
27 }
28

```

Figura 44 - Método para converter índice

No trecho de código acima na linha vinte e cinco é calculado o índice correspondente de uma matriz bidimensional para um vetor unidimensional. Logo em seguida na linha vinte e seis o índice é retornado.

O método *SearchColuna* é responsável por receber a imagem do usuário e repassar aos métodos responsáveis pelos processamentos do sentido do fluxo. A figura 45 abaixo demonstra o trecho de códigos deste método.

```

13 public short[] SearchColuna(short[] imagemUsuario, DepthImageFrame frame)
14 {
15     short[] direcao = new short[frame.PixelDataLength];
16     short[] fluxo = new short[frame.PixelDataLength];
17     short[] fluxoAcumulado = new short[frame.PixelDataLength];
18
19     direcao = Direcao(imagemUsuario, frame);
20     fluxo = Fluxo(direcao, frame);
21     fluxoAcumulado = FluxoAcumulado(direcao, fluxo, frame);
22
23     return fluxoAcumulado;
24 }
25

```

Figura 45 – Método *SearchColuna*

No trecho acima na linha dezenove é repassado a imagem do usuário e o quadro da imagem ao método *Direcao*, o retorno deste método é atribuído ao vetor do tipo *short* chamado *direcao*. Assim na linha vinte é repassado o vetor *direção* e o quadro de imagem ao método *Fluxo*, e o seu retorno é armazenado no vetor *fluxo*. Assim na linha vinte e um é repassados os vetores *direção*, *fluxo* e o quadro de imagem para o método *FluxoAcumulado*, e o seu retorno é armazenado no vetor *fluxoAcumulado* do tipo *short*. Na linha vinte e três é retornado o vetor *fluxoAcumulado*.

Como demonstrado acima o método *SearchColuna* utiliza-se dos três métodos que processam o sentido do fluxo, com a finalidade de buscar o vale da superfície que está localizada a coluna vertebral. Na figura 46 abaixo será demonstrado parte do método *Direcao*.

```

36 public short[] Direcao(short[] bitsUsuario, DepthImageFrame quadro)
37 {
38     Point p = new Point();
39     Point v = new Point();
40     //int max = 0;
41     Dictionary<int, int> dir = new Dictionary<int, int>();
42     //int index = 0;
43
44
45     short[] direcao = new short[quadro.PixelDataLength];
46     for (int i = 0; i < quadro.Height - 1; i++)
47     {
48         for (int j = 0; j < quadro.Width - 1; j++)
49         {
50             p.X = j;
51             p.Y = i;
52
53             if (bitsUsuario[ConverteCoordPixel(p, quadro)] != 0)
54             {
103                 //Verificar vizinhança
104                 v.X = j;
105                 v.Y = i - 1;
106                 dir.Add(64, bitsUsuario[ConverteCoordPixel(v, quadro)]);
107
108
109                 v.X = j + 1;
110                 v.Y = i - 1;
111                 dir.Add(128, bitsUsuario[ConverteCoordPixel(v, quadro)]);
144
145                 var m = dir.OrderByDescending(d => d.Value).First();
146                 Int16 key = Convert.ToInt16(m.Key);
147                 direcao[ConverteCoordPixel(p, quadro)] = key;
148                 dir.Clear();
149             }
150         }
151     }

```

Figura 46 - Método de mapa de direção

Na linha trinta e oito do trecho de código acima é declarado o objeto *p* do tipo *Point* que corresponde ao ponto que está sendo visitado. Na linha trinta e nove é declarado o objeto *v* do tipo *Point* que corresponde ao pixel vizinho. Na linha quarenta e um é declarada uma coleção do tipo *Dictionary* que armazenará os valores de toda a vizinhança do pixel. Na linha quarenta e cinco é declarado o vetor *direcao* do tipo *short* do tamanho do quadro de imagem. Assim nas quarenta e seis, e quarenta e oito existe dois laços de repetição, sendo que o primeiro percorre a imagem em linha e o segundo percorre a imagem em coluna. Na linha cinquenta e três é verificado se o pixel corresponde a um pixel do usuário.

A partir da linha cento e três inicia-se a verificação da vizinhança. Após na linha cento e quarenta e quatro é selecionado o pixel vizinho de maior profundidade. E este será atribuído ao pixel correspondente no mapa de direção.

O segundo passo do processo é gerar o mapa de quantidade de apontamentos dos vizinhos. Assim foi desenvolvido o método *Fluxo* que irá contar quantos pixels vizinhos apontam para o pixel visitado. A figura 47 abaixo demonstra parte do trecho de código.

```

157 public short[] Fluxo(short[] direcao, DepthImageFrame quadro)
158 {
159     Point p = new Point();
160     Point v = new Point();
161     int dn = 0;
162
163     short[] fluxo = new short[quadro.PixelDataLength];
164
165     for (int i = 0; i < quadro.Height - 1; i++)
166     {
167         for (int j = 0; j < quadro.Width - 1; j++)
168         {
169             p.X = j;
170             p.Y = i;
171             dn = 0;
172
173             if (direcao[ConverteCoordPixel(p, quadro)] != 0)
174             {
175                 //Verificar vizinhança
176                 v.X = j;
177                 v.Y = i - 1;
178                 if (direcao[ConverteCoordPixel(v, quadro)] == 4)
179                 {
180                     dn++;
181                 }
182             }

```

Figura 47 – Método para o mapa de apontamento

Este método recebe como parâmetro o vetor de direção e o quadro de imagem. Nas linhas cento e cinquenta e nove a cento e sessenta e um, foi declarado o objeto *p* para representar o pixel visitado e *v* para o pixel vizinho, a variável inteira *dn* contará o número de apontamentos dos vizinhos. Na linha cento e sessenta e três é declarado o vetor *fluxo* que representa o mapa contendo a quantidade de apontamentos dos vizinhos. Após os laços de repetições que percorrem linha e coluna nas linhas cento e sessenta e cinco e cento sessenta e sete, inicia-se a verificação da vizinhança. Na linha cento e setenta e oito é demonstrado a primeira verificação de vizinhança, caso este pixel aponta pra ele, a variável *dn* soma mais um. Ao final das verificações o pixel correspondente no mapa recebe a quantidade de apontamentos.

O método que processa o mapa de fluxo acumulado é semelhante ao método de quantidade de apontamento. A diferença que caso o pixel vizinho aponte para o pixel visitado, este terá o valor atual somado ao de todos que o apontam. A figura 48 abaixo demonstra parte do código deste método.

```

249 public short[] FluxoAcumulado(short[] direcao, short[] fluxo, DepthImageFrame quadro)
250 {
251     Point p = new Point();
252     Point v = new Point();
253     int dn = 0;
254
255     short[] fluxoAcumulado = new short[quadro.PixelDataLength];
256
257     for (int i = 0; i < quadro.Height - 1; i++)
258     {
259         for (int j = 0; j < quadro.Width - 1; j++)
260         {
261             p.X = j;
262             p.Y = i;
263             dn = 0;
264
265             if (direcao[ConverteCoordPixel(p, quadro)] != 0 & fluxo[ConverteCoordPixel(p, quadro)] != 0)
266             {
267                 //Verifica vizinhança
268                 v.X = j;
269                 v.Y = i - 1;
270                 if (direcao[ConverteCoordPixel(v, quadro)] == 4)
271                 {
272                     dn += fluxo[ConverteCoordPixel(v, quadro)];
273                 }
274             }

```

Figura 48 – Método *FluxoAcumulado*

O método *FluxoAcumulado* demonstrado acima recebe como parâmetro o mapa de direção, mapa de quantidade de apontamento e o quadro de imagem. Até a linha duzentos e setenta este método é semelhante ao *Fluxo*, descrito anteriormente, porém na verificação da vizinhança caso o pixel vizinho aponte é somado a variável *dn* também o valor do mapa de quantidade de apontamentos. Ao final é retornado o vetor do fluxo acumulado.

Durante a fase de testes não foi possível detectar a coluna vertebral por imagem de profundidade aplicando os métodos acima descritos. A partir do mapa de fluxo acumulado foi gerado uma imagem em escala de cinza do tipo *Gray16*, ou seja, cinza de 16 bits. A escala de cinza deste formato vai de 0 a 65536, onde 0 representa preto absoluto e 65536 representa a cor branca.

Assim percorrendo os mapas de fluxo acumulado em modo de *debug* percebeu-se que foram gerados valores dos apontamentos. Porém os valores acumulados são baixos e não representa o valor significativo para visualizar a visão humana. A figura 49 a seguir demonstra a tela do sistema executando a avaliação por profundidade.

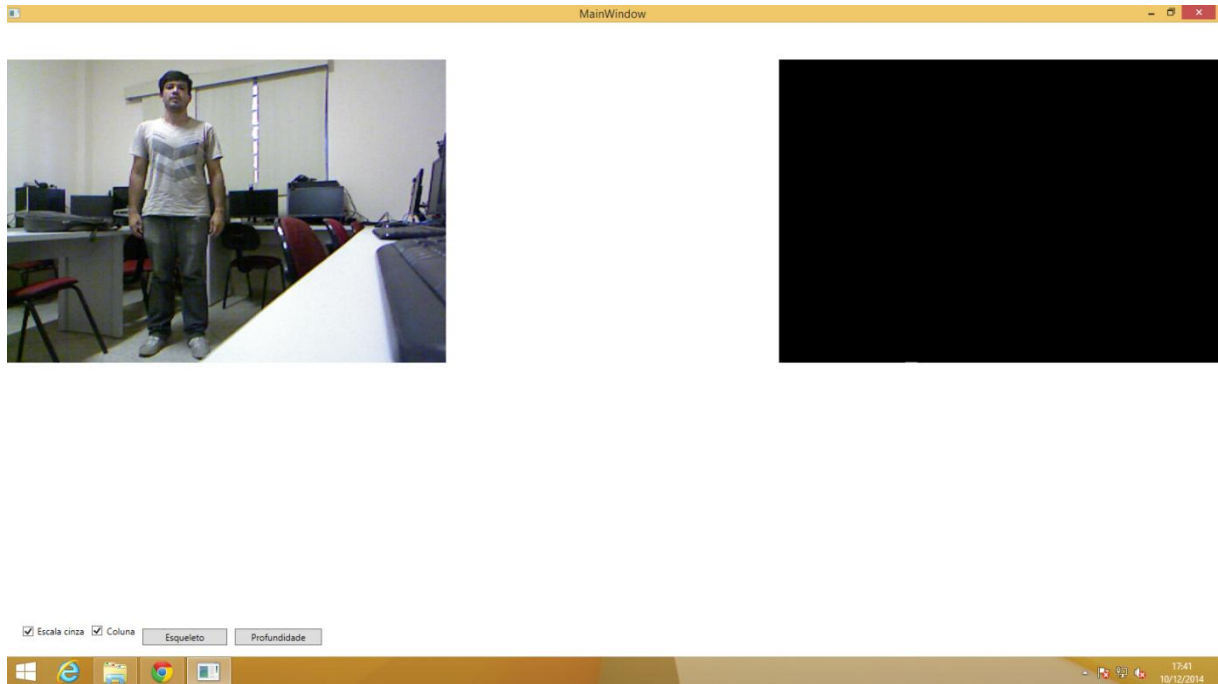


Figura 49 - Sistema gerando fluxo acumulado

Na figura acima demonstra que na imagem não é possível visualizar a região da coluna vertebral. No entanto esta funcionalidade retornou a imagem de profundidade filtrada apenas os pixels do usuário conforme demonstrado na figura 50 abaixo.

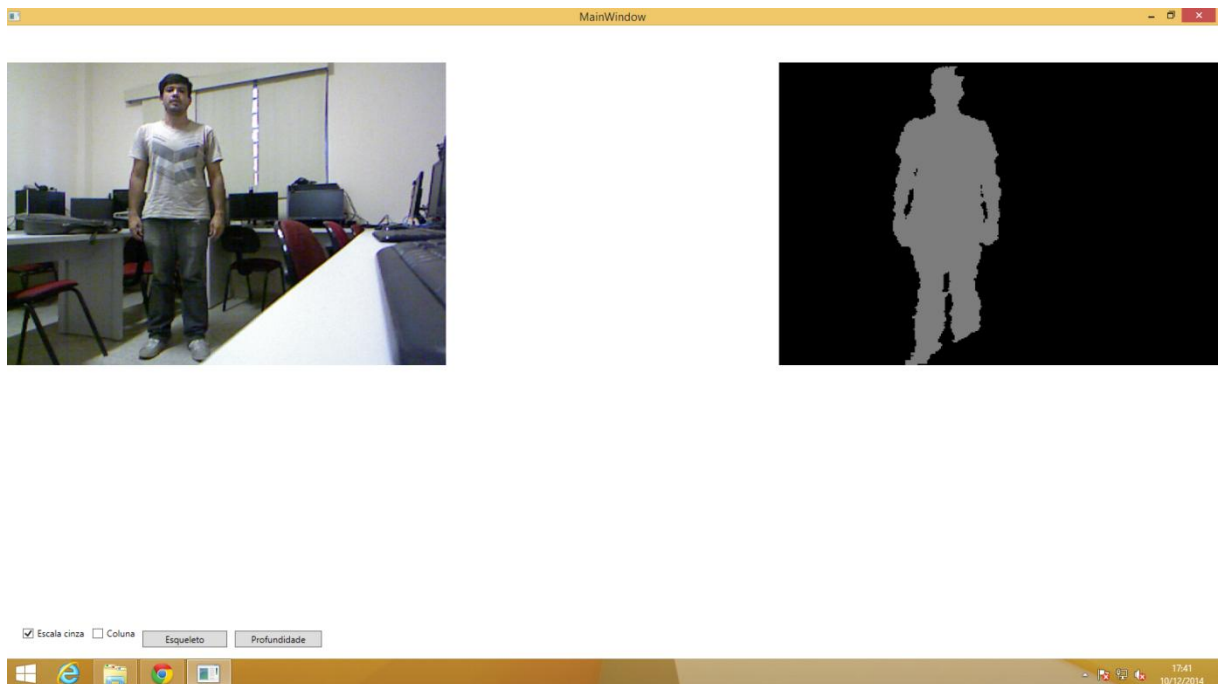


Figura 50 - Pixels do usuário em escala de cinza

Assim seria necessário aplicar contraste nas imagens geradas pelo fluxo acumulado para que se torne visível à visão humana. A seguir no próximo capítulo segue as considerações finais deste trabalho.

5 CONSIDERAÇÕES FINAIS

Este trabalho se deu início com estudo e pesquisas a respeito do funcionamento do sensor Kinect e levantar aspectos da avaliação postural praticada por fisioterapeutas. Assim foram estabelecidas algumas variáveis que poderiam ser mensuradas pela ferramenta.

Percebeu-se que o sensor Kinect em conjunto com seu SDK para desenvolvimento mostrou-se uma ótima ferramenta para aplicações no âmbito da fisioterapia. A partir de dados das articulações do esqueleto este projeto obteve o ângulo de inclinação central e dos ombros de indivíduos, que podem ser úteis no tratamento da escoliose.

Na avaliação por dados do esqueleto a implementação se torna de fácil entendimento, pois o SDK Kinect possui métodos que retornam valores exatos para serem utilizados e em formatos de fácil manipulação, tornando o desenvolvimento muito ágil.

Para obter o ângulo a partir das coordenadas das articulações se torna um pouco mais complexo, pois é necessário executar cálculos trigonométricos para obter as medidas, porém trata-se de um conteúdo bastante conhecido.

A funcionalidade que tinha por objetivo detectar a coluna vertebral por imagem de profundidade não apresentou resultado satisfatório, pois apesar dos métodos elaborados realizar o processamento corretamente, seria necessário aplicar contrastes na imagem ou outras técnicas de processamento de imagem para que assim fosse possível calcular deformidades.

Devido o conhecimento de visão computacional ser bastante amplo, não teria tempo hábil no cronograma deste projeto para executar tais processamentos na imagem de profundidade.

Assim novos trabalhos poderão surgir com objetivo de avaliar apenas por imagem de profundidade. Desta forma, surge novas oportunidade para trabalhos tanto no âmbito da fisioterapia como na computação.

Novos trabalhos podem avaliar a eficiência desta ferramenta na utilização da avaliação postural, assim aferir quanto preciso se torna o uso de sensores como Kinect para este tipo de aplicação.

Na computação novos trabalhos podem buscar novas formas de processar imagens utilizando o kinect aliando-se o sua utilização com bibliotecas disponíveis para processamento de imagens como o OpenCV. Outra possibilidade seria avaliar a utilização de outras SDK semelhantes a SDK Kinect para desenvolvimento em outras linguagens e plataformas operacionais.

6 REFERÊNCIAS BIBLIOGRÁFICAS

- BALLARD, D. H.; BROWN C.M. **Computer Vision**. New Jersey: Prentice-Hall, Inc., 1982.
- BARTL, Ján; FÍRA, Roman; HAIN, Miroslav. Inspection of surface by the Moiré method. **Measurement Science Review**, v. 1, n. 1, p. 29-32, 2001.
- BATOUICHE, M.; BENLAMRI, R.; KHOLLADI, M. K. A computer vision system for diagnosing scoliosis using moiré images. **Computers in biology and medicine**, v. 26, n. 4, p. 339-353, 1996.
- BORENSTEIN, Greg. **Making Things See: 3D vision with Kinect, Processing, Arduino, and MakerBot**. " O'Reilly Media, Inc.", 2012.
- BULLOCK-SAXTON, Joanne. Postural alignment in standing: a repeatability study. **Australian Journal of Physiotherapy**, v. 39, n. 1, p. 25-29, 1993.
- CATUHE, David. **Programming with the Kinect for Windows Software Development Kit**. Washington. Microsoft Press. 2012. 207 p.
- CHAN, Amanda CY et al. Intra-and Interobserver Reliability of the Cobb Angle–Vertebral Rotation Angle–Spinous Process Angle for Adolescent Idiopathic Scoliosis. **Spine Deformity**, v. 2, n. 3, p. 168-175, 2014.
- DEFINO, Helton LA; ARAÚJO, Paulo Henrique Mendes de. Estudo comparativo da medida da rotação vertebral pelos métodos de Nash & Moe e método de Raimondi; Comparative study of the measurements of the vertebral rotation using Nash & Moe and Raimondi methods. **Acta ortop. bras**, v. 12, n. 3, p. 167-173, 2004.
- DOMAGALSKA, Małgorzata E.; SZOPA, Andrzej J.; LEMBERT, Darius T. A descriptive analysis of abnormal postural patterns in children with hemiplegic cerebral palsy. **Medical science monitor: international medical journal of experimental and clinical research**, v. 17, n. 2, p. CR110, 2011.
- FA, TEIXEIRA; GA, CARVALHO. Confiabilidade e validade das medidas da cifose torácica através do método flexicurva. **Revista Brasileira de Fisioterapia**, v. 11, n. 3, p. 199-204, 2007.

FERREIRA, Dalva Minonroze Albuquerque et al. Avaliação da coluna vertebral: relação entre gibosidade e curvas sagitais por método não-invasivo. **Rev Bras Cineantropom Desempenho Hum**, v. 12, n. 4, p. 282-89, 2010.

FERREIRA, Dalva Minonroze Albuquerque et al. Rastreamento escolar da escoliose: medida para o diagnóstico precoce. **Revista brasileira de crescimento e desenvolvimento humano**, v. 19, n. 3, p. 357-368, 2009.

FERREIRA, Dalva Minonroze Albuquerque. **Estudo clínico da mensuração da gibosidade e suas correlações com medidas radiológicas na escoliose idiopática**. 1999. Tese de Doutorado. Universidade de São Paulo.

FERREIRA, Elizabeth Alves Gonçalves. **Postura e controle postural: desenvolvimento e aplicação de método quantitativo de avaliação postural**. 2005. Tese de Doutorado. Universidade de São Paulo.

HEBELA, Nader M.; TORTOLANI, P. Justin. Idiopathic scoliosis in adults: Classification, indications, and treatment options. In: **Seminars in Spine Surgery**. WB Saunders, 2009. p. 16-23.

IUNES, D. H. et al. Confiabilidade intra e interexaminadores e repetibilidade da avaliação postural pela fotogrametria. **Rev Bras Fisioter**, v. 9, n. 3, p. 327-34, 2005.

JENSON, S. K.; DOMINGUE, J. O. Extracting topographic structure from digital elevation data for geographic information system analysis. **Photogrammetric engineering and remote sensing**, v. 54, n. 11, p. 1593-1600, 1988.

Kendall FP, McCreary KE, Provence PG. *Músculos: provas e funções*. São

KING, Howard A. et al. The selection of fusion levels in thoracic idiopathic scoliosis. **The Journal of Bone & Joint Surgery**, v. 65, n. 9, p. 1302-1313, 1983.

KNOPLICH, José. *Endireite as costas: desvios da coluna, exercícios e prevenção*. São Paulo: Ibrasa, 1989.

KRAWCZKY, Bruna; PACHECO, Antonio G.; MAINENTI, Míriam RM. A Systematic Review of the Angular Values Obtained by Computerized Photogrammetry in Sagittal Plane: A Proposal for Reference Values. **Journal of manipulative and physiological therapeutics**, 2014.

LAY, Yun-Long et al. 3D face recognition by shadow moiré. **Optics & Laser Technology**, v. 44, n. 1, p. 148-152, 2012.

LENKE, Lawrence G. et al. Adolescent idiopathic scoliosis a new classification to determine extent of spinal arthrodesis. **The Journal of Bone & Joint Surgery**, v. 83, n. 8, p. 1169-1181, 2001.

LI, X. L. et al. A study on the digital moiré technique with circular and radial gratings. **Optics and lasers in engineering**, v. 45, n. 7, p. 783-788, 2007.

LINO, Antonio Carlos Loureiro. Técnica óptica de Moiré visando a aplicação no estudo de superfícies irregulares. 2002.

MARENGONI, Maurício; STRINGHINI, Stringhini. Tutorial: Introdução à visão computacional usando opencv. **Revista de Informática Teórica e Aplicada**, v. 16, n. 1, p. 125-160, 2009.

MICROSOFT, **Kinect for Windows SDK**. Version 1.8. [S.I.]. Microsoft Corporation. 2014. Disponível em : < <http://msdn.microsoft.com/en-us/library/hh855347.aspx> >. Acesso em: 12 de novembro 2014.

DE MILANO, Danilo; HONORATO, Luciano Barrozo. **VISÃO COMPUTACIONAL**.

Paulo: Manole; 1995.

PERDRIOLLE, RENE. **A Escoliose**. Summus Editorial, 1979.

REEDER, Jean M. Adult scoliosis: A personal experience. **AORN journal**, v. 33, n. 1, p. 35-50, 1981.

SACCO, I. C. N. et al. Confiabilidade da fotogrametria em relação a goniometria para avaliação postural de membros inferiores. **Rev Bras Fisioter**, v. 11, n. 5, p. 411-7, 2007.

SALMINGO, Remel Alingalan et al. Influence of implant rod curvature on sagittal correction of scoliosis deformity. **The Spine Journal**, 2013.

SHOTTON, Jamie et al. Real-time human pose recognition in parts from single depth images. **Communications of the ACM**, v. 56, n. 1, p. 116-124, 2013.

SOUCHARD, Philippe-Emmanuel; OLLIER, Marc. As escolioses: seu tratamento fisioterapêutico e ortopédico. **São Paulo: Realizações**, 2001.

SOUZA, Fabiano Inácio de et al. Epidemiologia da escoliose idiopática do adolescente em alunos da rede pública de Goiânia-GO. **Acta ortop. bras.** [online]. 2013, vol.21, n.4, pp. 223-225. ISSN 1413-7852.

TAKASAKI, Hiroshi. Moiré topography. **Applied optics**, v. 9, n. 6, p. 1467-1472, 1970.

WEBB, Jarrett; ASHLEY, James. **Beginning Kinect Programming with the Microsoft Kinect SDK**. Apress, 2012.

WICK, Jane Maureen et al. Infantile and juvenile scoliosis: the crooked path to diagnosis and treatment. **AORN journal**, v. 90, n. 3, p. 34.