CENTRO UNIVERSITÁRIO UNIVATES
CENTRO DE CIÊNCIAS EXATAS E TECNOLÓGICAS
CURSO DE ENGENHARIA DE CONTROLE E AUTOMAÇÃO

SISTEMA DE AUTOMAÇÃO COM VISÃO COMPUTACIONAL PARA INSPEÇÃO DE NÍVEL DO ENCHIMENTO DE EMBALAGENS

DOUGLAS ARNT

Lajeado, julho de 2017.

DOUGLAS ARNT

SISTEMA DE AUTOMAÇÃO COM VISÃO COMPUTACIONAL PARA INSPEÇÃO DE NÍVEL DO ENCHIMENTO DE EMBALAGENS

Trabalho de Conclusão de Curso apresentado ao Centro de Ciências Exatas e Tecnológicas do Centro Universitário UNIVATES, como parte dos requisitos para a obtenção do título de bacharel em Engenharia de Controle e Automação.
Área de concentração: Automação e Visão Computacional

ORIENTADOR: Anderson Antônio Giacomolli

CO-ORIENTADOR: Ronaldo Hüsemann

Lajeado, julho de 2017.

DOUGLAS ARNT

SISTEMA DE AUTOMAÇÃO COM VISÃO COMPUTACIONAL PARA INSPEÇÃO DE NÍVEL DO ENCHIMENTO DE EMBALAGENS

Este trabalho foi julgado adequado para a obtenção do título de bacharel em Engenharia de Controle e Automação do CETEC e aprovado em sua forma final pelo Orientador e pela Banca Examinadora.

Orientador: _____

Prof. Anderson Antônio Giacomolli, UNIVATES.

Mestre pelo PPGE/UFRGS – Porto Alegre, Brasil.

Banca Examinadora:

Prof. Anderson Antônio Giacomolli, UNIVATES.

Mestre pelo PPGE/UFRGS – Porto Alegre, Brasil.

Prof. Jaime Andre Back, UNIVATES.

Mestre pelo PPGE/UNISC – Santa Cruz do Sul, Brasil.

Prof. Juliano Schirmbeck, UNIVATES.

Mestre pelo PPGE/UNISINOS – São Leopoldo, Brasil.

Coordenador do Curso de Eng. de Controle e Automação :_____

Prof. M.Sc. Anderson Antônio Giacomolli

Lajeado, Julho de 2017.

Dedico este trabalho aos meus pais e minha esposa, em especial pela dedicação e apoio em todos os momentos difíceis.

AGRADECIMENTOS

Aos professores pela contribuição e oportunidade para realização deste trabalho, em especial aos professores Ronaldo Hüsemann e Anderson A. Giacomolli que orientaram e colaboraram para a elaboração desta monografia.

Aos colegas do curso e a todos que de alguma forma contribuíram para esta caminhada acadêmica e profissional.

RESUMO

Os sistemas de visão computacional e processamento digital de imagens vêm crescendo e sendo aplicados nas mais diferentes áreas. Nas industriais, muitos dos problemas relacionados com as inspeções e o monitoramento de processos realizados de forma manual, podem ser solucionados ou melhorados com a implementação de sistemas de visão computacional, visando maior eficiência e desempenho destes processos. As inspeções de embalagens são tarefas rotineiras, principalmente nas indústrias dos segmentos de bebidas e alimentos, onde muitos destes produtos são envasados em garrafas e embalagens plásticas, distribuídos e comercializados em unidades por volume. Este processo geralmente é mecanizado e automatizado em layout de esteiras, onde deve ser realizada a inspeção do nível de enchimento das embalagens antes da expedição ao consumidor final. O presente trabalho consiste no desenvolvimento de um sistema de automação com visão computacional para a inspeção do nível de enchimento em processos de envase de embalagens plásticas transparente. O sistema terá a funcionalidade de detectar as embalagens deslocadas por uma esteira, que na passagem da mesma por um sensor óptico, realiza a captura da imagem da embalagem por uma câmera digital, ambos conectados em um computador. O computador neste contexto realiza o processamento digital das imagens através de métodos de visão computacional, realizando o reconhecimento e a validação do nível correto de enchimento e acionando uma saída de alarme caso ocorrer um enchimento incorreto ou evento de não conformidade. Os testes funcionais com o protótipo foram realizados no laboratório de automação da instituição com a simulação de um ambiente industrial, apresentando resultados satisfatórios com uma boa precisão e eficiência na validação das inspeções. Resultados que ainda podem ser melhorados com alguns aperfeiçoamentos no hardware e software para atender aplicações simplificadas em ambiente industrial.

Palavras-chave: Sistema de Automação. Inspeção. Embalagens. Visão Computacional.

ABSTRACT

The systems of computer vision and digital processing of images have been growing and being applied in the most different areas. In manufacturing, many of the problems related to inspection and monitoring of processes performed manually, can be solved or improved with the implementation of computer vision systems, aiming at greater efficiency and performance of these processes. Packaging inspections are routine tasks, mainly in the beverage and food industries, where many of these products are bottled and plastic packaging, distributed and sold in units by volume. This process is usually mechanized and automated in the layout of mats, where inspection of the filling level of the packages must be carried out before dispatch to the final consumer. The present work consists in the development of an automation system with a computer vision for inspection of the level of filling in transparent plastic packaging processes. The system will have the functionality of detecting the packets displaced by a conveyor, which in the passage of the same by an optical sensor, captures the image of the packaging by a digital camera, both connected in a computer. The computer in this context performs the digital processing of the images through computational vision methods, performing the recognition and validation of the correct level of filling and triggering an alarm output in the event of an incorrect filling or non-compliance event. The functional tests with the prototype were performed in the automation laboratory of the institution with the simulation of an industrial environment, presenting satisfactory results with a good precision and efficiency in the validation of the inspections. Results that can still be improved with some improvements in hardware and software to meet simplified applications in an industrial environment.

Keywords: Automation System. Inspection. Packaging. Computer Vision.

LISTA DE FIGURAS

LISTA DE CÓDIGOS

LISTA DE TABELAS

LISTA DE ABREVIATURAS

CCD:	Charge Coupled Device
CMOS:	Complementary Metal-Oxide Semiconductor
CPU:	Central Processing Unit
DDR:	Double Data Rate
GIF:	Graphics Interchange Format
GPIO:	General Purpose Input Output
HDMI:	High-Definition Multimedia Interface
HSV:	Hue, Saturation and Value
I/O:	Input/Output
IDE:	Integrated Development Environment
IHM:	Human-Machine Interface
IP:	Ingress Protection
LED:	Light Emitting Diode
LCD:	Liquid Crystal Display
JPEG:	Joint Photographic Experts Group
NTSC:	National Television System Committee
PAL:	Phase Alternating Line
RAM:	Random Access Memory
RGB:	Red, Green and Blue
SD:	Secure Digital
ULA:	Unidade Lógica Aritmética
USB:	Universal Serial Bus
VCC:	Tensão Corrente Contínua
VCA	Tensão Corrente Alternada

SUMÁRIO

1 INTRODUÇÃO

Atualmente os sistemas de visão computacional e processamento digital de imagens vêm crescendo e sendo aplicados nas mais diferentes áreas. No processamento digital de imagens, as mesmas podem ser classificadas de acordo com a fonte onde as imagens são geradas na aquisição, como exemplo: imagens de raios-X e gama, infravermelhas e ultravioletas, ultrassom, feixes de luz, entre outras. Utilizadas nos campos da aeronáutica e o geoprocessamento de imagens, na medicina para análises de imagens, na biologia com imagens microscópicas, entre outros segmentos. Enquanto a visão computacional com a função de imitar a visão humana vem sendo amplamente aplicada nas áreas da automação industrial com sistemas automatizados, na metrologia em medições geométricas assistida por sensores de visão, em sistemas de segurança com o reconhecimento facial, entre outras aplicações (GONZALEZ, WOODS, 2008).

Segundo Keyence (2016), nas indústrias muitas aplicações necessitam de análises visuais para monitorar e controlar seus processos e sistemas. Muitos destes ainda são operados de forma manual, ou seja, necessitam da interação humana em fazer uma análise visual e interpretação para realizar uma determinada tarefa no processo. A operação manual por problemas como a fadiga, o descuido ou desconcentração do operador, podem comprometer a qualidade, a segurança e a confiabilidade do sistema. A figura 1 demonstra uma operação manual com inspeção visual do nível de enchimento de embalagens.

Figura 1 - Operação manual com inspeção visual do nível de enchimento de embalagens.

Fonte: S.N Puríssima (2017).

Os sistemas de visão computacional são equipamentos automatizados, desenvolvidos para substituir ou auxiliar a mão de obra humana, basicamente compostos por: câmeras de captura de imagens, computador, iluminação, softwares e hardwares de processamento digital de imagens. A utilização desta tecnologia pode substituir a análise visual humana, que quando comparada pode até ser muitas vezes mais rápida, precisa e consistente em atividades repetitivas, aumentado a confiabilidade, a qualidade e a segurança do processo com menores gastos financeiros no quadro operacional (KEYENCE, 2016).

Existem vários fabricantes de sistemas de visão computacional e processamento digital de imagens no mercado atual, visando melhorar a eficiência e o desempenho dos processos com a eliminação de defeitos, inspeções e monitoramento de produtos e montagens, aquisição de informações de fases de processo, entre outras funcionalidades para as mais diferentes aplicações, facilitando a automação dos mesmos (COGNEX, 2016). A figura 2 demonstra uma imagem ilustrativa de um sistema de visão computacional utilizado para inspeção do nível de enchimento de embalagens.

Figura 2 - Sistema de visão computacional aplicado para inspeção de embalagens.

Fonte: Omron (2016).

Neste contexto, analisando as embalagens dos mais diferentes produtos industrializados e comercializados, principalmente pelas indústrias de bebidas e alimentos. Observa-se que muitos destes produtos produzidos por estes setores são envasados e comercializados em garrafas e ou embalagens plásticas, geralmente distribuídas em unidades por volume. Segundo Omron (2017), os processos industriais de envase de garrafas e embalagens plásticas geralmente são mecanizados e automatizados em *layouts* na forma de

esteiras, e a inspeção do nível de enchimento dos produtos é realizado muitas vezes de forma manual ou através de amostragens por lotes. Tarefa que quando realizada de forma ineficiente pode possibilitar um enchimento irregular destes produtos no mesmo.

Um enchimento abaixo do nível pode ocasionar reclamações dos consumidores e até mesmo a rejeição dos produtos pela divergência do volume fornecido. Como também um enchimento acima do nível pode ocasionar perdas financeiras as empresas. Estes problemas podem ser solucionados ou melhorados, com a implementação de sistemas de automação com visão computacional, tornando as inspeções dos níveis de enchimento de embalagens mais eficiente e preciso, trazendo resultados satisfatórios.

Com o crescimento da utilização de tecnologias com visão computacional em aplicações industriais e o interesse de aprender mais sobre o tema, a proposta deste trabalho consiste no desenvolvimento de um sistema de automação com visão computacional para inspeção do nível de enchimento de embalagens plásticas transparente, em processos de envase de produtos líquidos e sólidos em movimentação por esteira motorizada em ambiente industrial, simulado em laboratório de automação. Apresentando um protótipo de baixo custo, que terá a funcionalidade de detectar as embalagens que se movem por uma esteira, através da passagem das mesmas por um sensor óptico, realizando a captura da imagem por meio de uma câmera digital, ambos conectados em um computador.

Após a captura da imagem, o computador executando o software do sistema embarcado terá a funcionalidade de realizar a inspeção do nível de enchimento das embalagens, através de métodos de visão computacional pré-programados, realizando o reconhecimento e a validação do enchimento correto, e a atuação de uma saída digital para alarme quando a embalagem estiver com o enchimento incorreto ou evento de não conformidade ocorrer. O software embarcado no computador foi escrito em linguagem de programação C/C++ com a utilização da biblioteca de visão computacional OpenCV e a biblioteca de aplicação Qt, executado em sistema operacional Linux.

O presente trabalho está estruturado em seis capítulos. No Capítulo 2 foi realizada uma revisão literária dos principais conceitos teóricos relacionados ao projeto. O Capitulo 3 descreve sobre a especificação do projeto, apresentando a proposta do trabalho e especificando os hardwares e softwares utilizados em seu desenvolvimento. O Capítulo 4 descreve detalhadamente o desenvolvimento do sistema. Os testes realizados e os resultados obtidos com a experimentação prática são apresentados no Capítulo 5. E por último é apresentado a conclusão do trabalho e as propostas futuras no Capitulo 6.

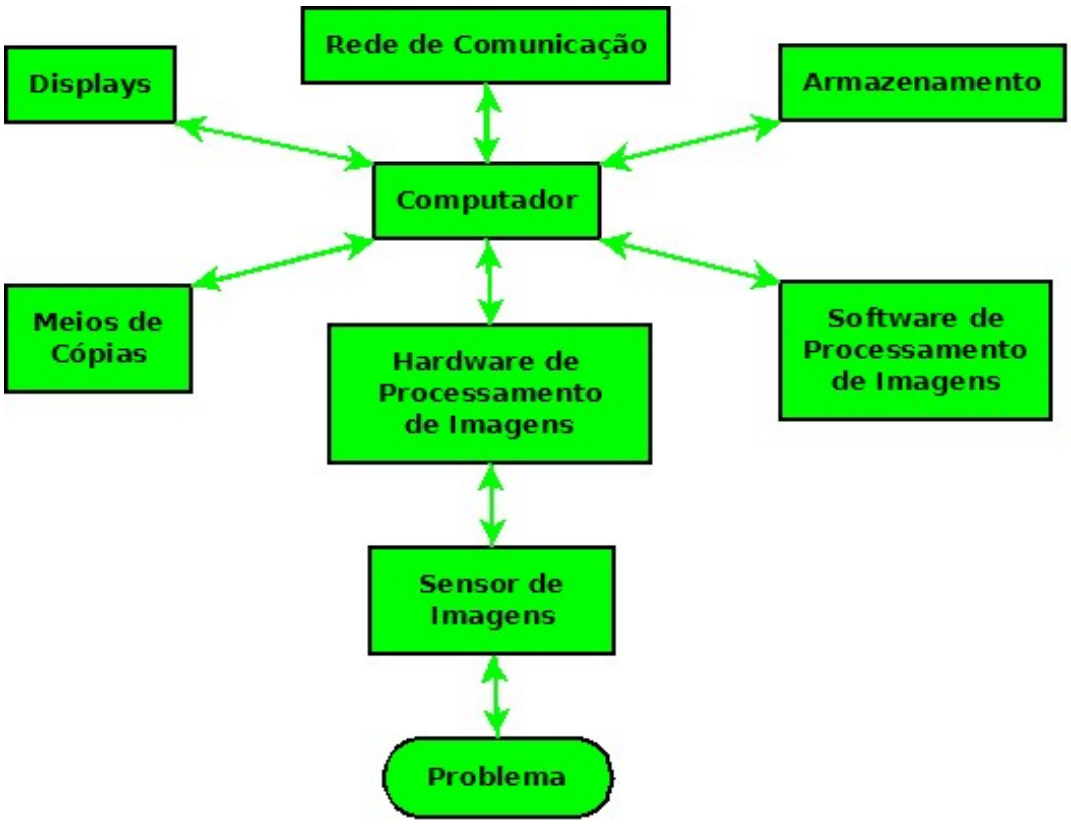
2 FUNDAMENTAÇÃO TEÓRICA

Os conceitos teóricos apresentados a seguir neste capitulo, são de fundamental importância para o entendimento deste trabalho acadêmico.

2.1 Sistemas de Processamento Digital de Imagens

Os sistemas de processamento digital de imagens já eram comercializados em meados de 1980, com periféricos conectados em computadores de grande porte. No inicio de 1990 o mercado de hardware de processamento de imagens começou a mudar e novas placas projetadas para estações de trabalho e computadores pessoais de baixo custo, impulsionaram o surgimento de novas empresas especializadas em sistemas de processamento de imagens, com a tendência de miniaturização destes sistemas com os computadores cada vez menores. Um sistema de processamento digital de imagens com seus principais componentes podem ser representados de forma genérica pelo diagrama mostrado na Figura 3 (GONZALEZ; WOODS, 2008).

Figura 3 - Diagrama dos componentes de um sistema de processamento digital de imagens.



Fonte: Adaptado de Gonzalez e Woods (2008).

Conforme Gonzalez e Woods (2008), os principais componentes de um sistema de processamento imagens são:

- **Sensor de Imagens:** São dispositivos utilizados para realizar a aquisição de imagens, como exemplo: as câmeras digitais e os scanners. Estes dispositivos são basicamente construídos por dois elementos principais: o sensor e o digitalizador. O sensor é um dispositivo físico sensível a energia irradiada pela imagem desejada, que produz em sua saída um sinal elétrico proporcional à intensidade da energia recebida do ambiente onde está a imagem. O digitalizador por sua vez converte este sinal elétrico vindo do sensor em informação digital, representado em sinal binário para o processamento digital da imagem.
- **Hardware de Processamento de Imagens:** Geralmente é constituído em um digitalizador com um hardware de Unidade Lógica Aritmética (ULA). Este dispositivo além de realizar a discretização da imagem recebida pelo sensor, pode reduzir ruídos, e alterar as propriedades de brilho e contrastes com operações matemáticas, tornando o processamento de imagens mais eficiente.
- **Computador:** é responsável por realizar o processamento digital de imagens, sendo um dos componentes mais importantes, que interage diretamente com quase todos os componentes do sistema. Este pode ser um computador de uso geral para aplicações mais simples como a aquisição e exibição de uma imagem 2D (bidimensional), ou até ser um supercomputador para aplicações especiais que necessitam de um processamento de imagens mais complexo, como o geoprocessamento de imagens.
- **Software de Processamento de Imagens:** São procedimentos escritos sob algoritmos em linguagens de programação implementados no computador para executar a maioria das funções do processamento de imagens.
- **Armazenamento:** é a capacidade de espaço de memória medida em bytes, dedicada para o armazenamento de imagens e dados para realizar o processamento de imagens. O armazenamento esta dividido em três categorias: armazenamento de curto prazo para o uso durante o processamento (utiliza memória RAM); o armazenamento de massa para recuperação relativamente rápida (utiliza o disco rígido ou memória SD - *Secure Disk*), e o armazenamento de arquivo que é caracterizado pelo acesso freqüente e armazenamento de quantidades maiores de dados, utilizando os discos rígidos e as memórias SD geralmente.

- **Meio de Cópias:** São dispositivos utilizados para realizar a cópia das informações comunicadas pelo sistema de processamento de imagens. Podem ser: impressoras, discos ópticos, pendrives, cartões de memória, entre outros.
- **Displays:** São dispositivos eletrônicos responsáveis em apresentar os resultados gráficos do processamento de imagens. Podem ser: monitores (LCD – *Liquid Cristal* Display, LED – *Light Emitter Diode*, tubo de raios catódicos), IHMs (Interface Homem Máquina), televisores entre outros.
- **Rede de Comunicação:** A rede de comunicação é um componente importante nos sistemas atuais que necessitam da comunicação remota entre computadores ou via internet. Este componente pode viabilizar o acesso remoto entre o local de operação do sistema e uma estação remota.

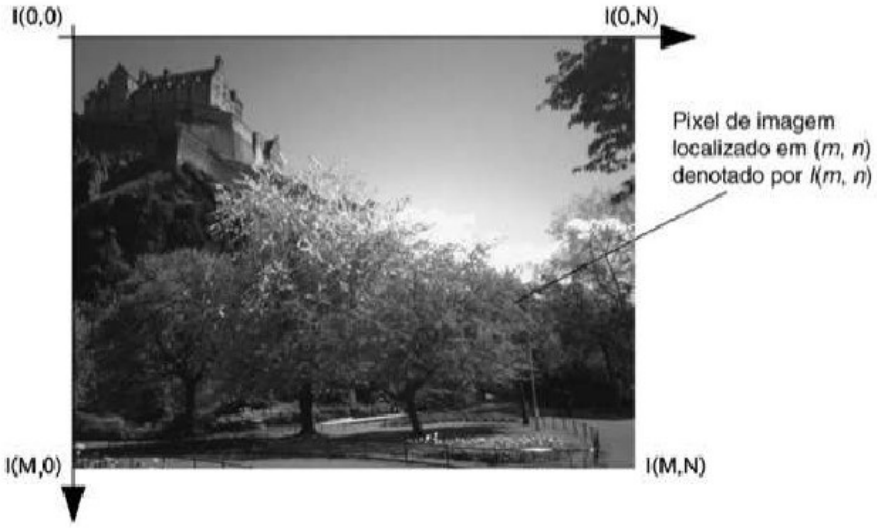
2.2 Fundamentos de Imagem Digital

Segundo Snyder e Qi (2010) a aquisição de uma imagem digital é realizada através de um sensor de imagens que realiza a conversão de um cenário 3D (tridimensional) em uma imagem eletrônica 2D. Geralmente estes sensores possuem um digitalizador que efetua uma média do sinal analógico gerado, através de uma amostragem finita, discretizando este em um sinal digital que pode ser representado de forma binária por uma matriz M por N. Os índices M e N desta matriz são respectivamente as linhas e colunas da imagem que determinam a resolução espacial da mesma, dada por M x N, como exemplo: resolução de imagem 640 x 480. Os pixels de uma imagem são os elementos individuais desta matriz. que são referidos pelos índices I (M, N), considerados como a menor unidade de uma imagem digital.

Uma imagem digital pode possuir uma ou mais cores, que junto com fatores, como: a iluminação do ambiente, o brilho e contraste, define a intensidade do sinal de um determinado pixel I (M, N). A intensidade de cada pixel pode ser representada digitalmente por um valor numérico, possibilitando uma conversão para uma imagem real através de um mapa de cor. O mapa de cor mais comum é a escala de cinza que aloca a cor preta como zero e cor branca como o valor máximo numérico (exemplo: 255 no caso em 8 bits), representando desta forma todos os tons de cinza nesta escala. A escala de cinza associa um único valor para cada posição de pixel definida pela intensidade do sinal da imagem. Enquanto para obtenção de imagens coloridas, cada pixel deve ser representado por um vetor com as três componentes: vermelho, verde e azul e com seus respectivos valores numérico, representando uma imagem RGB (*Red Green Blue*). Com a combinação destas três componentes fundamentais ou valores

associados em cada pixel, pode-se constituir uma imagem colorida com as demais cores existentes. Além desta representação, também é muito usual o mapa de cor HSV, onde: H ou *Hue* é a tonalidade, S ou *Saturation* é a saturação e V ou *Value* é o valor de intensidade da cor da imagem. A figura 4 apresenta uma imagem em escalas de cinza no plano 2D com a representação de um pixels em uma resolução M x N (SOLOMON; BRECKON, 2013).

Figura 4 - Imagem em escalas de cinza com resolução M x N.



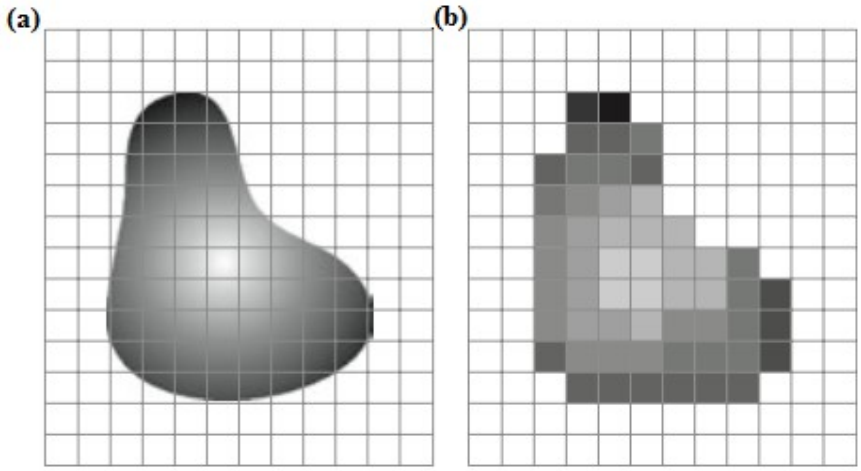
Fonte: Solomon e Breckon (2013).

Na digitalização de uma imagem, a quantização pode definir a qualidade da mesma capturada pelo sensor. A quantização espacial é o mapeamento do sinal analógico da cena em um número discreto de amostragem de pixels, organizados na forma de uma matriz, no qual cada um destes possui uma capacidade finita de representação em formato de dados. O tamanho de dados (em bits) armazenados em cada pixel determina a quantização de cor e também a resolução de bit de uma imagem. A resolução de bit define os valores possíveis de intensidade ou cor que podem estar relacionados a imagem, como exemplo: uma imagem binária de duas cores (preto e branco) possui dois bits, uma imagem na escala de cinza possui 8 bits que equivale a uma combinação de 256 diferentes tons de cinza, enquanto uma imagem colorida de 16 bits é equivalente a 65.536 níveis de cores (GONZALEZ; WOODS, 2008).

As imagens digitais necessitam de métodos eficientes para exibição e armazenamento em formatos padronizados, que consistem em um cabeçalho de arquivo com informações de como são armazenados os dados e os valores numéricos dos pixels da imagem. Entre estes formatos, pode-se destacar: GIF, JPEG, BMP, PNG, TIF, entre outros (SOLOMON; BRECKON, 2013).

A figura 5 demonstra a influência da quantização. A figura 5 (a) apresenta uma imagem contínua projetada no plano da matriz do sensor. A figura 5 (b) apresenta imagem após a digitalização com a amostragem e quantização.

Figura 5 - Imagens representando a influência da quantização.



Fonte: Adaptado de Gonzalez e Woods (2008).

2.3 Visão Computacional e Processamento de Imagens

O processamento de imagens refere-se ao processamento digital dos sinais gerados por um sensor de imagens para extração e exibição de uma imagem, realizado por um computador através da aplicação de métodos de processamento de sinal. O processamento de imagem é utilizado para várias aplicações, tais como: a representação de imagens e melhor nitidez, restauração das imagens por meio de filtragem, correções geométricas, entre outras. Estas aplicações são geralmente a primeira fase para a entrada de imagens em um sistema de visão computacional (SUAREZ; CARROBLES; ENANO; GARCÍA; GRACIA; INCERTIS; TERCERO, 2014).

Segundo Pajankar (2015) a visão computacional é um campo da ciência da computação, matemática, engenharia elétrica e áreas similares, que tem a finalidade de imitar a visão humana na forma de adquirir, processar, analisar e compreender imagens e vídeos de um determinado cenário. Um sistema de visão computacional pode receber diferentes formas de dados, como entradas de imagens, vídeos, sequência de imagens em diferentes formatos e extrair informações úteis para uma tomada decisão. Entre as aplicações mais típicas, pode-se destacar: o reconhecimento e classificação de objetos, a detecção e análise de movimento e a reconstrução de imagens.

Segundo Marques Filho e Viera Neto (1999), a aplicação de um sistema de visão computacional é baseada nas informações do conhecimento de um determinado problema a ser solucionado, sendo especificados hardwares para a solução deste e armazenando estas informações como base de conhecimento no sistema, guiando todas as funcionalidades, algoritmos computacionais e métodos das etapas da estrutura de um sistema de visão computacional. As principais etapas de uma estrutura de um sistema simplificado de visão computacional podem ser classificadas como a aquisição, pré-processamento, segmentação, extração de características, reconhecimento e interpretação. A figura 6 apresenta o diagrama da estrutura de um sistema de visão computacional.

Figura 6 - Diagrama da estrutura de um sistema de visão computacional.

Fonte: Marques Filho e Viera Neto (1999).

2.3.1 Aquisição

A aquisição de imagens é realizada por sensores e digitalizadores de imagens. Entre os sensores de imagens mais utilizados, pode-se destacar: os sensores CCD (*Charge Coupled Device*) e CMOS (*Complementary Metal-Oxide Semiconductor*). Entre os aspectos técnicos relacionados a um projeto de sistema de visão computacional, podem ser citados: a escolha do tipo de sensor de imagem, o conjunto de lentes, as condições de iluminação (estes em geral são predominantes sobre o sensor em si no ambiente utilizado), os requisitos de velocidade, resolução, quantização, tempo de resposta, entre outros. A aquisição de imagens pode ser

realizada de forma manual ou automática através de sensores que detectam o objeto cuja imagem será capturada (MARQUES FILHO; VIERA NETO, 1999).

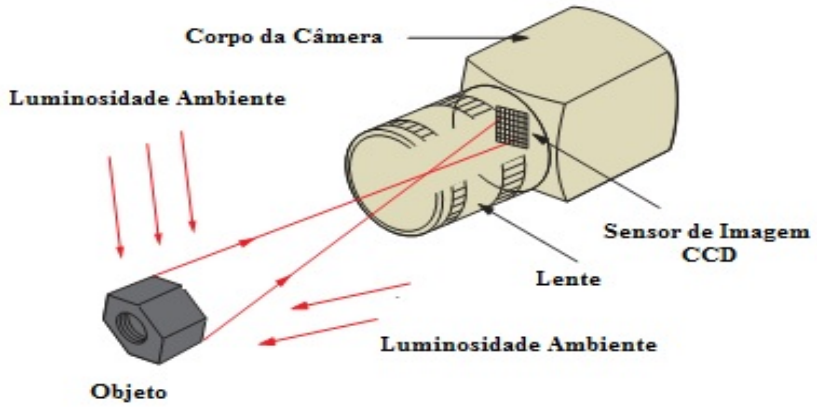
2.3.1.1 Iluminação

Conforme Nakamura (2006), a iluminação da cena é considerada um fator importantíssimo para realizar a aquisição de imagens. Uma boa iluminação da cena é referenciada na temperatura de cor de luz aproximada de 3200 K (Kelvin). Temperatura em que a maioria dos sensores de imagens atualmente fabricados possui seu melhor desempenho. A finalidade da iluminação é a obtenção de um melhor contraste e qualidade da imagem, pois a relação entre sinal por ruído é dependente da quantidade de fótons que chegam ao sensor de imagem, emitidos pela fonte de iluminação. A iluminação também deve atender as exigências da aplicação, em que na maioria dos casos se busca uma iluminação uniforme sobre a cena.

2.3.1.2 Sensor de Imagem CCD

Um dos sensores de imagens mais populares nas câmeras digitais de imagens é o sensor CCD. Este dispositivo consiste em uma matriz de células semicondutoras fotossensíveis com capacitores de metal-óxido semicondutor, que atuam armazenando uma carga elétrica proporcional à energia luminosa incidente da imagem desejada. O sinal elétrico produzido por este dispositivo é condicionado por um circuito condicionador especial, produzindo em sua saída analógica um sinal composto de vídeo (NAKAMURA, 2006). A figura 7 ilustra o funcionamento de uma câmera digital com um sensor de imagem CCD.

Figura 7 - Funcionamento de câmera digital com sensor de imagem CCD.



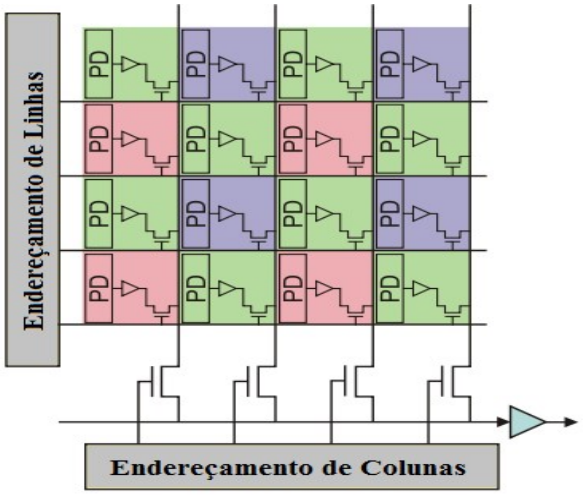
Fonte: Adaptado de Keyence (2016).

Conforme Marques Filho e Viera Neto (1999) uma câmera CCD monocromática consiste basicamente em: um conjunto de lentes para focalizar a imagem sobre a área fotossensível, o sensor CCD e um circuito condicionador para gerar o sinal de saída. Para realizar a aquisição de imagens coloridas, pode se utilizar três sensores (uma para cada componente) junto um conjunto de prismas e filtros de cores para realizar a decomposição da imagem em suas componentes: vermelho, verde e azul, que formam a representação RGB. Cada componente é capturada por um sensor CCD independente e os sinais elétricos gerados por cada sensor são combinados conforme o padrão de varredura de cor utilizado, como exemplo: o NTSC (*National Television Standards Committee*) utilizado nos Estados Unidos e muitos países do continente americano e PAL (*Phase Alternating Line*) largamente utilizados nos países europeus.

2.3.1.3 Sensor de Imagem CMOS

Segundo Nakamura (2006) a tecnologia CMOS é muito utilizada em circuitos integrados utilizados em aplicações digitais e analógicas, empregados nas fabricações de microcontroladores, memórias de estado sólido, circuitos integrados de lógica, sensores de imagens, conversores de sinais, entre outros. Os sensores de imagens CMOS vêm sendo muito empregados em câmeras digitais compactas, devido a seu encapsulamento possuir dimensões menores, permitindo a alimentação com tensões inferiores e com menor consumo de energia, comparados com outros tipos de sensores. A figura 8 ilustra uma matriz de pixels de um sensor de imagem CMOS.

Figura 8 - Matriz de pixels 4 x 4 de um sensor de imagem CMOS.



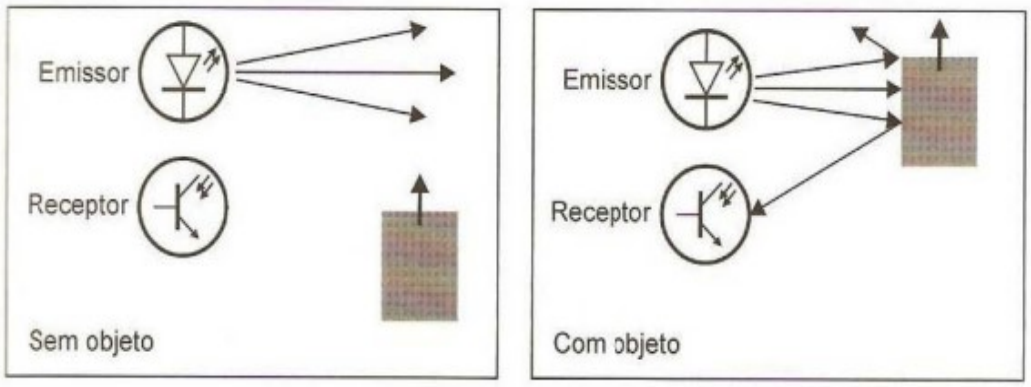
Fonte: Adaptado de Nakamura (2006).

O princípio de funcionamento de um sensor de imagem CMOS é semelhante à de um sensor CCD. Ambos utilizam uma matriz com elementos semicondutores fotossensíveis para obtenção de um sinal composto de vídeo. O sensor de imagem CMOS se difere na construção de sua matriz, onde cada pixel é constituído por um circuito basicamente formado por: um fotodiodo, um transistor de efeito de campo metal-óxido semicondutor e um amplificador de sinal. Os pixels deste sensor são endereçados em formato de linhas e colunas, que geralmente são varridos periodicamente por um conversor analógico-digital (NAKAMURA, 2006).

2.3.1.4 Sensor Óptico Difuso

Os sensores ópticos difusos podem ser utilizados para a detecção de objetos em sistemas de automação. Estes sensores possuem no mesmo dispositivo o emissor e o receptor. No local onde o sensor é instalado um feixe luminoso é enviado pelo emissor fazendo com que uma determinada área fique ativa. Quando ocorre a passagem de um objeto neste local, o feixe luminoso emitido pelo emissor é refletido de forma difusa entre o objeto e o receptor, com isso o receptor é comutado, detectando o objeto na passagem pelo dispositivo. A figura 9 demonstra o funcionamento de um sensor óptico difuso na detecção de um objeto (THOMAZINI; ALBUQUERQUE, 2007).

Figura 9 - Funcionamento de um sensor óptico difuso.



Fonte: Thomazini e Albuquerque (2007).

2.3.2 Pré-Processamento

Após a etapa da aquisição, as imagens extraídas podem conter elementos indesejáveis, como: contrastes e brilhos inadequados, distorções e ruídos de imagens; elementos que prejudicam a qualidade da imagem. O pré-processamento tem a função de melhorar ou corrigir estas imperfeições, ou modificar características e tratamento das imagens para as

etapas seguintes com métodos de manipulação de pixels. Entre estes, pode-se citar: os filtros de imagens, conversão de cores, erosão e dilatação, entre outros (MARQUES FILHO; VIERA NETO, 1999).

2.3.3 Segmentação

A segmentação é uma das etapas mais difíceis de ser executada em um sistema de visão computacional. Nesta etapa é recebida uma imagem de entrada representada na forma de matriz, na qual são utilizados métodos que tem a funcionalidade de localizar e identificar objetos e partes de interesse na imagem, segmentando estes em uma nova imagem. Entre os principais métodos de segmentação, pode-se citar: a limiarização, histogramas de imagens, detecção de linhas, círculos e retângulos, entre outros (GONZALEZ; WOODS, 2008).

2.3.4 Extração de Características

Após a etapa da segmentação, métodos e algoritmos descritores realizam a extração de características das imagens segmentadas. Estes algoritmos podem extrair informações importantes ou de interesse, para a diferenciação de um objeto de outro e o processamento dos dados correspondente das imagens para a etapa de reconhecimento e interpretação. Entre estes, pode-se citar: as extrações de contornos, componentes conectados e cascos convexos, preenchimento de regiões, medições de áreas geométricas, entre outros (MARQUES FILHO; VIERA NETO, 1999).

2.3.5 Reconhecimento e Interpretação

Por último o reconhecimento das imagens é uma atribuição de um rótulo ou mais de objetos, reconhecidos por suas características extraídas junto à base de conhecimento associada às mesmas. A interpretação é baseada na validação do reconhecimento de imagens no sistema de visão computacional, como exemplo: realizar a validação de uma imagem como verdadeira ou falsa em uma aplicação simples de reconhecimento de imagens em um determinado sistema (MARQUES FILHO; VIERA NETO, 1999).

2.4 Métodos de Visão Computacional e o OpenCV

Nas diferentes etapas de uma estrutura de sistema de visão computacional e processamento digital de imagens, várias operações matemáticas e algoritmos computacionais

são executados para realizar as funcionalidades de interesse (JOSHI; ESCRIVÁ; GODOY, 2016). Entre as ferramentas de desenvolvimento para visão computacional, o OpenCV se destaca por ser uma biblioteca de visão computacional de código livre e multi-plataforma, que possui atualmente mais de 2500 algoritmos e funções baseadas em métodos de visão computacional, como: filtragem de imagens, conversão de cores, histogramas de imagens, análises de formas, transformações geométricas e análises estruturais, segmentação de objetos, calibração de câmera, funções de aprendizado de máquina, entre outros (KAEHLER, BRADSKI, 2006).

O OpenCV originalmente foi desenvolvida pela Intel e atualmente é liberada sob a licença da BSD (*Berkeley Software Distribution*), sendo totalmente livre para fins acadêmicos e profissionais. Possui interfaces com várias linguagens de programação como: C/C++, Python, Java e suporte com os sistemas operacionais: Windows, Linux, Mac OS, IOS e Android. A biblioteca é escrita em linguagem de programação C/C++ e otimizada para aplicações em tempo real, contando com os recursos de multiprocessamento, módulos de entradas e saídas de imagens e vídeos, e até disponibliza a criação de interface gráfica aos usuários. Podendo ser aplicada nas mais diversas áreas, como: medicina, segurança, robótica, inspeção de produtos, reconhecimento facial, gravação de vídeos, entre outras (KAEHLER, BRADSKI, 2006).

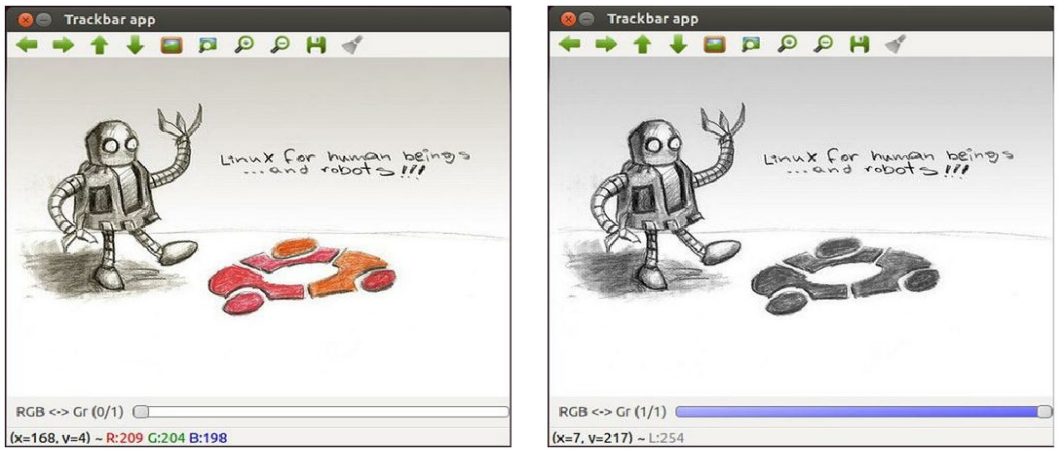
Os principais métodos de visão computacional utilizados para o desenvolvimento deste trabalho estão descritos a seguir.

2.4.1 Conversão de Cores

Uma imagem com representação espacial é basicamente uma matriz bidimensional formada de pixels, no qual cada pixel possui suas propriedades. A conversão de cores é um método que pode utilizar operações matemáticas simples, como: a adição, subtração e multiplicação de valores constantes ou escalares. Atribuídos com os valores de intensidade dos pixels no caso de uma imagem simples como em escalas de cinza, ou com os valores das componentes de cor RGB dos pixels em uma imagem colorida. Alterando os seus valores originais e convertendo os mesmos em outras cores na imagem (SZELISKI, 2010).

A figura 10 demonstra uma aplicação de conversão de cores com o método de visão computacional.

Figura 10 - Aplicação do método de conversão de cores.



Fonte: Brahmbhatt (2013).

O método de conversão de cores pode ser encontrado na biblioteca OpenCV com a função *cvtColor()*. Esta função converte uma imagem de entrada com determinado espaço de cor, em outra imagem com diferente espaço de cor em sua saída, de acordo com o código do tipo de conversão escolhido.

Listagem 1 - Função *cvtColor()*.

```
cvtColor(InputArray src, OutputArray dst, int code, int dstCn=0);
```

Fonte: Adaptado de OpenCV Manual (2016).

Onde:

src - Matriz de imagem em escalas de cinza ou colorida associada a entrada.

dst - Matriz de imagem associada à saída ou destino para conversão da imagem.

code - Código do tipo de conversão de espaço de cor (opcional).

dstCn - Valor inteiro do numero de canais na imagem de destino (opcional).

2.4.2 Filtragem de Imagens

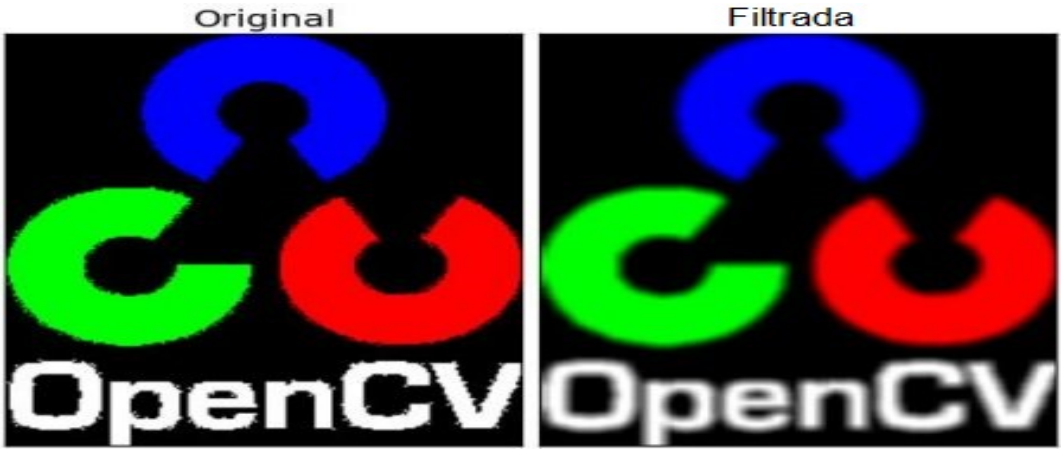
Segundo Davies (2012), a filtragem de imagens é utilizada para eliminar ruídos e elementos indesejáveis de uma imagem, melhorando a apresentação da mesma. Este método geralmente é aplicado no pré-processamento de imagens, utilizando técnicas e operações matemáticas para a remoção destes elementos indesejáveis da cena e iluminação do local.

Semelhante aos sinais elétricos, por exemplo, um ruído de um sinal analógico, pode ser removido por meio de filtros passa-baixa ou outras operações que operam no domínio da freqüência. No processamento de imagens pode ser aplicado também, porém em sinais digitais, e no domínio da frequência com tempo discreto. Entre os métodos de filtragem de imagens, pode-se destacar: Gaussiano, Gaussiano-médio, Passa-baixa, Passa-alta, Suavização

Linear, Transformada de Fourier, Transformada de Laplace, entre outros (GONZALEZ; WOODS, 2008).

A figura 11 demonstra uma aplicação com o método de filtragem de imagem através da suavização.

Figura 11 - Aplicação da filtragem de imagem com a suavização.



Fonte: Adaptado de Opencv Tutorials (2016).

Na biblioteca OpenCV podem ser encontradas várias funções de métodos de filtragem de imagens. Entre estas, a função *blur()* tem a funcionalidade de realizar a filtragem de uma imagem de entrada, por meio de suavização usando um tamanho de núcleo especificado, e alocando em sua saída a imagem filtrada.

Listagem 2 - Função *blur()*.

```
blur(InputArray src, OutputArray dst, Size ksize, Point
  anchor=Point(-1,-1), int borderType=BORDER_DEFAULT);
```

Fonte: Adaptado de OpenCV Manual (2016).

Onde:

src - Matriz de imagem colorida ou em escalas de cinza associada à entrada.

dst - Matriz de imagem de mesmo formato associada à saída ou destino.

ksize - Valor decimal referente ao tamanho do núcleo (opcional).

anchor - Pontos para as posições dentro dos núcleos (opcional).

borderType - Valor inteiro referente ao método de extrapolação de pixels (opcional).

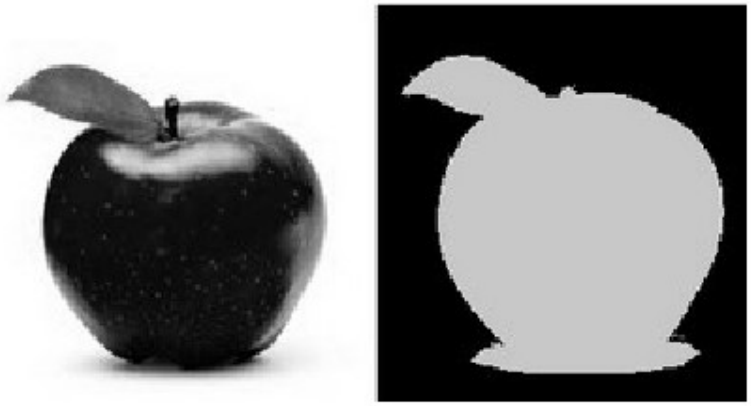
2.4.3 Limiarização

A limiarização é um dos métodos de segmentação mais utilizados no processamento digital de imagens. Este método geralmente é aplicado para facilitar a separação do objeto de interesse do fundo de imagem. Pode receber de entrada uma imagem colorida ou em escalas

de cinza, na qual uma função realiza a comparação de certos valores de intensidade de pixels com um limiar determinado, convertendo estes em preto e branco e resultando em uma nova imagem binária (KAEHLER, BRADSKI, 2006).

Davies (2012) comenta que se a iluminação do fundo de imagem for bastante uniforme e o objeto de escolha estiver bem nítido sobre este fundo, a segmentação da imagem pode ser simplesmente realizada com a limiarização, realizando um recorte do fundo de imagem e segmentando o objeto de interesse. A figura 12 ilustra uma aplicação com o método de limiarização de uma imagem em escalas de cinza.

Figura 12 - Aplicação do método de limiarização de uma imagem.



Fonte: OpenCV Tutorials (2016).

Na biblioteca do OpenCV podem ser encontradas funções para a limiarização de imagens. Entres os algoritmos mais utilizados podem-se destacar as funções: *threshold()* e *inRange()*. A função *threshold()* é aplicada para operações básicas de limiarização, este algoritmo recebe de entrada uma matriz de uma imagem em escalas de cinza, e compara a intensidade de cada pixel com um valor limiar, e transformando em sua saída uma matriz de imagem binária.

Listagem 3 - Função *threshold()*.

```
threshold(InputArray src, OutputArray dst, double thresh, double
                        maxval, int type);
```

Fonte: Adaptado de OpenCV Manual (2016).

Onde:

src - Matriz de imagem em escalas de cinza associada à entrada.

dst - Matriz de imagem associada à saída ou destino para a imagem binária.

thresh - Valor decimal do limiar desejado (8 bits ou 0 a 255).

maxval - Valor decimal associado ao fundo de escala (8 bits ou 0 a 255).

type - Valor inteiro associado ao tipo de limiarização (opcional).

A função *inRange()* é aplicada para a limiarização e segmentação de cores em uma imagem, diferente do algoritmo anterior, este recebe uma matriz colorida e compara a intensidade dos elementos RGB, ou seja, as cores: vermelho, verde e azul da imagem. Possibilitando encontrar qualquer tonalidade de cor para posteriormente segmentar apenas a coloração de interesse, e por fim converter esta segmentação em uma matriz de imagem binária em sua saída ou destino.

Listagem 4 - Função *inRange()*.

```
inRange(InputArray src, InputArray lowerb, InputArray upperb,
                        OutputArray dst);
```

Fonte: Adaptado de OpenCV Manual (2016).

Onde:

src - Matriz de imagem colorida associada à entrada.

lowerb - Vetor escalar de cores RGB com valores do limite inferior.

upperb - Vetor escalar de cores RGB com valores do limite superior.

dst - Matriz de imagem associada à saída ou destino para imagem binária.

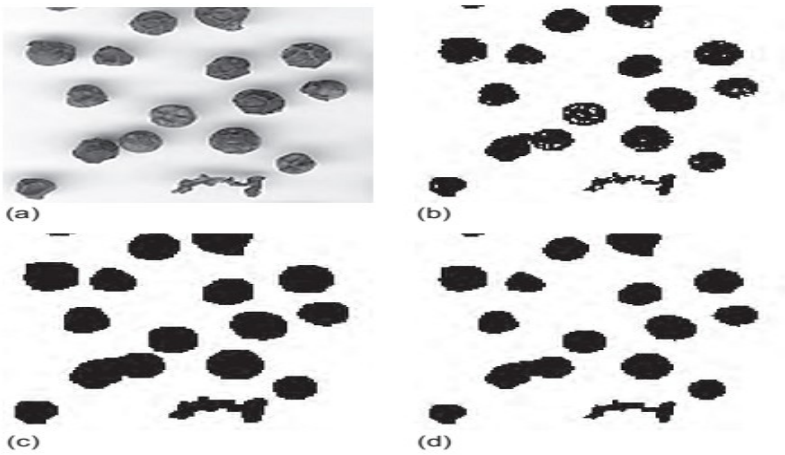
2.4.4 Erosão e Dilatação

A erosão e a dilatação são métodos morfológicos de processamento de imagens, que recebem uma imagem de entrada, realizando cálculos e comparações dos valores de intensidade dos pixels e suas vizinhanças, analisando a sua estrutura geométrica e resultando em uma nova imagem. A dilatação adiciona pixels entre o plano de fundo de uma imagem e as fronteiras dos objetos, enquanto a erosão remove pixels. Na dilatação, o valor da saída dos pixels é o valor máximo de todos os pixels na vizinhança, o que faz expandir os objetos. Já na erosão, o valor dos pixels de saída é o valor mínimo de todos da vizinhança, encolhendo os objetos na imagem (SUAREZ; CARROBLES; ENANO; GARCÍA; GRACIA; INCERTIS; TERCERO, 2014).

Segundo Davies (2012) a combinação destes dois métodos representa as operações de abertura e fechamento de imagens. A abertura é definida como a erosão, seguida de dilatação; enquanto o fechamento é o inverso, dilatação seguida de erosão. A abertura é utilizada para remover pequenos objetos de uma imagem, preservando as maiores e suavizando os contornos. O fechamento é utilizado para remover pequenos orifícios da imagem com o preenchimento, preservando também os objetos maiores de modo semelhante ao de abertura. A figura 13 ilustra a aplicação dos métodos de erosão e dilatação em uma imagem em escalas

de cinza. A figura 13 (a) apresenta a imagem original. A figura 13 (b) a imagem com limiarização. A figura 13 (c) a imagem com dilatação. A figura 13 (d) a imagem com erosão.

Figura 13 - Aplicações dos métodos de erosão e dilatação em uma imagem.



Fonte: Davies (2012).

Os métodos de erosão e dilatação podem ser encontrados na biblioteca do OpenCV com as funções: *erode() e dilate()*. Ambos os algoritmos recebem as mesmas variáveis em suas funções, no qual é atribuída uma matriz de imagem de entrada e um elemento de estruturação especificado, que determina a forma mínima de uma vizinhança de pixel para a operação de erosão ou dilatação sobre a imagem de origem, e salvando em sua saída uma matriz com a imagem modificada.

Listagem 5 - Função *erode().*

```
erode(InputArray src, OutputArray dst, InputArray kernel, Point
          anchor=Point(-1,-1), int iterations=1, int
           borderType=BORDER_CONSTANT, const Scalar&
         borderValue=morphologyDefaultBorderValue());
```

Fonte: Adaptado de OpenCV Manual (2016).

Onde:

src - Matriz de imagem colorida ou em escalas de cinza associada à entrada.

dst - Matriz de imagem de mesmo formato associada à saída ou destino.

kernel - Matriz para o tipo de formato dos elementos estruturais (opcional).

anchor - Pontos para as posições dentro dos elementos (opcional).

iterations - Valor inteiro referente ao número de vezes da aplicação (opcional).

borderType - Valor inteiro referente ao método de extrapolação de pixels (opcional).

borderValue - Vetor escalar associado ao valor de uma borda constante (opcional).

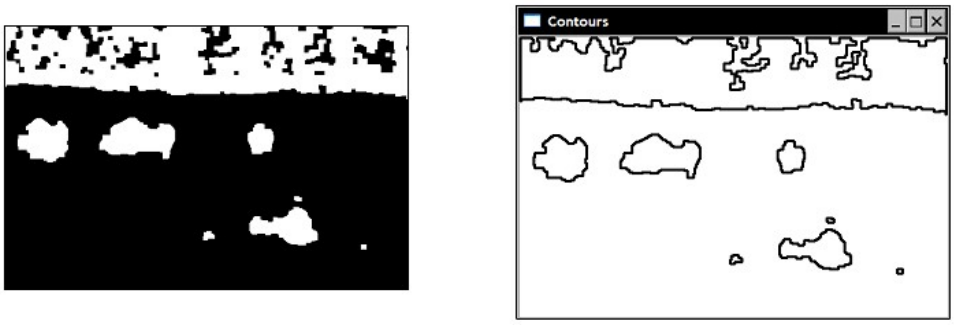
2.4.5 Extração de Contornos

Segundo Laganière (2011) os métodos de extração de contornos são muito utilizados para a extração de características e o reconhecimento de objetos em imagens. A extração de contornos é realizada através da detecção dos limites distintos dos gradientes de uma imagem, ou seja, detecta o limite entre a diferença de uma intensidade de cor e outra. A segmentação da imagem com métodos como a limiarização são geralmente aplicados antes da extração de contornos para se ter uma precisão melhor nos resultados, prevenindo assim a detecção de contornos falsos ou indesejáveis.

Existem vários métodos que utilizam técnicas matemáticas para encontrar os contornos em uma imagem, entre estes, pode-se destacar: a Transformada de Hough, Laplace e Sobel, que podem realizar a detecção de linhas, círculos, elipses e bordas. O método de Canny é um algoritmo de extração de contornos mais robusto e eficiente entre os outros métodos. Este algoritmo utiliza vários estágios de limiarização, com máscaras e filtros de imagens para a eliminação de ruídos e falsos contornos, sendo um dos métodos mais utilizados na extração de características de uma imagem (LAGANIÈRE, 2011).

A figura 14 ilustra a extração de contornos de uma imagem binarizada (em cores preto e branco).

Figura 14 - Extração de contornos de uma imagem binarizada.



Fonte: Laganière (2011).

O método de Canny é encontrado na biblioteca do OpenCV com a função *canny()*. Esta função encontra as bordas de uma matriz de imagem de entrada e as marca em um mapa de saída. O algoritmo de Canny realiza dois estágios de limiarização, no qual o menor valor encontrado entre a primeira e a segunda limiarização é usado para ligação da borda, e o maior valor é usado para encontrar os segmentos iniciais das bordas mais fortes.

Listagem 6 - Função *canny()*.

```
Canny(InputArray image, OutputArray edges, double threshold1, double
      threshold2, int apertureSize=3, bool L2gradient=false );
```

Fonte: Adaptado de OpenCV Manual (2016).

Onde:

image - Matriz de imagem em escalas de cinza associada à entrada.

edges - Matriz de imagem associada à saída ou destino para o mapa de bordas.

threshold1 - Valor decimal do limiar da primeira limiarização (8 bits ou 0 a 255).

threshold2 - Valor decimal do limiar da segunda limiarização (8 bits ou 0 a 255).

apertureSize - Valor inteiro para o tamanho da abertura do filtro de Sobel.

L2gradient - Variável booleana para sinalizar a precisão de cálculos de gradientes da imagem (opcional).

Para encontrar e recuperar contornos de uma imagem binária a biblioteca OpenCV conta com a função *findContours()*. Uma função e ferramenta útil para análises de formas, detecção e reconhecimento de objetos. Este algoritmo recebe uma matriz de imagem binária ou em escalas de cinza, detectando todos os contornos da mesma e armazenando os pontos de cada contorno encontrado em um vetor de saída.

Listagem 7 - Função *findContours()*.

```
findContours(InputOutputArray image, OutputArrayOfArrays contours,
OutputArray hierarchy, int mode, int method, Point offset=Point());
```

Fonte: Adaptado de OpenCV Manual (2016).

Onde:

image - Matriz de imagem em escalas de cinza associada à entrada.

contours - Vetor de saída contendo os pontos de todos os contornos armazenados.

hierarchy - Vetor de saída contendo informações sobre a topologia da imagem (opcional).

mode - Valor inteiro associado ao modo de recuperação dos contornos (opcional).

offset - Pontos para o deslocamento original dos contornos (opcional).

3 ESPECIFICAÇÃO DO PROJETO

Este capítulo descreve sobre a apresentação do projeto e as especificações técnicas dos hardwares e softwares utilizados para o desenvolvimento do mesmo. Primeiramente será apresentada a proposta do sistema de automação com visão computacional para inspeção do nível do enchimento de embalagens, e em seguida são apresentados os hardwares utilizados para a construção do protótipo e os softwares utilizados para o desenvolvimento do sistema.

3.1 Apresentação Geral do Projeto

Os problemas relacionados com um enchimento incorreto, principalmente de produtos dos setores das indústrias de alimentos e bebidas, como: reclamações e rejeições de produtos pela divergência do volume fornecido e perdas financeiras. Geralmente decorrentes de uma inspeção operacional ineficiente nos processos de envase, podem ser solucionados ou melhorados com a implementação de um sistema de automação com visão computacional.

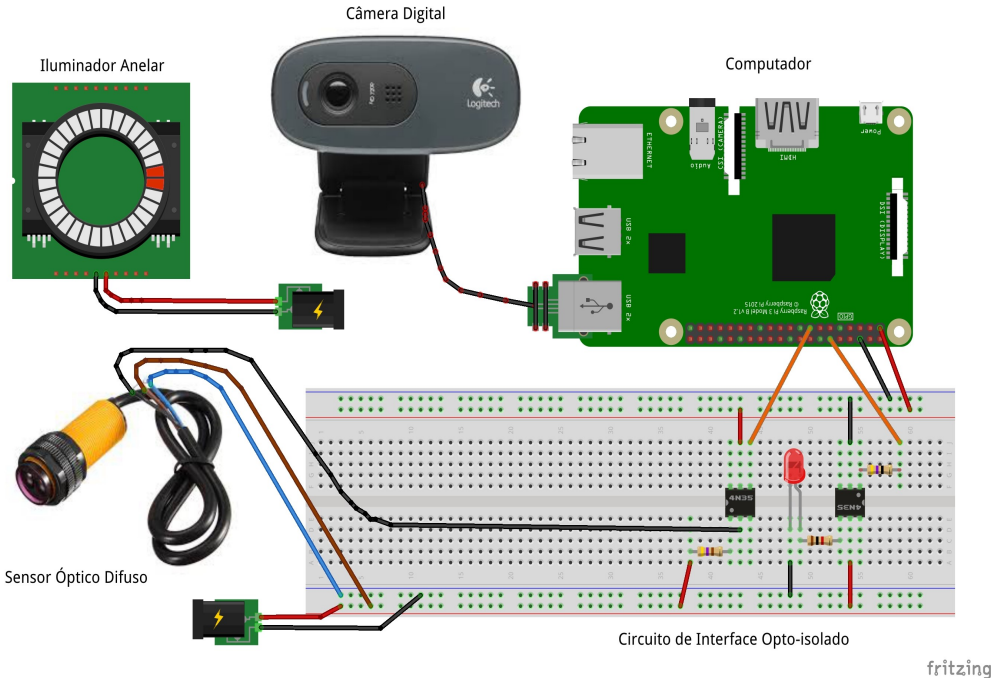
Um sistema tradicional de visão computacional pode ser constituído, por: um sensor óptico difuso para detecção das embalagens; uma câmera digital para a aquisição de imagens junto a um dispositivo de iluminação para controlar a luminosidade do cenário; e um computador com softwares para realizar o processamento de imagens e hardwares para interface de I/Os (*Input/Output*) e comunicação.

Nestes moldes, o projeto consiste no desenvolvimento de um protótipo de baixo custo de um sistema de automação para inspeção do nível do enchimento de embalagens transparente, para aplicação em processos de envase de produtos líquidos e sólidos transportados por esteira motorizada, testado e simulado em laboratório de automação. O sistema basicamente segue um funcionamento e construção tradicional, que é iniciado com a detecção da embalagem em movimento com um sensor óptico difuso, utilizado para detectar a aproximação da mesma do protótipo e posteriormente realizar a captura da imagem por meio de uma câmera digital. Um iluminador com ajuste de intensidade luminosa, acoplada junto à câmera digital pode ser utilizado para controle da iluminação do cenário e melhoramento na aquisição das imagens, com a eliminação de interferências de iluminações externas. Ambos os dispositivos conectados em um computador.

A imagem capturada pela câmera digital é processada pelo computador com o software do sistema de visão computacional embarcado, que por sua vez realizará a segmentação da imagem, a extração de características, o reconhecimento e a validação do nível do enchimento da embalagem, por meio de algoritmos baseados em métodos de visão

computacional. Extraindo dados úteis das imagens, como: a altura, a largura e ou a área do enchimento da embalagem segmentada, e comparações com um nível de enchimento padrão, calibrado e configurado inicialmente pelo o usuário com o software do sistema. Desta forma realiza-se a inspeção do nível do enchimento das embalagens, monitorando se as mesmas tiveram o enchimento correto ou não, e podendo acionar dispositivos de alarmes com a atuação de uma saída digital quando um evento de não conformidade ocorrer. A figura 15 ilustra o desenho dos componentes utilizados para o desenvolvimento do protótipo.

Figura 15 - Desenho ilustrativo dos componentes do projeto.



Fonte: Autor (2017).

Para o desenvolvimento do sistema de automação com visão computacional para inspeção de nível de enchimento de embalagens, o protótipo foi inicialmente construído e testado em laboratório de automação, com os seguintes componentes:

- Um sensor óptico difuso Omron E3F-DS30C1 para realizar a detecção das embalagens para a captura de imagem;
- Uma câmera digital Logitech C270 que é composta por um sensor e um hardware de processamento de imagens, para realizar a aquisição e o pré-processamento de imagens;
- Um iluminador anelar de *leds* com ajuste de intensidade marca Lucky Zoom, para controle da iluminação do cenário para a aquisição de imagens;
- Uma placa de desenvolvimento Raspberry Pi 3, que é um minicomputador com a funcionalidade de executar o software do sistema de visão embarcado, realizando o

processamento de imagens, o armazenamento de dados, e permitindo a exibição com a saída HDMI (*High-Definition Multimedia Interface*) por um monitor e os meios de copias e rede com as interfaces USB (*Universal Serial Bus*) e Ethernet, disponíveis na placa;

- Uma placa com circuito eletrônico opto-isolado para interface de I/O's para ligação do sensor óptico na entrada e os dispositivos a serem utilizados para alarme na saída.

Os hardwares utilizados para o desenvolvimento do projeto serão especificados detalhadamente a seguir. Basicamente a montagem do protótipo é centralizada na placa Raspberry, que possui quatro portas USB e 40 pinos de GPIO (*General Purpose Input Output*), permitindo a conexão com a câmera digital Logitech C270 em uma destas portas, e a conexão dos pinos de entrada e saída com a interface com circuito eletrônico opto-isolado, para as ligações do sensor óptico e a saída digital utilizada para alarme.

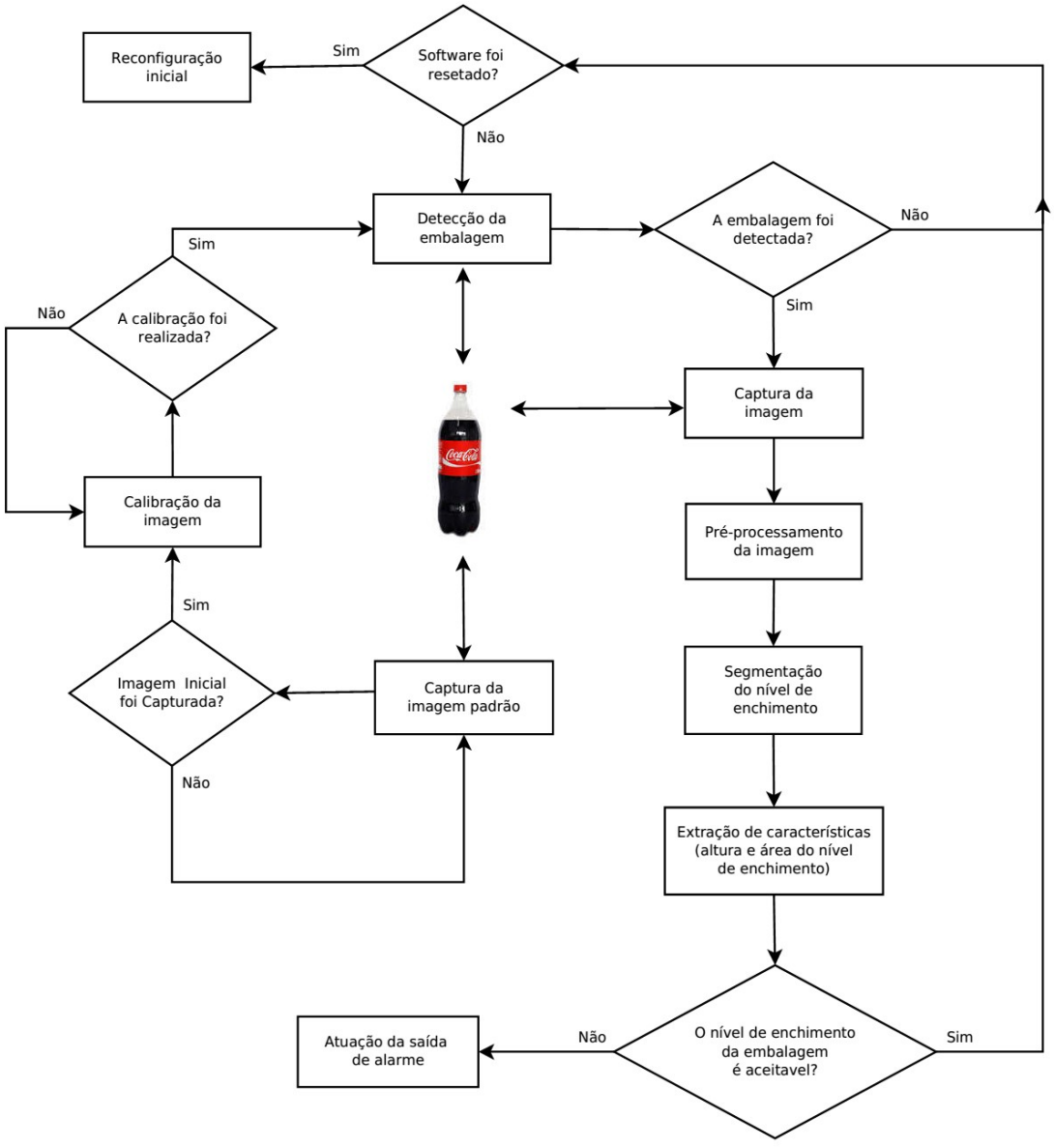
A Raspberry Pi 3 suporta o sistema operacional Raspbian, que foi utilizado junto com a IDE (*Integrated Development Environment*) Qt Creator para o desenvolvimento do software do sistema de visão, que foi escrito em linguagem de programação C/C++, utilizando a biblioteca de visão computacional do OpenCV e a biblioteca de aplicação do Qt. O software do sistema foi basicamente desenvolvido com uma tela de interface principal, que permite o usuário fazer interações com botões, campos de edições, painel para visualizar as imagens, entre outras ferramentas que possam facilitar as configurações e utilidades do sistema.

O sistema segue um fluxograma lógico de operações que pode ser visualizado na figura 16, iniciado com a execução do software. No primeiro momento o usuário deverá capturar a imagem da embalagem transparente com o nível de enchimento padrão, pressionando o botão Capturar. Posteriormente já com a imagem capturada e visível no painel, o usuário deverá realizar a calibração da imagem padrão, selecionando primeiramente o tipo de segmentação desejado: colorido ou em escalas de cinza, e pressionar o botão Calibrar. Duas telas serão abertas, no qual uma terá barras de rolagem para ajustar as ferramentas de limiarização e a outra para visualizar e encontrar as informações dos valores em pixels de altura e área do nível de enchimento da embalagem.

Após os ajustes e com a coleta das informações dos valores de altura e área, a etapa de calibração pode ser finalizada pressionando a tecla ESC. Finalizado a calibração, o usuário poderá inicializar a inspeção das embalagens, selecionando a variável a ser validada (altura ou área), e informando nos seus respectivos campos de edição, o seu valor e a tolerância em porcentagem e depois pressionar o botão Iniciar para começar a operação. Todas as imagens inspecionadas momentaneamente ficarão disponíveis para visualização no painel e totalizado

o numero das embalagens validas e invalidas e o total inspecionado. Pressionando o botão Resetar, a inspeção de embalagens é interrompida e todas as variáveis do sistema são resetadas com seus valores iniciais. O desenvolvimento do software do sistema será descrito mais detalhadamente no próximo capítulo.

Figura 16 - Fluxograma lógico de operações do sistema de visão.



Fonte: Autor (2017).

3.2 Hardwares

Nesta seção serão apresentados e especificados os principais hardwares utilizados para a construção do protótipo.

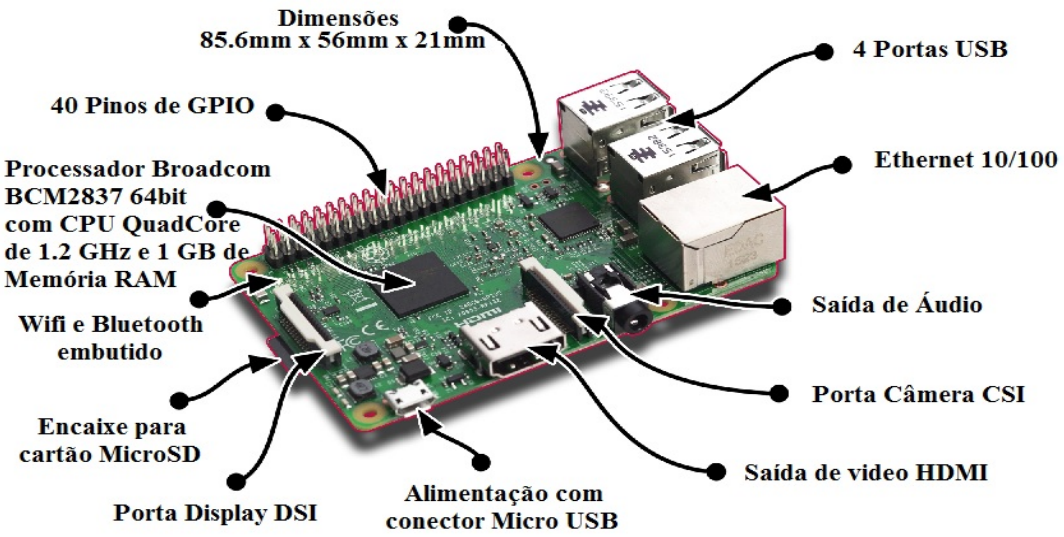
3.2.1 Raspberry Pi 3

A Raspberry Pi 3 é uma placa de desenvolvimento de baixo custo, que tem a funcionalidade de um minicomputador, criada e mantida por uma comunidade de desenvolvedores e entusiastas. No site da comunidade *www.raspberry.org* pode ser encontrada uma variada quantidade de materiais de apoio, desde tutoriais, projetos e até exemplos práticos. Esta placa é a terceira geração da Raspberry Pi, que foi lançada em fevereiro de 2016 para substituir a placa Raspberry Pi 2 modelo B. Entres as características que mais se destacam neste hardware, pode-se citar:

- CPU ARMv8 quad-core de 64 bits de 1.2 GHz;
- 1 GB de memória RAM DDR3;
- Wifi embutido;
- Bluetooth 4.1;
- 4 portas USB;
- 40 pinos GPIO;
- Saída de vídeo HDMI;
- Conexão de Rede porta Ethernet;
- Tomada de áudio combinada de 3,5 mm e vídeo composto;
- Interface para câmera;
- Ranhura para cartão Micro SD;
- Núcleo acelerador gráfico 3D VideoCore IV;

A Raspberry Pi 3 suporta vários sistemas operacionais, entre alguns: O Raspbian (recomendado), Android, Ubuntu, Debian, Angstrom, Windows IoT, entre outros. Consiste numa placa compacta com um hardware robusto e com processamento moderado, podendo ser aplicado em uma infinidade de aplicações, inclusive para aplicações com processamento de imagens e visão computacional. A figura 17 ilustra uma imagem da placa Raspberry Pi 3 com seus dados técnicos (RASPBERRY, 2016).

Figura 17 - Placa Raspberry Pi 3.



Fonte: Adaptado de Raspberry (2016).

3.2.2 Câmera Digital Logitech C270

A Logitech C270 é uma câmera digital com conexão USB, para captura de vídeos e imagens coloridas com representação RGB. Possui em seu hardware um sensor de imagem do tipo CMOS, com resolução nativa 1280 x 960 pixels ou 1.2 MP (megapixels), podendo realizar a captura de vídeos com até 30 fps com resolução de até 640 x 480 pixels e capturando imagens com até 3 MP de quantização espacial. A câmera é compatível com os sistemas operacionais Windows e Linux e necessita ter os seguintes requisitos básicos para o sistema: processador de 1 GHZ, 512 MB de memória RAM, 200 MB de espaço em disco rígido e porta USB 1.1. A figura 18 ilustra a imagem da câmera digital Logitech C270 (LOGITECH, 2016).

Figura 18 - Câmera digital Logitech C270.

Fonte: Logitech (2016).

3.2.3 Sensor Óptico Omron E3F-DS30C1

O sensor E3F–DS30C1 é um sensor óptico difuso da família E3F da Omron com detecção de luz infravermelha. Este sensor possui formato cilíndrico de 18 mm de diâmetro, construído em plástico e com grau de proteção IP-67, ou seja, é resistente a pulverizações com água. Pode ser alimentado com tensões de 10 a 30 VCC, realizando a detecção de objetos de 0,1 m até 0,3 m de distância, possuindo alta imunidade contra ruídos eletromagnéticos e a luz ambiente. Pode-se destacar ainda entre as especificações deste dispositivo: o tempo de resposta máximo de 2,5 ms, a saída de controle a transistor NPN normalmente aberto com até 100 mA de corrente e podendo operar em temperaturas que podem variar de -25 °C a 55 °C. Características que tornam este sensor uma boa opção para aplicações industriais. A figura 19 apresenta imagem ilustrativa do sensor óptico difuso Omron E3F-DS30C1 (OMRON E3F, 2017).

Figura 19 - Sensor óptico difuso Omron E3F-DS30C1.

Fonte: Omron E3F (2016).

3.2.4 Iluminador Anelar Lucky Zoom

O iluminador Anelar da marca Lucky Zoom é uma sistema de iluminação ajustável para aplicação em microscópios. O iluminador tem formato anelar que possui 60 *leds* com luz branca de alto brilho, de 3500 mcd (milicandela) e temperatura de cor de 5500 K, que pode ser ajustável com um dispositivo de ajuste de intensidade de brilho incluído junto. Pode-se destacar ainda entre as especificações técnicas do equipamento: Alimentação da fonte 100 - 240 VCA em 50 – 60 Hz, consumo de 4,5 W, diâmetro interno do iluminador de 60 mm e externo de 94 mm e a melhor distancia de trabalho entre 50 a 150 mm do alvo. A figura 20 apresenta imagem ilustrativa do iluminador anelar Lucky Zoom com o controle de intensidade luminosa (LUCKY ZOOM, 2017).

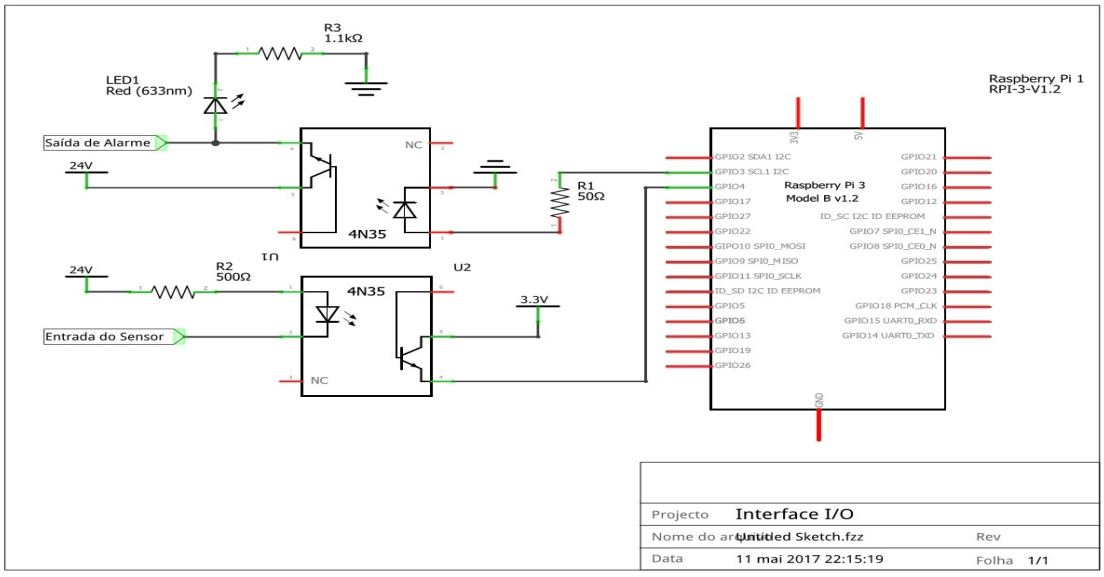
Figura 20 - Iluminador anelar Lucky Zoom.

Fonte: Lucky Zoom (2017).

3.2.5 Circuito de Interface Opto-isolado

Os pinos de I/O da placa Raspberry trabalham com níveis de tensão de até 3,3 VCC, o que impossibilita a ligação de outros dispositivos que não trabalha com essa faixa de tensão. Nas indústrias geralmente utiliza-se 24 VCC para alimentação de sensores, controladores e dispositivos em geral. Para solucionar este problema, foi necessário dimensionar um circuito de interface opto-isolado para fazer a ligação entre os pinos de I/O da placa e o sensor óptico difuso e a saída de alarme, ambos dimensionados para 24 VCC. Na saída de alarme foi colocado um led entre os conectores da saída para visualização local. A figura 21 apresenta o desenho do esquema elétrico do circuito de interface opto-isolado.

Figura 21 - Esquema elétrico do circuito de interface opto-isolado.



Fonte: Autor (2017).

Para a montagem do circuito foi utilizado dois opto acopladores 4N35, um resistor de 50 Ω, um resistor de 1100 Ω, um resistor de 500 Ω, um led de cor vermelha e uma placa padrão perfurada. Para dimensionar os componentes eletrônicos, foram extraídos os dados técnicos de seus respectivos *datasheets* e utilizada a lei de ohms com a aplicação da equação 1, apresentada abaixo.

$$R = \frac{Vin - Vf}{If} \tag{1}$$

Onde:

R é o resistor dimensionado para o componente;

Vin é a tensão de alimentação;

Vf é a tensão nominal do componente;

If é a corrente nominal do componente;

$$R(Led) - \frac{24V - 2V}{20mA} - 1100\,\Omega$$

$$R(4N25) = \frac{24V - 1.3V}{50mA} = 454\,\Omega\ ou\ 500\,\Omega\ (comercial)$$

$$R(4N25) = \frac{3.3V - 1.3V}{50mA} = 40\,\Omega\ ou\ 50\,\Omega\ (comercial)$$

3.3 Softwares

Nesta seção serão apresentados os principais softwares e bibliotecas utilizados para o desenvolvimento do sistema, com exceção da biblioteca de visão computacional do Opencv, apresentado anteriomente.

3.3.1 Qt e o Qt Creator

O Qt é uma biblioteca multiplataforma para o desenvolvimento de interfaces gráficas em C++, criado pela empresa norueguesa Trolltech. Atualmente o Qt é mantido pelo Qt Project, uma iniciativa de software livre envolvendo desenvolvedores individuais e provenientes de empresas como Nokia, Digia, entre outras. Com ele é possível desenvolver aplicativos para diversas plataformas de desktop, dispositivos móveis e embarcados, sem que seja necessário alterar o código fonte e também ser usado por vários tipos de linguagens de programação através de bibliotecas de ligação (QT, 2017).

O Qt Creator é uma IDE multiplataforma de código livre que atualmente esta na sua versão 5.7. Esta IDE traz consigo a biblioteca Qt para o desenvolvimento de aplicações multi-plataforma de maneira fácil e rápida. Com ele é possivel criar interfaces gráficas com várias ferramentas de design, disponibilizadas junto com a integração dos seus principais modulos: Qt Widgets, Qt Gui, Qt Designer, Qt Linguist, Qt Network, entre outros (QT, 2017).

3.3.2 WiringPi

O WiringPi é uma biblioteca de codigo livre, escrita em linguagem de programação C, aplicada para facilitar o acesso aos pinos de GPIO da familia de placas de desenvolvimento Raspberry. Esta biblioteca permite que os usuarios possam fazer operações de leitura e escrita nos pinos de suas placas de maneira simples e intuitiva, que faz familiarizar e lembrar a programação e utilização do Arduino (WIRINGPI, 2017).

4 DESENVOLVIMENTO DO SISTEMA

Neste capítulo será descrito o desenvolvimento do software do sistema de visão embarcado, escrito em linguagem de programação C/C++, com a utilização das ferramentas da IDE Qt Creator e as bibliotecas OpenCV, Qt e WiringPI.

4.1 O Desenvolvimento do Software do Sistema e a Estruturação

A instalação da biblioteca OpenCV para as distribuições Linux, pode ser encontrada no *site opencv.org* com o passo a passo de como realizar a mesma detalhadamente. Nesta mesma página também é disponível uma documentação completa, com exemplos práticos, explicações das funções da biblioteca e um manual do OpenCV, que foram utilizados como materiais de apoio neste trabalho. O software Qt Creator foi instalado pelos repositórios do sistema operacional, e sua documentação completa, tutoriais e até exemplos podem ser encontrados no *site doc.qt.io/qt-5/qtexamplesandtutorials.html.* A biblioteca WiringPi pode ser encontrada no *site wiringpi.com,* nesta pagina também está disponível instruções de como realizar a instalação da mesma, documentações e exemplos de suas funções.

A programação do software do sistema foi iniciada após serem instalados todos os softwares e bibliotecas especificados para o projeto. Na inicialização foi utilizado um computador pessoal com arquitetura x64 para o desenvolvimento do software, no qual foram realizados vários experimentos práticos com as funções das bibliotecas utilizadas, com a escrita de códigos fonte executados separadamente. A fim de testar e avaliar os algoritmos das bibliotecas para cada etapa de uma estrutura de sistema de visão computacional, e posteriormente a adição e inclusão das funções e códigos fontes que tiveram resultados satisfatórios, para o código fonte do software do sistema embarcado no minicomputador Raspberry Pi 3.

O software do sistema foi basicamente estruturado em seis arquivos, descritos a seguir:

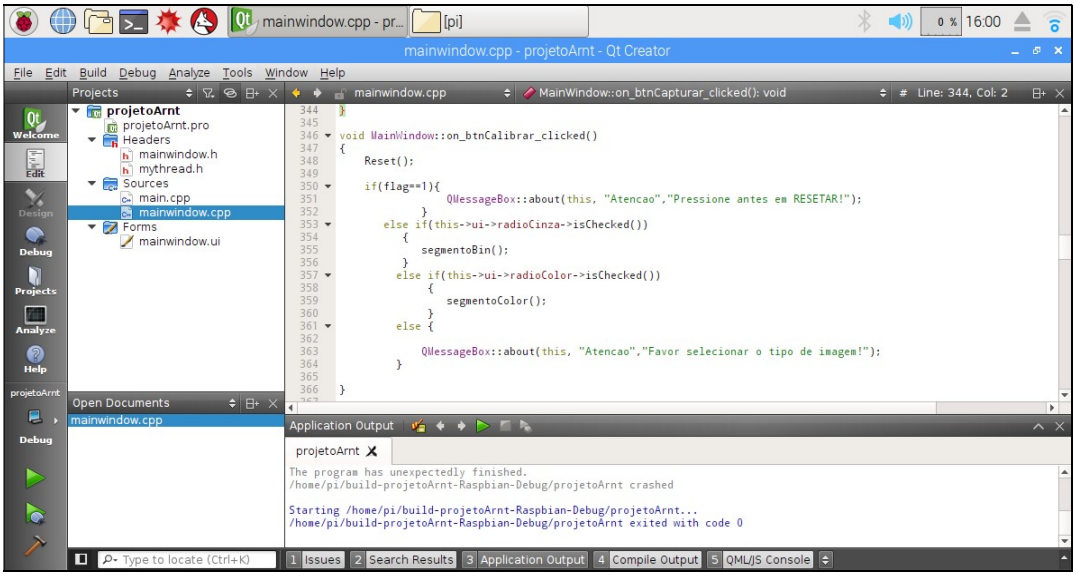
- ***main.cpp*** – É o arquivo que possui a função global, que realiza a execução do programa, inicializando a tela de interface principal e todo o fluxo de controle. Neste arquivo são incluídas todas as bibliotecas principais de interface ou cabeçalho, como os arquivos: *mainwindow.h* e *mythread.h*.
- ***mainwindow.cpp*** – É um arquivo de protótipos de funções, neste arquivo estão todas as funções principais do programa, que permitem a funcionalidade do sistema. No

inicio deste arquivo estão declaradas todas as bibliotecas de cabeçalho do Opencv, Qt e WiringPi utilizadas para o desenvolvimento.

- ***mainwindow.h*** – É um arquivo que possui a declaração de todas as classes e funções dos protótipos: *mainwindow.cpp* e *mainwindow.ui*, permitindo a reutilização e portabilidade dos mesmos.
- ***mythread.h*** – Semelhante ao arquivo *mainwindow.h*, este contém a declaração de todas as classes e funções de multiprocessamento do protótipo *mainwindow.cpp*.
- ***mainwindow.ui*** – É o arquivo que contém todos os formulários de construção da interface gráfica do programa.
- ***projetoArnt.pro*** - É o aquivo que possui todos os ficheiros e suas configurações básicas, junto com os caminhos e declarações das bibliotecas externas como: o Opencv e o WiringPi.

A figura 22 demonstra a tela da IDE do Qt Creator no desenvolvimento do sistema e a estruturação do software.

Figura 22 - IDE do Qt Creator e a estruturação do software.



Fonte: Autor (2017).

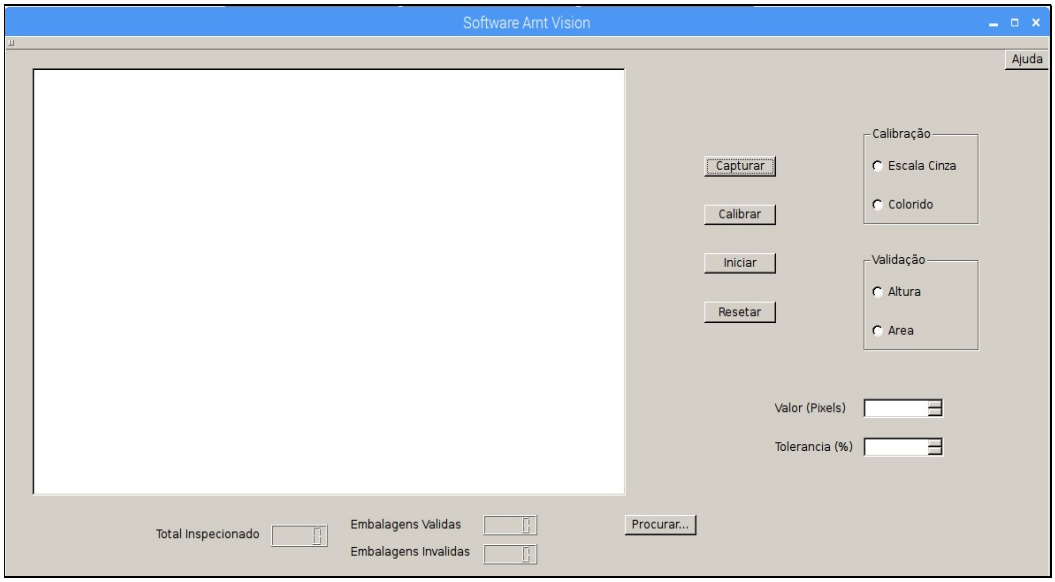
A utilização da IDE Qt Creator facilitou o desenvolvimento do software com a interface gráfica, utilizando suas ferramentas de design que permitiram a criação de botões, paineis, campos de edição e seleção, de forma simples e automática. O Qt Creator funciona através de uma comunicação de objetos baseadas em sinais e encaixes (*signals and slots)*. Os sinais são emitidos quando é alterado o estado de um objeto, como exemplo: Um evento quando um botão for pressionado. Já os encaixes são funções que são chamadas quando um sinal conectado a eles é emitido, exemplo: O botão fechar foi pressionado, logo o encaixe ou a

função fechar() será chamado. Desta forma foi desenvolvido um ambiente para que o usuário possa fazer a interação com o sistema de visão computacional de maneira simples e prática, com uma tela principal composta com os seguintes componentes:

- **Painel de Imagem** - É um painel utilizado para a visualização das imagens do sistema.
- **Botão Ajuda** - É um botão de pressionar utilizado para apresentar informações do software e um passo a passo das funcionalidades em uma caixa de mensagem.
- **Botão Capturar** – É um botão de pressionar utilizado para realizar a captura inicial da imagem da embalagem, com o nivel de enchimento padrão.
- **Botão Calibrar** – É um botão de pressionar utilizado para a funcionalidade de calibração do nivel de enchimento padrão, permitindo as configurações e ajustes, para a inspeção das embalagens.
- **Grupo de Botões Calibração** – É um grupo com dois botões de rádio utilizados para a selecionar o tipo de segmentação da imagem na calibração: escalas de cinza ou colorido.
- **Botão Iniciar** – É um botão de pressionar utilizado para iniciar a inspeção das embalagens, após ser realizado a calibração.
- **Grupo de Botões Validação** - É um grupo com dois botões de rádio utilizados para selecionar o tipo de variável a ser comparada na validação da inspeção de embalagens: Área ou Altura.
- **Botão Resetar** - É um botão de pressionar utilizado para interromper a inspeção de embalagens e resetar todas as configurações iniciais.
- **Botão Procurar** – É um botão de pressionar utilizado para abrir imagens de interesse no painel de imagem.
- **Campos de Edição Validação** – São dois campos de edição de texto utilizados para editar os valores em pixels de altura ou área do nivel de enchimento segmentado e o valor de tolerância em porcentagem.
- **Display Totalizador** – São três displays de digitos utilizados para a visualização da totalização das embalagens validas, invalidas e o total inspecionado.

A figura 23 demonstra a tela principal do sistema desenvolvido junto com todos seus componentes para interação.

Figura 23 - Tela principal do sistema desenvolvido.

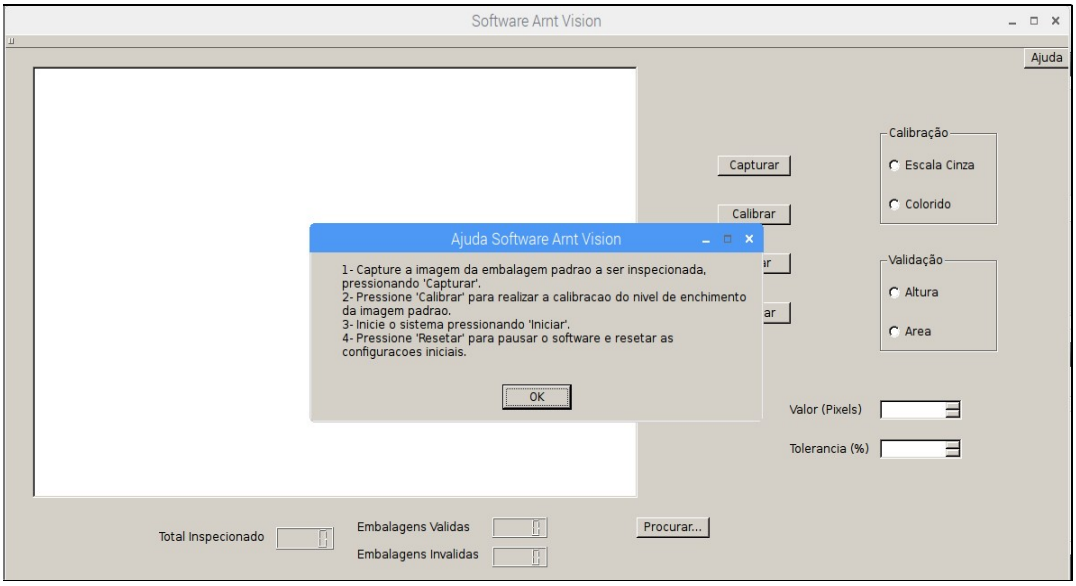


Fonte: Autor (2017).

4.1.1 Captura Inicial da Imagem Padrão

No desenvolvimento do sistema, ao executar o software, a tela principal é criada para que o usuário possa interagir com os seus componentes. Para realizar a inspeção do nível de enchimento das embalagens, o usuário deve seguir uma sequência de operações até iniciar a mesma. Esta sequência de operações pode ser visualizada em uma caixa de mensagem ao pressionar o botão Ajuda. A figura 24 ilustra a caixa de mensagem com informações de ajuda.

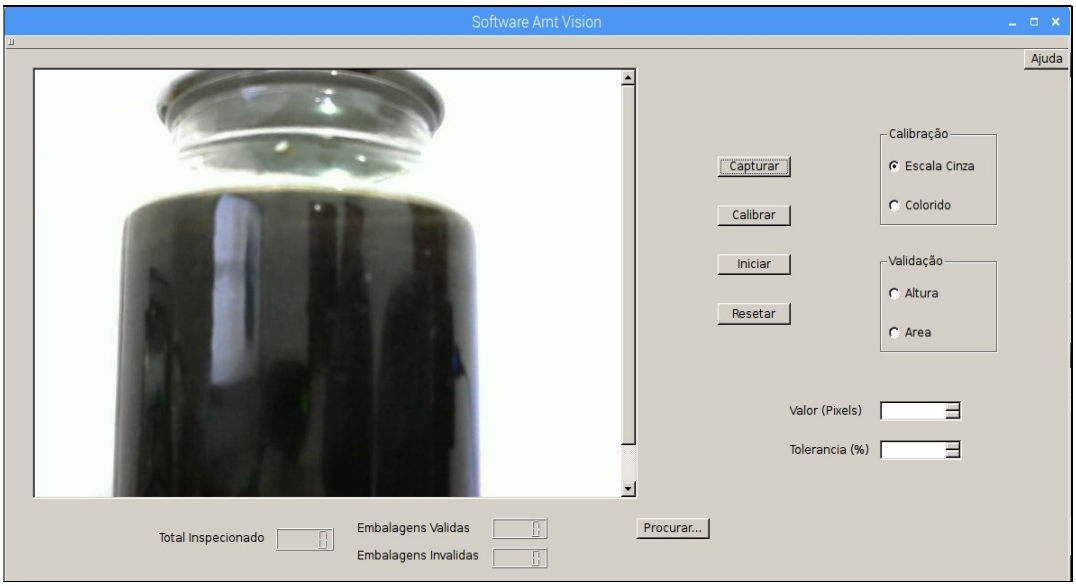
Figura 24 - Caixa de mensagem com informações de ajuda.



Fonte: Autor (2017).

A primeira operação a ser realizada é a captura inicial da imagem padrão, operação em que o usuário define uma embalagem transparente com um nível de enchimento padrão de um determinado produto, seja de propriedade líquida ou sólida, e posiciona a mesma ao foco da câmera digital para posterior captura da imagem, que pode ser executada e visualizada no painel, pressionando o botão Capturar. Neste momento a aquisição da imagem que é a primeira etapa do sistema de visão computacional é realizada. O evento do botão Capturar pressionado, chama a função *on_btnCapturar_clicked()* que possui internamente uma função da classe *VideoCapture* da biblioteca *videoio* do OpenCV, responsável em abrir o periférico da câmera digital, para que na sequência a função continue realizando a captura, a gravação e a visualização da imagem no painel. A figura 25 apresenta a captura inicial da imagem padrão.

Figura 25 - Captura inicial da imagem padrão.



Fonte: Autor (2017).

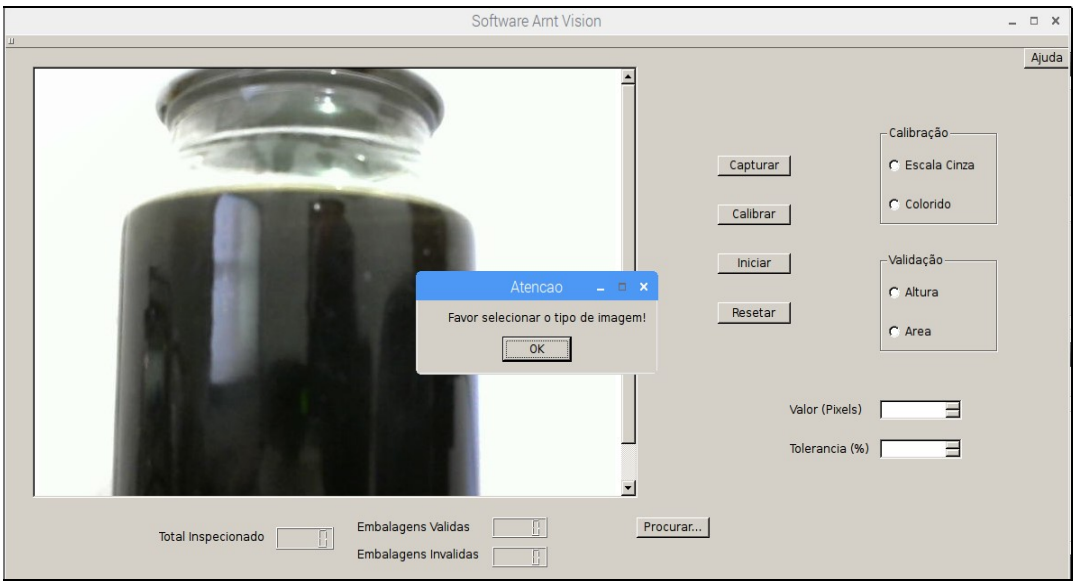
4.1.2 Calibração do Nível de Enchimento da Embalagem

Após ser realizada a captura da imagem padrão, o usuário pode prosseguir para o próximo passo, a calibração do nível de enchimento padrão da embalagem. Nesta etapa deve-se selecionar primeiramente o tipo de segmentação de imagem, selecionando um dos botões de rádio no grupo Calibração; Escalas de cinza ou Colorido. Esta seleção inicial permite duas opções. Uma segmentação de imagem simplificada, como em escalas de cinza para aplicações em que o produto envasado possui uma tonalidade de cor ou contraste mais forte; ou uma

segmentação colorida para determinada cor especifica ou com coloração mais fraca e translúcida.

Ao pressionar o botão Calibrar a função *on_btnCalibrar_clicked()* é chamada, e direciona para a função do tipo de segmentação selecionado: *segmentoBin()* para segmentação em escalas de cinza ou *segmentoColor()* para segmentação colorida. Nota-se que a seleção do tipo de segmentação no grupo de calibração é obrigatória, caso não esteja selecionada, ou ainda, não tenha sido realizado a captura da imagem padrão, uma caixa de mensagem alertando a ausência será criada na tela principal, bloqueando o acesso da funcionalidade por persistência, até que ocorra a convergência com o sistema. A figura 26 apresenta a caixa de mensagem informando a ausência da seleção obrigatória.

Figura 26 - Caixa de mensagem informando a ausência da seleção.



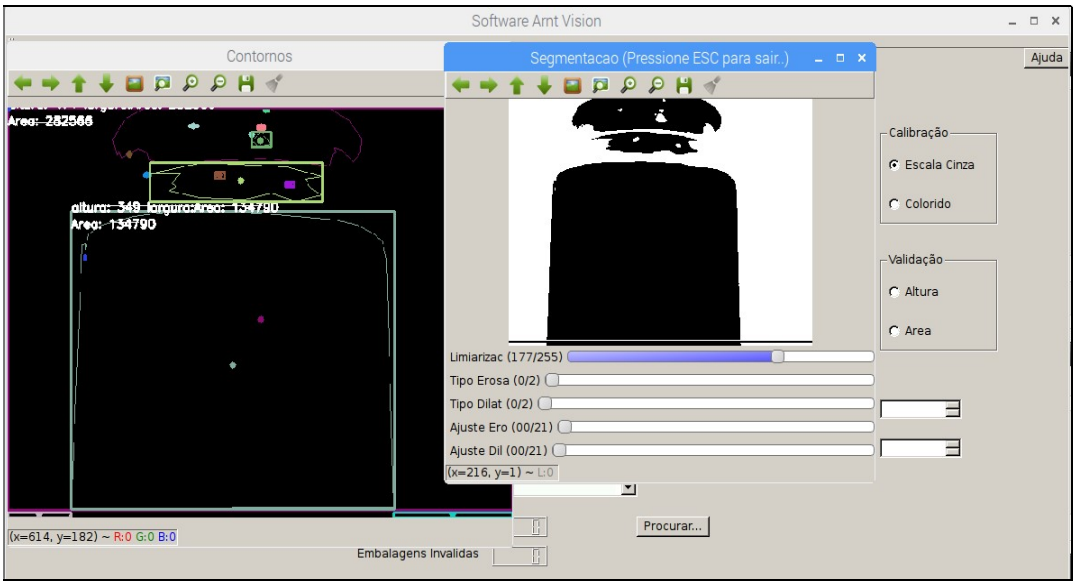
Fonte: Autor (2017).

De acordo com o tipo de segmentação selecionado, a função respectiva será executada criando duas telas auxiliares sobre a tela principal, graças às funções de interface gráfica *imshow()* e *CreateTrackbar()* da biblioteca *highgui* do OpenCV, presente internamente em ambas as funções. Nestas duas telas, uma possui varias barras de rolagem que são utilizadas para os ajustes da segmentação do nível de enchimento da embalagem e também a visualização da imagem binária segmentada, e a outra é utilizada para visualização dos contornos da imagem e a impressão das informações dos valores de altura e área do nível de enchimento computado. Valores que devem ser coletados para posteriormente ser utilizados para realizar a validação nas inspeções das embalagens. Após os ajustes da calibração com a segmentação desejável do nível de enchimento da embalagem e seus respectivos valores

característicos informados, o usuário pode finalizar a operação pressionando a tecla Esc. Momento em que as telas auxiliares são fechadas e os valores de ajustes das barras de rolagem são salvos para o próximo passo.

A função *segmentoBin()* tem a funcionalidade de realizar o pré-processamento, a segmentação em escalas de cinza e a extração de características da imagem padrão inicialmente capturada. A função inicialmente lê e carrega a imagem salva na operação anterior em uma matriz, para posteriormente converter a mesma em escalas de cinza com a função *cvtColor()* e filtrar eliminando ruídos com a função *blur()*, ambas da biblioteca OpenCV. Após ser realizado o pré-processamento da imagem, a função *calcularAreaBin()* realiza a segmentação da imagem em escalas de cinza e a computação dos valores de área e altura, a serem impressos e visualizados nas telas auxiliares. Esta função utiliza internamente as funções: *threshold()* para realizar a limiarização da imagem em escalas de cinza em imagem binária, *erode()* para realizar a erosão e *dilate()* para realizar a dilatação da imagem, ambas as funções associadas aos valores da interação das barras de rolagem. Os contornos da imagem são encontrados com a função *findContours()*, que finalmente são computados para encontrar os valores de área e altura do nível de enchimento da embalagem. A figura 27 demonstra a calibração com a segmentação em escalas de cinza.

Figura 27 - Calibração com a segmentação em escalas de cinza.

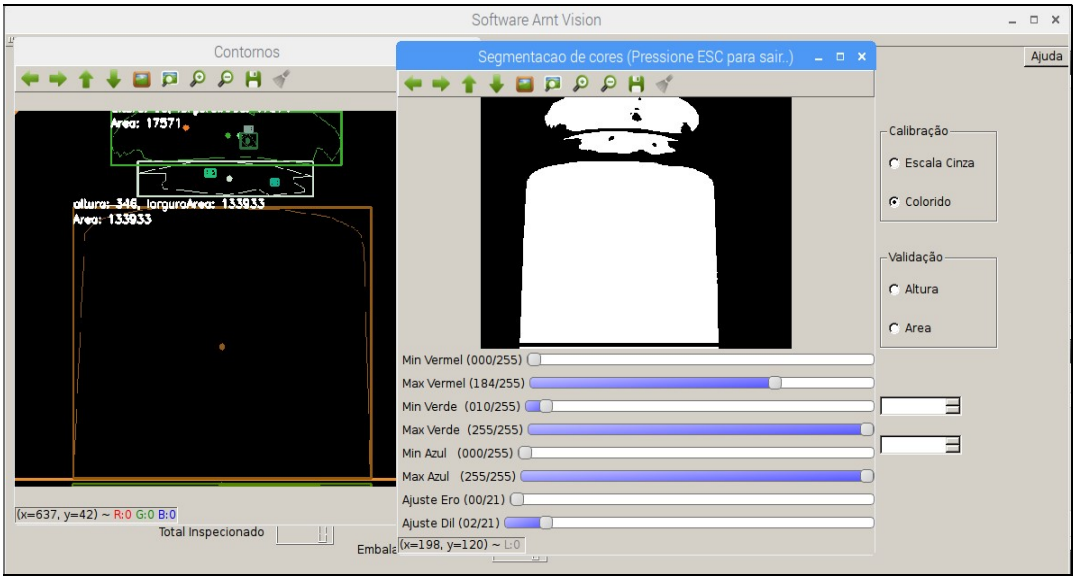


Fonte: Autor (2017).

A função *segmentoColor()* é semelhante à função *segmentoBin()*, porém tem a funcionalidade de realizar a segmentação colorida da imagem padrão inicialmente capturada. Esta função inicialmente lê e carrega a imagem inicial capturada e salva-a em uma matriz,

filtrada com a função *blur()*. Após isso a função *calcularAreaColor()* realiza a segmentação da imagem colorida e a computação dos valores de área e altura, a serem impressos e visualizados em suas respectivas telas auxiliares. Semelhante a função descrita anteriormente, esta utiliza internamente as funções: *inRange()* para realizar a limiarização da imagem colorida em imagem binária, *erode()* e *dilate()* para realizar a erosão e a dilatação da imagem, ambas as funções também associadas aos valores da interação das barras de rolagem. A função *findContours()* é utilizada para encontrar os contornos da imagem, que são computados para encontrar os valores de área e altura do nível de enchimento. A figura 28 demonstra a calibração com a segmentação colorida.

Figura 28 - Calibração com a segmentação colorida.



Fonte: Autor (2017).

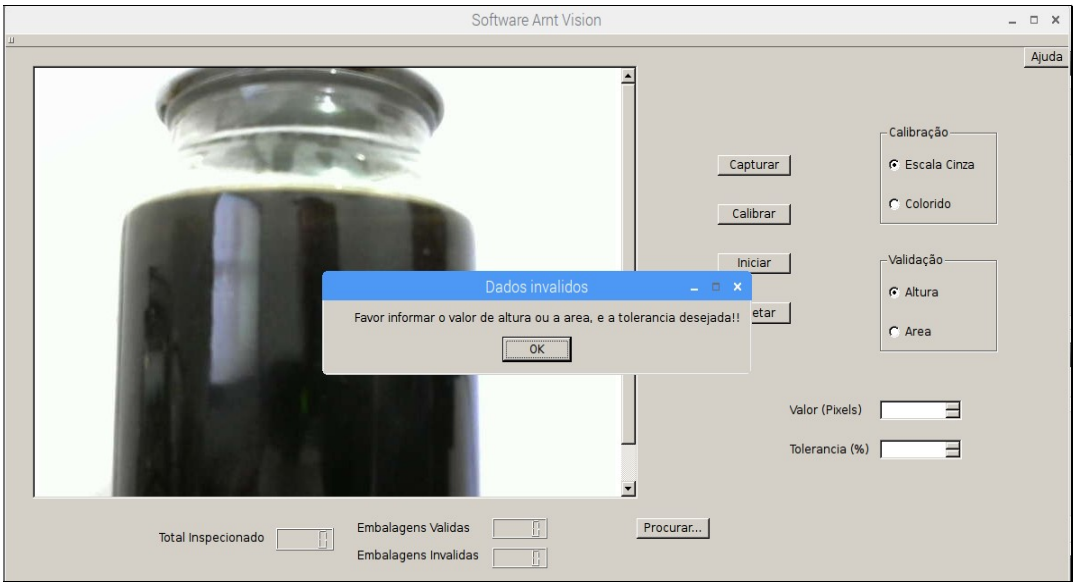
4.1.3 Inicialização da Inspeção Automática

A inspeção do nível de enchimento das embalagens pode ser inicializada após serem realizadas as etapas da captura inicial da imagem padrão e a calibração no nível de enchimento da embalagem, com as informações dos valores de área e altura encontrados. Além destas etapas que devem ser inicialmente concluídas, o usuário deve selecionar qual a variável a ser comparada para a inspeção, com os botões de radio do grupo Validação; Área ou Altura. Estas opções permitem que o usuário possa fazer inspeções do nível de enchimento de embalagens transparente com produtos líquidos e sólidos. A opção Altura é recomendada para produtos líquidos, no qual o enchimento geralmente é uniforme, e a opção Área para

produtos sólidos em que o nível de enchimento muitas vezes não é uniforme e desregular por suas propriedades físicas.

Após a seleção da variável para a validação, é necessário realizar a informação dos valores nos campos de edição e pressionar o botão Iniciar para inicializar a inspeção do nível de enchimento das embalagens. Informando no campo de edição Valor, o valor de área ou altura coletado na etapa anterior de calibração; e no campo de edição Tolerância, o valor aceitável em porcentagem do nível de enchimento medido. Ao pressionar o botão Iniciar a função *on_btnIniciar_clicked()* é chamada, e com isso uma nova instância do objeto *MyThread()* da biblioteca Qt é criada para realizar o multiprocessamento da funcionalidade em paralelo com a tela principal, executando a função *MyThread::run()*. Nota-se que a seleção da variável para validação e a edição dos valores nos campos de edição é obrigatório, juntamente com a conclusão das etapas anteriores da aquisição e calibração. Caso alguma destas condições não tenha sido realizada, uma caixa de mensagem alertando a ausência da mesma será criada sobre a tela principal, bloqueando o acesso da funcionalidade por persistência, até que todas as condições sejam atendidas. A figura 29 demonstra a caixa de mensagem informando a ausência da edição dos campos do valor e tolerância.

Figura 29 - Caixa de mensagem informando a ausência da edição dos campos.

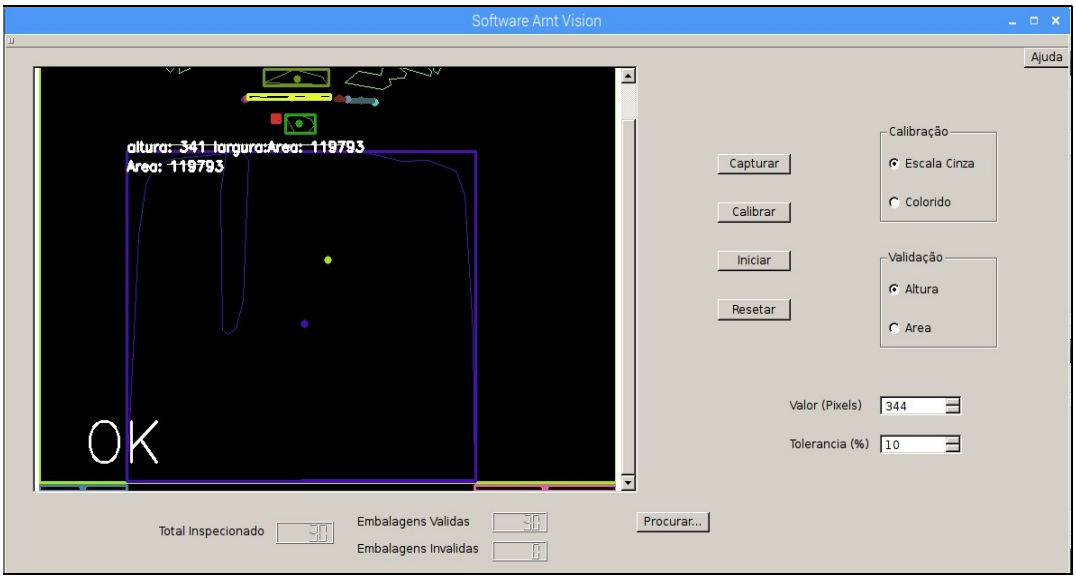


Fonte: Autor (2017).

Na função *MyThread::run()* é executado todas as funcionalidades desta etapa, que inicia com a leitura do tipo de variável selecionado para validação e também os valores dos campos de edição da tela principal, para posteriormente compará-los com os valores computados de cada embalagem inspecionada. Após a leitura dos dados, a função entra em

um laço repetitivo condicionado por uma variável booleana de uma função associada ao botão Resetar. Neste momento a passagem das embalagens pelo sensor óptico difuso, pode ser detectada com a leitura do pino de GPIO da placa Raspberry em que o mesmo esta conectado, realizado com a função *digitalRead()* da biblioteca WiringPi. Quando a embalagem é detectada, o periférico da câmera digital é aberto para a captura da imagem da mesma, com uma nova instancia da função *VideoCapture* e carregada em uma matriz. A figura 30 demonstra a validação de uma embalagem inspecionada com nível de enchimento conforme.

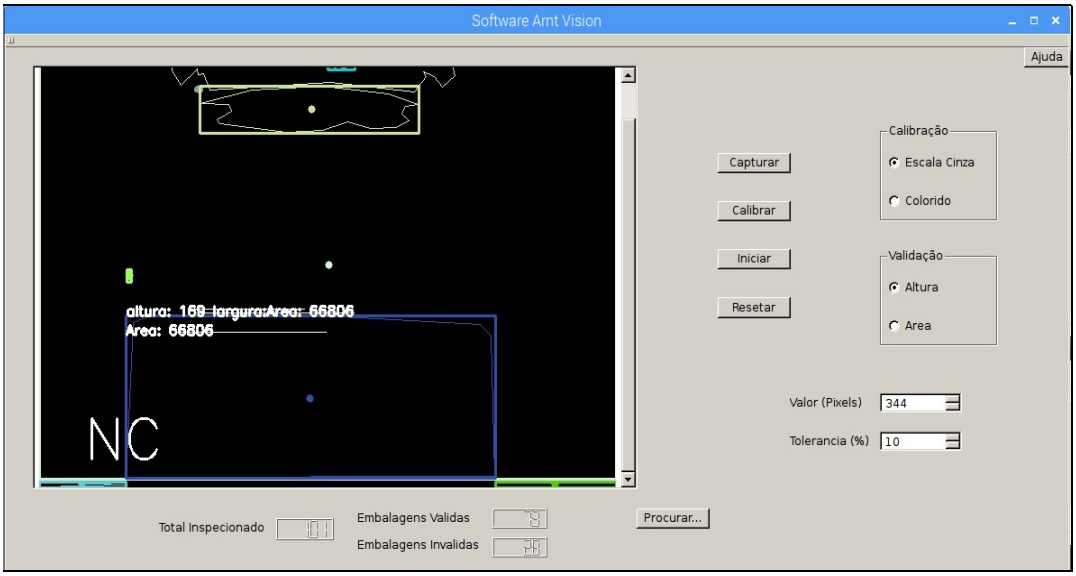
Figura 30 - Validação de uma embalagem inspecionada.



Fonte: Autor (2017).

Baseada nas informações e variáveis salva da calibração do nível de enchimento padrão anteriormente realizada, a função direciona para uma das segmentações de imagem; escalas de cinza ou colorida, computando os seus valores de altura e área do nível de enchimento segmentado, para compará-los com os valores de validação configurados na tela. Nesta comparação, se os valores computados estiverem dentro do intervalo de mais ou menos o valor da variável com sua tolerância especificada, o nível de enchimento da embalagem inspecionada é valida. Caso os valores forem inferiores ou superiores ao intervalo especificado, o nível de enchimento da embalagem é invalido, e a saída de alarme é acionada com função *digitalWrite()* também da biblioteca WiringPi, realizando a escrita do pino da placa em que a mesma se encontra conectada e mantendo-a acionada por 500 milissegundos. A figura 31 demonstra a invalidação de uma embalagem inspecionada com o nível de enchimento não conforme.

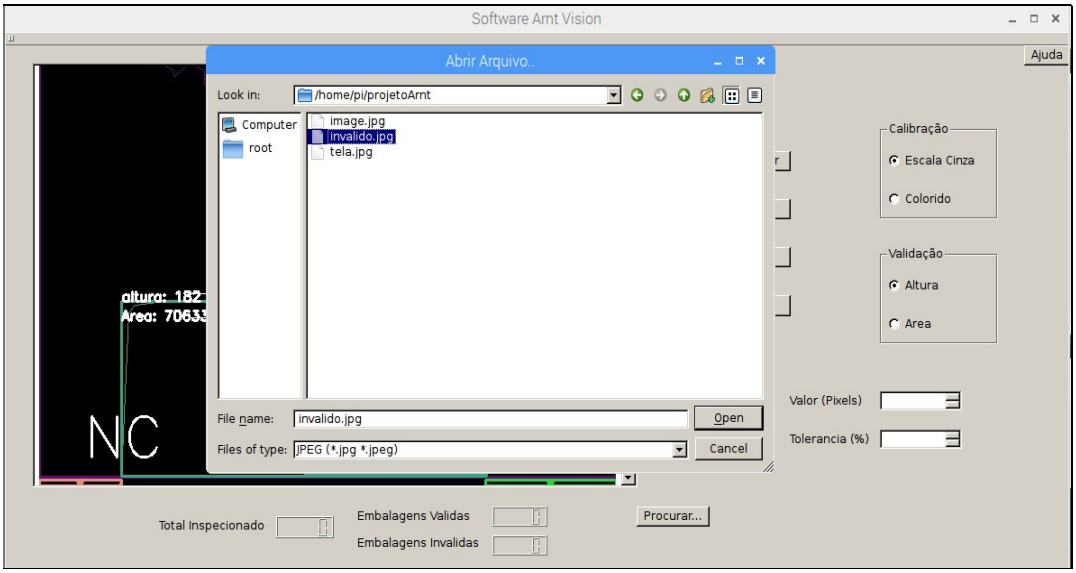
Figura 31 - Invalidação de uma embalagem inspecionada.



Fonte: Autor (2017).

Em ambos os casos da comparação, a imagem do nível de enchimento segmentado com seus respectivos valores e status de validação, é disponibilizada para a visualização instantânea no painel de imagem, juntamente com a atualização e totalização de seus respectivos displays na tela principal. Quando a inspeção do nível de enchimento da embalagem é valida, a imagem é impressa com o status OK no canto esquerdo inferior da mesma, e também no mesmo local quando invalida, mas com a impressão do status NC (Não Conforme), alem de salva-la em um arquivo do programa para posteriormente ser analisada e aberta com o botão Procurar. A figura 32 apresenta a ferramenta de procura do sistema.

Figura 32 - Ferramenta de procura do sistema.

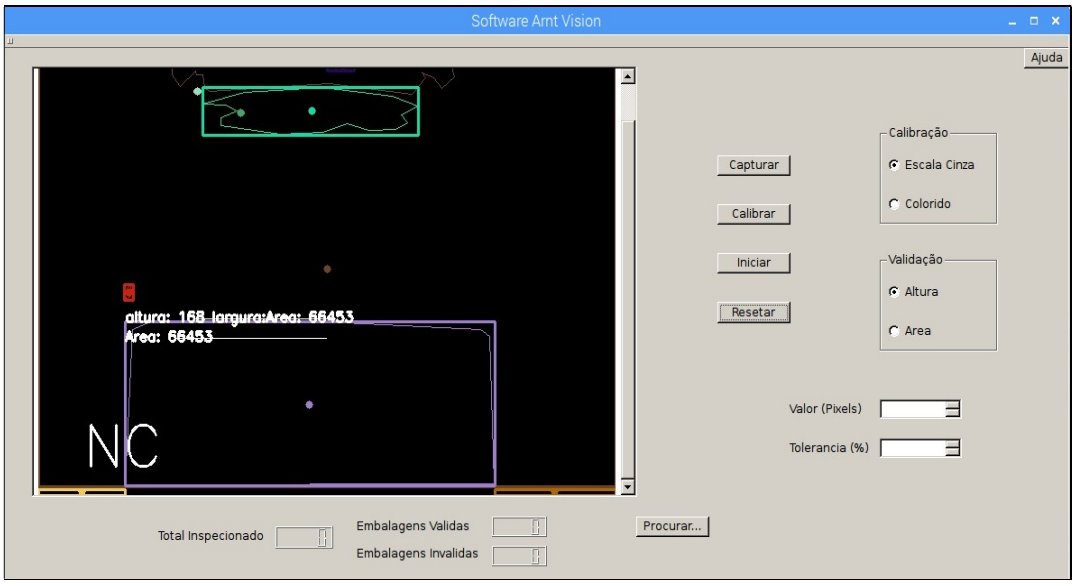


Fonte: Autor (2017).

4.1.4 Interrupção da Inspeção e Configurações Iniciais

Quando iniciado a inspeção do nível de enchimento das embalagens, a função *MyThread::run()* fica presa em um laço repetitivo, condicionado por uma variável booleana de uma função associada ao botão Resetar. Para finalizar a inspeção e resetar as configurações iniciais do sistema, o usuário deve pressionar este botão para que a função *on_btnResetar_clicked()* seja chamada. Na execução desta função, a variável booleana é alterada de estado e todas as variáveis do programa recebem seus valores iniciais, fazendo com que a função *MyThread::run()* seja finalizada, destruindo aquela instancia de multiprocessamento. A figura 33 demonstra a interrupção da inspeção das embalagens e a reconfiguração inicial das variáveis do sistema.

Figura 33 - Interrupção da inspeção e reconfiguração inicial.



Fonte: Autor (2017).

5 RESULTADOS E DISCUSSÃO

Neste capitulo será apresentado os resultados obtidos com a montagem e os testes de inspeção do protótipo do projeto, realizados com o enchimento de produtos líquidos e sólidos em embalagens transparentes. Transportados por uma esteira motorizada em ambiente industrial simulado no laboratório de automação da instituição, e posteriormente uma discussão sobre estes resultados.

5.1 Montagem do Protótipo do Sistema

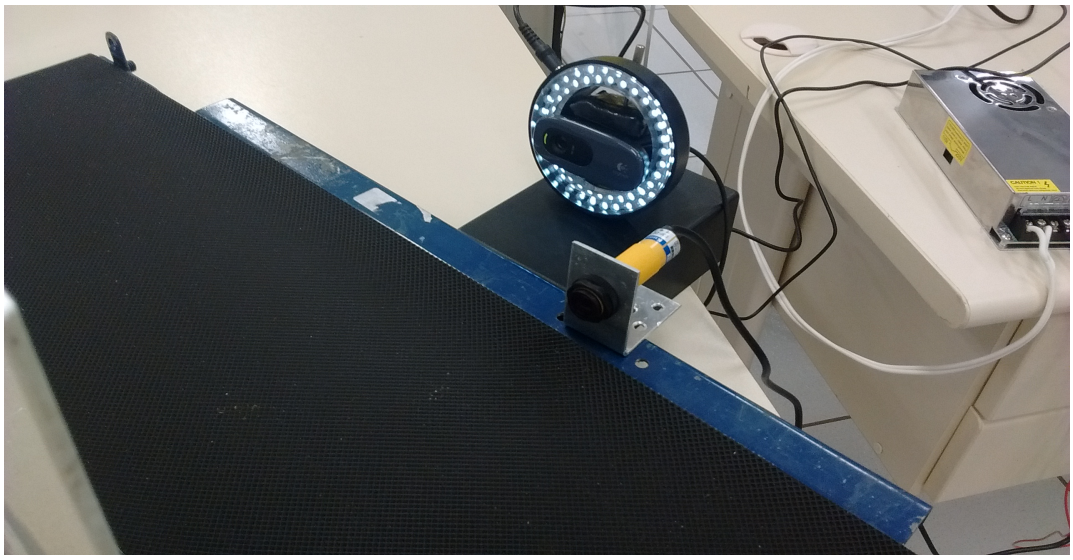
A montagem do protótipo do sistema de automação para inspeção do nível de enchimento de embalagens foi realizada no laboratório de automação da instituição de ensino. Para a montagem do sistema, além dos componentes descritos anteriormente, foram utilizados: uma esteira motorizada para realizar o transporte das embalagens; uma fonte de alimentação 24 VCC para alimentar o motor da esteira, o sensor óptico difuso e a saída de alarme; um monitor com teclado e mouse para interação com o sistema; e um painel plástico de cor branca utilizada como cenário para facilitar o processamento de imagens.

Figura 34 - Protótipo do sistema montado no laboratório.

Fonte: Autor (2017).

Na figura 34 é possível visualizar na imagem à direita, o controle do iluminador anelar utilizado para melhorar a iluminação da cena e o minicomputador Raspberry Pi 3 em uma caixa plástica acrílica. Com a conexão de todos os periféricos utilizados no protótipo: teclado, mouse e câmera digital conectado em suas portas USB; saída de vídeo HDMI conectado com o cabo do monitor; e o cabo de alimentação conectado no minicomputador. Na parte superior da caixa que contem a Raspberry, pode ser visualizada a placa do circuito de interface opto-isolada. Esta placa esta conectado com os pinos GPIO0 e GPIO2 ambos configurados respectivamente como entrada e saída, e também os conectores do sensor óptico difuso e a saída de alarme, ambos alimentados com 24 VCC derivado da fonte de alimentação, visível na imagem à esquerda. A figura 35 demonstra câmera digital e o sistema de aquisição de imagens do protótipo.

Figura 35 - Câmera digital e o sistema de aquisição do protótipo.



Fonte: Autor (2017).

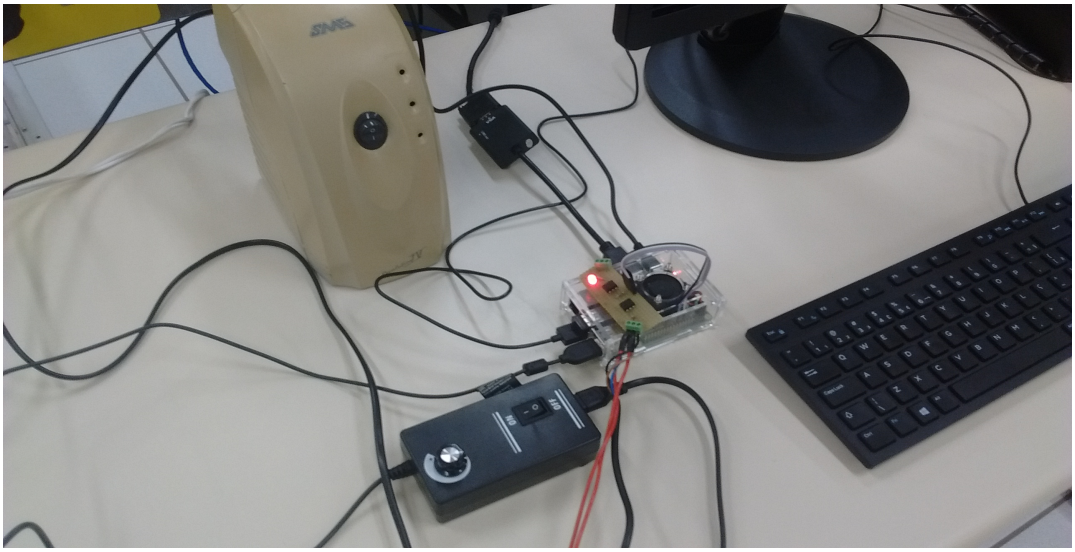
5.2 Testes e Ajustes Iniciais

No desenvolvimento do protótipo e do software do sistema, os testes com as embalagens eram realizados com a captura das imagens das mesmas estáticas, ou seja, em repouso na frente da câmera. Nesta etapa os resultados de repetibilidade foram muito satisfatórios, mais de 99% de acertos com a tolerância de 1% configurado. Porem para os testes finais no laboratório com a esteira motorizada foi necessário realizar alguns testes e ajustes no software e protótipo do sistema. A velocidade aproximada em que a esteira opera é de 60 embalagens por minuto, ou seja, aproximadamente uma embalagem por segundo, dado coletado com um cronômetro. Com isso foi necessário ajustar a posição da câmera digital

fisicamente e também criar um atraso no software entre a passagem da embalagem pelo sensor óptico e a captura da imagem com a câmera digital.

Além disso, também foi ajustada a intensidade do iluminador anelar com a iluminação do ambiente local, ajuste que é recomendado realizar toda vez que for inicializado uma inspeção com o sistema, para uma melhor qualidade na aquisição das imagens na cena. E também foram realizados testes iniciais de validação com a inspeção de embalagens transparente com produtos líquidos e sólidos, realizando ajustes no software do sistema e testando as melhores configurações e as atuações físicas do protótipo. A figura 36 ilustra o teste de validação do sistema com a saída de alarme.

Figura 36 - Teste de validação com a saída de alarme.

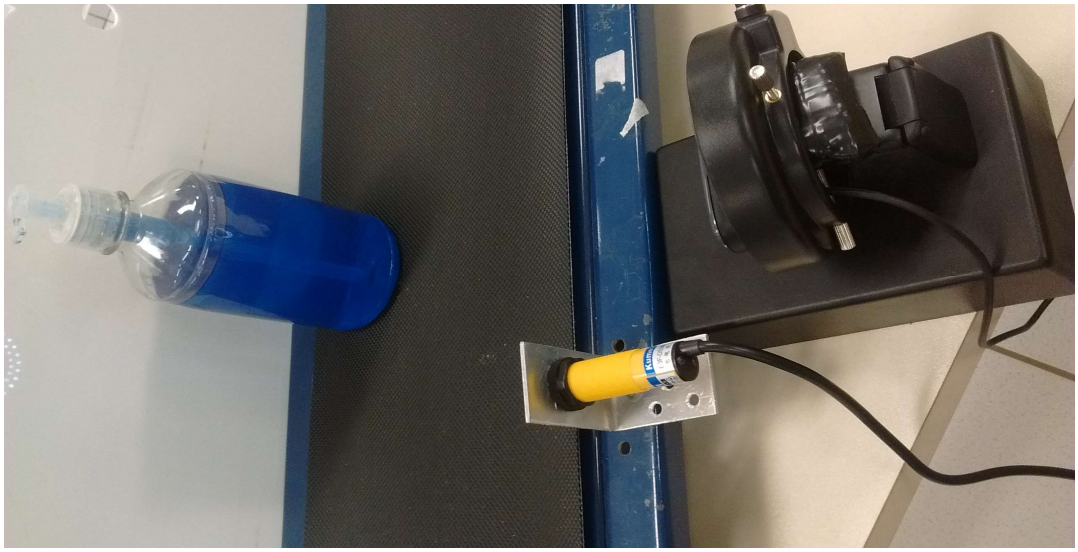


Fonte: Autor (2017).

5.3 Inspeção do Nível de Enchimento com Produto Líquido

Após ser realizada a montagem, os testes e ajustes iniciais no software e protótipo do sistema, iniciaram-se os testes de validação com a inspeção do nível de enchimento de embalagens com produto líquido. Foi selecionada uma embalagem transparente com determinado nível de enchimento de um produto na forma líquida com coloração, utilizada como padrão de enchimento nas inspeções do protótipo. Este padrão foi posicionado de forma estática em frente à câmera digital na distancia de operação do transporte das embalagens pela esteira, para em seguida ser realizado a captura da imagem e os ajustes de intensidade luminosa na cena com o controle do iluminador e a calibração da imagem padrão, ajustados conforme a melhor segmentação do nível de enchimento na mesma. A figura 37 demonstra a inspeção do nível de enchimento das embalagens com produto líquido.

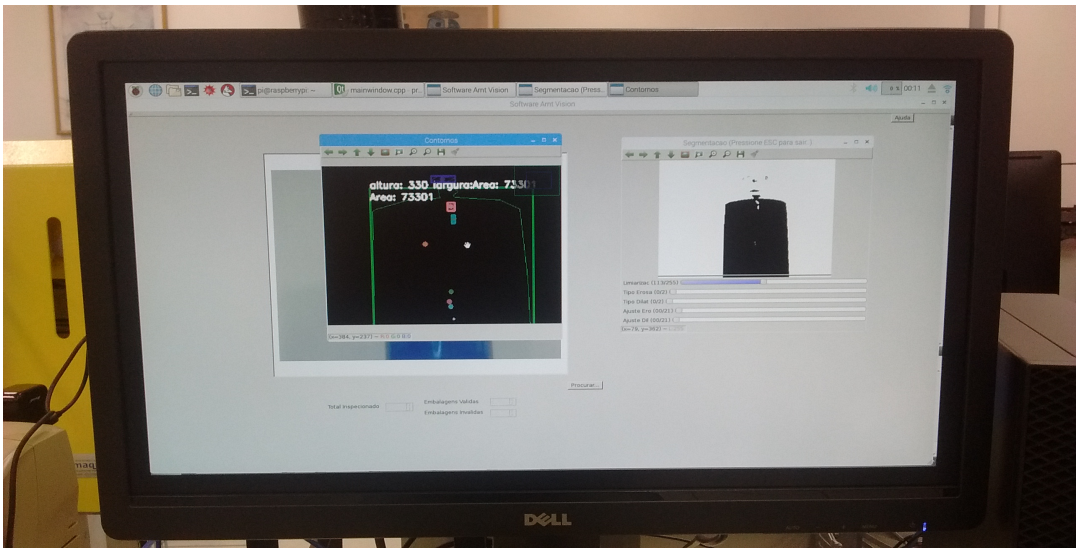
Figura 37 - Inspeção de embalagens com produto líquido.



Fonte: Autor (2017).

A montagem da cena com o painel plástico de cor branca permitiu uma calibração genérica, em que qualquer coloração com tonalidade razoavelmente forte, poderia ser facilmente segmentada com escalas de cinza. Entre tanto se fosse desejável a detecção de cores ou aplicações em produtos com tonalidade de cor fraca e translúcida, poderá ser utilizado a segmentação colorida. Nestes testes, a embalagem com produto líquido selecionada foi calibrada com segmentação em escalas de cinza, com pequenos ajustes na barra rolagem de limiarização, com a obtenção dos valores de altura e área em pixels do nível de enchimento segmentado da imagem. A figura 38 demonstra a calibração da embalagem.

Figura 38 - Calibração da embalagem com produto líquido.

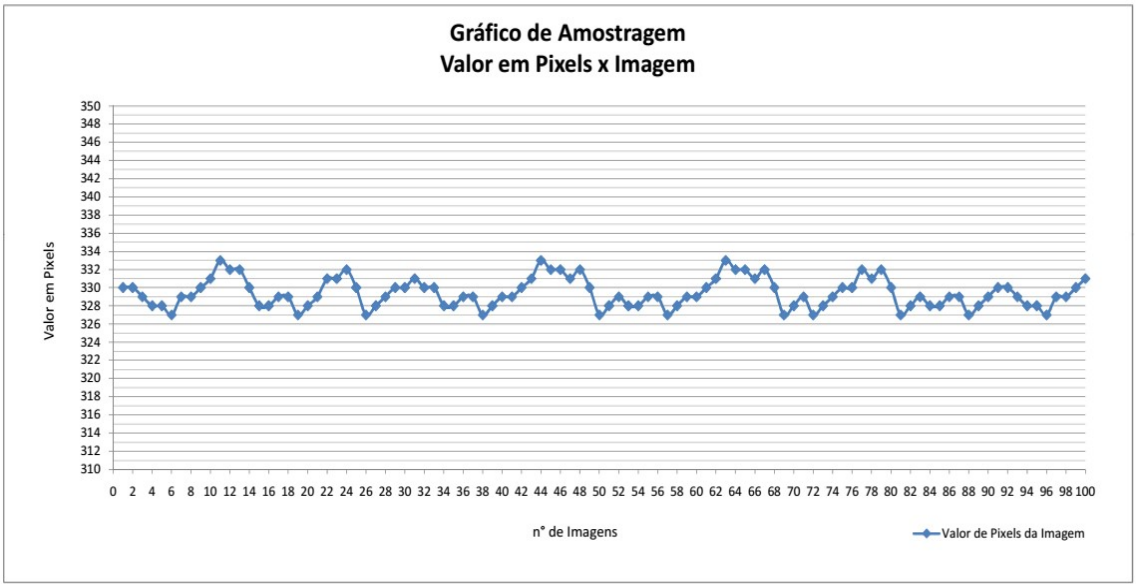


Fonte: Autor (2017).

Como o produto líquido tem a propriedade física de ocupar todo o espaço do nível de enchimento, permite o uso da variável altura para a realização das inspeções. Nos campos de validação foi selecionada a variável altura e editado o valor de 330 pixels (valor de altura encontrado na calibração), junto com a tolerância de 5% em seus respectivos campos de edição e inicializado as inspeções com o protótipo com a esteira em movimento. Nas inspeções das embalagens foi realizado um teste de repetibilidade com a amostragem de cem inspeções com o padrão de enchimento em movimentação, para verificar as oscilações das medições encontradas com o protótipo e posteriormente a passagem de embalagens com diferentes níveis avaliando a validação do sistema.

Na figura 39 apresentada a seguir, é possível visualizar o gráfico de amostragem com as oscilações das medições dos valores de altura do nível de enchimento segmentado em pixels por imagem amostrada.

Figura 39 - Gráfico de amostragem das inspeções das embalagens com produto líquido.



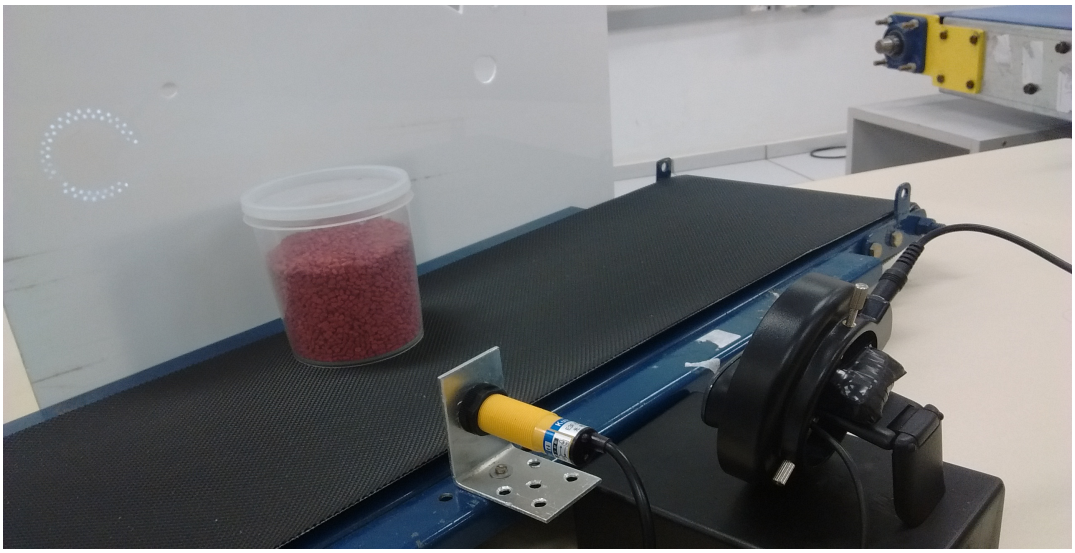
Fonte: Autor (2017).

No teste de repetibilidade com a inspeção repetitiva do padrão de enchimento em movimento, foi observado picos de até 3 pixels de oscilação comparados ao valor de referencia, que equivalem aproximadamente 1 % do valor medido. O desvio padrão da oscilação deste teste foi de 1,63 pixels, valor que pode ser visualizado na tabela 1 de cálculos disponibilizada no Apêndice deste trabalho. Nos testes de validação com a tolerância de 5 % com diferentes níveis de enchimento nas embalagens, os resultados foram todos satisfatórios com nenhum erro sistemático de validação.

5.4 Inspeção do Nível de Enchimento com Produto Sólido

Depois de serem realizados os testes com a inspeção do nível de enchimento com produto líquido e a coleta dos resultados, o sistema foi resetado com as configurações iniciais para realizar os testes com a inspeção das embalagens com o nível de enchimento com produto sólido. Os testes de validação foram realizados com uma embalagem transparente com determinado nível de enchimento de polímero granulado, utilizado como padrão de enchimento para as inspeções com o protótipo. Semelhante aos procedimentos realizados na inspeção anterior, a embalagem padrão foi posicionada em frente da câmera digital, para captura da imagem, ajustes da iluminação e calibração da imagem padrão. A figura 40 demonstra a inspeção de embalagens com produto sólido.

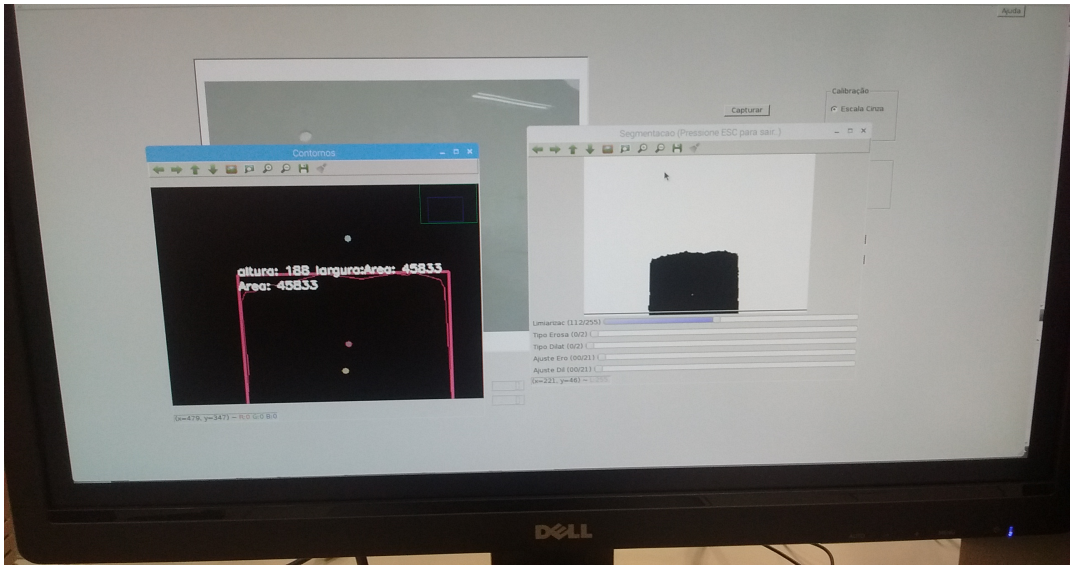
Figura 40 - Inspeção de embalagens com produto sólido.



Fonte: Autor (2017).

O produto sólido na forma granulada pode ter um enchimento com o nível desuniforme, com ocupações irregulares no interior das embalagens, diferente dos produtos líquidos. Neste caso recomenda-se a utilização da variável área para realizar a validação e inspeções das embalagens. A calibração para esta inspeção demonstrada na figura 41 foi realizada com a segmentação em escalas de cinza, com a seleção da variável área e editada 45833 pixels (valor de área encontrado na calibração), junto com a tolerância de 5 % em seus respectivos campos de edição no grupo validação, e inicializado as inspeções das embalagens com a esteira em movimento. Foram também realizados os testes de repetibilidade com a passagem do padrão de enchimento conforme o procedimento já realizado anteriormente, junto com os testes de validação com a passagem de embalagens com diferentes níveis de enchimento, a fim de avaliar as oscilações das medições e as validações do sistema.

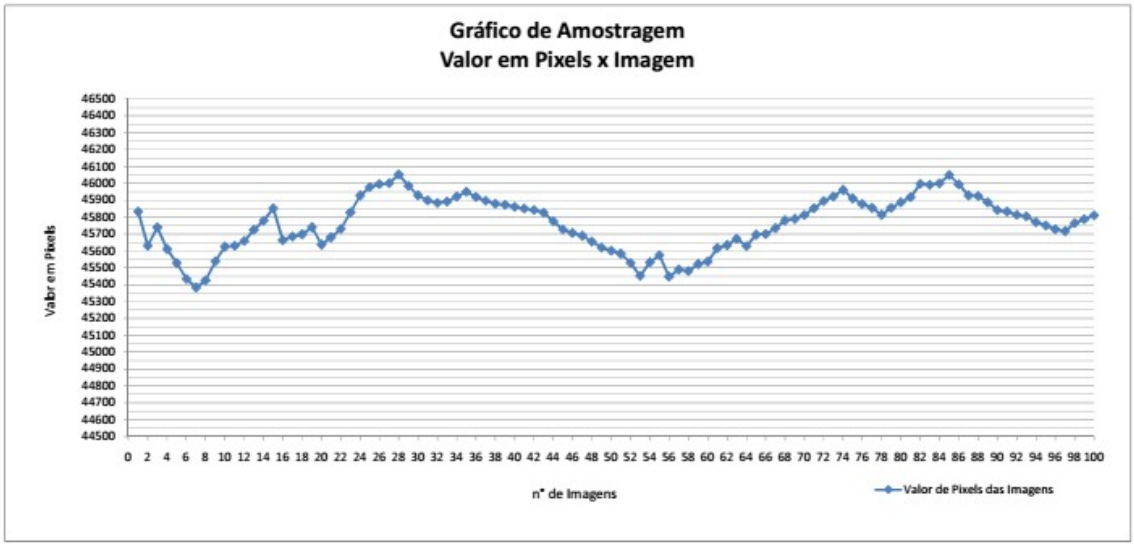
Figura 41 - Calibração da embalagem com produto sólido.



Fonte: Autor (2017).

Na inspeção das embalagens com enchimento de produto sólido, foi possível observar no teste de repetibilidade, picos de até 450 pixels de oscilação comparados ao valor de referencia, que equivalem aproximadamente 1 % do valor medido, visível no gráfico de amostragem da figura 42. O desvio padrão da oscilação deste teste foi de 249,7 pixels, valor que pode ser visualizado na tabela 2 de cálculos disponibilizada no Apêndice. Nos testes de validação com diferentes níveis de enchimento nas embalagens, o sistema mostrou-se eficiente, não apresentando erros de validação, e novamente os resultados foram satisfatórios.

Figura 42 - Gráfico de amostragem da inspeção das embalagens com produto sólido.



Fonte: Autor (2017).

5.5 Discussão dos Resultados

Os indicadores apresentados neste capitulo, foram os dados dos melhores resultados obtidos de uma série de testes e ajustes realizados com o protótipo montado no laboratório de automação. Para conseguir um satisfatório desempenho com o sistema, alguns fatores devem ser cuidadosamente ajustados antes de inicializar a inspeção das embalagens:

- A iluminação sobre o objeto a ser realizada a aquisição de imagens não deve ser muito intensa, pois o brilho excessivo prejudica a segmentação da imagem.
- A iluminação também não pode ser muito fraca, pois a iluminação do ambiente pode interferir na qualidade da aquisição de imagens.
- Ao realizar a calibração deve-se procurar a melhor segmentação possível do nível de enchimento da embalagem, eliminando contornos falsos e elementos indesejáveis com os ajustes nas barras de ferramentas do sistema.
- Recomenda-se o uso de guias nas laterais da esteira, para evitar que a embalagem transportada não se aproxime e nem se afaste da câmera digital no momento que é realizado a captura da imagem, evitando assim oscilações nas medições dos pixels.

Os resultados em geral obtidos com os testes do protótipo do sistema foram satisfatórios. Os testes de repetibilidade em ambas as inspeções, com produtos: líquidos e sólidos apresentaram boa precisão, com poucos pixels de oscilação comparada com o valor de referência, alcançando picos máximos de oscilação de aproximadamente 1 % do valor medido. Os testes de validação mostraram que em ambas as inspeções, o sistema foi eficiente e confiável, não apresentando erros sistemáticos dentro da faixa de tolerância configurado. Resultados que validam parcialmente o protótipo do sistema desenvolvido, para aplicações industriais simples que não tenham muita exigência de precisão e robustez quanto ao ambiente.

A utilização de uma câmera digital com maior resolução e quantização podem melhorar ainda mais os resultados. Como as comparações do sistema são realizadas por pixels, uma imagem com resolução maior teria maior quantidade dos mesmos, o que diminuiria o percentual de oscilação em relação ao valor de referencia. Para melhores resultados em uma implementação do sistema em ambiente industrial, alguns aperfeiçoamentos no software e melhoramentos no hardware ainda devem ser realizados, como: blindagem do hardware, conectores especiais, aterramento, entre outros. Além da realização de outros testes e ensaios elétricos, para garantir robustez e confiabilidade na aplicação de interesse.

6 CONCLUSÕES

A utilização da visão computacional vem crescendo cada vez mais e se expandindo nas mais diferentes áreas. Um dos fatores que motivaram e despertaram o interesse de estudo destas tecnologias e o desenvolvimento deste trabalho. Nas indústrias em geral é fundamental a utilização destas tecnologias para a automatização de seus processos, não com a intenção de eliminar a mão de obra humana e suas funções, mas sim auxiliar de forma mais eficiente, rápida e precisa, reduzindo custos e otimizando os sistemas.

Todo o estudo realizado foi de fundamental importância para o entendimento da tecnologia e o desenvolvimento do projeto. A utilização da biblioteca OpenCV foi essencial para o desenvolvimento do sistema, disponibilizando um grande numero de funções e recursos de visão computacional que facilitaram a sua construção junto com a biblioteca de aplicações do Qt. A escolha da placa Raspberry Pi 3 foi pontual com seu bom desempenho e com seu processamento robusto e eficaz, permitindo o embarque do software do sistema e a interação com os demais componentes com a sua execução em tempo real.

No geral os resultados obtidos com os testes de repetibilidade e validação nas inspeções do nível de enchimento de embalagens transparente com produtos líquidos e sólidos foram satisfatórios. Em ambas as inspeções os resultados apresentaram uma boa precisão com uma validação eficiente, atendendo as expectativas com o projeto. De forma a validar parcialmente o sistema para aplicações industriais simples com exigências menores, que pode ser melhorado com aperfeiçoamentos no software e melhoramentos no hardware, atribuindo mais confiabilidade e robustez para as aplicações.

No aperfeiçoamento do projeto futuro, pretende-se realizar um melhoramento no hardware, com uma blindagem com conectores especiais para acondicionar o protótipo para maior resistência em ambientes mais agressivos. Realizar a implementação de mais funcionalidades de visão computacional no software do sistema, como a detecção de rótulos e reconhecimentos de caracteres para aplicações mais complexas e também incluir uma comunicação serial com algum protocolo livre como o Modbus, permitindo o acesso aos dados do sistema e até configurações e ajustes de forma remota. Aperfeiçoamentos que visam otimizar o projeto.

REFERÊNCIAS

BRAHMBHATT, Samarth. **Pratical OpenCV.** 1º Ed. New York, EUA: Springer, 2013.

COGNEX, Cognex: **Guia Especializado: Comprando Um Sistema de visão: 10 Perguntas Que Você Deve Fazer**. Disponível em: <http://www.cognex.com/ExploreLearn/FindOutMore/WhitePapersArticles/WhitePaperArticle.aspx?id=6314>. Acessado em 18 de setembro de 2016.

DATASHEET 4N35, Datasheet 4n35. **Optocoupler 4n35**. Disponível em: <https://www.vishay.com/docs/81181/4n35.pdf>. Acessado em 02 de dezembro de 2016.

DAVIES, Roy. **Computer and Machine Vision: Theory, Algorithms, Practicalities,** 4° Ed. Egham, Surrey, GB: AP, 2012.

GARCÍA, Gloria B.; SUAREZ, Oscar D.; ARANDA, José Luis E.; TERCERO, Jesus S.; GRACIA, Ismael S.; ENANO, Noelia V.; **Learning Image Processing with OpenCV**, 1° Ed. Birmingham, GB: Packt, 2015.

GONZALEZ, Rafael C.; WOODS, Richard E.; **Digital Image Processing**, 3° Ed. Upper Saddle River, NJ, US: Prentice-Hall, 2008.

JOSHI, Prateek; ESCRIVÁ, David M.; GODOY, Vinícius. **OpenCV By Example**, 1° Ed. Birmingham, GB: Packet Publishing, 2016.

KAEHLER, Adrian; BRADSKI, Gary. **Learning OpenCV**, 2º Ed. Sebastopol, US: O'Reilly Media, 2006.

KEYENCE, Keyence: **Reduza os Custos e Elimine as Reclamações de Produtos Utilizando Sistemas de Visão**. Disponível em: <http://www-search.keyence.com.br/br/ptbr/downloadcon/search.x>. Acessado em 20 de setembro de 2016.

KEYENCE, Keyence: **Guia Técnico de Sistemas de Visão**. Disponível em: <http://www-search.keyence.com.br/br/ptbr/downloadcon/search.x>. Acessado em 02 de outubro de 2016.

LAGANIÈRE, Robert. **OpenCV 2 Computer Vision Application Programming Cookbook**, 1° Ed. Birmingham, GB: Packt Publishing, 2011.

LOGITECH, Logitech: **HD Webcam Logitech C270**. Disponível em: <http://www.logitech.com/pt-br/product/hd-webcam-c270>. Acessado em 01 de maio de 2016.

LUCKY ZOOM, LLucky Zoom: **Iluminador Anelar Lucky Zoom**. Disponível em: < http://www.luckyfilm.com/html/Product/www/bdlkjckmyyxgs/html/en/product/jckenproducts/index.html>. Acessado em 01 de abril de 2017.

MARQUES FILHO, Ogê. VIERA NETO, Hugo. **Processamento Digital de Imagens**, 1º Ed. Rio de Janeiro: Brasport, 1999.

NAKAMURA, Junichi. **Image Sensors and Signal Processing for Digital Still Cameras**, 3° Ed. New York, US: LTC, 2006.

NEGUS, Christopher. **Linux - A Bíblia - o Mais Abrangente e Definitivo Guia Sobre Linux**, 8° Ed. Rio de Janeiro: Alta Books, 2014.

PAJANKAR, Ashwin. **Raspberry Pi Computer Vision Programming**, 1° Ed. Birmingham, GB: Packt Publishing, 2015.

QT, Qt: **Qt Examples And Tutorials | Qt 5.8 - Qt Documentation.** Disponível em: <htpp://doc.qt.io/qt-5/qtexamplesandtutorials.html. >. Acessado em 02 de dezembro de 2016.

RASPBERRY, Raspberry Pi: **Raspberry Pi - Teach, Learn, and Make with Raspberry Pi.** Disponível em: < https://www.raspberrypi.org/>. Acessado em 15 de novembro de 2016.

OMRON E3F, Omron E3F: **Sensores Fotoelétricos E3F**. Disponível em: < http://industrial.omron.com.br/uploads/arquivos/Datasheet-E3F2.pdf>. Acessado em 20 de outubro de 2016.

OMRON, Omron: **Quality Inspection Solutions.** Disponível em: <https://inspection.omron.us/en/home>. Acessado em 03 dezembro de 2016.

OPENCV MANUAL, OpenCV Manual: **The OpenCV Reference Manual - OpenCV Documentation.** Disponível em: <http://docs.opencv.org/3.0-beta/opencv2refman.pdf>. Acessado em 10 de agosto de 2016.

OPENCV TUTORIALS, OpenCV Tutorials: **The OpenCV Tutorials: Release 2.4.13.0**. Disponível em: <http://docs.opencv.org/2.4/doc/tutorials/tutorials.html>. Acessado em 10 de agosto de 2016.

SNPURRISSIMA: Serra Negra Puríssima: **Controle de Qualidade Puríssima.** Disponível em: < http://www.snpurissima.com.br/#purissima>. Acessado em 11 de abril de 2017.

SOLOMON, Chris; BRECKON, Toby. **Fundamentos de Processamento Digital de Imagens: Uma Abordagem Prática com Exemplos em Matlab**, 1° Ed. Rio de Janeiro: LTC, 2013.

SUAREZ, Oscar D.; CARROBLES, Milagro F.; ENANO, Noelia V.; GARCÍA, Gloria B.; GRACIA, Ismael S.; INCERTIS, Julio Alberto P.; TERCERO Jesus S. **OpenCV Essentials**, 1° Ed. Birmingham, GB: Packt Publishing, 2014.

SNYDER, Wesley E; QI, Hairong; **Machine Vision.** New York, EUA: Cambridge University Press, 2010.

SZELISKI, Richard. **Computer Vision: Algorithms and Applications**. 1° Ed. Washington, EUA: Springer, 2010.

THOMAZINI, Daniel. ALBUQUERQUE, Pedro U. Braga. **Sensores industriais: Fundamentos e Aplicações**, 3º Ed. São Paulo: Erica, 2007.

WIRINGPI, WiringPi: **WiringPi, GPIO Interface Library for the Raspberry Pi.** Disponível em: < http://wiringpi.com/>. Acessado em15 de novembro de 2016.

APÊNDICE

Tabela 1 - Tabela de cálculos das inspeções das embalagens com produto líquido.

Amostragem dos Valores de Pixels (Altura)							
Amostra	**Medição (Pixels)**	**Média Aritm. (Pixels)**	**Desvio (Pixels)**	**Desvio Médio (Pixels)**	**Módulo do Desvio**	**Quadrado do Desvio**	**Desvio Padrão**
1	330	329,1	0,9	0,31	0,9	0,81	1,63
2	330		0,9		0,9	0,81	
3	329		-0,1		0,1	0,01	
4	328		-1,1		1,1	1,21	
5	328		-1,1		1,1	1,21	
6	327		-2,1		2,1	4,41	
7	329		-0,1		0,1	0,01	
8	329		-0,1		0,1	0,01	
9	330		0,9		0,9	0,81	
10	331		1,9		1,9	3,61	
11	333		3,9		3,9	15,21	
12	332		2,9		2,9	8,41	
13	332		2,9		2,9	8,41	
14	330		0,9		0,9	0,81	
15	328		-1,1		1,1	1,21	
16	328		-1,1		1,1	1,21	
17	329		-0,1		0,1	0,01	
18	329		-0,1		0,1	0,01	
19	327		-2,1		2,1	4,41	
20	328		-1,1		1,1	1,21	
21	329		-0,1		0,1	0,01	
22	331		1,9		1,9	3,61	
23	331		1,9		1,9	3,61	
24	332		2,9		2,9	8,41	
25	330		0,9		0,9	0,81	
26	327		-2,1		2,1	4,41	
27	328		-1,1		1,1	1,21	
28	329		-0,1		0,1	0,01	
29	330		0,9		0,9	0,81	
30	330		0,9		0,9	0,81	
31	331		1,9		1,9	3,61	
32	330		0,9		0,9	0,81	
33	330		0,9		0,9	0,81	
34	328		-1,1		1,1	1,21	
35	328		-1,1		1,1	1,21	
36	329		-0,1		0,1	0,01	

37	329		-0,1		0,1	0,01	
38	327		-2,1		2,1	4,41	
39	328		-1,1		1,1	1,21	
40	329		-0,1		0,1	0,01	
41	329		-0,1		0,1	0,01	
42	330		0,9		0,9	0,81	
43	331		1,9		1,9	3,61	
44	333		3,9		3,9	15,21	
45	332		2,9		2,9	8,41	
46	332		2,9		2,9	8,41	
47	331		1,9		1,9	3,61	
48	332		2,9		2,9	8,41	
49	330		0,9		0,9	0,81	
50	327		-2,1		2,1	4,41	
51	328		-1,1		1,1	1,21	
52	329		-0,1		0,1	0,01	
53	328		-1,1		1,1	1,21	
54	328		-1,1		1,1	1,21	
55	329		-0,1		0,1	0,01	
56	329		-0,1		0,1	0,01	
57	327		-2,1		2,1	4,41	
58	328		-1,1		1,1	1,21	
59	329		-0,1		0,1	0,01	
60	329		-0,1		0,1	0,01	
61	330		0,9		0,9	0,81	
62	331		1,9		1,9	3,61	
63	333		3,9		3,9	15,21	
64	332		2,9		2,9	8,41	
65	332		2,9		2,9	8,41	
66	331		1,9		1,9	3,61	
67	332		2,9		2,9	8,41	
68	330		0,9		0,9	0,81	
69	327		-2,1		2,1	4,41	
70	328		-1,1		1,1	1,21	
71	329		-0,1		0,1	0,01	
72	327		-2,1		2,1	4,41	
73	328		-1,1		1,1	1,21	
74	329		-0,1		0,1	0,01	
75	330		0,9		0,9	0,81	
76	330		0,9		0,9	0,81	
77	332		2,9		2,9	8,41	
78	331		1,9		1,9	3,61	
79	332		2,9		2,9	8,41	

80	330		0,9		0,9	0,81	
81	327		-2,1		2,1	4,41	
82	328		-1,1		1,1	1,21	
83	329		-0,1		0,1	0,01	
84	328		-1,1		1,1	1,21	
85	328		-1,1		1,1	1,21	
86	329		-0,1		0,1	0,01	
87	329		-0,1		0,1	0,01	
88	327		-2,1		2,1	4,41	
89	328		-1,1		1,1	1,21	
90	329		-0,1		0,1	0,01	
91	330		0,9		0,9	0,81	
92	330		0,9		0,9	0,81	
93	329		-0,1		0,1	0,01	
94	328		-1,1		1,1	1,21	
95	328		-1,1		1,1	1,21	
96	327		-2,1		2,1	4,41	
97	329		-0,1		0,1	0,01	
98	329		-0,1		0,1	0,01	
99	330		0,9		0,9	0,81	
100	331		1,9		1,9	3,61	
			0,31	-	**1,28**	**2,66**	-

Fonte: Autor (2017).

Tabela 2 - Tabela de cálculos das inspeções das embalagens com produto sólido.

Amostragem dos Valores de Pixels (Área)							
Amostra	**Medição (Pixels)**	**Média Aritm. (Pixels)**	**Desvio (Pixels)**	**Desvio Médio (Pixels)**	**Módulo do Desvio**	**Quadrado do Desvio**	**Desvio Padrão**
1	45833		258,1		258,1	66615,61	
2	45630		55,1		55,1	3036,01	
3	45741		166,1		166,1	27589,21	
4	45610		35,1		35,1	1232,01	
5	45528		-46,9		46,9	2199,61	
6	45434		-140,9		140,9	19852,81	
7	45383	45574,9	-191,9	190,76	191,9	36825,61	249,69
8	45426		-148,9		148,9	22171,21	
9	45539		-35,9		35,9	1288,81	
10	45625		50,1		50,1	2510,01	
11	45630		55,1		55,1	3036,01	
12	45658		83,1		83,1	6905,61	
13	45726		151,1		151,1	22831,21	

14	45779		204,1		204,1	41656,81	
15	45852		277,1		277,1	76784,41	
16	45663		88,1		88,1	7761,61	
17	45685		110,1		110,1	12122,01	
18	45698		123,1		123,1	15153,61	
19	45740		165,1		165,1	27258,01	
20	45637		62,1		62,1	3856,41	
21	45680		105,1		105,1	11046,01	
22	45730		155,1		155,1	24056,01	
23	45829		254,1		254,1	64566,81	
24	45928		353,1		353,1	124679,61	
25	45978		403,1		403,1	162489,61	
26	45996		421,1		421,1	177325,21	
27	46002		427,1		427,1	182414,41	
28	46053		478,1		478,1	228579,61	
29	45986		411,1		411,1	169003,21	
30	45928		353,1		353,1	124679,61	
31	45899		324,1		324,1	105040,81	
32	45885		310,1		310,1	96162,01	
33	45893		318,1		318,1	101187,61	
34	45922		347,1		347,1	120478,41	
35	45951		376,1		376,1	141451,21	
36	45920		345,1		345,1	119094,01	
37	45897		322,1		322,1	103748,41	
38	45880		305,1		305,1	93086,01	
39	45874		299,1		299,1	89460,81	
40	45862		287,1		287,1	82426,41	
41	45851		276,1		276,1	76231,21	
42	45842		267,1		267,1	71342,41	
43	45827		252,1		252,1	63554,41	
44	45775		200,1		200,1	40040,01	
45	45727		152,1		152,1	23134,41	
46	45706		131,1		131,1	17187,21	
47	45689		114,1		114,1	13018,81	
48	45655		80,1		80,1	6416,01	
49	45620		45,1		45,1	2034,01	
50	45602		27,1		27,1	734,41	
51	45583		8,1		8,1	65,61	
52	45529		-45,9		45,9	2106,81	
53	45452		-122,9		122,9	15104,41	
54	45533		-41,9		41,9	1755,61	
55	45575		0,1		0,1	0,01	
56	45448		-126,9		126,9	16103,61	

57	45489		-85,9		85,9	7378,81	
58	45481		-93,9		93,9	8817,21	
59	45522		-52,9		52,9	2798,41	
60	45538		-36,9		36,9	1361,61	
61	45616		41,1		41,1	1689,21	
62	45634		59,1		59,1	3492,81	
63	45671		96,1		96,1	9235,21	
64	45629		54,1		54,1	2926,81	
65	45695		120,1		120,1	14424,01	
66	45700		125,1		125,1	15650,01	
67	45734		159,1		159,1	25312,81	
68	45780		205,1		205,1	42066,01	
69	45789		214,1		214,1	45838,81	
70	45812		237,1		237,1	56216,41	
71	45853		278,1		278,1	77339,61	
72	45896		321,1		321,1	103105,21	
73	45923		348,1		348,1	121173,61	
74	45961		386,1		386,1	149073,21	
75	45910		335,1		335,1	112292,01	
76	45878		303,1		303,1	91869,61	
77	45855		280,1		280,1	78456,01	
78	45813		238,1		238,1	56691,61	
79	45856		281,1		281,1	79017,21	
80	45889		314,1		314,1	98658,81	
81	45918		343,1		343,1	117717,61	
82	45997		422,1		422,1	178168,41	
83	45991		416,1		416,1	173139,21	
84	46001		426,1		426,1	181561,21	
85	46050		475,1		475,1	225720,01	
86	45995		420,1		420,1	176484,01	
87	45928		353,1		353,1	124679,61	
88	45926		351,1		351,1	123271,21	
89	45889		314,1		314,1	98658,81	
90	45842		267,1		267,1	71342,41	
91	45833		258,1		258,1	66615,61	
92	45814		239,1		239,1	57168,81	
93	45805		230,1		230,1	52946,01	
94	45770		195,1		195,1	38064,01	
95	45751		176,1		176,1	31011,21	
96	45729		154,1		154,1	23746,81	
97	45716		141,1		141,1	19909,21	
98	45764		189,1		189,1	35758,81	
99	45788		213,1		213,1	45411,61	

100	45811		236,1		236,1	55743,21	
			190,76	-	**214,19**	**62348,42**	-

Fonte: Autor (2017).