UNIVERSIDADE DO EXTREMO SUL CATARINENSE - UNESC

CURSO DE CIÊNCIA DA COMPUTAÇÃO

BRUNO DE MATTIA AMARAL

IDENTIFICAÇÃO DE OBSTÁCULOS FÍSICOS BASEADA EM IMAGEM APLICADA AO AUXÍLIO NA MOBILIDADE DE PORTADORES DE DEFICIÊNCIA VISUAL

CRICIÚMA

2015

BRUNO DE MATTIA AMARAL

IDENTIFICAÇÃO DE OBSTÁCULOS FÍSICOS BASEADA EM IMAGEM APLICADA AO AUXÍLIO NA MOBILIDADE DE PORTADORES DE DEFICIÊNCIA VISUAL

Trabalho de Conclusão de Curso, apresentado para obtenção do grau de Bacharel no Curso de Ciência da Computação da Universidade do Extremo Sul Catarinense, UNESC.

Orientador: Prof. MSc. Gustavo Bisognin

CRICIÚMA

2015

BRUNO DE MATTIA AMARAL

IDENTIFICAÇÃO DE OBSTÁCULOS FÍSICOS BASEADA EM IMAGEM APLICADA AO AUXÍLIO NA MOBILIDADE DE PORTADORES DE DEFICIÊNCIA VISUAL

Trabalho de Conclusão de Curso aprovado pela Banca Examinadora para obtenção do Grau de Bacharel, no Curso de Ciência da Computação da Universidade do Extremo Sul Catarinense, UNESC, com Linha de Pesquisa em Engenharia de Software.

Criciúma, 25 de junho de 2015

BANCA EXAMINADORA

Prof. Gustavo Bisognin - MSc - (UNESC) - Orientador

Prof. Evânio Ramos Nicoleit - MEng - (UNESC)

Prof. Fabrício Giordani - Esp - (UNESC)

AGRADECIMENTOS

Aos meus pais, Ubirajara de Neres do Amaral e Olívia de Mattia do Amaral pela dedicação, educação, por sempre me incentivarem nos estudos e contribuírem para minha formação pessoal.

Ao meu irmão Eduardo de Mattia do Amaral por sempre estar presente me apoiando e incentivando em minhas escolhas.

A minha namorada Fernanda Nascimento Dagostin pela alegria e carinho nos momentos durante essa jornada.

Ao meu orientador e amigo Gustavo Bisognin por depositar confiança em mim, pela competência, disponibilidade e companheirismo na execução deste projeto.

Meus amigos Diogo Anphiloquio e Renato Mafioletti Macarini que iniciaram essa jornada comigo, pelo companheirismo e colaboração durante todo o curso, e pelos momentos de descontração.

Ao meu grande amigo David Batista Gesuino pelo apoio, parceria, amizade, e por estar presente nos momentos difíceis.

E a todo corpo docente do curso de Ciência da Computação, que contribuiu para que eu chagasse até aqui.

“Take my eyes for what I've seen. I will give my sight to you.”

Bruce Dickinson

RESUMO

Aproximadamente 18% da população brasileira possui algum grau de deficiência visual, sendo que uma quinta parte destes não são capazes de distinguir formas ou luz. As tecnologias assistivas buscam auxiliar portadores de deficiências nas dificuldades causadas por elas, tornando-os aptos a desempenhar certas atividades com mais facilidade. A evolução da tecnologia em placas de desenvolvimento e sistemas embarcados, possibilitou a execução de uma aplicação que realiza o processamento de imagens em tempo real, em um dispositivo portátil. A partir do estudo de visão computacional, estereocopia visual, acessibilidade, síntese de voz, foi possível criar um protótipo de tecnologia assistiva, que auxilia na mobilidade do portador de deficiência visual. O protótipo desenvolvido faz uso da biblioteca OpenCV para processar imagens advindas de um par de câmeras, formando um par de imagens estéreo. Assim a partir deste par é possível, por meio de algoritmos, determinar objetos próximos e detectar obstáculos a frente. A síntese de voz foi acoplada ao protótipo permitindo que o usuário, deficiente visual, se torne ciente dos obstáculos à sua frente. Dessa forma, este trabalho tem o intuito de promover inclusão social aos portadores de deficiência visual, facilitando sua mobilidade, tornando-os mais independentes e como consequência, melhorar sua qualidade de vida.

Palavras-chave: Tecnologia assistiva. Visão computacional. OpenCV. Estereoscopia visual. Síntese de voz.

ABSTRACT

Nearly 18% of the brazilian population has some degree of visual impairment, and a fifth of these are not able to distinguish shapes or light. Assistive technologies goal is to help people with disabilities in difficulties caused by it, making them able to perform certain activities easier. The technological evolution of development boards and embedded systems, allowed the execution of an application that performs image processing in real time, on a portable device. Through the study of computer vision, stereo vision, accessibility, speech synthesis, it was possible to create a prototype assistive technology that aids in mobility of the visually impaired. The prototype uses the OpenCV library to process the resulting images of a pair of cameras, forming a pair of stereo images. Therefore, through this to pair, it is possible, through algorithms, determine nearest objects and detect obstacles ahead. The speech synthesis was embedded on the system allowing the visually impaired user, become aware of the obstacles ahead. Thus, this work aims to promote social inclusion for persons with visual disabilities, assisting in their mobility, making them more independent and as a result, improve their quality of life.

Palavras-chave: Assistive technology. Computer vision. OpenCV. Stereo vision. Speech synthesis.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

ABNT	Associação Brasileira de Normas Técnicas
ARM	*Advanced RISC Machine*
DV	Deficiente Visual
F0	Frequência fundamental
GB	*Gygabytes*
IBGE	Instituto Brasileiro de Geografia e Estatística
IoT	*Internet of Things*
ISO	International Organization for Standardization
JDK	Java SE *Development Kit*
JPEG	Joint Photographic Experts Group
JSON	*JavaScript Object Notation*
OpenCV	Open-source Computer Vision Library
RAM	*Random Access Memory*
RISC	*Reduced Instruction Set Computing*
SO	Sistema Operacional
SoC	*System-on-a-chip*
TA	Tecnologia Assitiva
TTS	*Text-to-speech*
USB	*Universal Serial Bus*

SUMÁRIO

1 INTRODUÇÃO

De acordo com Cook e Hussey (1995), Tecnologia Assistiva é um termo ainda novo, utilizado para identificar todo o arsenal de recursos e serviços que contribuem para proporcionar ou ampliar habilidades funcionais de pessoas com deficiência e, consequentemente, promover vida independente e inclusão. Em outras palavras, pode também ser definida como sendo uma ampla gama de equipamentos, serviços, estratégias e práticas concebidas e aplicadas para diminuir a complexidade dos problemas encontrados pelos indivíduos com algum tipo de deficiência.

Ao se procurar o conceito básico de tecnologias assistivas, encontra-se a definição de que esta, pode ser dividida em conceito, recursos e serviços, dentre as quais o presente trabalho pretende abordar. A divisão de recursos, que, segundo a *American with Disabilities Act* (ADA), podem variar de uma simples bengala a um complexo sistema computadorizado. Estão incluídos ainda, brinquedos e roupas adaptadas, computadores, softwares e hardwares especiais, que contemplam questões de acessibilidade, dispositivos para adequação da postura sentada, recursos para mobilidade manual e elétrica, equipamentos de comunicação alternativa, chaves e acionadores especiais, aparelhos de escuta assistida, auxílios visuais, materiais protéticos e milhares de outros itens confeccionados ou disponíveis comercialmente.

O ser humano é dotado de dois olhos, os quais estão a uma certa distância um do outro, assim cada olho capta uma imagem diferente, já que cada um vê o ambiente de um ângulo diferente. Graças a estas diferenças entre imagens captadas pelo olho direito e pelo olho esquerdo o cérebro humano é capaz de processá-las e dar a noção de profundidade do ambiente, ou seja, ele permite que se saiba a distância dos objetos ao redor (RAPOSO et al, 2004), isso se chama estereoscopia visual, ou então visão estereoscópica.

> Uma parte significativa das informações que obtemos do mundo, enquanto estamos acordados, é através da visão. Nossos olhos fazem um trabalho maravilhoso de girar incessantemente e mudar o foco conforme a necessidade de ver as coisas. Nosso cérebro, um trabalho ainda mais maravilhoso de processar o fluxo de informações de ambos os olhos e criar um mapa 3D do mundo ao seu redor e nos fazer cientes de nossa posição e orientação neste mapa. (BRAHMBHATT, 2013, p. 3, tradução nossa).

A visão computacional é a área da computação que visa o processamento e análise de imagens através de uma máquina. *Open-source Computer Vision Library*

(OpenCV) é uma biblioteca que inclui uma série de algoritmos, métodos e módulos que propiciam ao desenvolvedor uma boa infraestrutura para tratamento e processamento de imagens e vídeos, com ela se pode recortar imagens, alterar o brilho, contrastes, detectar formas e objetos em movimento entre outras funções (BRAHMBHATT, 2013, tradução nossa).

A mobilidade para pessoas com deficiência visual é um fator bastante crítico, e tem chamado a atenção de diversos projetos de inclusão social nos últimos tempos. O avanço da informática associado às políticas públicas e aos incentivos mundiais às tecnologias assistivas, têm direcionado uma série de trabalhos científicos que visam a aplicação da tecnologia computacional à melhoria de vida da população com algum tipo de deficiência. Neste contexto, surge a tecnologia de identificação de obstáculos como sendo um apoio importante para quem possui algum tipo de limitação visual. Tendo em vista o que foi argumentado acima, o presente trabalho, tem como proposta a criação de um protótipo de identificação de obstáculos e distância dos mesmos, que irá informar ao usuário por meio de áudio se há obstáculo à frente, evitando assim, tropeços ou quedas, e consequentemente, melhorando a mobilidade deste usuário.

1.1 OBJETIVO GERAL

Desenvolver um protótipo computacional de reconhecimento de obstáculos através de imagem como auxílio à mobilidade para portadores de deficiência visual.

1.2 OBJETIVOS ESPECÍFICOS

Os objetivos específicos deste trabalho são:

a) explanar sobre a deficiência visual e descrever critérios de acessibilidade, usabilidade e mobilidade para construção de tecnologia auxiliar ao deficiente;
b) descrever o funcionamento de um sistema de visão computacional e processamento de imagens digitais;
c) expor técnicas utilizadas para obtenção de um mapa de profundidade e reconhecimento de obstáculos;

d) utilizar algoritmos e métodos da biblioteca OpenCV para processar imagens adquiridas;
e) desenvolver um sistema capaz de reconhecer obstáculos e assim auxiliar na mobilidade de deficientes visuais.
f) utilizar um *System-on-a-chip* (SoC) para executar e controlar as tarefas do sistema desenvolvido.

1.3 JUSTIFICATIVA

Segundo o Censo demográfico de 2010 (IBGE, 2012), dentre as deficiências investigadas, a mais frequente na população brasileira é a visual. Cerca de 35 milhões de pessoas, 18,8% da população, declararam ter dificuldade de enxergar mesmo com óculos ou lentes de contato. Entre elas, 6,5 milhões disseram ter a dificuldade de forma severa, e mais de 506 mil informaram serem cegas. Hoje existem em torno de 70 deficientes visuais no Brasil que possuem um cão guia para auxiliar no dia-a-dia, e há mais de 2000 na fila para se obter um.

De acordo com estes números, podemos identificar claramente a grande necessidade de criação de tecnologias que possam ajudar estas pessoas a melhorar sua qualidade de vida. Neste contexto, surgem as Tecnologias Assistivas (TA), que apoiam pessoas com deficiência a realizar tarefas cotidianas. Segundo Bersch (2008) a TA deve ser entendida como o “recurso do usuário” e não como “recurso do profissional”, ou de alguma área específica de atuação. Isto se justifica pelo fato de que ela serve à pessoa com deficiência que necessita desempenhar funções do cotidiano de forma independente. Por exemplo, um sistema de identificação de objetos é da pessoa cega ou que precisa de apoio para a locomoção, a cadeira de rodas de quem possui uma deficiência física, a lente servirá a quem tem baixa visão. Em vista disso, acredita-se que a utilização de hardware associado ao conhecimento de computação aplicada, pode ser a chave para a criação de sistemas que consigam identificar obstáculos e avisar ao usuário sobre as condições do ambiente em que está se locomovendo.

Neste trabalho, propomos a utilização de um SoC/microcontrolador associado a *webcams* para a construção do protótipo. Segundo Mazidi, Mckinlay e Causey (2008), um microcontrolador não tem apenas uma *Central Processing Unit*

(CPU), mas também vem equipado com um valor fixo de memória *Random Access Memory* (RAM) e *Read Only Memory* (ROM), e portas de entrada e saída de dados, tudo isso em um único chip. Como estes valores são fixos, e cabem em um único chip, isto torna os SoCs ideais para o desenvolvimento de aplicações que necessitem de baixo custo e operem em um espaço diminuto. Também serão utilizados Softwares Livres, ou seja, softwares que podem ser utilizados tanto em âmbito acadêmico como comercial de forma gratuita, e que possuem seu código aberto para modificações, caso necessário. OpenCV é uma biblioteca que faz parte dessa classe de softwares, e será utilizada para auxiliar no processamento das imagens captadas por câmeras. Através do hardware e da tecnologia atual, é possível captar e processar detalhes do ambiente em que estamos inseridos, com um custo não muito elevado.

Com base no que foi apresentado acima, foi avaliado que existe uma demanda significativa de pessoas que poderiam ser beneficiadas com a solução proposta, bem como a associação de tecnologias existentes de baixo custo, justifica o desenvolvimento, uma vez que está se trabalhando não somente em um projeto aplicado, mas sim um conceito de tecnologia assistiva que deve impactar positivamente na sociedade e cuja implementação pretende melhorar a qualidade de vida das pessoas portadoras de deficiência visual, no que tange a sua locomoção.

1.4 ESTRUTURA DO TRABALHO

A estrutura do trabalho é dividida em sete capítulos.

No primeiro capítulo é feito a introdução do trabalho realizado, apresentado seus objetivos e sua justificativa.

O segundo capítulo trata dos estudos relacionados ao deficiente visual, bem como suas características e dificuldades, e dentro deste contexto é abordado os assuntos de acessibilidade, mobilidade, usabilidade e autonomia. Também são apresentadas as principais características de tecnologias assistivas voltada ao auxílio de portadores de deficiencia visual.

O terceiro capítulo apresenta estudos realizados sobre a visão computacional, explanando sua finalidade e se aprofundando um pouco no processamento de imagens digitais, e no processo de esteroescopia visual.

O quarto capítulo trata do processamento de linguagens naturais, mais

especificamente da tecnologia de sintetização de voz e sua importância para o auxílio ao DV.

O quinto capítulo trata do estudo realizado sobre a biblioteca OpenCV, e apresenta alguns de seus métodos e funcionalidades relevantes para o projeto.

No capítulo sexto são apresentados trabalhos correlatos ao estudo desenvolvido.

O capítulo sétimo apresenta o trabalho proposto, no qual é descrito a metodologia a ser utilizada para o desenvolvimento do mesmo, e os resultados obtidos neste trabalho.

O capítulo oitavo conclui este trabalho, decorrendo sobre os obstáculos encontrados durante o seu desenvolvimento, as soluções aplicadas, os objetivos alcançados, bem como possíveis maneiras de se aperfeiçoar este trabalho.

2 DEFICIÊNCIA VISUAL

O termo deficiência visual refere-se a uma situação irreversível de diminuição da resposta visual, em virtude de causas congênitas ou hereditárias, mesmo após o tratamento clínico e/ou cirúrgico e uso de óculos convencionais. (CASTRO, 1993).

Hoffmann (2002) salienta que a deficiência visual envolve desde pessoas que têm pouca acuidade visual (visão subnormal), que têm capacidade de distinguir a luz mas têm dificuldade de distinguir formas, até a cegueira absoluta, que sequer distinguem a luz.

Os seres humanos necessitam de seus sentidos para interagir com o ambiente ao seu redor. A visão é seu principal meio de recepção de imagens, e por meio desta é possível compreender melhor o ambiente em que se está inserido. No deficiente visual a visão não funciona em sua plenitude, para se ter noção das dificuldades de uma pessoa com deficiência visual, basta fechar os olhos e então tentar se locomover sem nenhum auxílio, mesmo assim, haverá a vantagem de se recordar das características do ambiente que você esteve, o que facilita a orientação (CAZÉ; OLIVEIRA, 2008).

2.1 ACESSIBILIDADE VOLTADA PARA DEFICIENTES VISUAIS

De acordo com a ISO-9050, o conceito de acessibilidade é: a possibilidade de acesso, alcance, percepção, e entendimento para que se possa utilizar de forma correta e com segurança, equipamentos, serviços, edificações e outros elementos (ABNT, 2004).

Este conceito está relacionado à estruturação do ambiente, bem como as oportunidades disponíveis no mesmo, o que muitas vezes é confundido com o conceito de mobilidade, que por sua vez está relacionada com a capacidade do próprio indivíduo de se locomover (AGUIAR, 2010).

A acessibilidade pode ser aplicada tanto em ambientes físicos, quanto a ambiente informacionais. Todo ambiente público deveria ser acessível a todos de forma igual, porém esta tarefa se torna quase impossível, a partir do momento em que ela deve atender diferentes tipos e graus de deficiências, logo, seria necessária uma

grande gama de recursos para ser concretizada. Recursos estes como, softwares, sintetizadores de voz, modificações no ambiente, ferramentas, entre outros, que são importantes meios para propiciar acessibilidade aos deficientes e criam possibilidades para melhorar o seu convívio social (NICHOLL, 2001).

Uma ferramenta que auxilie o deficiente visual a identificar obstáculos ao seu redor, tem o propósito de fazer com que ele possa se movimentar pelo ambiente físico de forma mais prática e acessível, melhorando também sua vida social. Fitz (2008), relata que, tal ferramenta deve ser apresentada de forma agradável, intuitiva, e que de preferência, possa ser operada sem um treinamento prévio.

2.2 USABILIDADE PARA PESSOAS COM DEFICIÊNCIA VISUAL

Para que a ferramenta possa ter as características contextualizadas anteriormente, alguns critérios devem ser observados na sua construção (DIAS, PASSERINO, 2009):

a) as informações e componentes devem ser apresentados ao usuário de forma que ele possa identificar todo o contexto disponível para utilização da ferramenta;
b) os componentes oferecidos devem ser de fácil operação, navegação e localização dentro da ferramenta;
c) a informação e as operações disponíveis devem ser apresentadas ao usuário de uma forma compreensível;
d) o conteúdo apresentado deve ser completo e inteligível para que possa ser interpretado de forma clara pelo usuário.

Os sintetizadores de voz possibilitam que deficientes visuais, de certa forma, leiam os sinais produzidos por uma ferramenta. Segundo Hoffmann (2002), um software leitor de tela, pode passar informações através de áudio para o deficiente visual, assim o mesmo pode usufruir das possibilidades dispostas na tela de um computador como, menus, ícones, textos entre outros. Logo, a sintetização de voz é uma tecnologia importante para promover tanto acessibilidade, quanto usabilidade para deficientes visuais.

Nielsen (2000, tradução nossa) afirma que, estar atento as diretrizes de usabilidade ao desenvolver uma ferramenta permite melhorar significativamente as

tarefas realizadas pelos usuários, tornando mais agradável, rápido e fácil de realizá-las.

2.3 AUTONOMIA E MOBILIDADE PARA PESSOAS COM DEFICIÊNCIA VISUAL

Autonomia e independência são características inerentes a cidadania, e estão diretamente ligadas ao bem-estar do indivíduo. A grande maioria dos ambientes possuem barreiras visíveis, como obstáculos físicos que dificultam a locomoção ou a utilização do ambiente, ou invisíveis, que estão relacionadas à visão que a sociedade tem de um portador de deficiência (PRADO, 1997).

Ter condições de ir a algum lugar, conquistar um emprego, ter um salário digno, poder suprir as próprias necessidades e levar uma vida com relativa autonomia, muitas vezes são dificuldades que o deficiente visual tem de enfrentar no seu dia a dia (HOFFMANN, 2002). Dessa forma o uso de ferramentas e adaptações no ambiente, devem ser incentivadas para que o portador da deficiência possa realizar atividades com mais autonomia.

Segundo Hoffmann (1998), a mobilidade e orientação de uma pessoa que tem um déficit visual vai além de reconhecer os objetos a sua volta, um deficiente visual, faz uso de suas habilidades cognitivas, motoras, afetivas e sociais para se localizar, além de também poder fazer o uso de técnicas ou de ferramentas, como bengala, cão-guia ou mesmo um guia humano.

Tais ferramentas e estratégias junto ao uso de tecnologia assistiva visam maior independência, e possibilitam a realização de tarefas com mais facilidade, tornando-os mais autônomos, uma vez que elas os auxiliam a realizar tarefas por si mesmos sem necessidade de ajuda (BERSCH, 2008).

2.4 TECNOLOGIA ASSISTIVA VOLTADA PARA DEFICIENTES VISUAIS

Tecnologia Assistiva é um termo ainda recente que representa todo recurso ou serviço que tenha o intuito de facilitar, melhorar ou propiciar o exercício de uma função ao deficiente, assim promovendo uma vida mais autônoma e inclusão social (BERSCH, 2008).

Cook e Hussey (1995) citam a tecnologia assistiva como, estratégias,

práticas, serviços e equipamentos que visam diminuir problemas funcionais em pessoas com deficiências.

A Tecnologia assistiva é classificada em categorias. Há a categoria de auxílios para cegos ou com visão subnormal, que refere-se a deficiência visual, nesta categoria enquadra-se qualquer recurso que vise a independência do deficiente visual, e o auxilie na realização de tarefas como, identificação cores, leitura de textos, consultar o relógio, em ter maior mobilidade, entre outros (BERSCH, 2008).

Visto os conceitos de acessibilidade, usabilidade, mobilidade e autonomia, é de grande importância a aplicação de tais conceitos no desenvolvimento da tecnologia para identificação de obstáculos, objeto deste estudo, sendo o produto final uma tecnologia assistiva que auxiliará o deficiente visual. Esta ferramenta de auxilio fará uso da visão computacional, disposta no capítulo seguinte, para reconhecer obstáculos no ambiente em que seu usuário estiver inserido.

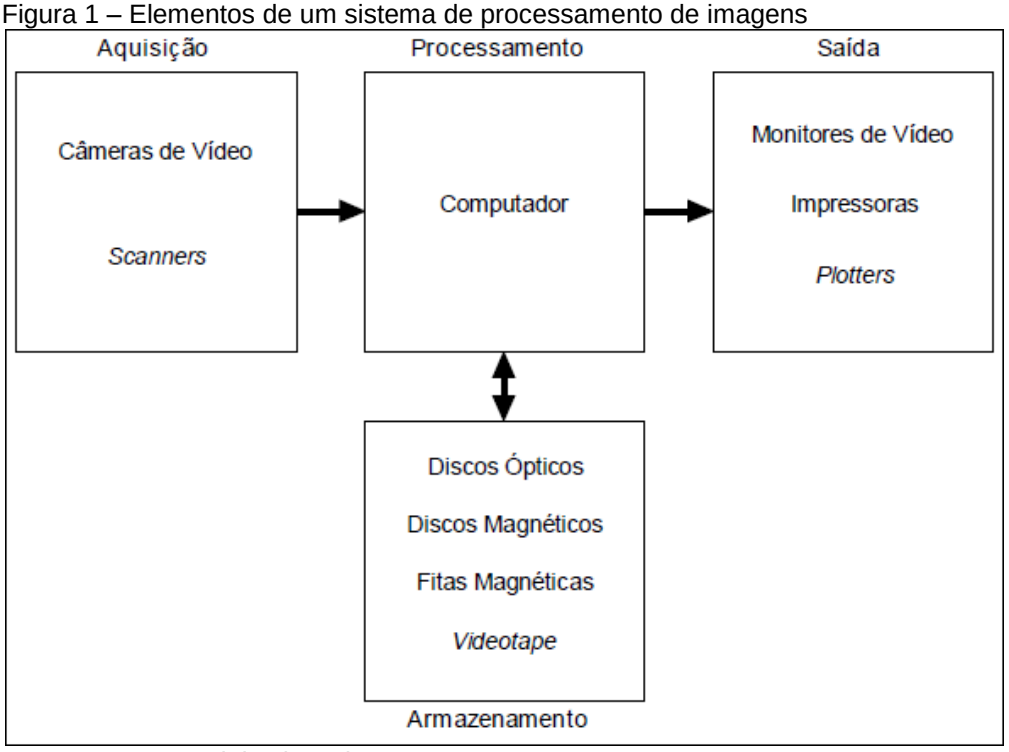
3 VISÃO COMPUTACIONAL

O processamento de imagens refere-se a realização de operações, manipulação e tratamento em imagens afim de extrair informações. A visão computacional surge aplicando computação ao processamento de imagens, onde é possível obter novas informações, representações ou conclusões, transformando imagens em dados, e interpretando-os (FORSYTH; PONCE, 2012, tradução nossa).

O processamento de imagens geralmente está ligado ao poder de processamento da máquina, porém paralelamente os processadores vêm evoluindo continuamente, o que forma um cenário propício ao aparecimento de novas tecnologias e aplicações da visão computacional.

Com o uso de hardware e software, a visão computacional pode ser aplicada a funções em que apenas a visão humana não consegue ou tem dificuldades de realizar, como exemplo, ela vem sendo aplicada ao auxílio em diagnósticos médicos (FORSYTH; PONCE, 2012, tradução nossa).

Figura 1 – Elementos de um sistema de processamento de imagens



Fonte: Marques, Vieira (1999).

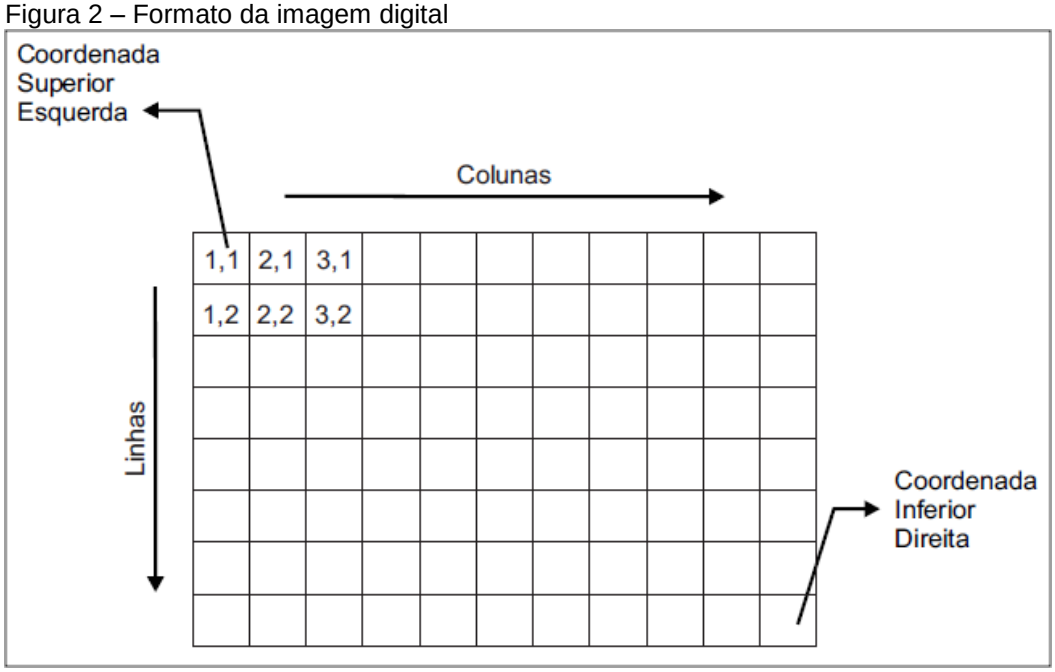
Marques Filho e Vieira Neto (1999) relatam que, as principais operações que se pode efetuar sobre uma imagem são: aquisição, armazenamento, processamento e exibição (figura 1).

3.1 PROCESSAMENTO DE IMAGENS DIGITAIS

A imagem digital é constituída por um conjunto de elementos organizados na forma de grade, onde cada célula desta grade possui uma coordenada definida em um plano (x,y), representado em linhas (y) e colunas (x), a origem da imagem, ou seja, o ponto por onde a imagem começa a ser “construída”, por convenção, é sempre no canto superior esquerdo, como mostra a figura 2 (IBGE, 2001).

De acordo com IBGE (2001) e Martin (2013), uma imagem é composta por um número determinado de elementos, o menor elemento da grade que compõe a imagem é chamado de *pixel* (contração de *picture element*).

Figura 2 – Formato da imagem digital

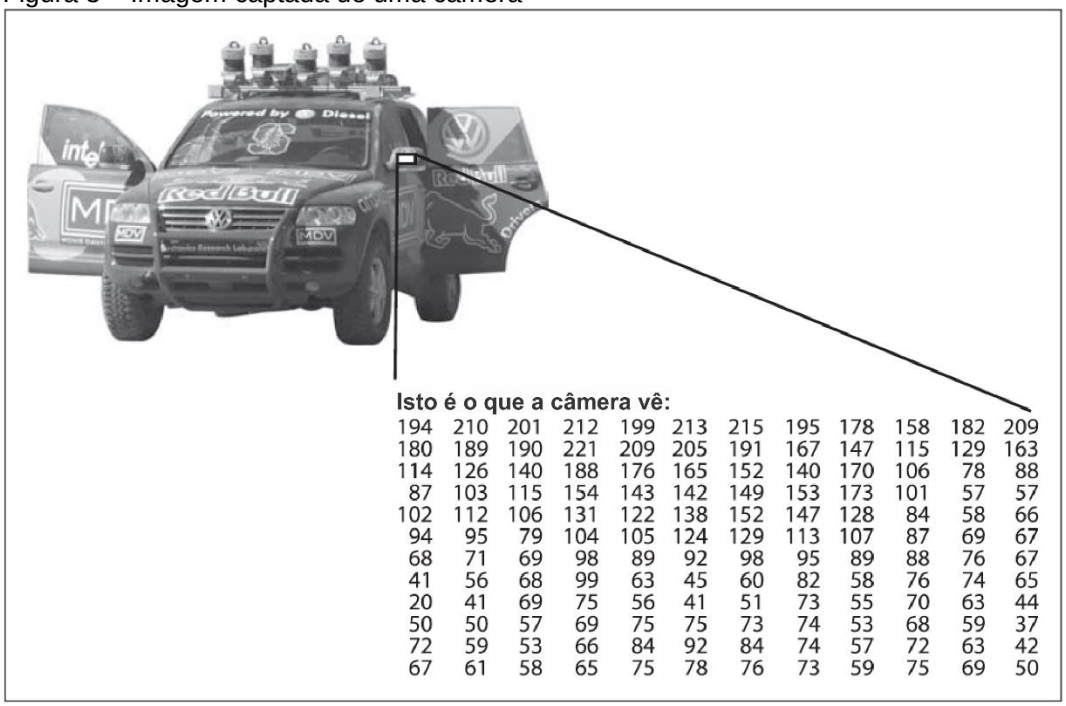


Fonte: IBGE (2001).

Quando uma imagem é captada através de uma câmera, existem interferências que podem distorcer os dados captados, tais interferências podem ser causadas pela luz, movimentos, reflexos, por imperfeições nas lentes da câmera, ruído elétrico causado por outros eletrônicos, entre outros. Tais distorções na imagem

final tornam mais árduo o trabalho da máquina em interpretá-la e processá-la (BRADSKI; KAEHLER, 2008, tradução nossa).

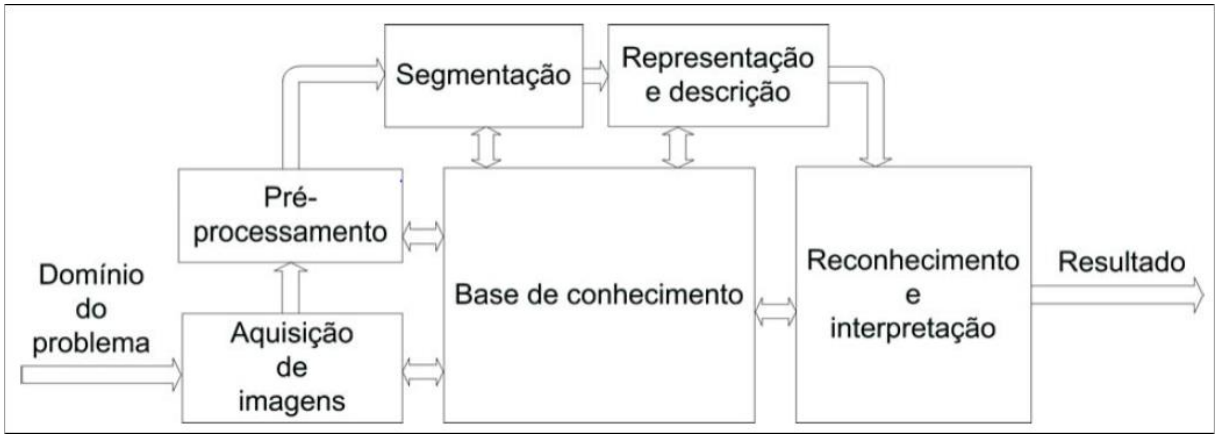
Figura 3 – Imagem captada de uma câmera



Fonte: Bradski e Kaehler (2008, tradução nossa).

Como visto na figura 3, para um computador, uma imagem é apenas uma tabela preenchida com números, é comum se pensar que assim como é fácil para um ser humano identificar que existe um carro na imagem, também será para uma máquina, porém isto é um erro, a máquina precisa processar todos esses números, considerar os erros gerados pelas distorções causadas por interferências externas na imagem, achar padrões comparando com outras imagens já previamente conhecidas de carros, e uma série de outros processos, para então poder afirmar que existe um carro na imagem (BRADSKI; KAEHLER, 2008, tradução nossa).

Figura 4 – Componentes de um sistema de processamento de imagens.



Fonte: Gonzalez e Woods (2000).

De acordo com o diagrama da figura 4, uma imagem passa por algumas etapas durante seu processamento. O primeiro passo neste processo é a aquisição da imagem, que pode ser feito por meio de dispositivos como câmeras, dispositivos de armazenamento, scanners etc. Após obter a imagem ela passa pela etapa de pré-processamento, esta etapa pode envolver técnicas como remoção de ruídos, alteração de contraste, entre outras. O objetivo principal em se pré-processar a imagem é melhorá-la (figura 5) para otimizar o resultado do processamento em si (GONZALEZ; WOODS, 2000).

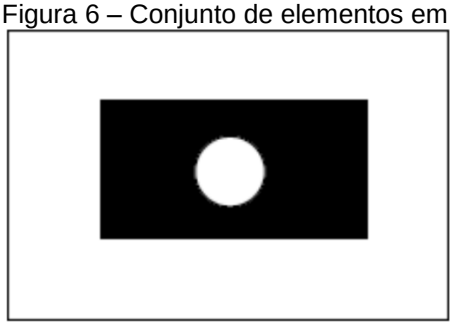
Figura 5 – Antes e após o pré-processamento da imagem

Fonte: Marengoni e Stringhini (2009).

A etapa de *segmentação* é um processo, em que, se tenta extrair ou destacar as informações importantes na imagem. Há duas abordagens principais desempenhados pela segmentação, a primeira é o agrupamento de elementos que se parecem uns com os outros, isto permite montar grupos de pixel semelhantes, estes

grupos são chamados de regiões. A segunda abordagem é formar conjuntos de elementos conforme sua relação a outros grupos, um exemplo é quando se olha para figura 6, pode-se notar a relação entre o círculo interno com o retângulo preto, onde pode-se interpretar como um círculo sobreposto no retângulo, ou um furo no retângulo, em ambas há uma relação entre os dois (FORSYTH; PONCE, 2012, tradução nossa).

Figura 6 – Conjunto de elementos em uma imagem



Fonte: Forsyth e Ponce (2012).

A *segmentação* produz uma saída de dados na forma de *pixels*, contendo as fronteiras e/ou os pixels contidos nas regiões, estes dados devem ser convertidos conforme a necessidade da aplicação. A representação dos dados em forma de fronteiras é mais adequada em aplicações que necessitam processar melhor as características externas, como o formato de um objeto, já a representação por regiões é mais utilizada para se obter informações da estrutura interna, como texturas (GONZALEZ; WOODS, 2007, tradução nossa).

Com a representação dos dados terminada, deve ser descrito como os dados da imagem serão enfatizados (GONZALEZ; WOODS, 2007, tradução nossa). Segundo Forsyth e Ponce (2012, tradução nossa), conforme o foco de cada aplicação, é necessário obter um conjunto de características que represente a aparência da imagem que se busca. Para se descrever características de um modo eficaz, deve-se observar onde existem diferenças em outras instâncias, porém que não sejam afetadas por transformações na imagem, como redimensionamento ou rotação.

A descrição auxilia na interpretação, assim, ao se processar a imagem de um quarto se espera ver uma cama e um travesseiro, porém não uma bola de praia ou uma torradeira (FORSYTH; PONCE, 2012, tradução nossa).

O reconhecimento é responsável por rotular um objeto de acordo com o que foi descrito na etapa anterior (GONZALEZ; WOODS, 2007, tradução nossa). Sendo assim, o processo de reconhecimento deve ser capaz de diferenciar objetos através de padrões, que nada mais são do que arranjos de descrição, porém essa é uma tarefa complexa, pois para executá-la a estrutura de dados deve estar organizada de maneira a facilitar a busca pelas características descritas anteriormente. Humanos podem reconhecer uma cadeira quando veem uma, mesmo que não tenham visto aquele modelo de cadeira antes, e um sistema ideal de reconhecimento deveria ser capaz de fazer o mesmo (FORSYTH; PONCE, 2012, tradução nossa).

O conhecimento em um sistema de processamento de imagens é codificado em forma de base de conhecimento. Esta base de conhecimento pode conter desde informações simples, como detalhamento da imagem (figura 7), até algo complexo, como um conjunto de imagens de satélite conectadas a uma aplicação que detecta mudanças. Dessa forma a base de conhecimento é importante em cada uma das etapas de processamento, pois ela auxilia e realimenta cada etapa do sistema (GONZALEZ; WOODS, 2007, tradução nossa).

Figura 7 – Atributos em classificação de objetos

Fonte: Forsyth e Ponce (2012, tradução nossa).

O resultado do processamento da imagem pode ocorrer em qualquer etapa, por exemplo, focar uma imagem na interpretação humana muitas vezes não vai além da etapa de pré-processamento (GONZALEZ; WOODS, 2000). Este resultado pode ser alguma informação obtida durante o processo, ou ser exibido em forma de imagem.

3.2 VISÃO ESTEREOSCÓPICA

O principal objetivo do processamento de imagens é, portanto, extrair informações de objetos do mundo real e representá-los virtualmente. Apesar de parecer uma tarefa simples, não há sistema que se aproxime da capacidade visual humana, pois ainda há um grande problema na visão computacional, a fase da aquisição da imagem, ou seja, a entrada do sistema, como visto anteriormente, é

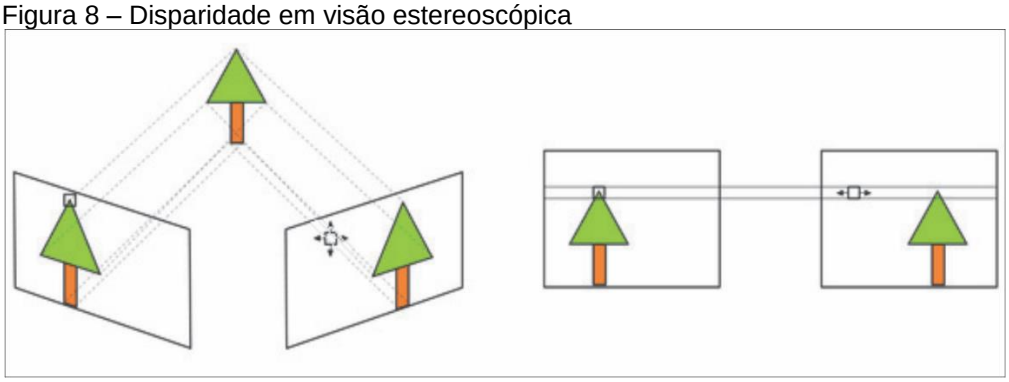
suscetível a variações, e uma pequena variação na entrada pode se tornar uma grande na saída, a ponto de inutilizar o resultado (DOMÍNGUEZ-MORALES, 2012, tradução nossa).

Devido à evolução das tecnologias de áudio que estão cada vez mais presentes no cotidiano, é comum relacionar a palavra *estéreo* ao som. Canais de áudio independentes ligados a caixas de som, ou fones de ouvido, fazem com que o som chegue aos ouvidos de forma ligeiramente diferente, assim o cérebro humano processa-os dando a sensação de imersão ao ambiente (RAPOSO et al, 2004).

Uma câmera estéreo é composta por dois sensores de captura separados por uma distância fixa, chamada de distância interaxial, apontadas na mesma direção. Ela é capaz de captar duas imagens de ângulos diferentes de um mesmo ambiente ou objeto (BRAHMBHATT, 2013, tradução nossa).

Um importante avanço na visão computacional foi a captura de múltiplos pontos de visão, uma vez que a captação de imagens com um único ponto de visão demonstrava limitações. Logo uma câmera estéreo mostra vantagens como maior sensibilidade a profundidade do ambiente, e permite a reconstrução 3D do cenário com mais eficiência (DOMÍNGUEZ-MORALES, 2012, tradução nossa).

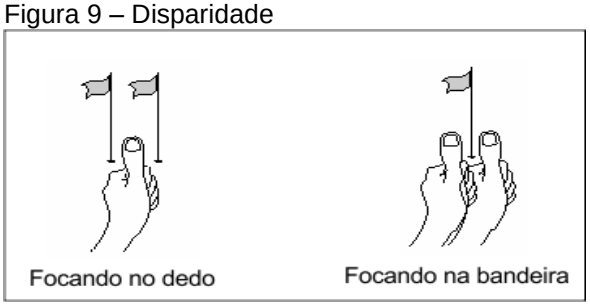
Figura 8 – Disparidade em visão estereoscópica



Fonte: Domínguez-Morales (2012).

Ao se capturar a imagem de um cenário com uma câmera estéreo, é possível encontrar o mesmo ponto nas duas imagens geradas, porém estes estarão em posições diferentes em suas respectivas imagens (figura 8), a distância entre os pontos é chamada de disparidade (FORSYTH; PONCE, 2012, tradução nossa).

Figura 9 – Disparidade

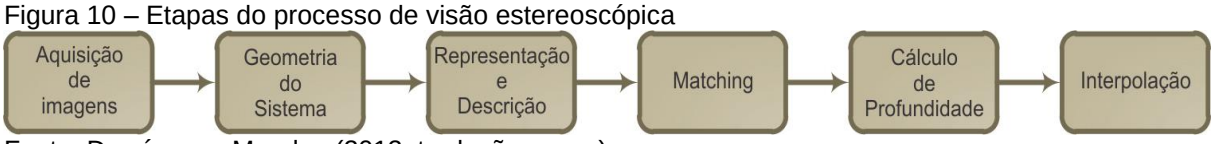


Fonte: Raposo et al (2004).

Para se compreender melhor a disparidade (figura 9), basta estender o polegar a sua frente e focar a visão nele, ao se tentar visualizar um objeto mais ao longe, com um olho de cada vez, cobrindo a visão do outro, pode-se notar que o objeto muda de posição a medida que se alterna a visão de um olho para outro (RAPOSO et al, 2004).

Outra característica inerente a um sistema de visão estereoscópica é a geometria, dependendo da aplicação do sistema a sua geometria física pode ser convergente ou paralela, a visão humana é convergente, pois os olhos podem convergir em um ponto e focar nele (DOMÍNGUEZ-MORALES, 2012, tradução nossa).

Figura 10 – Etapas do processo de visão estereoscópica

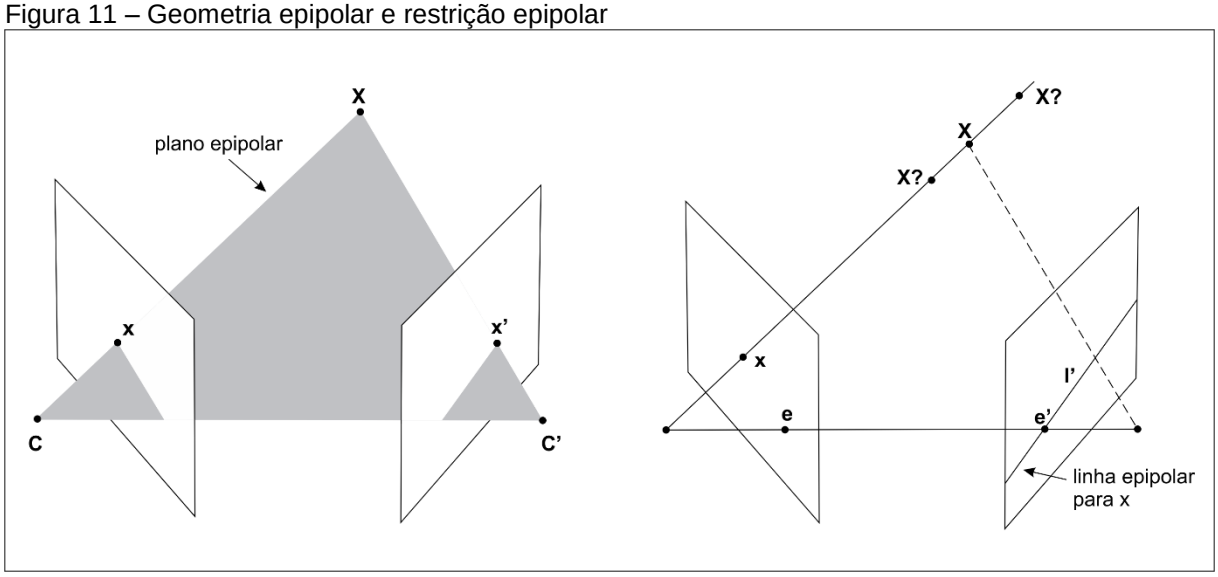


Fonte: Domínguez-Morales (2012, tradução nossa).

Como se pode observar, a figura 10 demonstra que existem duas etapas que são comuns entre o processamento de imagens digitais e o processo da visão estereoscópica: a etapa de *aquisição de imagens* e a de *representação e descrição*. Ambas funcionam da mesma maneira exposta anteriormente.

A princípio, ao se capturar um par de imagens de uma câmera estéreo, cada ponto de uma das imagens deve ter um ponto correspondente na outra. Para se encontrar o ponto em outra imagem correspondente se restringe a região de busca através da geometria epipolar e restrição epipolar (figura 11) (FORSYTH; PONCE, 2012, tradução nossa).

Figura 11 – Geometria epipolar e restrição epipolar



Fonte: Domínguez-Morales (2012, tradução nossa).

O plano epipolar é formado pelo ponto *X*, que é um ponto físico no cenário e as lentes das câmeras *C* e *C'*. Os pontos *x* e *x'* são a representação da projeção do ponto *X*, gerados pelas câmeras, e fazem parte do plano epipolar (FORSYTH; PONCE, 2012, tradução nossa).

Os pontos *e* e *e'* são chamados de epipoles e ficam situados na intersecção da projeção das câmeras com a linha interaxial, que é uma linha imaginária que une o centro das duas câmeras. A linha epipolar é formada ligando o ponto *x* ao ponto *e*, ao se projetar esta linha na projeção gerada pela outra câmera, pode se dizer que x' deve estar em algum ponto da linha *l'* (BRADSKI; KAEHLER, 2008, tradução nossa).

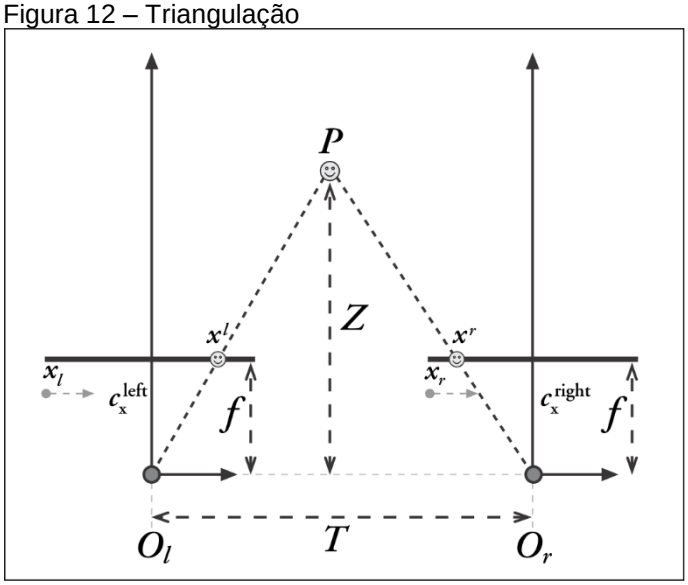
O processo de *matching* consiste em achar cada ponto correspondente de uma imagem na outra, para isso se impõem algumas restrições na área de procura de acordo com a geométrica física do conjunto de câmeras, com a geometria epipolar e com a descrição e representação dos objetos que ocorre na etapa anterior (DOMÍNGUEZ-MORALES, 2012, tradução nossa).

Vale ressaltar que, segundo Domínguez-Morales (2012, tradução nossa), dentre as seis etapas apresentadas na figura 10, a mais importante e que tem maior impacto no resultado final, é o processo de *matching*.

O cálculo de profundidade pode ser feito através do processo de triangulação (figura 12), sabendo a distância interaxial, a angulação das câmeras, e obtido o os pontos correspondentes através do processo de matching, é possível computar a distância de onde o ponto está no cenário até a câmera, logo o resultado

da triangulação é um mapa de profundidade do ambiente (BRADSKI; KAEHLER, 2008, tradução nossa).

Figura 12 – Triangulação



Fonte: Bradski e Kaehler (2008).

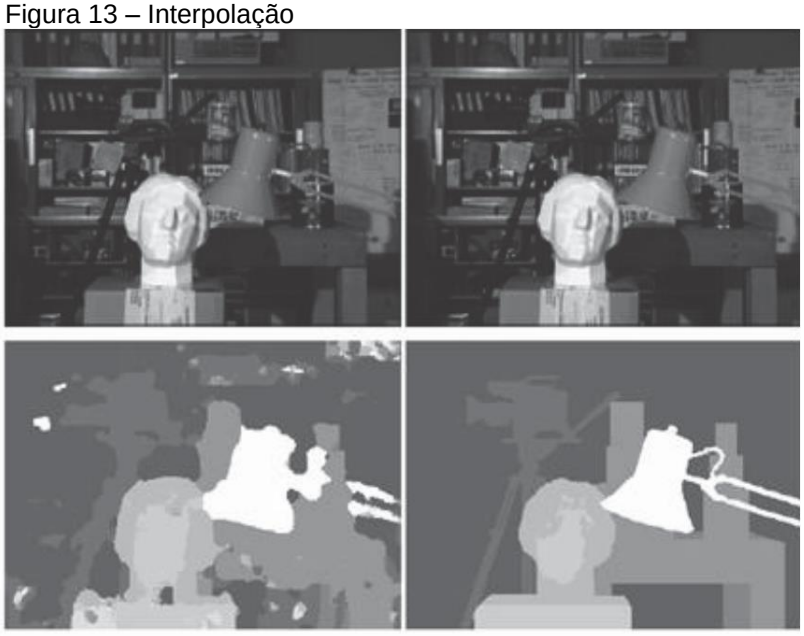
É possível calcular a disparidade pela formula $d = x^l - x^r$, com isso podemos aplicar o teorema dos triângulos semelhantes, onde T é a distância interaxial, Z é a profundidade do ponto P, ou seja, a distância entre P e o ponto médio entre as duas câmeras , x^l e x^r são os pontos projetados de P, e f é a distância focal inerente a câmera , que é a distância entre a sua lente na câmera até seu sensor ou filme (BRADSKI; KAEHLER, 2008, tradução nossa).

$$\frac{T - (x^l - x^r)}{Z - f} = \frac{T}{Z} \quad \Rightarrow \quad Z = \frac{fT}{x^l - x^r}$$

A etapa de interpolação nem sempre é aplicada, ela depende dos mecanismos utilizados nos processos anteriores e do problema que a aplicação tenta resolver, pois em alguns casos, os resultados obtidos no final do processo de cálculo de profundidade são suficientes. Em outros casos, os resultados mostram uma grande quantidade de pares de pontos com as suas correspondências em ambas as imagens, porém estes pontos ficam dispersos, e se torna necessário a interpolação entre eles (DOMÍNGUEZ-MORALES, 2012, tradução nossa).

Para se resolver o problema da interpolação geralmente se usa a interpretação do mapa de disparidade que é gerado pelo processo de *matching,* com isso o sistema extrai uma função contínua para obter a profundidade de qualquer ponto nas projeções de ambas as câmeras (DOMÍNGUEZ-MORALES, 2012, tradução nossa).

Figura 13 – Interpolação



Fonte: Domínguez-Morales (2012, tradução nossa).

A figura 13 apresenta duas imagens acima, tiradas do mesmo cenário, de posições diferentes, e duas abaixo, a imagem à esquerda antes do processo de interpolação e a direita após o processo.

A partir das informações interpretadas na visão computacional, é possível reconhecer obstáculos mais próximos, e então se faz necessário tornar o usuário ciente deles por meio da síntese de voz, contextualizada no capítulo seguinte.

4 PROCESSAMENTO DE LINGUAGENS NATURAIS

As linguagens utilizadas pelo ser humano com propósito de comunicação em geral, seja por fala ou escrita, são chamadas de linguagens naturais. Uma linguagem é uma estrutura e sistema descrito por regras, no caso da forma escrita ela é composta por cadeias de caracteres que formam símbolos, abreviaturas, siglas, acrónimos, numerais, fórmulas matemáticas, entre outros (SILVA, 2008).

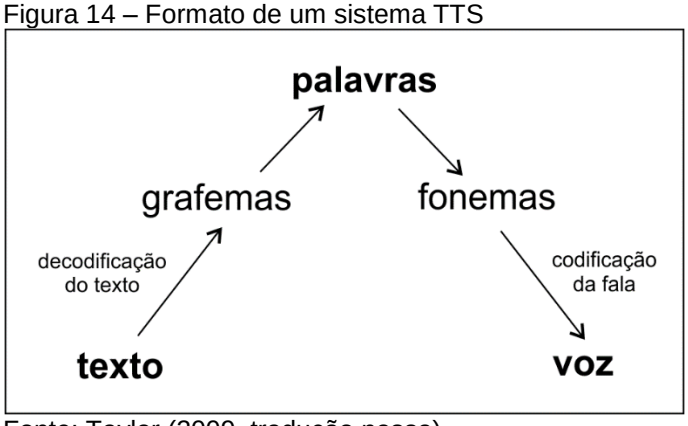
O processamento de fala e de linguagem engloba técnicas computacionais que processam a linguagem escrita ou falada. Tais técnicas podem estar presentes em aplicações simples e comuns, como a contagem de palavras, até aplicações mais complexas, como um sistema que responda questões online, ou que faça tradução em tempo real através do reconhecimento da fala (JURAFSKY; MARTIN, 2008, tradução nossa).

4.1 SINTETIZADORES DE VOZ

Algumas ferramentas garantem que os portadores de deficiência visual possam receber as informações do ambiente. A síntese de voz é uma destas ferramentas, e compreende os processos pelos quais a voz humana é reproduzida artificialmente.

A ocorrência mais comum de síntese de voz, é o chamado *text-to-speech* (TTS). Em um sistema TTS (figura 14) a sua entrada de dados é um texto, ou seja, um conjunto de caracteres, e o resultado em sua saída é o áudio em forma de fala do conteúdo extraído do texto (TAYLOR, 2009, tradução nossa).

Figura 14 – Formato de um sistema TTS



Fonte: Taylor (2009, tradução nossa).

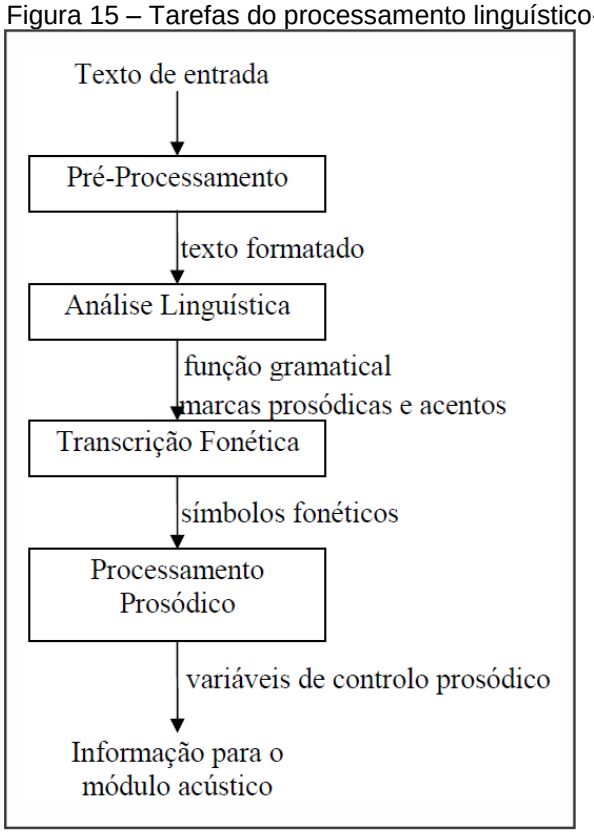
Este tipo de sistema tem em especial o interesse de tornar acessível sistemas de informação de qualquer natureza, como em estações de transporte e em comércios, e tem particular utilidade para os utilizadores que tenham dificuldade em ler, como cegos ou analfabetos (TEIXEIRA; BARROS; FREITAS, 2003).

Teixeira, Barros e Freitas (2003) dividem o processo de síntese de voz em dois blocos principais: o processamento linguístico-prosódico e processamento acústico.

4.1.1 Processamento linguístico-prosódico

O processamento linguístico-prosódico apresenta alguns processos realizados ao nível linguístico, como o pré-processamento do texto para a conversão de acrónimos, abreviaturas e numerais em palavras. Todo o processamento deste bloco (figura 15) tem por objetivo filtrar e normalizar o texto, para então entregar ao próximo bloco informações essenciais ao processamento acústico (TEIXEIRA; BARROS; FREITAS, 2003).

Figura 15 – Tarefas do processamento linguístico-prosódico



Fonte: Teixeira, Barros e Freitas (2003).

Palavras são parte fundamental das linguagens naturais, sejam elas faladas ou escritas, e por isso qualquer área do processamento de linguagens naturais, como a síntese ou reconhecimento de voz, necessita de um vasto conhecimento das palavras referentes à língua trabalhada (JURAFSKY; MARTIN, 2008, tradução nossa).

O pré-processamento ou normalização de texto converte em palavras toda a espécie de símbolos, abreviaturas, siglas, acrónimos, numerais, e fórmulas matemáticas. Comumente é usado dicionários ou léxicos para resolvê-lo, ou seja, uma lista de entradas e de suas respectivas expansões ortográficas, ou imediata transcrição fonética (SILVA, 2008).

Decodificar a linguagem natural é uma tarefa complexa, uma das dificuldades enfrentadas são as chamadas ambiguidades (TAYLOR, 2009, tradução nossa). As ambiguidades acontecem quando, em uma entrada do sistema, existem múltiplas alternativas que podem ser construídas a partir da estrutura linguística (JURAFSKY; MARTIN, 2008, tradução nossa).

O primeiro passo é organizar o texto. Para isso o sistema começa separando o texto em frases, ao identificar caracteres como “ponto (.)”, “interrogação (?)”, “exclamação (!)”, ou “reticências (…)”, o sistema reconhece o final de uma frase, porém existem exceções que o sistema deve ignorar como, por exemplo, a abreviatura da palavra “Senhor” indicada por “Sr.”, onde o caractere de “ponto (.)” não representa final de frase (SILVA, 2008).

Na fase de análise linguística é feita a divisão das frases ou sentenças em palavras, que é uma tarefa relativamente fácil, sendo que as palavras são divididas por espaços em branco ou hifens (SILVA, 2008).

Em seguida é realizada a separação silábica, marcada a sílaba acentuada, ou eventuais graus de acento. Porém esta etapa não é simples como a divisão de uma frase em palavras, nela é necessário a aplicação de um algoritmo que utilize as regras da linguagem trabalhada para efetuar esta tarefa (TEIXEIRA; BARROS; FREITAS, 2003).

A transcrição fonética tem um papel importante na síntese da fala, porém é um problema ainda longe de estar solucionado (SILVA, 2008). Nesta etapa o texto deve ser transcrito em sequências de fones (TEIXEIRA; BARROS; FREITAS, 2003).

O termo fone é utilizado para descrever o som único produzido pela fala, e pode ser entendido como uma unidade básica da fala (TAYLOR, 2009, tradução nossa).

A transcrição envolve praticamente todos os aspectos de estudo da fala humana, uma vez que é necessário nela relacionar os dados transmitidos pela fala com sua forma escrita (TAYLOR, 2009, tradução nossa).

Ao se realizar a transcrição fonética pode ser utilizado um dicionário contendo a sequência de fones das palavras, ou máquinas de estados. Mas é importante ressaltar que, qualquer que seja o método usado, é fundamental se ter a informação sobre a sílaba tônica da palavra, que mesmo em palavras sem acentuação, podem diferir no sentido de uma palavra a outra, como em “espeto”, que se lido com o “e” aberto é considerado um verbo, porém se lido com o “e” fechado é considerado um substantivo (TEIXEIRA; BARROS; FREITAS, 2003).

A transcrição da ortografia para fones produz um dos componentes mais importante para o sistema TTS. Outro componente importante é a especificação da prosódia. Prosódia é o termo usado para se referir a aspectos da pronúncia, como ritmo, duração e entonação (JURAFSKY; MARTIN, 2008, tradução nossa).

O processamento prosódico é o responsável por tornar o aspecto da voz sintética mais próximo da natural possível (TEIXEIRA; BARROS; FREITAS, 2003).

4.1.2 Processamento acústico

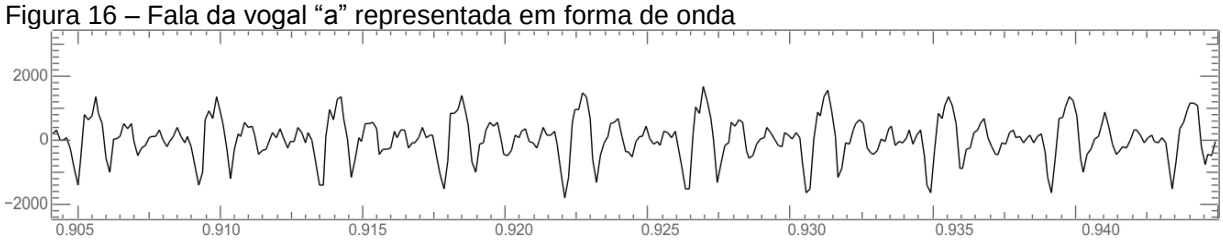
O processo de fala do ser humano é um processo complexo, e ainda não compreendido totalmente. É possível chegar a um modelo aproximado da produção da fala humana, porém a falta de compreensão de todos os mecanismos que formam a fala, impossibilita a criação de um modelo capaz de reproduzi-la com uma boa precisão (TAYLOR, 2009, tradução nossa).

Nesta parte do sistema, os dados recolhidos do processamento prosódico são reunidos e concatenados de acordo com modelos acústicos, afim de se extrair ao final o sinal da fala em forma de onda (TEIXEIRA; BARROS; FREITAS, 2003).

Para se compreender melhor a produção da fala, Jurafsky e Martin (2008, tradução nossa) explicam que, a fala é gerada por mudanças na pressão do ar através de um alto-falante ou de um orador, estas mudanças podem ser representas em um

gráfico onde o eixo y representa o grau de compressão ou rarefação do ar, e o eixo x o tempo (figura 15).

Figura 16 – Fala da vogal “a” representada em forma de onda



Fonte: Jurafsky e Martin (2008).

Pode se notar, no gráfico (figura 16), que a forma de onda se repete por 9 ciclos. A quantidade de vezes que a onda se repete em 1 segundo é chamada de frequência. Também pode-se observar que a duração do sinal analisado foi de 0.036 segundos, logo é possível calcular a frequência: $9 \div 0.036$, então obtém-se o resultado de 250 ciclos por segundo, cuja a unidade representativa é Hertz (JURAFSKY; MARTIN, 2008, tradução nossa).

A maior parte do processamento acústico foca na síntese da entonação da fala. A entonação é representada em sua maior parte pela frequência fundamental (F0), tanto que muitas vezes a entonação é definida como a manipulação da F0. A segunda parte mais importante no processamento acústico é o tempo em que os fones são executados, do tempo de execução dos fones provem a afinação e o ritmo da fala (TAYLOR, 2009, tradução nossa).

Toda forma de onda complexa pode ser decomposta em ondas menores e mais simples, uma analogia musical a isto, seria um acorde, que é formado por algumas notas, e cada nota com seu som característico. Assim é o que acontece com a fala, os fones são como as notas de um acorde, e uma sequência de fones é como o acorde, formando a onda sonora de uma palavra por exemplo (JURAFSKY; MARTIN, 2008, tradução nossa).

Uma vez conhecida e processada a entrada, os dados gerados são submetidos a um modelo acústico. Existem vários tipos de modelos, o objetivo principal deles é criar a partir da entrada a F0 da fala, porém como dito anteriormente nenhum deles conseguem reproduzir com perfeição a F0, devido à complexidade do problema e também as inúmeras variações possíveis das saídas que podem ser

geradas pela entrada. Porém os modelos conseguem extrair e salientar aspectos que são linguisticamente importantes preservando-os e descartando os demais (TAYLOR, 2009, tradução nossa).

5 BIBLIOTECA OPENCV

Os avanços tecnológicos que giram em torno da economia geralmente criam a impossibilidade de todos obterem acesso a estudo, a utilização e desenvolvimento de softwares. Os custos de utilização e da compra dos softwares, bem como a alteração de seus códigos internos de programação não estão obviamente disponíveis a maioria, que acabam por ficar a cargo de grandes empresas investidoras e universidades, tornando lento o desenvolvimento da tecnologia caso estes fossem mais acessíveis.

Nesse contexto surgiram os softwares livres, que tiveram seu início com Richard Stallman, que durante o ano de 1983, no *Massachusets Institute of Technology* (MIT), fundou o projeto *GNU*. Stallman tinha por objetivo, a criação de uma modalidade de software de sistema operacional, que facilitasse o desenvolvimento da tecnologia, bem como sua acessibilidade, e que fizesse o mesmo que um sistema já existente fazia. O sistema existente se chamava *Unix*, e o *GNU* em contrapartida, objetivava ter todas as potencialidades do mesmo, de forma livre, aberta e grátis, sem proprietários e licenças (SILVEIRA, 2004).

OpenCV é uma biblioteca de programação livre, de código aberto, desenvolvida inicialmente pela Intel Corporation. Sua versão alpha foi apresentada no ano de 2000 na *IEEE Conference on Computer Vision and Pattern Recognition.* Atualmente ela está em posse de uma fundação sem fins lucrativos chamada openCV.org (BRAHMBHATT, 2013, tradução nossa).

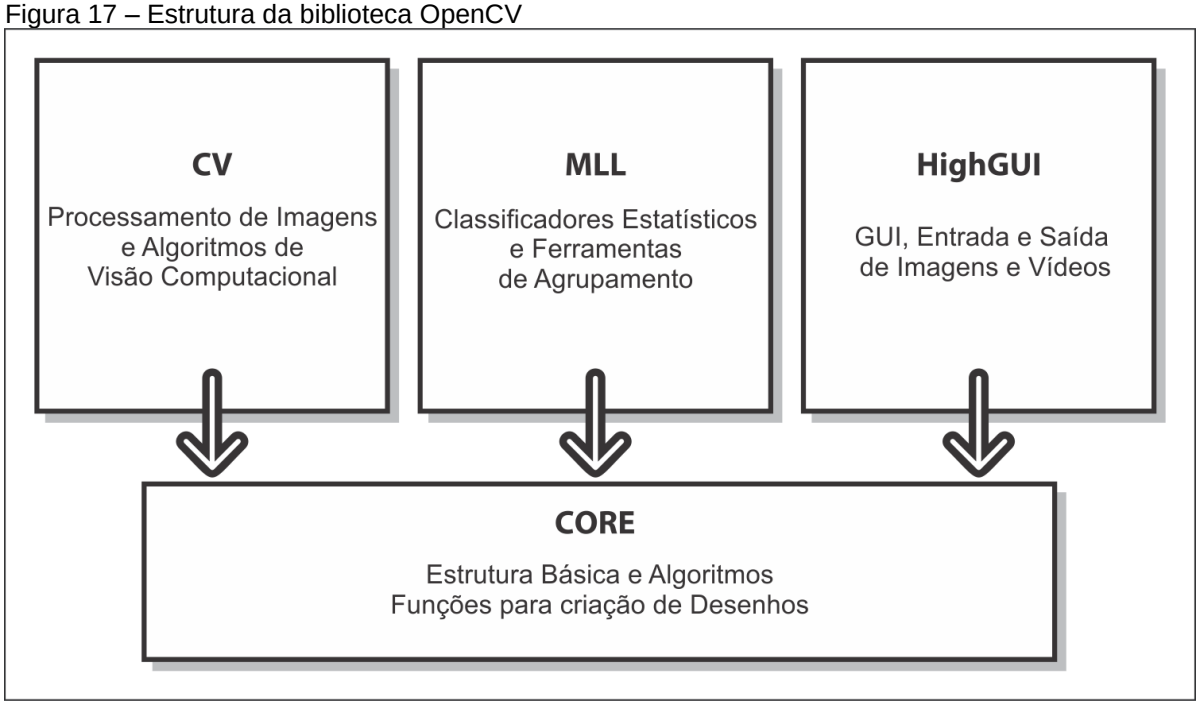
O OpenCV implementa uma variedade de ferramentas de interpretação de imagens, indo desde operações simples como um filtro de ruído, até operações complexas, tais como, a análise de movimentos, reconhecimento de padrões e reconstrução em 3D (MARENGONI; STRINGHINI, 2009).

A biblioteca é distribuída sob a licença BSD e tem mais de 47 mil pessoas em sua comunidade (OPENCV, 2014, tradução nossa). De acordo com Laurent (2004, tradução nossa), esta licença é considerada domínio público, podendo ser modificada sem nenhuma restrição. Para atender as necessidades dos criadores da licença, apenas é necessária a referência dos termos para "*University of California, Berkeley*" e "*Regents*" pelo nome do próprio indivíduo ou organização que a utiliza. Ou seja, todos os trabalhos derivados de softwares licenciados como BSD, devem incluir uma

citação no código fonte, afirmando que o produto contém software desenvolvido pela Universidade de Berkeley.

A biblioteca pode ser usada em várias linguagens de programação tais como, C++, C, Python e Java. E também tem suporte para vários sistemas operacionais como, Windows, Linux, Mac OS, iOS e Android. A biblioteca em si foi escrita em C/C++, e foi projetada para ter mais eficiência computacional e ter o principal foco em aplicações de tempo real, tirando vantagens de processadores *multi-core* (OPENCV, 2014, tradução nossa).

Figura 17 – Estrutura da biblioteca OpenCV



Fonte: Bradski e Kaehler (2008, tradução nossa).

A biblioteca é estruturada em 5 principais componentes, dos quais 4 são apresentados na figura 17, onde o componente CV que significa C*omputer Vision*, representa os módulos responsáveis pelo processamento de imagens e algoritmos de visão computacional, o componente MLL é uma sigla que provem de *Machine Learning Library* e realiza todos processos que envolve conhecimento externo no sistema, como classificadores e agrupamentos, HighGUI contém as rotinas de entrada e saída, e por fim o componente principal CORE é onde fica localizada a estrutura básica da biblioteca, algoritmos e funções essenciais a outros módulos. (BRADSKI; KAEHLER, 2008, tradução nossa).

O quinto componente que não está presente na figura 17, é o CvAux, nela contém funções experimentais e outros algoritmos já depreciados (BRADSKI; KAEHLER, 2008, tradução nossa).

A estrutura da biblioteca contém os seguintes módulos (OPENCV, 2014, tradução nossa):

a) *core*: é um módulo compacto que define a estrutura básica da biblioteca e contém outros dados usados por funções básicas de todos os outros módulos;
b) *imgproc*: módulo responsável pelo processamento de imagens, incluindo transformações geométricas como, redimensionamento e rotação, e muitas outras funções relacionadas ao tratamento de imagens;
c) *video*: é um módulo de análise de vídeos, inclui funções como estimativa de movimento, subtração do fundo do cenário, rastreamento de objetos;
d) *calib3d:* módulo que inclui funções para uso de câmeras com múltiplos pontos de visão, assim como câmeras estéreo, reconstrução de cenários em 3d, entre outros;
e) *features2d:* responsável pela extração de informações de imagens, realiza as funções descritas nos processos de *matching*, representação de descrição;
f) *objdetect:* módulo responsável pelo reconhecimento de objetos pré-definidos em classes;
g) *highgui:* uma interface de fácil uso para capturar ou carregar vídeos e imagens;
h) *gpu:* módulo feito para tirar proveito de placas de vídeo para acelerar o processo feito nos outros módulos.

5.1 VISÃO ESTEROSCÓPICA COM OPENCV

Para se utilizar uma câmera estéreo junto a biblioteca OpenCV é necessário calibrá-la primeiro, isto pode ser feito por meio de um processo chamado *stereo calibration.* O processo de calibração envolve o cálculo da geometria de uma

câmera em relação a outra, para isso é necessário saber a rotação, ou seja, o ângulo que a câmera está virada, e a translação em relação a outra câmera. Em ambas as camêras é feito o cálculo através da função `cvStereoCalibrate()` (BRADSKI; KAEHLER, 2008, tradução nossa).

Após calibradas é necessário obter as imagens das câmeras, a classe `VideoCapture` provê o método `grab()` que é utilizado para capturar os *frames* advindos das câmeras num formato ainda bruto, um *frame* é uma fração de uma sequência de imagens, após utiliza-se o método `retrieve()` para decodificá-los em imagens (BRAHMBHATT, 2013, tradução nossa).

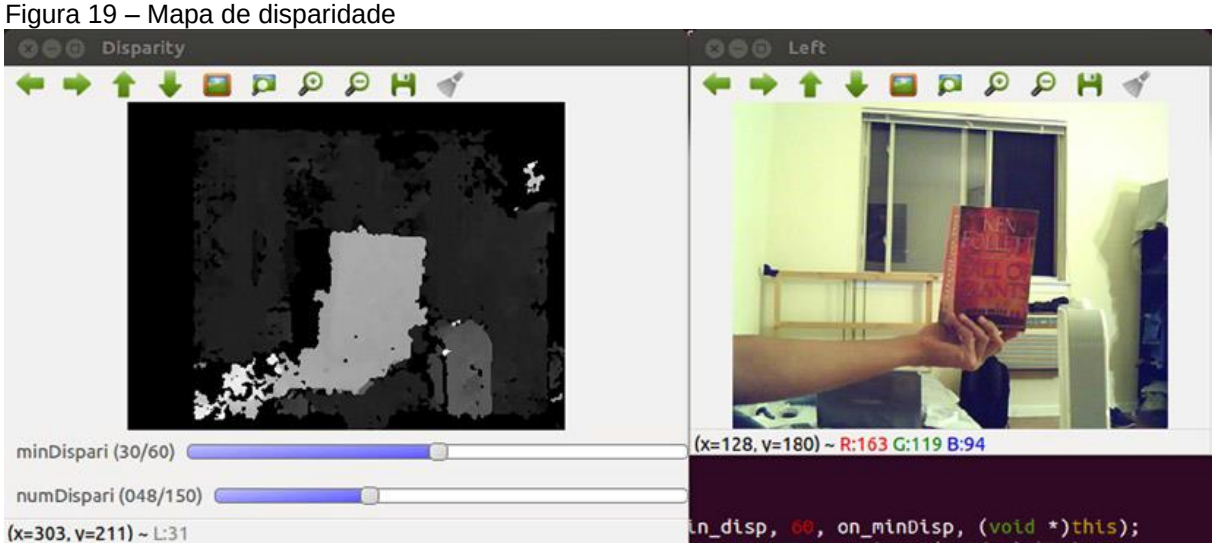
Então é aplicado o processo de *matching,* para isso o OpenCV disponibiliza uma função que alinha as imagens das câmeras `cvInitUndistortRectifyMap()`, esta função retorna um mapa de pixels que são correspondentes aos pixel da imagem não retificada, logo utiliza-se o método `remap()` para retificá-las, ao aplicar esta sequência de métodos em ambas imagens, estas ficam alinhadas horizontalmente (figura 18), então é possível achar pontos correspondentes de uma na outra, de forma mais fácil (BRAHMBHATT, 2013, tradução nossa).

Figura 18 – Imagens retificadas

Fonte: Brahmbhatt (2013).

A função `stereo()` da classe `StereoSGBM`, gera um mapa de disparidade, ou seja, um mapa de disparidade do cenário a partir da imagens retificadas na etapa anterior (BRAHMBHATT, 2013, tradução nossa).

Figura 19 – Mapa de disparidade



Fonte: Brahmbhatt (2013).

A partir do mapa de disparidade demonstrado na figura 19, é possível extrair a distância aproximada entre a câmera e os objetos no cenário.

6 TRABALHOS CORRELATOS

A pesquisa de trabalhos correlatos se fez necessária para que se tenha contato com metodologias utilizadas em outras experiências dentro da mesma área de pesquisa, bem como tornar possível que se constate os resultados obtidos nestas experiências e com isso direcionar melhor o objeto de estudo.

Neste capítulo são apresentados alguns trabalhos que fazem uso das tecnologias abordadas na fundamentação teórica, sua metodologia de pesquisa e os resultados obtidos em cada trabalho.

6.1 FERRAMENTA PARA LOCALIZAÇÃO EM AMBIENTE CONTROLADO UTILIZANDO MAPA DE SINAIS WIRELESS

Este trabalho de conclusão de curso teve como propósito desenvolver uma aplicação para dispositivos móveis, utilizando uma ferramenta de mapa de sinais wireless, buscando localizar e orientar indivíduos com deficiência visual dentro de um ambiente mapeado (FIERA, 2013).

Realizou-se uma pesquisa sobre tecnologia assistiva que busca oferecer autonomia, inclusão social e consequentemente, melhor qualidade de vida aos deficientes visuais, visando ampliar as habilidades e tornando os produtos desenvolvidos universais, sem qualquer necessidade de adaptação para seus usuários.

Foi realizada duas entrevistas com um usuário deficiente visual, sendo uma estruturada e outra desestruturada.

Primeiramente, foi feito levantamento de requisitos sobre localização e orientação. Foi utilizado a teoria dos grafos e algoritmos para cálculo da posição no ambiente, após foi criado um mapa fictício de um mercado, onde foi utilizado antenas de rádio frequência *acess point* (AP).

Foi criado um mapa de sinal em um ambiente onde existiam corredores, neste foram instalados quatro APs, e então marcados no piso 16 pontos numerados com uma distância variando de 1 a 3 metros entre cada ponto. Foram selecionados seis pontos representando regiões no ambiente. Para cada uma dessas regiões foram

associados pontos os quais o usuário deveria se dirigir caso pretendesse ir até uma delas.

Também, com o auxílio do sintetizador de voz nativo do *Android* foi possível a conversão do texto em linguagem natural, sendo a leitura em inglês. Assim, a instalação do pacote de voz Svox Luciana possibilitou a leitura em português, facilitando a compreensão pelo usuário.

O trabalho atingiu os objetivos propostos, por meio de um protótipo desenvolvido que, para sua utilização em ambiente real, ainda necessita de alguns refinamentos. A ferramenta que foi desenvolvida oferece recursos de acessibilidade voltados a pessoas com deficiência visual, utilizando os padrões de acessibilidade já existentes no mercado, e facilita a mobilidade do deficiente visual em um ambiente preparado previamente.

6.2 METODOLOGIA PARA DETECÇÃO DE OBSTÁCULOS PARA NAVEGAÇÃO DE EMBARCAÇÕES AUTÔNOMAS USANDO VISÃO COMPUTACIONAL

O projeto exposto nesta dissertação de mestrado tem como objetivo apresentar novos métodos para detecção de obstáculos utilizados na navegação de barcos, sendo desenvolvido e baseado em visão computacional para medir a distância de obstáculos através de câmeras de vídeo. Onde a imagem desejada é a da perspectiva da superfície do corpo de água, aonde a embarcação navegará, com as câmeras apoiadas sobre o próprio barco.

Os obstáculos a serem detectados são os fortuitos, tendo em vista que o projeto é integrado a outras rotinas de navegação. As rotinas em si não são responsáveis em determinar um plano de fuga, realizando a comunicação com o programa de planejamento de rota, determinando assim a direção e distância do obstáculo para que o programa principal possa tomar as decisões necessárias (MUNHOZ, 2010).

Para captura e processamento das imagens, utilizou-se o ambiente de desenvolvimento Matlab, onde foram executadas em um microcomputador portátil, da marca Toshiba com processador AMD64 Turion com 2.0 Ghz. As imagens foram obtidas através de um par de câmeras *webcam* modelo 11123 fabricados pela Clone.

As câmeras foram apoiadas em uma chapa de aço 16 por ser resistente e leve, tendo dimensões de 1500 mm x 556mm dobrada em 90 graus para eliminar a possibilidade de flexão. Neste suporte foram colocadas as duas *web-cams*, tendo uma distância de 1,4m entre elas.

Os resultados obtidos nos testes e validações foram satisfatórios onde o sistema detectou corretamente todos os obstáculos das imagens testadas. Para as imagens coletada contra o sol o método proposto em comparação ao método de Snyder é mais eficiente.

6.3 UMA APLICAÇÃO DE NAVEGAÇÃO ROBÓTICA AUTÔNOMA ATRAVÉS DE VISÃO COMPUTACIONAL ESTÉREO

Nesta dissertação de mestrado foi desenvolvida uma técnica para navegação autônoma, sendo utilizado imagens estereoscópicas, para estimar o movimento de um robô em um ambiente desconhecido. São abordados métodos de correlação de imagens para o desenvolvimento de uma aplicação de robótica móvel autônoma, sendo capturadas duas imagens de posições diferentes, por duas câmeras fixadas no robô, aplicando o método de correlação de pontos, modificando as imagens unidimensionais para o processamento dos dados e obtenção do mapa de profundidade.

A metodologia do projeto se deu em três etapas, começando com a escolha do método de correlação de pontos entre as imagens, sendo aplicado o método simplificado para imagens unidimensionais e testados com imagens capturadas com câmeras em ambientes virtuais. A segunda etapa ocorre com o uso do algoritmo de correlação de pontos robustos, que resolve o problema de mapeamento de pontos correlatos em um mapa métrico, sendo a coordenada absoluta do sistema de mapeamento a primeira posição do robô. A terceira e última etapa estima-se o movimento que houve nas duas posições, para a análise dos resultados obtidos por simulação e testes experimentais com o robô Pioneer 3. Sendo que para o desenvolvimento dos métodos e processamento foram utilizados o Matlab e o software Blender para a simulação do ambiente virtual.

O método de correlação de imagens unidimensionais identificou as características principais nos objetos, mesmo com a diminuição da medida que influi

nas procuras das mesmas características nas outras imagens em cena (ESPINOSA, 2010).

Os resultados obtidos em ambientes virtuais e reais mostraram um erro médio entre 1% e 3% correspondentemente na profundidade dos objetos, sendo indicado a eficiência e robustez do método de correlação de imagens unidimensional e na determinação do mapa de profundidade. Com a metodologia empregada foi possível obter o mapa para navegação robótica identificando e calculando a profundidade do objeto, estimando o movimento que houve entre duas posições. Os estudos permitiram guiar e situar o robô móvel utilizando informações das câmeras, além de que, os métodos utilizados na navegação do robô, permitiram extrair as informações de distância dos objetos, aplicando estratégias de controle em paralelo aos métodos para prevenção e identificação de obstáculos.

6.4 USO DE VISÃO COMPUTACIONAL EM DISPOSITIVOS MÓVEIS PARA AUXÍLIO À TRAVESSIA DE PEDESTRES COM DEFICIÊNCIA VISUAL

Esta dissertação de mestrado tem por objetivo a elaboração de uma solução utilizando visão computacional para identificar faixas de pedestres, com o propósito de auxiliar deficientes visuais na travessia de ruas utilizando dispositivos móveis, buscando a acessibilidade da interface implementada para o dispositivo móvel.

O sistema desenvolvido é acionado a partir do toque na tela do dispositivo, e então ele começa a efetuar o reconhecimento do ambiente tentando detectar a faixa de pedestres. Se detectada a faixa, o usuário é informado através de sinal de voz pelo do sintetizador SVOX, indicando ao usuário a ação a ser realizada, ou pela vibração do aparelho (SOUSA, 2013).

Para que o sistema reconheça as faixas de pedestres foram estudados o formato das faixas disposto pela ABNT, e então a imagem da faixas é tratada de forma a evidenciar a faixa no cenário, após isso o sistema valida se os contornos são pertencentes a faixa de pedestres ou não.

Foi escolhida a plataforma Android para o desenvolvimento, por ter sido usada a biblioteca OpenCV que dá a vantagem de possuir suporte nativo a plataforma, também pela plataforma Android possuir um *kit* de desenvolvimento que facilita a

criação de aplicações acessíveis, e por ela possuir cerca de metade do mercado mundial.

Souza (2013), destaca que a solução ainda carece de melhorias para então realizar testes em situações reais. No entanto, foram realizadas entrevistas e testes com deficientes visuais, os quais não tiveram dificuldades no uso da interface, muitos estão acostumados ao uso de dispositivos móveis.

Os resultados do projeto foram satisfatórios, foram realizados 64 testes e cenários diferentes, dos quais 55 tiveram êxito em detectar a faixa. Porém, é necessário observar que o sistema deve passar segurança ao usuário, portanto se faz necessário melhorias futuras no projeto.

6.5 AVALIAÇÃO DE SINTETIZADORES DE VOZ PARA LEITURA EM LIVROS DIGITAIS

Segundo Silva Neto e Araújo (2013), da mesma forma que se busca promover a acessibilidade aos ambientes físicos, se trabalha com este conceito aplicado aos ambientes digitais. Portanto este artigo científico objetiva a avalição de sintetizadores de voz com o propósito de tornar o meio digital mais acessível ao deficiente visual.

Os critérios de seleção dos *softwares* avaliados para este trabalho visaram a compatibilidade do formato digital com os leitores utilizados, uma vez que alguns formatos necessitam de leitores específicos. Foi utilizado o método multicritérios para comparar os sintetizadores usados em páginas de navegadores e em livros digitais. Os sintetizadores de voz que passaram nos critérios foram: DOSVOX 4.3, Jaws 9.0 e Virtual Vision 2.0.

O DOSVOX foi desenvolvido pela Universidade Federal do Rio de Janeiro, e possui um sintetizador de voz para computadores, que possibilita a comunicação do usuário deficiente visual.

O *software* Jaws 9.0 é um sintetizador de voz por ser usado em diferentes sistemas operacionais, ele verbaliza todos os eventos que ocorrem no sistema.

O Virtual Vision permite ao usuário deficiente visual a interação com todos os aplicativos do Windows colhendo informações que podem ser lidas, possibilitando assim a navegação por menus, telas e textos (SILVA NETO; ARAÚJO, 2013).

A conclusão do autor foi que das tecnologias avaliadas, sejam elas sintetizadores de voz ou leitores de *e-books,* mesmo que com limitações, facilitam o acesso dos deficientes visuais ao meio digital. E dos sintetizadores de voz estudados o que mais atendeu as necessidades do deficiente visual foi o Virtual Vision, porém os leitores de livros digitais ainda necessitam de evolução e oferecem maior suporte a língua portuguesa.

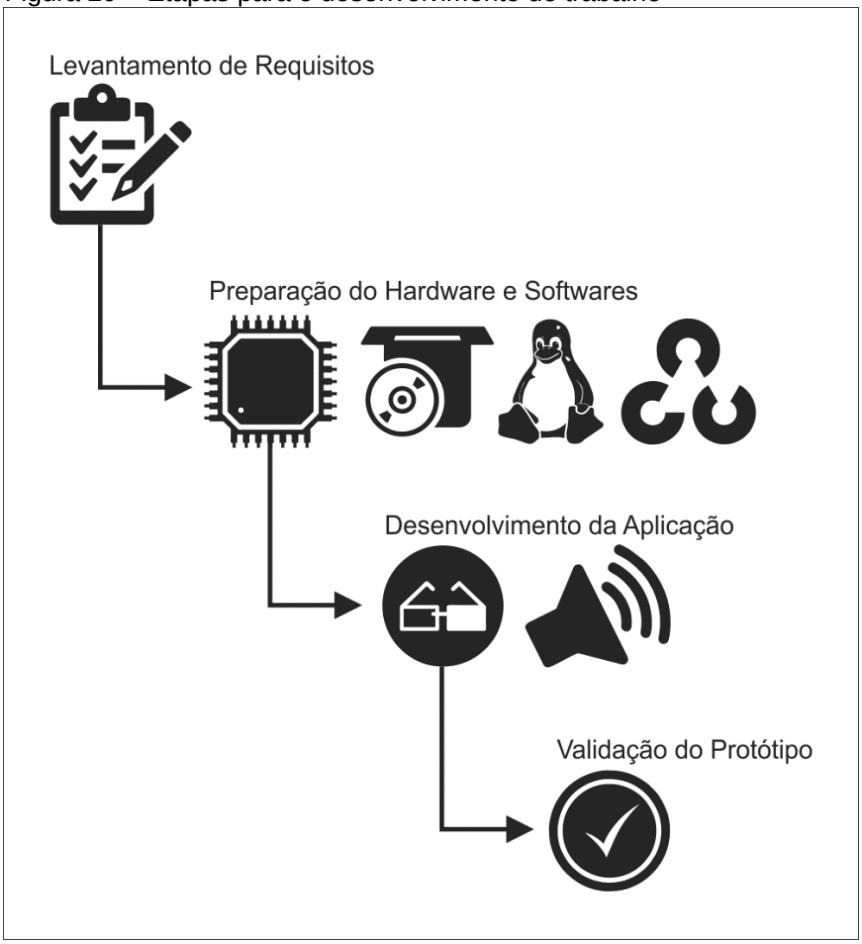
7 SISTEMA DE RECONHECIMENTO DE OBSTÁCULOS FÍSICOS

Por meio do conhecimento construído no decorrer deste trabalho, o sistema que reconhece obstáculos a partir do processamento de imagens capturadas por câmeras, tem o intuito de promover maior inclusão social, e melhor mobilidade para portadores de deficiência visual. A partir do estudo sobre processamento de imagens e visão estereoscópica se possibilitou a implementação do protótipo deste sistema, que, no momento em que detecta um obstáculo informa o usuário através de voz sintetizada, assim tornando-o apto a desviar do mesmo a tempo. A metodologia utilizada no desenvolvimento deste protótipo é descrita no capítulo 7.1.

7.1 METODOLOGIA

O desenvolvimento deste trabalho se seguiu de acordo como é demonstrado no diagrama da figura 20.

Figura 20 – Etapas para o desenvolvimento do trabalho



Fonte: Do autor.

Durante o primeiro período foi realizado o levantamento de requisitos, se baseando em outros projetos e trabalhos envolvendo deficientes visuais e/ou estereoscopia visual, e também em necessidades próprias deste trabalho. Logo na segunda etapa foi necessário preparar tanto a parte de hardware quanto a de software para fornecer um ambiente adequado para o desenvolvimento do protótipo. Em seguida foi iniciado o desenvolvimento em si partindo dos estudos realizados, fora implementado o algoritmo para o processamento das imagens, e unido a ele o sintetizador de voz para se comunicar com o usuário do sistema, e então na sequência a validação do protótipo.

7.1.2 Levantamento de requisitos

Com base na leitura e pesquisa de trabalhos e projetos desenvolvidos para o auxílio de deficientes visuais, foi identificado que mesmo com o uso da bengala, ainda existem obstáculos que a mesma não detecta durante a movimentação do portador da deficiência, geralmente estes são objetos que ficam suspensos acima do nível da cintura, como por exemplo cestos de lixos e telefones públicos. Sendo assim admitiu-se que o foco deste trabalho seria identificar obstáculos à frente do usuário e que ficam ao nível acima da cintura.

Também foi constatado que, o meio mais comum usado para interação com deficientes visuais em sistemas ou ferramentas virtuais é a síntese de voz, sendo assim foi implementada esta funcionalidade para informar ao usuário dos obstáculos.

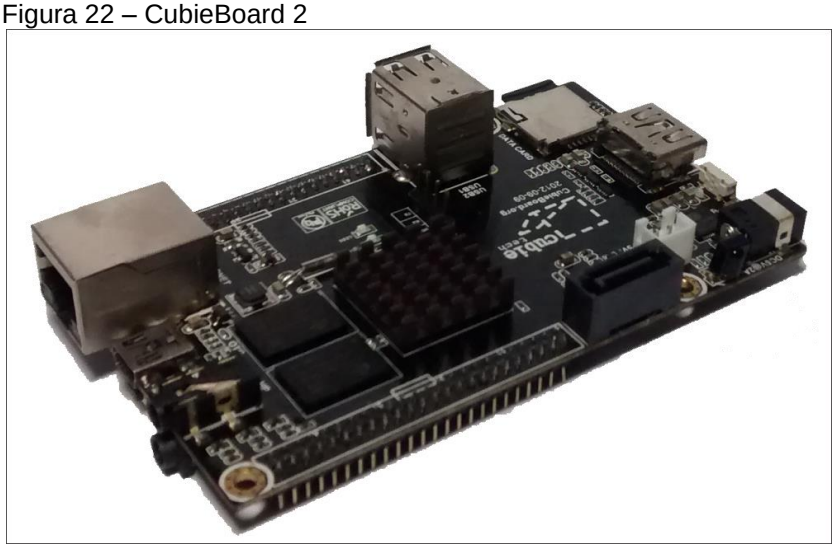
Para o desenvolvimento da aplicação fez-se necessário a aquisição de duas webcams com interface USB (figura 21), utilizadas para a captura das imagens e formação da estereoscopia.

Figura 21 – *Webcams*

Fonte: Do autor.

A fim de manter o protótipo portátil, para que o usuário possa carregá-lo com maior facilidade, foi preciso uma placa System-on-a-chip (SoC) com capacidade computacional para suportar a aplicação desenvolvida, para tal foi adquirido um CubieBoard 2 (figura 22), que foi escolhido após pesquisa por placas do gênero capazes de executar uma distribuição do sistema operacional Linux e também a biblioteca OpenCV, a partir disso foi levado em consideração os atributos da placa, como processador e quantidade de memória RAM, sendo também que deveria conter no mínimo duas entras USB para suportar as câmeras a serem utilizadas e uma saída de áudio para a execução da síntese de voz. A placa CubieBoard 2 da empresa de mesmo nome, se adequou aos requisitos com um processador ARM cortex-A7 dual-core de 1GHz, 1 Gb de memória RAM. Outro detalhe que foi levado em consideração na escolha da placa, é o fato da arquitetura da placa ser baseada no Raspberry Pi, que é uma placa popular e com uma comunidade de desenvolvedores grande e ativa.

Figura 22 – CubieBoard 2



Fonte: Do autor.

Para abastecer a placa com energia, foi utilizada uma bateria portátil do tipo *Power Bank,* que são utilizadas para recarregar *smartphones*, assim garantindo certo tempo de autonomia ao protótipo.

Como a aplicação será desenvolvida fora da placa, optou-se por utilizar a linguagem Java, pelo fato do *software* poder ser portado de uma máquina para outra sem a necessidade de se alterar o código fonte ou de se recompilar o *software*.

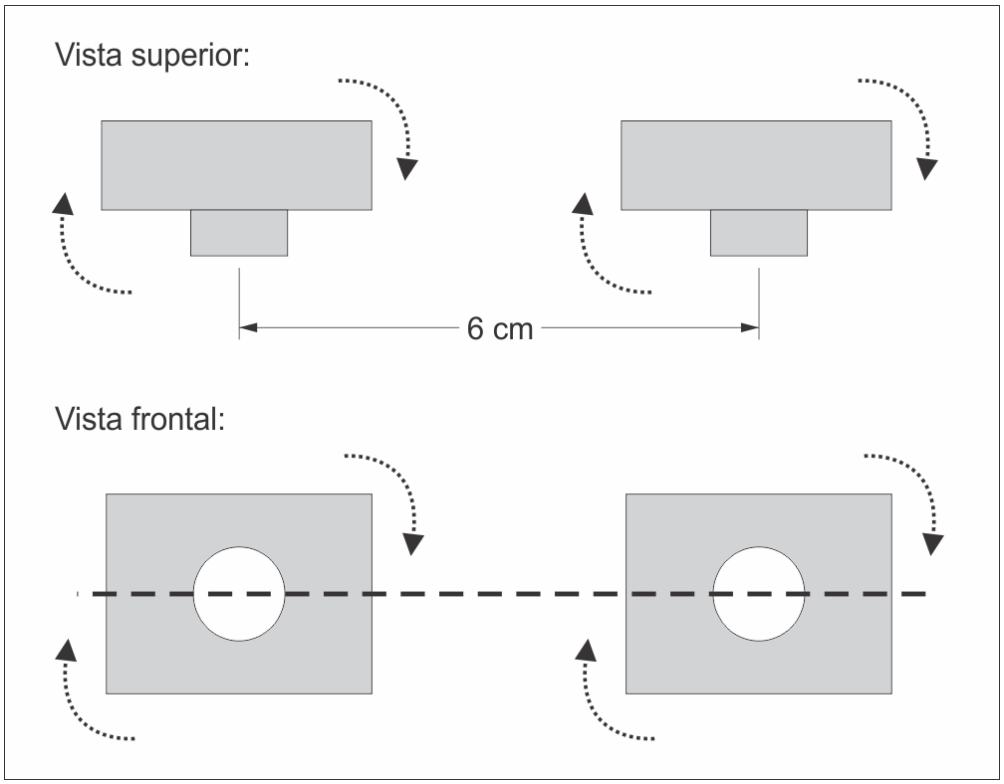
7.1.3 Preparação do hardware e softwares

Antes de iniciar o desenvolvimento deste trabalho o primeiro passo foi a preparação da estrutura de hardware, usada para executar a aplicação, e também para entrada de dados. Em seguida, foi preparado o ambiente de software, envolvendo a instalação do sistema operacional, bibliotecas e softwares necessários para execução.

7.1.3.1 Preparando o hardware

O CubieBoard 2 vem de fábrica com Android Jelly Bean instalado em sua memória *flash*, assim foi necessário a instalação de uma distribuição Linux nele. Porém, a memória interna da placa dispõe de apenas 4 *Gygabytes (GB)*, inviabilizando a instalação do Linux somado aos demais softwares necessários, sendo que a placa tem suporte para cartões de memória de até 32 GBs, foi utilizado um cartão de 8 GB para instalação dos softwares.

Figura 23 – Posicionamento das câmeras



Fonte: Do autor.

Uma parte importante do trabalho realizado foi a construção do par de câmeras estéreo. Para se obter imagens no formato estéreo com uma qualidade aceitável ambas as câmeras precisam ficar alinhadas na horizontal com a maior precisão possível, e então fixadas a uma distância arbitraria de 6cm. Como é demostrado na figura 23, a rotação tanto no eixo x, quanto no eixo y das câmeras também influenciam no resultado da estereoscopia, o ideal seria que ambas ficassem em ângulo reto nos dois eixos.

Para construção da estrutura das câmeras fora removido os circuitos internos das webcams e então feito uma *case* com acrílico e ajustado o alinhamento com parafusos no momento de fixação dos circuitos na case, o resultado pode ser observado na figura 24.

Figura 24 – Câmera estéreo

Fonte: Do autor.

A construção desta foi necessário para otimizar o campo de visão disponível no mapa de disparidade, isso se deve ao fato de que o mapa de disparidade é gerado com base nos pixels que ambas imagens tem em comum, então quanto maior a diferença no deslocamento de uma imagem para outra menor será o campo de visão, apesar do deslocamento na horizontal ser necessário para se obter o efeito estéreo, este não precisa ser muito extenso, e deslocamentos em outros eixos são desnecessários e prejudicam a qualidade do mapa gerado.

7.1.4.2 Preparação do ambiente de software

Após pesquisa por distribuições do Linux compatíveis com a placa, e instalado algumas para testes, foi escolhida a Cubian X1, que foi baseada no Debian e projetada especialmente para rodar em placas da empresa CubieBoard, apresentando maior compatibilidade com o hardware da placa do que as demais distribuições testadas.

Com o Sistema Operacional (SO) instalado, o próximo passo foi instalar o Java SE Development Kit (JDK) para arquitetura ARM, para isso foi feito *download* da instalação do JDK 8 no site oficial da Oracle, e após instalado o JDK na placa foi necessário configurar o sistema para utilizar a versão instalada como padrão, pois o SO possui o JDK 6 pré-instalado, o qual não é compatível com a biblioteca Open-CV que também é necessária para o desenvolvimento deste trabalho. Sendo que o SO possui um gerenciador de softwares instalados, esta configuração foi feita por meio dele em linha de comando como mostrado a seguir.

Figura 25 – Configurando o Java no Cubian X1

```
$ sudo update-alternatives --install /usr/bin/java java /opt/jdk1.8.0/bin/java 1500
$ sudo update-alternatives --install /usr/bin/javac javac /opt/jdk1.8.0/bin/javac 1500
```

Fonte: Do autor.

A figura 25 demonstra a configuração de novas alternativas do java e javac para o sistema utilizar, estes são, o interpretador e compilador da linguagem respectivamente, o valor 1500 é o valor que o sistema utiliza para escolher qual alternativa usar, no caso ele escolhe o maior valor entre as alternativas instaladas de cada software.

A próxima etapa foi instalar as dependências da biblioteca OpenCV, foi utilizado o instalador de pacotes do SO para isso. Foram instalados os seguintes pacotes:

a) gcc-arm-linux-gnueabihf: este é um pacote de ferramenta para utilizar o compilador gcc que vem pré-instalado no SO, empregando instruções para processadores do tipo ARM;
b) Cmake: sistema de *build*, testes e gerenciador de softwares;
c) libjpg-turbo: é uma biblioteca que auxilia e acelera a manipulação de imagens em JPEG para processadores do tipo ARM;
d) pkg-config*:* fornece uma interface para consultar bibliotecas instaladas e compilação a partir do seu código-fonte;
e) libgtk*:* biblioteca de manipulação e edição de imagens;
f) libav: foram instalados os pacotes libavcodec e libavformat desta biblioteca de processamento de vídeo que tem suporte para processadores do tipo ARM;
g) libswscale: biblioteca que otimiza o redimensionamento de imagens e operações de conversão de cores.

Em seguida foi feito download da versão 2.4.10 do código fonte biblioteca OpenCV pelo site oficial, extraído em um diretório e na raiz do mesmo foi executado o comando exibido na figura 26 para configurar a biblioteca.

Figura 26 – Configurando a biblioteca OpenCV

```
$ cmake
-D SOFTFP=ON
-D CMAKE_TOOLCHAIN_FILE=opencv-2.4.10/platforms/linux/arm-gnueabi.toolchain.cmake opencv-2.4.10
-D CMAKE_C_FLAGS="-O3 -mfpu=neon -mfloat-abi=hard"
-D CMAKE_CXX_FLAGS="-O3 -mfpu=neon -mfloat-abi=hard"
-D CMAKE_BUILD_TYPE=RELEASE
-D CMAKE_INSTALL_PREFIX=/usr/local
-D BUILD_NEW_PYTHON_SUPPORT=ON
-DWITH_JPEG=ON
-DBUILD_JPEG=OFF
-DJPEG_INCLUDE_DIR=/opt/libjpeg-turbo/include/
-DJPEG_LIBRARY=/opt/libjpeg-turbo/lib/libjpeg.a
-DWITH_V4L=ON
-D WITH_TBB=ON
-DBUILD_TBB=ON
-DENABLE_VFPV3=ON
-DENABLE_NEON=ON
```

Fonte: Do autor.

Neste comando foi configurado para que na compilação da biblioteca sejam utilizadas as bibliotecas descritas anteriormente, também foram adicionadas configurações para o uso de instruções VFPv3 e NEON, uma vez que o processador da placa tem suporte para estas tecnologias, assim otimizando a execução dos algoritmos da biblioteca. As configurações -DWITH_TBB=ON e -DBUILD_TBB=ON permitem que a biblioteca tire proveito de processadores ARM que sejam *multi core.* Então, após configurada a biblioteca ela foi compilada pelo comando `make`.

Por fim, para a síntese de voz foi escolhido o software eSpeak, por ele ter suporte tanto para Windows, onde a aplicação foi desenvolvida, assim facilitando a fase de testes, quanto para Linux onde a aplicação final será executada, e também por disponibilizar a língua portuguesa. Na plataforma Linux a instalação do sintetizador tem a dependência da biblioteca de áudio `libportaudio0` que foi instalada pelo gerenciador de pacotes do sistema como as demais anteriormente. Também foi necessário configurar uma variável de ambiente para o sintetizador, pois o mesmo possui uma funcionalidade utilizada neste trabalho, a de permitir que se execute sínteses através de linha de comando, necessitando que o sistema reconheça o `espeak` como comando.

7.1.4 Desenvolvimento da aplicação

A aplicação foi desenvolvida no NetBeans IDE 8.0.2 na linguagem Java, ela foi dividida em três pacotes:

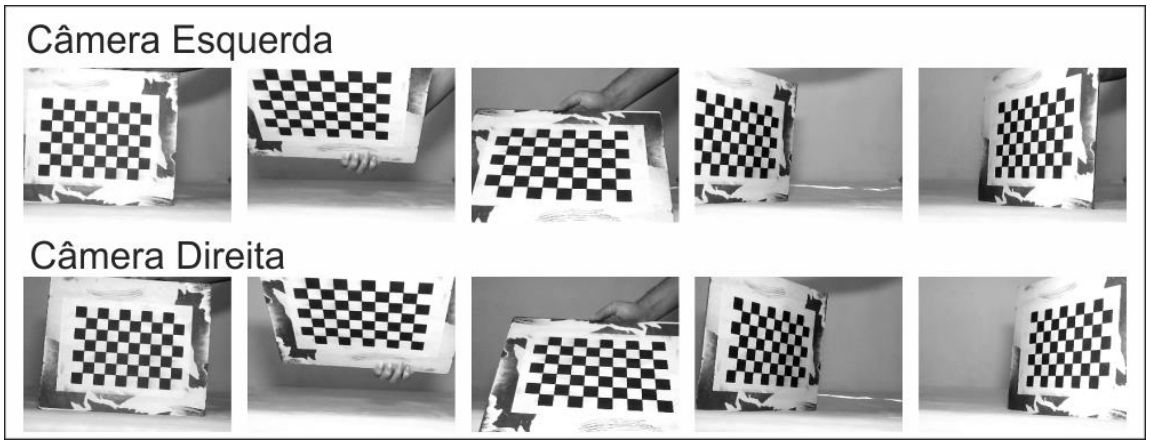
a) output: nele estão as classes que se destinam a exibir as imagens capturadas pelas câmeras, o resultado do mapa de disparidade, e síntese de voz;
b) processing: responsável por capturar as imagens, processá-las, remover distorções, alinhá-las, criar o mapa de disparidade e reconhecer obstáculos;
c) utils: este pacote possui classes de apoio aos outros módulos, como armazenamento de matrizes, conversores entre outros, também se encontra neste pacote a classe que efetua a calibração das câmeras.

7.1.4.1 Calibração do par estéreo

Primeiramente para que seja possível extrair o mapa de disparidade a aplicação precisa retificar as imagens, sendo que as imagens captadas pelas câmeras possuem distorções geradas pelas lentes ou pelos próprios sensores, apesar de mínimas, estas distorções prejudicam a qualidade do mapa de disparidade, já que o algoritmo que o gera faz a busca a nível de pixels, por semelhanças entre duas imagens. Após retificadas é necessário alinhá-las verticalmente para facilitar o esforço do algoritmo, assim a busca pelos pixels é feita apenas no sentido horizontal.

Este procedimento é comumente chamado de calibração estéreo, o OpenCV fornece métodos para realizar esta calibração, para isso no próprio site da biblioteca se encontra arquivos, para serem impressos, com padrões desenhados. Esta impressão é utilizada na calibração, tirando fotos ou capturando em vídeo a impressão, a biblioteca reconhece os padrões na imagem capturada e consegue realizar a calibração com base nas distancias entre os padrões, na figura 27 é exibido uma parte do conjunto de fotos capturadas para a calibração do par estéreo usado neste trabalho, as imagens foram convertidas em escala de cinza no momento em que são carregadas, pois torna mais fácil o reconhecimento dos padrões já que este foi impresso em preto e branco.

Figura 27 – Conjunto de capturas para calibração



Fonte: Do autor.

Neste trabalho foi utilizado o padrão de tabuleiro de xadrez, ao todo foram utilizadas 33 capturas de cada câmera para calibração, sendo que quanto maior o número de amostras melhor a qualidade da calibração.

O resultado da calibração são algumas matrizes, que serão utilizadas para retificar e alinhar as imagens captadas posteriormente:

a) cameraMatrix1 e cameraMatrix2: mantêm as propriedades referentes aos sensores das câmeras 1 e 2;
b) distCoeffs1 e distCoeffs2: coeficientes de distorção referentes as lentes das câmeras e sua distância focal (zoom);
c) R1 e R2: matrizes usadas para alinhar as imagens em relação a rotação de uma câmera para a outra;
d) P1 e P2: matrizes de projeção para as novas coordenadas das imagens retificadas referentes as câmeras 1 e 2, em outras palavras, é utilizada para alinhar a imagens no sentido vertical e horizontal.

Apesar da biblioteca OpenCV disponibilizar uma solução pronta para o uso, quando se trata de salvar uma matriz de uma imagem em arquivos no disco, esta funcionalidade não está presente na versão Java da biblioteca na versão 2.4.10, portanto, foi implementado uma classe, presente no módulo *utils,* que permite descarregar matrizes para arquivos tipo texto convertendo as matrizes para o formato JSON[1], e carregá-las novamente convertendo de JSON para matriz.

[1] JavaScript Object Notation (JSON) é uma estrutura de armazenamento de dados, tipo chave valor, no formato de texto.

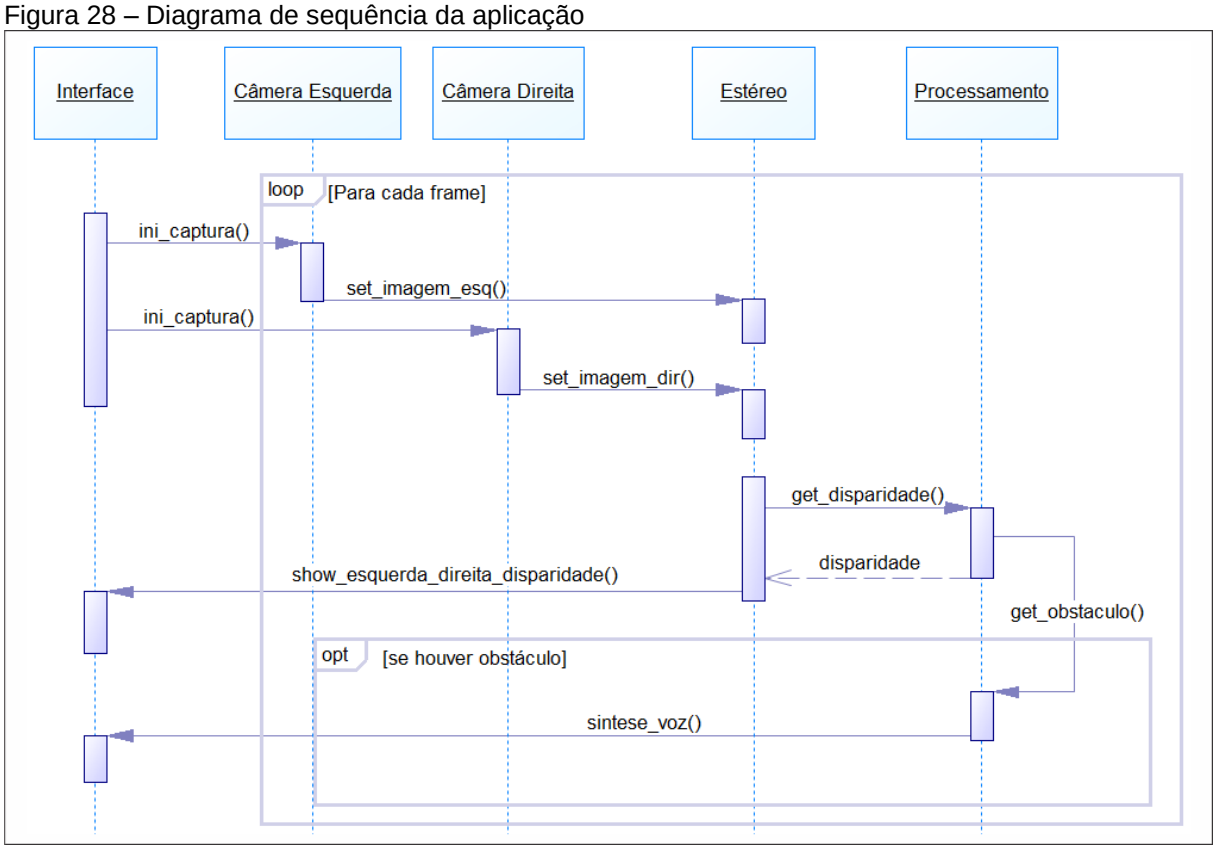
Lembrando que a calibração é válida para o estado em que o par de câmeras estava durante a calibração, caso seja alterada a rotação, translação ou o foco (zoom) de alguma das câmeras, uma nova calibração deve ser feita.

Desta maneira o par de câmeras é calibrado, e então são salvas as matrizes relacionadas acima em arquivo, e a partir disto é possível carregar as matrizes no início da execução da aplicação e aplicá-las em cada *frame* capturado para obter as imagens retificadas e alinhadas.

7.1.4.2 Processamento de imagens

O procedimento de calibração é feito separadamente da execução principal da aplicação, sendo necessário apenas num primeiro momento, ou em caso das câmeras forem desreguladas. A sequência de execução principal pode ser observada na figura 28.

Figura 28 – Diagrama de sequência da aplicação



Fonte: Do autor.

Foi criada uma interface (figura 29) visual, apenas a título de teste, para se poder visualizar o resultado gerado, porém para o funcionamento pleno da aplicação esta não é necessária uma vez que aplicação é feita para deficientes visuais, e esta se comunicará por síntese de voz. O botão capturar que pode ser visualizado na figura é utilizado para salvar as imagens, esquerda e central, na pasta raiz da aplicação, empregadas na fase de calibração.

Figura 29 – Interface da aplicação

Fonte: Do autor.

Ao iniciar a aplicação, o software reconhece as câmeras plugadas e inicia a captura em forma de vídeo rodando a 30 quadros por segundos, a biblioteca OpenCV entrega cada quadro no formato de matriz de 320x240 e 3 canais, que é a representação de uma imagem, com resolução de 320 pixels de largura por 240 de altura e 3 canais de cores, *blue*, *green e red* (BGR), formato padrão da biblioteca. Ao mesmo tempo o software carrega as matrizes de calibração, e após isso a cada quadro recebido as imagens são retificadas e alinhadas através delas, antes de serem exibidas na tela.

Para evitar que o fluxo de entrada de dados, e o alinhamento das imagens de uma câmera concorra com a outra pelo uso do processador, cada captura de vídeo é executada em uma *Thread*[2] separada.

A cada *frame* capturado a matriz que representa ele é enviada para outra classe que roda na *Thread* principal da aplicação, e mantém duas matrizes uma para imagens capturadas da câmera esquerda e outra para imagens capturadas pela câmera direita, uma vez que a classe tenha as duas matrizes preenchidas é extraído o mapa de disparidade a partir das duas.

Para a obtenção do mapa de disparidade foi utilizada classe StereoSGBM do OpenCV, ao se instanciar esta classe é passado parâmetros que vão regular e refinar o mapa que será obtido ao chamar a função `compute` da classe, os parâmetros são:

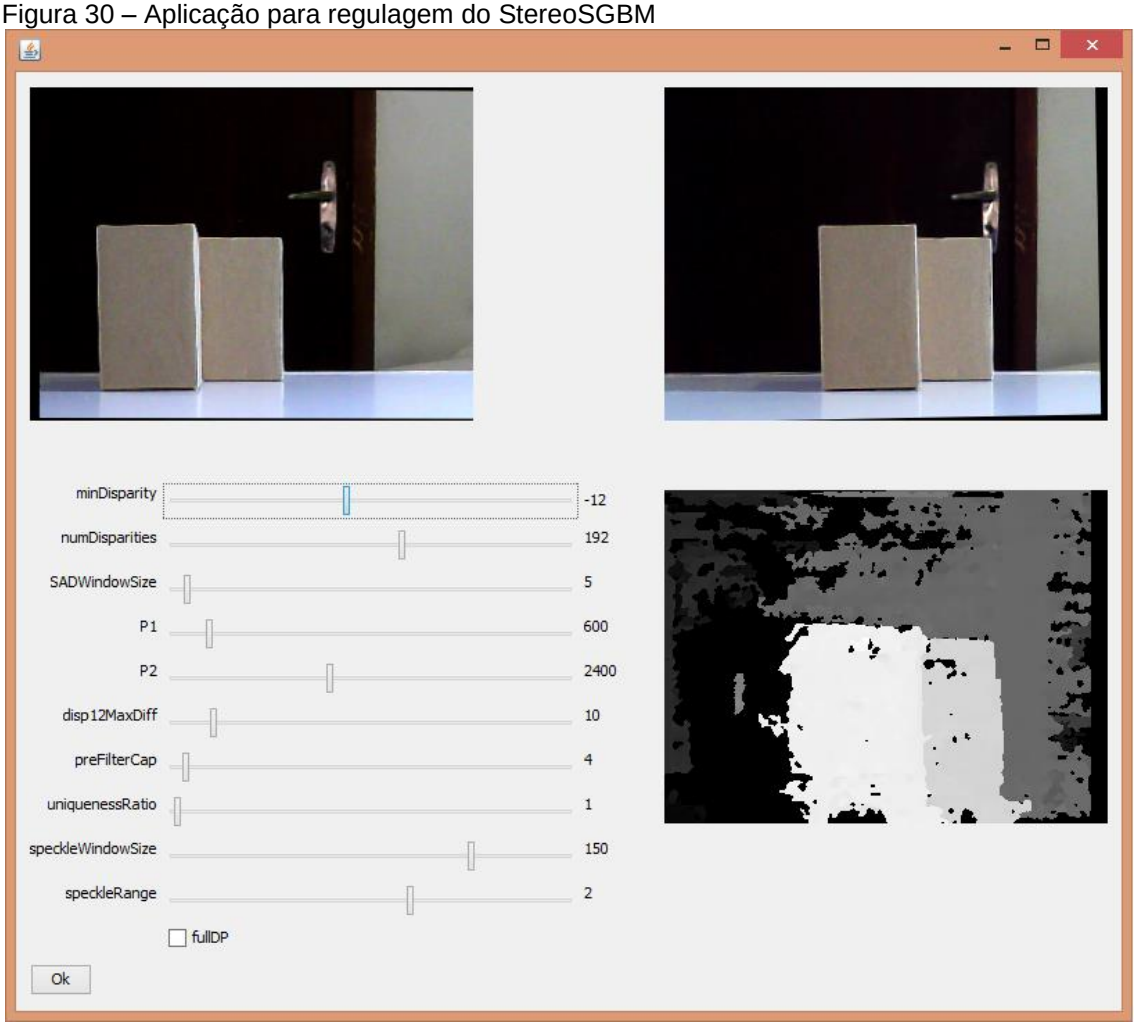
a) minDisparity: valor mínimo de disparidade, ajustado de acordo com o deslocamento na imagem provocado pela calibração, caso não haja deslocamento este valor deve ser zero;
b) numDisparities: máximo de disparidade subtraído o mínimo de disparidade, este valor deve ser múltiplo de 16 e maior que zero;
c) SADWindowSize: tamanho da área que o algoritmo vai procurar pelo pixel, no processo de *matching*, deve ser um valor ímpar;
d) P1: primeiro parâmetro de controle de suavidade da disparidade;

[2] *Thread* é forma usada pelo sistema operacional para divisão das tarefas a serem executadas pelo processador.

e) P2: segundo parâmetro de controle da suavidade, sendo quanto maior o valor mais suave será o resultado, o algoritmo necessita que o valor de P1 seja menor que o de P2;
f) disp12MaxDiff: distância máxima permitida entre pixels na checagem de esquerda-para-direita. Um valor negativo desabilita a checagem;
g) preFilterCap: valor de truncamento para pixels pré-filtrados da imagem. O algoritmo deriva cada pixel e trunca seu valor no intervalo de -preFilterCap até preFilterCap;
h) uniquenessRatio: margem de porcentagem em que o primeiro valor (pixel), deve ser melhor que o segundo, para que seja escolhido no processo de *matching*;
i) speckleWindowSize: valor máximo (de 50 a 200) a ser considerado como mancha de ruído no mapa de disparidade, e então invalidado. O valor 0 desabilita o cancelamento de manchas;
j) speckleRange: define a diferença de disparidade que é considerado como mancha;
k) fullDP: habilita o algoritmo completo para obtenção do mapa de disparidade, consome quantidade maior de processamento e memória.

Para a escolha dos valores, destes parâmetros, mais apropriados para este trabalho, foi implementado uma aplicação (figura 30) a parte que permite visualizar em tempo real a mudança feita nestes valores.

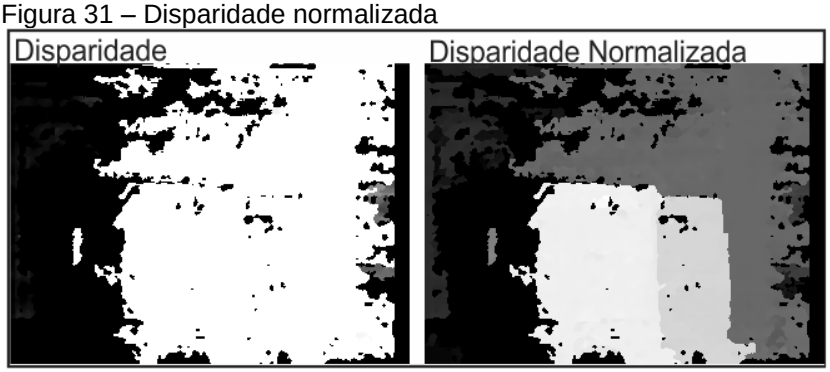
Figura 30 – Aplicação para regulagem do StereoSGBM



Fonte: Do autor.

Com esta aplicação foi possível produzir um mapa de disparidade mais nítido. Contudo, o resultado gerado pela classe StereoSGBM precisa ser normalizado para se tornar inteligível, na figura 31 é demonstrado a disparidade normalizada na escala de cinza, onde é possível distinguir que objetos mais próximos ficam em tons mais claros, enquanto objetos mais distantes ficam em tons mais escuros.

Figura 31 – Disparidade normalizada



Fonte: Do autor.

7.1.4.3 Reconhecimento de obstáculos

O reconhecimento de obstáculos foi feito com base na matriz da disparidade normalizada, sabendo que a matriz tem as mesmas proporções das imagens capturadas pelas câmeras, 320x240, pode-se calcular a quantidade total de pixels presentes na imagem, simplesmente multiplicando os valores, obtém-se então o valor de 76800 pixels.

Foi estipulado que um objeto que ocupe 20% da imagem será considerado obstáculo, para isso primeiro foi definido a proximidade com que o objeto será detectado, como a imagem foi normalizada em escala de cinza seus pixels assumem valores entre 0 e 255, quanto maior este valor mais claro é o tom de cinza, em outra palavra mais próximo estará o objeto que conter o pixel com valor alto.

Então foi utilizado o valor de 120 como limite, este valor representa objetos em torno de um metro de distância, porém o algoritmo lê pixels na faixa de 120 a 250, reconhecendo objetos a um metro ou menos. O algoritmo percorre toda a matriz registrando em um contador a quantidade de pixels cujo valor seja maior que 120, se ao final da contagem a quantidade de pixel for maior que 20% de 76800, provavelmente terá um objeto sólido a frente do usuário.

Ainda assim, como o algoritmo percorre a matriz de apenas um frame, ou seja, a imagem que é utilizada no algoritmo representa um curto espaço de tempo, sendo que uma captura é tirada a cada 33 milissegundos, então não é possível garantir que o obstáculo realmente esteja presente com base em um único frame. Neste caso foi implementado um vetor, este armazena o resultado do algoritmo de reconhecimento durante os últimos 2 frames capturados, e por fim um segundo algoritmo analisa se existe um obstáculo nos frames guardados no vetor, se sim é feita a chamada para síntese de voz.

Como o software eSpeak usado para síntese de voz deste trabalho permite sua execução por linha de comando, não foi necessária integração direta com a aplicação. A linguagem Java permite chamar/criar processos do sistema operacional, logo para avisar o usuário sobre o obstáculo foi feito uma simples chamada desta maneira: `Process p = rt.exec(espeak -vpt+f3 "Obstáculo a frente");`

onde o parametro `-vpt+f3` indica ao eSpeak que a linguagem usada é o português e a voz utilizada será o `f3`, voz feminina 3, seguido então pela mensagem que será sintetizada.

7.1.5 Validação do protótipo

A validação do protótipo ocorreu de duas maneiras, a primeira de forma técnica, validando o comportamento do reconhecimento de obstáculos, a segunda maneira foi analisando a possibilidade de se guiar pelo protótipo e o tempo de resposta entre a aparição do obstáculo e a detecção do mesmo pela aplicação.

Para avaliar o reconhecimento de obstáculos, a aplicação foi executada em um notebook, por ser mais fácil de visualizar a saída de dados, e o processo ser mais rápido e prático, foi mantido as câmeras em posição fixa, e em seguida adicionado objetos em seu campo de visão.

A primeira informação relevante extraída na avaliação foi que o campo de visão da câmera é pequeno, supondo um quadro que meça 50 centímetros de largura por 37 centímetros de altura, a um metro de distância da câmera este quadro ocuparia a imagem inteira justamente, na mesma analogia a um metro e meio de distância, a câmera teria a visão de um quadro de 78 cm por 61 cm.

Para simulação dos obstáculos foram utilizadas caixas de tamanhos variados. As dimensões de uma das faces de cada caixa utilizada para simulação dos obstáculos estão dispostas na tabela 1, foi disposto apenas a face que ficou voltada para a câmera durante a validação.

Tabela 1 – Dimensões dos objetos

Objeto	Altura	Largura	Área
Caixa 1	9 cm	11 cm	99 cm²
Caixa 2	11 cm	17 cm	187 cm²
Caixa 3	14 cm	20 cm	280 cm²
Caixa 4	28 cm	28 cm	784 cm²
Caixa 5	32 cm	40 cm	1280 cm²

Fonte: Do autor.

Os objetos foram dispostos individualmente um de cada vez, nas distancias de 50cm, 90cm e 120cm das câmeras, e avaliado se o protótipo realizava o reconhecimento corretamente, lembrando que a distância implica no tamanho que o

objeto é identificado pelas câmeras, sendo que mesmo objetos grandes, porém distantes, não devem ser detectados como obstáculos. Nos dados observados na tabela 2, é exibido quais objetos, de acordo com a tabela 1, foram reconhecidos marcados com o símbolo ✓, e não reconhecidos marcados com um X, e em quais distâncias ocorreu.

Tabela 2 – Obstáculos reconhecidos

Distância / Objeto	50 cm	90 cm	120cm
Caixa 1	✓	X	X
Caixa 2	✓	X	X
Caixa 3	✓	✓	X
Caixa 4	✓	✓	✓
Caixa 5	✓	✓	✓

Fonte: Do autor.

Como pôde ser observado a medida que os obstáculos menores se afastam, a aplicação não os detecta mais, este era o resultado esperado, pois quanto mais distante o objeto está menor é sua representação na imagem, então em certa distância o objeto já não representa risco imediato de colisão com o usuário. Já os objetos maiores acabam continuando a serem detectados mesmo estando mais distante por conta de ainda tomar grande parte da imagem capturada, isso possibilita a detecção de paredes por exemplo.

Durante a avaliação do reconhecimento de obstáculos foi cronometrado, sem muita exatidão, apenas para se ter noção se houvesse alguma diferença perceptível no tempo de detecção de um objeto para outro, ou de uma distância para outra. Todas as detecções de obstáculos ocorreram em cerca de 1 segundo após o objeto ser posicionado, por frações de segundos objetos maiores foram reconhecidos antes dos demais, isso se explica pelo fato deles ocuparem maior parte da imagem captada, desse modo o algoritmo não precisa percorrer grande parte da matriz para chegar a região onde está o objeto, sendo assim objetos posicionados no canto superior esquerdo da imagem captada, são reconhecidos com mais facilidade e agilidade pelo algoritmo, porém a diferença de tempo é mínima.

Em seguida foi avaliado o reconhecimento dos obstáculos com o protótipo em movimento. O reconhecimento ocorreu de maneira bastante similar aos resultados obtidos anteriormente, porém foi verificado que ao se fazer curvas as imagens ficam

borradas pela velocidade com que o plano de fundo se desloca durante a curva, isso em algumas situações geraram falsos positivos, detectando obstáculos onde na verdade não existia algum.

7.2 RESULTADOS OBTIDOS

O trabalho desenvolvido tem por objetivo expandir a inclusão social de deficientes visuais por meio da tecnologia assistiva criada neste, favorecendo sua mobilidade em qualquer ambiente, tornando-o mais independente, deste modo também melhorando sua qualidade de vida.

Uma das limitações do protótipo foi que, devido à abertura focal da câmera não foi possível detectar obstáculos na área esperada, sendo necessário escolher qual a faixa de altura é mais interessante para um deficiente visual estar ciente de um obstáculo na mesma. Deste modo foi escolhido a área acima da cintura pois é onde a bengala, instrumento habitualmente usado pelos deficientes visuais, não alcança.

Outra limitação no protótipo desenvolvido foi a qualidade das câmeras, com a quantidade máxima de 30 *frames* por segundo em movimentos rápidos as imagens se tornam borradas. O algoritmo de *matching* tem grandes dificuldades em encontrar os pixels correspondentes em imagens borradas, assim podendo gerar resultados errôneos.

Também relacionado a câmera e ao algoritmo de *matching,* pela falta de qualidade da imagem em si, quando ambas as câmeras estão apontadas para uma superfície lisa da mesma cor e mesma luz incidindo sobre, a câmera não gera detalhes suficientes para que o algoritmo de *matching* encontre o pixel mais provável de ser correspondente na outra imagem, pois existem muitos pixels semelhantes ou iguais, e isto confunde um pouco o algoritmo, assim mais uma vez podendo gerar resultados errôneos.

Apesar das limitações, o protótipo desenvolvido no decorrer deste trabalho conseguiu atender satisfatoriamente o seu objetivo, foi possível detectar obstáculos a curta distância, como paredes, cadeiras e entre outros, a tempo de informar o usuário através da síntese de voz.

Um ponto interessante no resultado deste trabalho, foi o fato de que mesmo o processamento de imagens sendo um trabalho relativamente pesado, que demanda bastante esforço do processador, atualmente a evolução de *smartphones*, e

popularização da IoT[3], vêm incrementando os processadores para plataformas pequenas como a utilizada neste trabalho, e tornando possível a execução de algoritmos custosos, como o desenvolvido neste trabalho, em dispositivos portáteis, a um custo mais acessível.

[3] Internet of Things (IoT), termo utilizado para definir objetos (coisas) que estão conectados à internet e podem ser acessados por ela, como geladeiras, sensores entre outros.

8 CONCLUSÃO

Com o mercado aquecido de SoCs, vem surgindo placas cada vez mais robustas e eficientes, possibilitando a realização de ideias, como a proposta neste trabalho. A aplicação desenvolvida tem grande apelo social, buscando maior acessibilidade a pessoas com grau de deficiência visual mais avançado, tornando-as mais independentes. Tal aplicação acarreta em inclusão social para seus usuários, possibilitando que realizem atividades cotidianas com mais facilidade e autonomia.

Este trabalho de conclusão de curso possibilitou ao autor aprofundar seu conhecimento na área de processamento de imagens, sistemas embarcados e tecnologias assistivas. O conhecimento construído possibilitou compreender as etapas do processamento de imagens voltado à estereoscopia visual, às dificuldades encontradas no dia a dia de deficientes visuais, padrões de acessibilidade, e também a lidar com os limites impostos pelo hardware comumente encontrados em SoCs.

O algoritmo da aplicação desenvolvida foi aperfeiçoado durante o desenvolvimento deste trabalho. No início foi verificado que a entrada de dados de ambas as câmeras não poderiam ser processadas pela mesma *Thread*, pois isso causava lentidão da captura de uma delas. Durante a construção do algoritmo para obter o mapa de disparidade, também se verificou necessário a implementação de uma aplicação paralela que auxiliasse no ajuste dos parâmetros a se adequarem aos aspectos do protótipo desenvolvido.

Constatou-se neste trabalho, assim como abordado nos trabalhos correlatos a ele, a dificuldade de se encontrar softwares de síntese de voz na linguagem portuguesa. O software utilizado para este fim neste trabalho, apesar de apresentar a linguagem portuguesa possui um tom muito artificial, o que com o tempo de uso pode se tornar cansativo.

A princípio a ideia do protótipo era de informar a direção, ou posição relativa do obstáculo no mundo real, porém foi observado que o campo de visão obtido pelas *webcams* utilizadas eram demasiados pequenos para esta funcionalidade. Portanto tornando a posição do obstáculo na imagem quase irrelevante, em geral os obstáculos detectados ficam sempre a frente devido a câmera não “enxergar” muito além disso.

Este trabalho possui vários pontos em que pode ser expandido e aprimorado. Pode ser adaptado para uso de lentes olho de peixe nas câmeras, o que possibilitaria um campo de visão muito mais amplo, possibilitando contornar o problema mencionado no parágrafo anterior. Pode-se também ser incrementado, implementando reconhecimento dos objetos comuns, tal como cadeiras, mesas, portas entres outros, assim podendo guiar o usuário até o encontro do objeto. Também pode-se trabalhar o software de síntese de voz, afim de melhorar a qualidade da comunicação com o usuário.

O SoC utilizado neste trabalho possui pinos de entrada e saída de dados, que também possibilita enriquecer o protótipo com outros tipos de sensores, ou até mesmo um controle externo para manipular a aplicação, uma vez que essa possa exercer mais de uma função.

REFERÊNCIAS

AGUIAR, F. de O. **Acessibilidade relativa dos espaços urbanos pedestres com restrições de mobilidade**. São Carlos, 2010. Tese (Doutorado em Engenharia). Escola de Engenharia de São Carlos, Universidade de São Paulo.

ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. 2004. Rio de Janeiro. ABNT 2004: Acessibilidade a edificações, mobiliário, espaços e equipamentos urbanos. Disponível em: <http://www.aracaju.se.gov.br/userfiles/emurb/2011/07/Normas_NBR9050_AcessibilidadeEdificacoes.pdf> Acesso em 02 set. 2014.

BERSCH, Rita. **Introdução à tecnologia assistiva**. Porto Alegre: Centro Especializado em Desenvolvimento Infantil [CEDI], 2008.

BRADSKI, Gary; KAEHLER, Adrian. **Learning OpenCV.** Beijing: O'reilly Media, 2008. 580 p.

BRAHMBHATT, Samarth. **Practical Opencv:** Hands on Projects For Computer Vision on the Windows, Linux, And Raspberry Pi Plataforms. New York: Apress, 2013. 244 p.

CASTRO, José Alberto Barbosa de Moura e. **Estudo da influência da capacidade de resistência aeróbia na orientação e mobilidade do cego.** 1993. 148 f. Tese (Doutorado) - Curso de Ciências do Desporto, Universidade do Porto, Porto, 1993.

CAZÉ, C. M. J. O; OLIVEIRA, A. S. **Dança Além da Visão:** Possibilidades do Corpo Cego. Disponível em: <http://www.revistas.ufg.br/index.php/fef/article/view/3592/4263> Acesso em 01 set. 2014.

COOK, Albert M.; HUSSEY, Susan M. **Assistive technologies:** principles and practice. St. Louis, Missouri. Mosby - Year Book, Inc., 1995. 712 p.

DIAS, C. O; PASSERINO, L. M. **Uma Proposta de Metodologia para Adoção de OA Usando Critérios de Acessibilidade.** 2009. 11 f. Universidade Federal do Rio Grande do Sul, Porto Alegre, 2009.

DOMÍNGUEZ-MORALES, M. et al. Stereo Matching: From the Basis to Neuromorphic Engineering. In: BHATTI, Asim (Ed.). **Current Advancements in Stereo Vision.** Intech, 2012, cap.1. Disponível em: <http://www.intechopen.com/books/current-advancements-in-stereo-vision/stereo-matching-from-the-basis-to-neuromorphic-engineering>. Acesso em: 8 nov.2014.

ESPINOSA, Carlos Andrés Diaz. **Uma aplicação de navegação robótica autônoma através de visão computacional estéreo.** 2010. 133 f. Dissertação (Mestrado) - Curso de Engenharia Mecânica, Universidade Estadual de Campinas, Campinas, 2010.

FIERA, Luiz Ricardo. **Ferramenta Para Localização Em Ambiente Controlado Utilizando Mapa De Sinais Wireless.** 2013. 90 f. TCC (Graduação) - Curso de Ciência da Computação, Universidade do Extremo Sul Catarinense, Criciúma, 2013.

FITZ, Paulo Roberto. **Geoprocessamento sem complicação**. São Paulo: Oficina de Textos, 2008. 160p.

FORSYTH, David A.; PONCE, Jean. **Computer Vision:** A Modern Approach. 2. ed. Boston: Pearson Prentice Hall, 2012. 792 p.

GONZALEZ, Rafael C.; WOODS, Richard E. **Processamento De Imagens Digitais**, São Paulo: Edgard Blücher, 2000.

GONZALEZ, Rafael C.; WOODS, Richard E. **Digital Image Processing.** 3. ed. New Jersey: Pearson Prentice Hall, 2007.

HOFFMANN, Lia Terezinha. **Abordagem ergonômica para a inserção laboral dos portadores de deficiência visual em estúdios de gravação:** um estudo de caso. 2002. 111 f. Dissertação (Mestrado) - Curso de Engenharia, Universidade Federal do Rio Grande do Sul, Porto Alegre, 2002.

HOFFMANN, S.B. **Orientação e Mobilidade: um processo de alteração positiva no desenvolvimento integral de criança portadora de cegueira congênita: estudo intercultural entre Brasil e Portugal. Dissertação (Mestrado) – Ciências do Movimento Humano,** Universidade Federal do Rio Grande do Sul, Porto Alegre, 1998.

IBGE. **Introdução ao processamento digital de imagens**. Rio de Janeiro: IBGE, 2000. 92p.

INSTITUTO BRASILEIRO DE GEOGRAFIA E PESQUISA. 2012. Rio de Janeiro. Censo Demográfico 2010. Disponível em: <http://www.ibge.gov.br/home/presidencia/noticias/imprensa/ppts/00000008473104122012315727483985.pdf> Acesso em: 16 maio 2014.

JURAFSKY, Daniel; MARTIN, James H. **Speech and Language Processing**. New Jersey: Pearson Prentice Hall, 2008. 988 p.

LAURENT, Andrew M. St.. **Understanding Open Source and Free Software Licensing.** Beijing: O'reilly Media, 2004. 224 p.

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

MARQUES FILHO, Ogê; VIEIRA NETO, Hugo. **Processamento Digital de Imagens.** Rio de Janeiro: Brasport, 1999. 410 p.

MARTINS, Alecsander Pereira. **Inspeção dimensional utilizando visão computacional em plataforma móvel.** 2013. 96 f. Dissertação (Mestrado) - Curso de Engenharia Elétrica, Universidade Estadual de Londrina, Londrina, 2013.

MAZIDI, Muhammad Ali; MCKINLAY, Rolin D.; CAUSEY, Danny. **PIC Microcontroller and Embedded Systems:** Using Assembly and C for PIC18. New Jersey: Pearson Prentice Hall, 2008. 832 p.

MUNHOZ, Alexandre. **Metodologia para detecção de obstáculos para navegação de embarcações autônomas usando visão computacional.** 2010. 110 f. Dissertação (Mestrado) - Curso de Engenharia Elétrica, Universidade de São Paulo, São Carlos, 2010.

NICHOLL, A.R.J. **O Ambiente que Promove a Inclusão**: Conceitos de Acessibilidade e Usabilidade. *Revista Assentamentos Humanos*. Marília, v. 3, n. 2, p. 49-60, 2001.

NIELSEN, J. **Designing Web Usability: The Practice of Simplicy.** New Riders Publishing. Indianapolis, 2000.

OPENCV. **Introduction.** Disponível em: <http://docs.opencv.org/modules/core/doc/intro.html>. Acesso em: 15 nov. 2014.

OPENCV. **OpenCV (open source computer vision).** Disponível em: <http://opencv.org/>. Acesso em: 15 nov. 2014.

PRADO, A.R de A. **Ambientes Acessíveis**. Disponível em: <http://www.entreamigos.com.br/sites/default/files/textos/Ambientes%20Acessiveis_2.pdf> Acesso em: 05 set. 2014.

RAPOSO, Alberto B. et al. **Visão Estereoscópica, Realidade Virtual, Realidade Aumentada e Colaboração**. In: CONGRESSO DA SOCIEDADE BRASILEIRA DE

COMPUTAÇÃO, 14., 2004, Salvador. **Anais...** Gávea: SBC, 2004, v. 2, XXIII JAI - Livro Texto, cap. 7, p. 289 – 331.

SILVA, Daniela Filipa M. B. M. da. **Algoritmos De Processamento Da Linguagem Natural Para Sistemas De Conversão Texto-Fala Em Português.** 2008. 202 f. Tese (Doutorado) - Curso de FiloloxÍa, Galego-portugués, Francés e Lingüística, Universidade da Coruña, A Coruña, 2008.

SILVA NETO, Sérvulo Fernandes da; ARAÚJO, Wagner Junqueira de. Avaliação de sintetizadores de voz para leitura em livros digitais. **Biblios**, v. 1, n. 51, p.78-90, jun. 2013. Disponível em: <http://www.redalyc.org/articulo.oa?id=16128807007>. Acesso em: 16 nov. 2014.

SILVEIRA, Sérgio Amadeu da. **Software livre:** A luta pela liberdade do conhecimento. São Paulo: Fundação Perseu Abramo, 2004. 79 p.

SOUSA, Kelly Aparecida Oliveira. **Uso de visão computacional em dispositivos móveis para Auxílio à travessia de pedestres com deficiência visual.** 2013. 85 f. Dissertação (Mestrado) - Curso de Engenharia Elétrica, Universidade Presbiteriana Mackenzie, São Paulo, 2013.

TAYLOR, Paul. **Text-to-Speech Synthesis.** Cambridge: Cambridge University Press, 2009.

TEIXEIRA, João Paulo; BARROS, Maria João; FREITAS, D.. Sistemas de conversão texto-fala. In: CONGRESSO LUSO-MOÇAMBICANO DE ENGENHARIA, 3., 2003, Maputo. **Proceedings... .** Porto: Edições Inegi, 2003. p. 1361 - 1374.

APÊNDICE(S)

APÊNDICE A - ARTIGO CIENTÍFICO

Identificação De Obstáculos Baseada Em Imagem Aplicada Ao Auxílio Na Mobilidade De Deficientes Visuais

Bruno de M. Amaral[1], Gustavo Bisognin[2]

[1]Acadêmico do Curso de Ciência da Computação – Unidade Acadêmica de Ciências, Engenharias e Tecnologias (UnaCET) – Universidade do Extremo Sul Catarinense (UNESC) Av. Universitária, 1105 - Bairro Universitário – Criciúma – SC – Brasil

[2]Professor do Curso de Ciência da Computação – Unidade Acadêmica de Ciências, Engenharias e Tecnologias (UnaCET) – Universidade do Extremo Sul Catarinense (UNESC) Av. Universitária, 1105 - Bairro Universitário – Criciúma – SC – Brasil

bruno-mattia@hotmail.com, gustavo@unesc.net

Abstract. *Through the large number of visually impaired people in society, more and more technologies are been researched to promote social inclusion for them; nevertheless still exists major barriers and limitations to their mobility. In order to extend the autonomy of the disabled population was developed a prototype of assistive technology. This article describes the construction of the prototype that helps visually impaired in their mobility through images processing application, for recognition of obstacles, and speech synthesis for the communication.*

Resumo. *Com o grande número de portadores de deficiência visual na sociedade, busca-se cada vez mais tecnologias que promovam inclusão social para estes, apesar disso ainda encontram-se grandes barreiras e limitações à sua locomoção. Com o objetivo de estender a autonomia da população portadora da deficiência foi desenvolvido um protótipo de tecnologia assistiva. Este artigo descreve a construção do protótipo que auxilia a mobilidade de deficientes visuais por meio da aplicação de processamento de imagens para reconhecimento de obstáculos, e síntese de voz para comunicação.*

1. Introdução

Em grande parte as informações que obtemos do ambiente ao nosso redor provém da visão (BRAHMBHATT, 2013), em deficientes visuais esta característica é limitada, impondo limites no seu dia a dia. A mobilidade para pessoas com deficiência visual é um fator bastante crítico, e tem chamado a atenção de diversos projetos de inclusão social nos últimos tempos.

O avanço da informática associado às políticas públicas e aos incentivos mundiais às tecnologias assistivas, têm direcionado uma série de trabalhos científicos que visam a aplicação da tecnologia computacional à melhoria de vida da população com algum tipo de deficiência.

A partir de um dispositivo portátil que proporcione ao deficiente visual maior autonomia, uma vez que o mesmo auxilie durante sua locomoção, é possível ampliar a acessibilidade a ambientes externos e internos sem necessidade de adaptação.

A constante evolução de dispositivos portáteis unida ao processamento de imagens, possibilitou o surgimento da tecnologia de identificação de obstáculos como sendo um apoio importante para quem possui algum tipo de limitação visual.

2. Acessibilidade a Deficientes Visuais

De acordo com a ISO-9050, o conceito de acessibilidade é: a possibilidade de acesso, alcance, percepção, e entendimento para que se possa utilizar de forma correta e com segurança, equipamentos, serviços, edificações e outros elementos (ABNT, 2004).

Todo ambiente público deveria ser acessível a todos de forma igual, porém esta tarefa se torna quase impossível, a partir do momento em que ela deve atender diferentes tipos e graus de deficiências, logo, seria necessária uma grande gama de recursos para ser concretizada (NICHOLL, 2001).

Uma ferramenta que auxilie o deficiente visual a identificar obstáculos ao seu redor, tem o propósito de fazer com que ele possa se movimentar pelo ambiente físico de forma mais prática e acessível, melhorando também sua vida social. Fitz (2008), relata que, tal ferramenta deve ser apresentada de forma agradável, intuitiva, e que de preferência, possa ser operada sem um treinamento prévio.

Os sintetizadores de voz possibilitam que deficientes visuais, de certa forma, leiam os sinais produzidos por uma ferramenta. Segundo Hoffmann (2002), um software leitor de tela, pode passar informações através de áudio para o deficiente visual, assim o mesmo pode usufruir das possibilidades dispostas na tela de um computador como, menus, ícones, textos entre outros. Logo, a sintetização de voz é uma tecnologia importante para promover tanto acessibilidade, quanto usabilidade para deficientes visuais.

3. Sistema de Reconhecimento de Obstáculos

A metodologia utilizada no desenvolvimento deste trabalho envolveu:

a) estudo de característica de acessibilidade para deficientes visuais, síntese de voz, técnicas de processamento de imagens incluindo estereoscopia visual, e métodos para reconhecimentos de obstáculos;

b) levantamento de requisitos, baseado em outros projetos e trabalhos envolvendo deficientes visuais e/ou estereoscopia visual, e também em necessidades próprias deste trabalho;

c) preparação do hardware utilizado, e de software, para fornecer um ambiente adequado para o desenvolvimento do protótipo;

d) implementação do algoritmo de calibração e a calibração do par de câmeras;

e) implementação do algoritmo de reconhecimento de obstáculos, e unido a ele o sintetizador de voz para comunicação com o usuário;

f) validação do protótipo.

3.1. Levantamento de Requisitos

Com base na leitura e pesquisa de trabalhos e projetos desenvolvidos para o auxílio de deficientes visuais, foi identificado que mesmo com o uso da bengala, ainda existem obstáculos que a mesma não detecta durante a locomoção do portador da deficiência, geralmente estes são objetos que ficam suspensos acima do nível da cintura, como por exemplo cestos de lixos e telefones públicos. Sendo assim admitiu-se que o foco deste trabalho seria identificar obstáculos à frente do usuário e que ficam ao nível acima da cintura.

Também foi constatado que, o meio mais comum usado para interação com deficientes visuais em sistemas ou ferramentas virtuais é a síntese de voz, sendo assim foi apurada a

necessidade de um software que realiza a síntese de voz em português e que possuísse características necessárias para ser embutido na aplicação.

Para o desenvolvimento da aplicação fez-se necessário a aquisição de duas webcams com interface USB, utilizadas para a captura das imagens e uso da técnica de estereoscopia visual.

A fim de manter o protótipo portátil, para que o usuário possa carregá-lo com maior facilidade, foi adquirida uma placa System-on-a-chip (SoC) com capacidade computacional para suportar a aplicação desenvolvida, para tal foi adotado o CubieBoard 2.

Optou-se por utilizar a linguagem Java, pelo fato do *software* poder ser portado de uma máquina para outra sem a necessidade de se alterar o código fonte ou de se recompilar o *software*.

3.2. Preparação do Hardware e Softwares

Uma parte importante do trabalho realizado foi a construção do par de câmeras estéreo. Para se obter imagens no formato estéreo com uma qualidade aceitável ambas as câmeras precisam ficar alinhadas na horizontal com a maior precisão possível, e então fixadas a uma distância arbitraria de 6 centímetros. Como é demostrado na Figura 1, a rotação tanto no eixo x, quanto no eixo y das câmeras também influenciam no resultado da estereoscopia, o ideal seria que ambas ficassem em ângulo reto nos dois eixos.

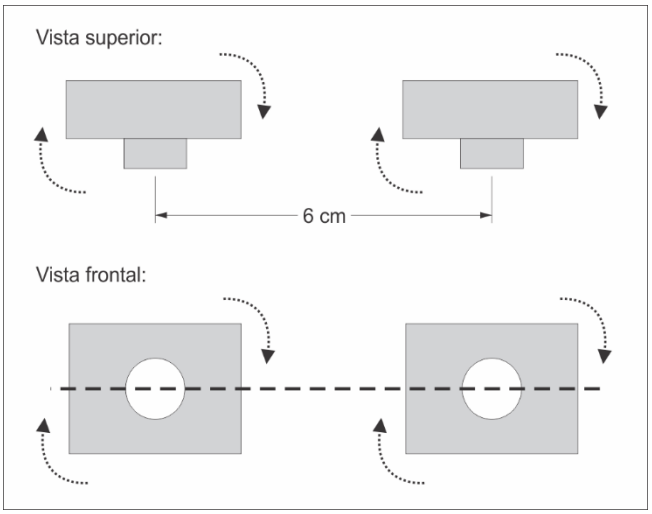


Figura 1. Posicionamento das câmeras

A construção do par foi necessário para otimizar o campo de visão disponível no mapa de disparidade, isso se deve ao fato de que o mapa de disparidade é gerado com base nos *pixels* que ambas imagens tem em comum, logo quanto maior a diferença no deslocamento de uma imagem para outra menor será o campo de visão.

Em seguida fora trabalhado o ambiente virtual, tratando as necessidades de softwares, bibliotecas e pacotes para o desenvolvimento e execução da aplicação.

Após pesquisa por distribuições do Linux compatíveis com a placa escolhida para este trabalho, foi escolhida a Cubian X1, que foi baseada no Debian e projetada especialmente para rodar em placas da empresa CubieBoard, apresentando maior compatibilidade com o hardware da placa do que as demais distribuições testadas.

Foi instalado e configurado o Java SE Development Kit 8 (JDK) para arquitetura ARM, obtido através do site oficial da Oracle. Foi instalado também dependências da biblioteca

OpenCV, foi utilizado o instalador de pacotes do sistema operacional para isso. Foram instalados os seguintes pacotes:

a) gcc-arm-linux-gnueabihf: este é um pacote de ferramenta para utilizar o compilador gcc que vem pré-instalado no SO, empregando instruções para processadores do tipo ARM;

b) Cmake: sistema de build, testes e gerenciador de softwares;

c) libjpg-turbo: é uma biblioteca que auxilia e acelera a manipulação de imagens em JPEG para processadores do tipo ARM;

d) pkg-config: fornece uma interface para consultar bibliotecas instaladas e compilação a partir do seu código-fonte;

e) libgtk: biblioteca de manipulação e edição de imagens;

f) libav: foram instalados os pacotes libavcodec e libavformat desta biblioteca de processamento de vídeo que tem suporte para processadores do tipo ARM;

g) libswscale: biblioteca que otimiza o redimensionamento de imagens e operações de conversão de cores.

Em seguida foi feito *download* do código fonte da versão 2.4.10 da biblioteca OpenCV, e compilada a versão para a linguagem Java da biblioteca, otimizada para processadores ARM.

Por fim, para a síntese de voz foi escolhido o software eSpeak, por ser multiplataforma, e também por disponibilizar a língua portuguesa. Também foi necessário configurar uma variável de ambiente para o sintetizador, pois o mesmo possui uma funcionalidade utilizada neste trabalho, a de permitir que se execute sínteses através de linha de comando, necessitando que o sistema reconheça o `espeak` como comando.

3.3. Calibração do par estéreo

Primeiramente para que seja possível extrair o mapa de disparidade a aplicação precisa retificar as imagens, sendo que as imagens captadas pelas câmeras possuem distorções geradas pelas lentes ou pelos próprios sensores, apesar de mínimas, estas distorções prejudicam a qualidade do mapa de disparidade, já que o algoritmo que o gera faz a busca, a nível de *pixels*, por semelhanças entre duas imagens. Após retificadas é necessário alinhá-las verticalmente para facilitar o esforço do algoritmo, assim a busca pelos *pixels* é feita apenas no sentido horizontal.

Este procedimento é comumente chamado de calibração estéreo, o OpenCV fornece métodos para realizar esta calibração. Um quadro ou uma folha impressa com o padrão de desenho de um tabuleiro de xadrez é utilizado para calibração tirando fotos deste ou o capturando em vídeo, a biblioteca reconhece os padrões na imagem capturada e consegue realizar a calibração com base nas distâncias entre os padrões, as capturas são convertidas em escala de cinza no momento em que são carregadas, pois torna mais fácil o reconhecimento dos padrões já que estes são apresentados em preto e branco.

O resultado da calibração são algumas matrizes, que serão utilizadas para retificar e alinhar as imagens captadas posteriormente, estas matrizes são armazenas em arquivo de texto, e serão carregadas na aplicação que fara uso delas para formatar as imagens capturadas pelo par calibrado.

Lembrando que a calibração é válida para o estado em que o par de câmeras estava durante a calibração, caso seja alterada a rotação, translação ou o foco (zoom) de alguma das câmeras, uma nova calibração deve ser feita.

3.4. Algoritmo de Reconhecimento de Obstáculos

A aplicação foi desenvolvida no NetBeans IDE 8.0.2 na linguagem Java, ela foi dividida em três pacotes:

a) output: nele estão as classes que se destinam a exibir as imagens capturadas pelas câmeras, o resultado do mapa de disparidade, e síntese de voz;

b) processing: responsável por capturar as imagens, processá-las, remover distorções, alinhá-las, criar o mapa de disparidade e reconhecer obstáculos;

c) utils: este pacote possui classes de apoio aos outros módulos, como armazenamento de matrizes, conversores entre outros, também se encontra neste pacote a classe que efetua a calibração das câmeras.

Foi criada uma interface visual (Figura 2) para se poder visualizar o resultado gerado, porém para o funcionamento pleno da aplicação esta não é necessária uma vez que aplicação é feita para deficientes visuais, e esta se comunicará por síntese de voz. Por meio da interface é possível capturar as imagens, esquerda e central (Figura 2), e salvá-las na pasta raiz da aplicação, a serem empregadas na fase de calibração.

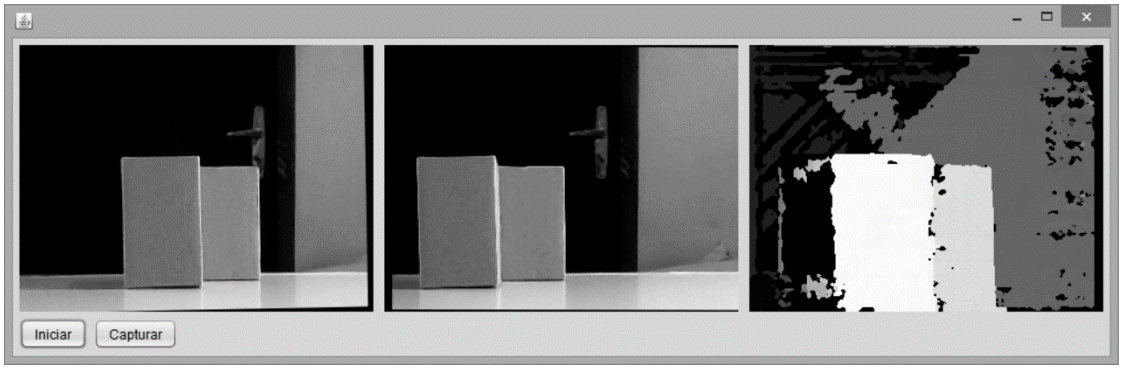


Figura 2. Interface da aplicação

Ao iniciar a aplicação, o software reconhece as câmeras plugadas e inicia a captura em forma de vídeo rodando a 30 quadros por segundos, a biblioteca OpenCV entrega cada quadro no formato de matriz de 320x240 e 3 canais, que é a representação de uma imagem, com resolução de 320 pixels de largura por 240 de altura e 3 canais de cores, *blue*, *green e red* (BGR), formato padrão da biblioteca. Ao mesmo tempo o software carrega as matrizes de calibração, e após isso a cada quadro recebido as imagens são retificadas e alinhadas através delas, antes de serem exibidas na tela.

Para evitar que o fluxo de entrada de dados, e o alinhamento das imagens de uma câmera concorra com a outra pelo uso do processador, cada captura de vídeo é executada em uma *Thread* separada.

A cada *frame* capturado a matriz que representa ele é enviada para outra classe que roda na *Thread* principal da aplicação, e mantém duas matrizes uma para imagens capturadas da câmera esquerda e outra para imagens capturadas pela câmera direita, uma vez que a classe tenha as duas matrizes preenchidas é extraído o mapa de disparidade a partir das duas.

Para a obtenção do mapa de disparidade foi utilizada classe StereoSGBM do OpenCV, ao se instanciar esta classe é passado parâmetros que vão regular e refinar o mapa que será obtido ao chamar a função *compute* da classe, os parâmetros são:

a) minDisparity: valor mínimo de disparidade, ajustado de acordo com o deslocamento na imagem provocado pela calibração, caso não haja deslocamento este valor deve ser zero;

b) numDisparities: máximo de disparidade subtraído o mínimo de disparidade, este valor deve ser múltiplo de 16 e maior que zero;

c) SADWindowSize: tamanho da área que o algoritmo vai procurar pelo pixel, no processo de matching, deve ser um valor ímpar;

d) P1: primeiro parâmetro de controle de suavidade da disparidade;

e) P2: segundo parâmetro de controle da suavidade, sendo quanto maior o valor mais suave será o resultado, o algoritmo necessita que o valor de P1 seja menor que o de P2;

f) disp12MaxDiff: distância máxima permitida entre pixels na checagem de esquerda-para-direita. Um valor negativo desabilita a checagem;

g) preFilterCap: valor de truncamento para pixels pré-filtrados da imagem. O algoritmo deriva cada pixel e trunca seu valor no intervalo de -preFilterCap até preFilterCap;

h) uniquenessRatio: margem de porcentagem em que o primeiro valor (pixel), deve ser melhor que o segundo, para que seja escolhido no processo de matching;

i) speckleWindowSize: valor máximo (de 50 a 200) a ser considerado como mancha de ruído no mapa de disparidade, e então invalidado. O valor 0 desabilita o cancelamento de manchas;

j) speckleRange: define a diferença de disparidade que é considerado como mancha;

k) fullDP: habilita o algoritmo completo para obtenção do mapa de disparidade, consome quantidade maior de processamento e memória.

O resultado gerado pela classe StereoSGBM precisa ser normalizado para se tornar inteligível, na Figura 3 é demonstrado a disparidade normalizada na escala de cinza, onde é possível distinguir que objetos mais próximos ficam em tons mais claros, enquanto objetos mais distantes ficam em tons mais escuros.

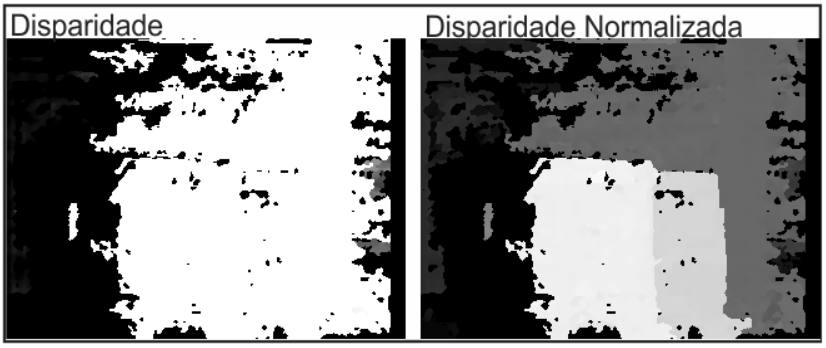


Figura 3. Normalização do mapa de disparidade

O reconhecimento de obstáculos foi feito com base na matriz da disparidade normalizada, sabendo que a matriz tem as mesmas proporções das imagens capturadas pelas câmeras, 320x240 *pixels*, pode-se calcular a quantidade total de *pixels* presentes na imagem, simplesmente multiplicando os valores, obtém-se então o valor de 76800 *pixels*.

Foi estipulado que um objeto que ocupe 20% da imagem será considerado obstáculo, para isso primeiro foi definido a proximidade com que o objeto será detectado, como a imagem foi normalizada em escala de cinza seus pixels assumem valores entre 0 e 255, quanto maior este valor mais claro é o tom de cinza, em outras palavras, mais próximo estará o objeto que conter um *pixel* com valor alto.

Utilizando o valor de 120 como limite, o valor representa objetos em torno de um metro de distância, logo o algoritmo admitirá *pixels* com valores na faixa de 120 a 250, reconhecendo objetos a um metro ou menos. O algoritmo percorre toda a matriz registrando em um contador a quantidade de *pixels* cujo valor seja maior que 120, se ao final da contagem a quantidade de pixel for maior que 20% de 76800, admite-se que há um objeto sólido a frente do usuário.

Como o software eSpeak usado para síntese de voz deste trabalho permite sua execução por linha de comando, não foi necessária integração direta com a aplicação. A linguagem Java permite chamar/criar processos do sistema operacional, logo para informar o usuário sobre o obstáculo foi feito uma simples chamada desta maneira: *Process p = rt.exec(espeak -vpt+f3 "Obstáculo a frente");* onde o parametro *-vpt+f3* indica ao eSpeak que a linguagem usada é o português e a voz utilizada será o *f3*, voz feminina 3, seguido então pela mensagem que será sintetizada.

3.5. Validação do Protótipo

A validação do protótipo ocorreu de duas maneiras, a primeira de forma técnica, validando o comportamento do reconhecimento de obstáculos, a segunda maneira, de forma empírica, foi analisado a possibilidade de se guiar pelo protótipo e o tempo de resposta entre a aparição do obstáculo e a detecção do mesmo pela aplicação.

Para avaliar o reconhecimento de obstáculos, a aplicação foi executada em um notebook, por ser mais fácil de visualizar a saída de dados, e o processo ser mais rápido e prático, foi mantido as câmeras em posição fixa, e em seguida adicionado objetos em seu campo de visão.

Para simulação dos obstáculos foram utilizadas caixas de tamanhos variados. As dimensões de uma das faces de cada caixa utilizada para simulação dos obstáculos estão dispostas na Tabela 1, foi disposto apenas a face que ficou voltada para a câmera durante a validação.

Tabela 1. Dimensões dos objetos

Objeto	Altura	Largura	Área
Caixa 1	9 cm	11 cm	99 cm²
Caixa 2	11 cm	17 cm	187 cm²
Caixa 3	14 cm	20 cm	280 cm²
Caixa 4	28 cm	28 cm	784 cm²
Caixa 5	32 cm	40 cm	1280 cm²

Os objetos foram dispostos individualmente um de cada vez, nas distâncias de 50cm, 90cm e 120cm das câmeras, e avaliado se o protótipo realizava o reconhecimento corretamente, lembrando que a distância implica no tamanho que o objeto é identificado pelas câmeras, sendo que mesmo objetos grandes, porém distantes, não devem ser detectados como obstáculos. Nos dados observados na Tabela 2, é exibido quais objetos, de acordo com a Tabela 1, foram reconhecidos marcados com o símbolo ✓, e não reconhecidos marcados com um X, e em quais distâncias ocorreu.

Tabela 2. Obstáculos reconhecidos

Distância / Objeto	50 cm	90 cm	120cm
Caixa 1	✓	X	X
Caixa 2	✓	X	X
Caixa 3	✓	✓	X
Caixa 4	✓	✓	✓
Caixa 5	✓	✓	✓

Como pôde ser observado a medida que os obstáculos menores se afastam, a aplicação não os detecta mais, este era o resultado esperado, pois quanto mais distante o objeto está menor é sua representação na imagem, então em certa distância o objeto já não representa risco imediato de colisão com o usuário. Já os objetos maiores acabam continuando a serem detectados mesmo estando mais distante por conta de ainda tomar grande parte da imagem capturada, isso possibilita a detecção de paredes por exemplo. Todas as detecções de obstáculos ocorreram em cerca de 1 segundo após o objeto ser posicionado.

Em seguida foi avaliado o reconhecimento dos obstáculos com o protótipo em movimento. O reconhecimento ocorreu de maneira bastante similar aos resultados obtidos anteriormente, porém foi verificado que ao se fazer movimentos rápidos as imagens ficam borradas pela velocidade com que o plano de fundo se desloca, isso em algumas situações geraram falsos positivos, detectando obstáculos erroneamente.

4. Conclusão

A aplicação desenvolvida tem grande apelo social, buscando maior acessibilidade a pessoas com grau de deficiência visual mais avançado, tornando-as mais independentes. Tal aplicação acarreta em inclusão social para seus usuários, possibilitando que realizem atividades cotidianas com mais facilidade e autonomia.

Com o mercado aquecido de SoCs, vem surgindo placas cada vez mais robustas e eficientes, possibilitando a realização de ideias, como a proposta neste trabalho. Apesar das limitações de hardware, o protótipo desenvolvido no decorrer deste trabalho conseguiu atender satisfatoriamente o seu objetivo, foi possível detectar obstáculos a curta distância, como paredes, cadeiras e entre outros, a tempo de informar o usuário através da síntese de voz.

O algoritmo da aplicação desenvolvida foi aperfeiçoado durante o desenvolvimento deste trabalho. No início foi verificado que a entrada de dados de ambas as câmeras não poderiam ser processadas pela mesma *Thread*, pois isso causava lentidão da captura de uma delas. Durante a construção do algoritmo para obter o mapa de disparidade, também se verificou necessário a implementação de uma aplicação paralela que auxiliasse no ajuste dos parâmetros a se adequarem aos aspectos do protótipo desenvolvido.

Este trabalho possui vários pontos em que pode ser expandido e aprimorado. Pode ser adaptado para uso de lentes olho de peixe nas câmeras, o que possibilitaria um campo de visão muito mais amplo. Pode-se também ser incrementado, implementando reconhecimento de objetos mais comuns, tal como cadeiras, mesas, portas entres outros, assim podendo guiar o usuário até o encontro do objeto. Também pode-se trabalhar o software de síntese de voz, afim de melhorar a qualidade da comunicação com o usuário.

O SoC utilizado neste trabalho possui pinos de entrada e saída de dados, que também possibilita enriquecer o protótipo com outros tipos de sensores, ou até mesmo um controle externo para manipular a aplicação, uma vez que essa possa exercer mais de uma função.

Referências

ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. 2004. Rio de Janeiro. ABNT 2004: Acessibilidade a edificações, mobiliário, espaços e equipamentos urbanos. Disponível em: <http://www.aracaju.se.gov.br/userfiles/emurb/2011/07/Normas_NBR9050_Acessibilidade Edificacoes.pdf> Acesso em 02 set. 2014.

BRAHMBHATT, Samarth. Practical Opencv: Hands on Projects For Computer Vision on the Windows, Linux, And Raspberry Pi Plataforms. New York: Apress, 2013. 244p.

FITZ, Paulo Roberto. Geoprocessamento sem complicação. São Paulo: Oficina de Textos, 2008. 160p.

HOFFMANN, Lia Terezinha. Abordagem ergonômica para a inserção laboral dos portadores de deficiência visual em estúdios de gravação: um estudo de caso. 2002. 111 f. Dissertação (Mestrado) - Curso de Engenharia, Universidade Federal do Rio Grande do Sul, Porto Alegre, 2002.

NICHOLL, A.R.J. O Ambiente que Promove a Inclusão: Conceitos de Acessibilidade e Usabilidade. Revista Assentamentos Humanos. Marília, v. 3, n. 2, p. 49-60, 2001.