UNIVERSIDADE FEDERAL DA FRONTEIRA SUL
CAMPUS DE CHAPECÓ
CURSO DE CIÊNCIA DA COMPUTAÇÃO

JONATHAN TERHORST RAUBER

SISTEMA DE VISÃO COMPUTACIONAL APLICADO À CONTAGEM DE SEMENTES DE SOJA MOVIMENTADAS POR UMA TRANSPORTADORA DE SUCÇÃO PNEUMÁTICA

CHAPECÓ
2019

JONATHAN TERHORST RAUBER

SISTEMA DE VISÃO COMPUTACIONAL APLICADO À CONTAGEM DE SEMENTES DE SOJA MOVIMENTADAS POR UMA TRANSPORTADORA DE SUCÇÃO PNEUMÁTICA

Trabalho de conclusão de curso apresentado como requisito para obtenção do grau de Bacharel em Ciência da Computação da Universidade Federal da Fronteira Sul.
Orientador: Me. Adriano Sanick Padilha

CHAPECÓ
2019

Rauber, Jonathan Terhorst

Sistema de visão computacional aplicado à contagem de sementes de soja movimentadas por uma transportadora de sucção pneumática / Jonathan Terhorst Rauber. – 2019.
55 f.: il.

Orientador: Me. Adriano Sanick Padilha.
Trabalho de conclusão de curso (graduação) – Universidade Federal da Fronteira Sul, curso de Ciência da Computação, Chapecó, SC, 2019.

1. Visão computacional. 2. Processamento de imagem. 3. Contagem de objetos. 4. Clusterização. I. Padilha, Me. Adriano Sanick, orientador. II. Universidade Federal da Fronteira Sul. III. Título.

E-mail: jonathan.rauber@uffs.edu.br

JONATHAN TERHORST RAUBER

SISTEMA DE VISÃO COMPUTACIONAL APLICADO À CONTAGEM DE SEMENTES DE SOJA MOVIMENTADAS POR UMA TRANSPORTADORA DE SUCÇÃO PNEUMÁTICA

Trabalho de conclusão de curso apresentado como requisito para obtenção do grau de Bacharel em Ciência da Computação da Universidade Federal da Fronteira Sul.

Orientador: Me. Adriano Sanick Padilha

Este trabalho de conclusão de curso foi defendido e aprovado pela banca avaliadora em: 04 de julho de 2019.

BANCA AVALIADORA

Me. Adriano Sanick Padilha – UFFS

Dr. Emilio Wuerges – UFFS

Dr. José Carlos Bins Filho – UFFS

Dr. Fernando Bevilacqua – UFFS

AGRADECIMENTOS

Agradeço...

... aos meus pais José Vidal e Vera, por sempre incentivarem os meus estudos e concederem todo o suporte possível.

... ao meu irmão Thiago, por ser o meu primeiro professor e pela mais pura amizade existente.

... à minha namorada Thalia, por ser minha companheira a graduação inteira, nas horas boas e ruins, pela compreensão nas noites de estudo passadas praticamente em claro e pela atenção, sempre em forma de amor.

... aos professores pelos ensinamentos repassados; especialmente ao Adriano pela confiança, amizade e oportunidades a mim concedidas, ao Milton e à Lucia, todos meus orientadores.

... ao Fabiano pelo suporte, e aos membros da banca avaliadora pelas colaborações neste trabalho de conclusão de curso.

... aos colegas da Ciência da Computação, com os quais passei anos memoráveis.

... aos colegas da Diretoria de Sistemas de Informação da UFFS, pessoas com quem aprendi muito, pessoal e profissionalmente.

... à instituição Universidade Federal da Fronteira Sul, por ser a minha segunda casa durante toda a graduação e maior parte da minha até então carreira profissional.

Levo no coração o carinho de todos.

“Software and cathedrals are much the same – first we build them, then we pray.”

— Samuel T. Redwine, Jr.

RESUMO

Os sistemas de contagem de sementes disponíveis para a indústria ainda são relativamente caros e possuem desempenho limitado. Um empecilho conhecido é o de segmentar os objetos quando estes constituem cenários desafiadores de sobreposição. Este trabalho propõe um sistema de contagem em ambiente controlado, no qual sementes de soja em movimento através de uma turbina de sucção pneumática são inspecionadas. Os objetos são segmentados do fundo, têm seus contornos detectados e calculam-se os seus cascos convexos. A partir da extração da quantidade de defeitos de convexidade, bem como de outras características morfológicas do objeto, é elaborado um clusterizador *K-Means* clássico, que rotula cada contorno com a quantidade de sementes nele existente. Por fim, as sementes são rastreadas pela análise do movimento quadro a quadro. Produzindo acurácia de cerca de 99%, constitui um método de contagem para grandes volumes de baixo custo computacional e com potencial para especialização em cada uma de suas etapas.

Palavras-chave: Visão computacional. Processamento de imagem. Contagem de objetos. Clusterização.

ABSTRACT

The seed counting systems available to the industry are still relatively expensive and have limited counting speed. One known difficulty is the challenge of segmenting objects when they are overlapping each other. This work proposes a counting system having a controlled environment in which soybean seeds are transported in an air conveyor pneumatic turbine and are inspected. The objects are segmented from the background, have their contours detected and their convex hulls are calculated. Thereafter, the number of convexity defects is extracted, as well as other morphological characteristics of the object. Thus, a classic K-Means clusterer is drawn, labelling each detected contour with the actual quantity of seeds. Finally the seeds are tracked by a frame-to-frame motion analysis. Although it produces accuracy that is still insufficient for industrial use, it constitutes a large throughput counting method with low overhead and is open for improvement in each one of its stages.

Keywords: Computer vision. Image processing. Object counting, Clustering.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

A contagem de pequenos objetos é uma atividade importante para a indústria e laboratórios de pesquisa. Nas empresas do agronegócio, por exemplo, é uma prática muito comum para a venda de sementes. Sem um processo de contagem, é necessário que se pesem os grãos para fazer aproximações da quantidade existente. No entanto, devido aos erros causados pela estimativa, adiciona-se uma margem de segurança à quantidade aproximada. Esta margem acrescenta custo a cada lote e consequentemente reduz o lucro, evidenciando a necessidade de um sistema que conte as sementes de forma mais precisa.

Outra aplicação da contagem de sementes é na pesquisa de melhoramento de plantas. O cenário da segurança nutricional das próximas décadas será um desafio. De acordo com Neilsen et al. (2017) "a população global vai superar os 9 bilhões e a demanda por comida vai aumentar mais de 50%", o que torna necessário que a produção seja elevada e muito disso passa pelo melhoramento genético dos grãos. Além disso, através da contagem pode-se extrair indicadores da qualidade das sementes, como pesar um lote e determinar o peso médio da quantia (PAIM et al., 2016).

A automação do processo de contagem ainda possui alto custo, fazendo com que algumas empresas optem por realizar a tarefa manualmente. Assim, um grande número de pessoas é envolvido no processo, gerando probabilidade de erro maior e estendendo o tempo gasto com a tarefa (PAIM et al., 2016). Para reduzir a carga humana associada à contagem de objetos, é desejável que haja uma automação desse procedimento através da visão computacional (KOBAYASHI et al., 2008).

Sistemas que automatizam o processo de contagem tem a desafiadora necessidade de unir dois aspectos: precisão de contagem e bom desempenho computacional. A precisão é uma característica imprescindível, sendo que os contadores automatizados surgem justamente para melhorar o processo que era realizado manualmente. Já o alto desempenho, ou seja, a contagem de grandes volumes de pequenos objetos em um curto espaço de tempo, possibilita que o processo tome menos tempo da linha de produção das indústrias e melhore o escoamento da produção. Para tentar suprir essas necessidades e as de outros problemas de contagem e inspeção, a visão computacional e a biblioteca OpenCV (OPENCV, 2018) vêm sendo exploradas e apresentam-se como potencial solução para a implementação de sistemas robustos (BRADSKI; KAEHLER, 2008).

Os métodos de contagem disponíveis no mercado não possuem o desempenho necessário para se contar integralmente um *big bag*[1] contendo cinco milhões de sementes. As indústrias necessitam de mais velocidade, dado que a capacidade de contagem pelos métodos atuais chega a ser de apenas algumas unidades de sementes por segundo. Por outro lado, a margem de erro deve ser mantida próxima de zero e para isso são necessárias soluções de contagem superiores

[1] Os big bags são sacos industriais feitos em polipropileno com material flexível para transportar produtos em grandes volumes a granel (JOTABASSO, 2019).

aos limitados sistemas atuais.

O campo da visão computacional tem se expandido com a melhoria na tecnologia das câmeras, com a expansão de bibliotecas como a OpenCV e outras do campo da Inteligência Artificial. O aumento do poder de processamento dos computadores também tem tornado possível a execução de aplicações de visão que décadas atrás eram inviáveis para uso em tempo real. Sistemas de visão computacional são aplicados no reconhecimento de pessoas, de assinaturas e de objetos; na inspeção de peças em linhas de montagem; na orientação de movimentos de robôs em indústrias automatizadas; nos carros autônomos; dentre muitas outras tarefas (CONCI; AZEVEDO; LETA, 2008). Neste cenário, a estimativa do número de certos tipos de objetos em uma imagem também é uma das principais aplicações da área (BARBEDO, 2012).

Dadas as variadas aplicações de contagem por visão computacional existentes e o campo de estudo ser relativamente recente, diferentes problemas acabam apresentando algumas características em comum. Assim, métodos de segmentação, processamento e classificação utilizados para resolver um problema de contagem podem incorporar soluções úteis para outros casos (BARBEDO, 2012). Nesse sentido, o presente trabalho busca avaliar um método de contagem de sementes transportadas por uma turbina de sucção pneumática, utilizando técnicas de visão computacional e aprendizado de máquina. O método propõe o emprego de um clusterizador para a solução do problema da sobreposição de objetos, diferentemente das abordagens tradicionais nas quais geralmente essa verificação é tratada na etapa da segmentação.

As turbinas de sucção de ar podem transportar uma quantidade grande de objetos em pouco tempo, além de configurar um sistema robusto que pode ser projetado em paralelo para multiplicar a capacidade de contagem. Esta pesquisa avalia o quanto deste potencial pode ser aproveitado utilizando técnicas de visão computacional, explorando quais as dificuldades e os limites existentes ao usar técnicas de baixo custo computacional para a contagem de sementes de soja.

2 REVISÃO BIBLIOGRÁFICA

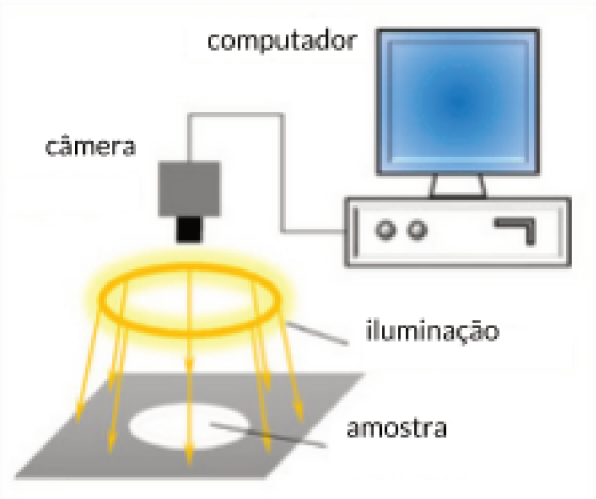
A Visão Computacional é uma área relacionada à análise de imagens, tratando da extração de informações e da identificação e classificação dos objetos presentes nelas. Basicamente, toma imagens como entrada e produz outros tipos de dados como saída (CONCI; AZEVEDO; LETA, 2008). Para tal, também envolve o uso de técnicas de inteligência artificial. Nesse contexto, a visão computacional incorpora a simulação da cognição humana, para "entender" uma cena ou as características de uma imagem.

Nas seções a seguir são apresentados os conceitos utilizados na concepção de sistemas de visão computacional, bem como a OpenCV, uma biblioteca de funções disponível na web para essas aplicações.

2.1 SISTEMAS DE VISÃO COMPUTACIONAL

De acordo com Brosnan e Sun (2004), sistemas de visão computacional em ambiente controlado geralmente são compostos por um sistema de iluminação, uma ou mais câmeras, além de hardware e software especializado (Figura 1). Em Marengoni e Stringhini (2009) é destacado que softwares desse tipo comumente incluem algoritmos e técnicas de processamento de imagens e Inteligência Artificial, de modo a interpretar e classificar as imagens ou realizar a tarefa que for desejada.

Figura 1 – Elementos essenciais de um sistema de visão de máquina.

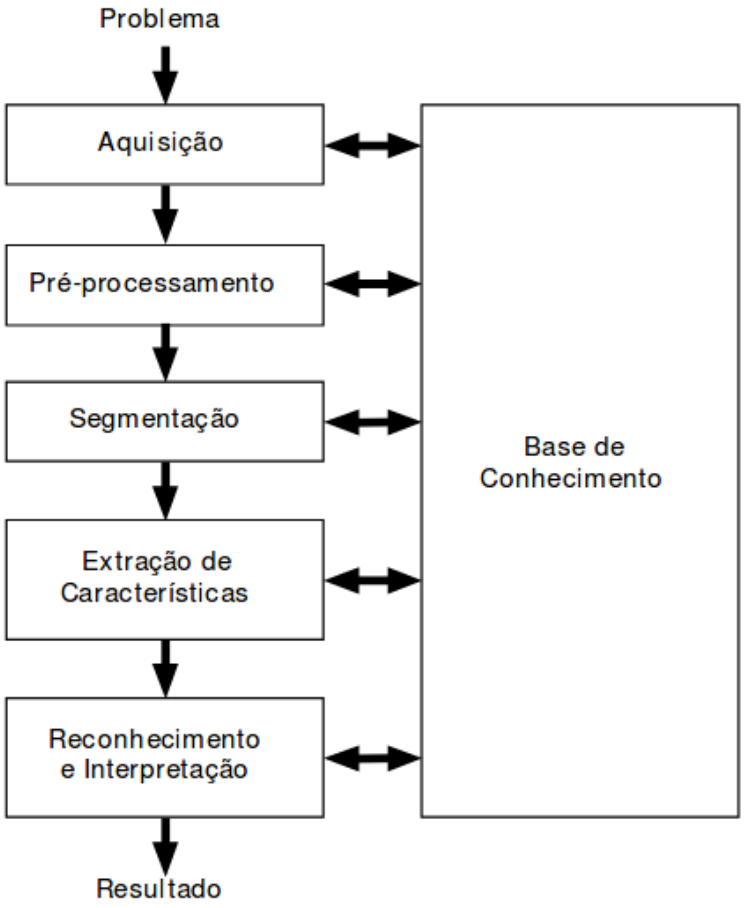


Fonte: Adaptado de Hong et al. (2014)

Um modelo genérico de um Sistema de Visão Computacional é constituído por etapas. A Figura 2 apresenta a organização destas etapas numa execução sequencial. Destaca-se a existência de uma base de conhecimento sobre o problema a ser resolvido (disposta à direita na imagem), que guia o funcionamento de cada etapa do sistema. Ela é responsável por integrar

as diferentes fases, identificando possíveis falhas em cada uma delas e podendo exigir que o processo retroceda para corrigi-las (MARQUES FILHO; NETO, 1999).

Figura 2 – Etapas de um sistema de visão artificial.



Fonte: Reproduzido de Marques Filho e Neto (1999)

2.1.1 Aquisição

A etapa inicial do sistema é a aquisição das imagens. Nesse momento ocorre uma redução de dimensionalidade, que consiste na transformação de uma cena real tridimensional em uma representação bidimensional através de uma câmera. A imagem formada na câmera é expressa na Equação 2.1, na qual f informa a intensidade da luz na coordenada espacial (x, y) e i e r indicam, respectivamente, a iluminação e a refletância no ponto (CONCI; AZEVEDO; LETA, 2008).

$$f(x, y) = i(x, y) * r(x, y) \tag{2.1}$$

De acordo com Conci, Azevedo e Leta (2008), existem dois conceitos importantes relacionados à imagem digital: a amostragem e a quantização de seus pixels (pontos (x, y)).

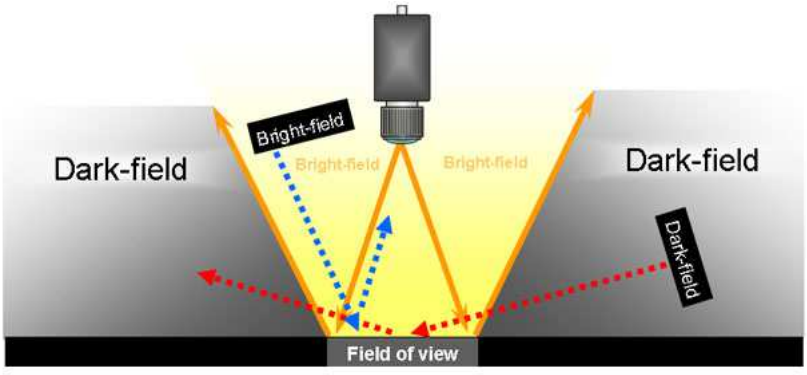
Amostragem é a quantidade de pontos representados da imagem, comumente referida como sua resolução. Aumentando a resolução, aumentam-se os níveis de detalhe de uma imagem. A quantização, por sua vez, indica a quantidade de níveis de tons que podem ser atribuídos a cada pixel. A câmera *Basler acA1300-200um*, por exemplo, possui resolução máxima de 1280x1024 pixels e escala de cor monocromática de até 10 bits, ou seja, pode representar a intensidade de cada pixel numa escala de 0 a 1023.

O posicionamento dos componentes de aquisição do sistema de visão computacional deve ser estratégico, de maneira a reduzir os problemas gerados para as etapas seguintes. Uma configuração indesejada da câmera pode gerar sobreposições excessivas de objetos, o que dificulta a etapa de segmentação e provoca falhas no reconhecimento, por conta de características como texturas e contornos dos objetos encontrarem-se oclusas. Completando, as condições de iluminação da cena também tem papel importante (MARQUES FILHO; NETO, 1999).

2.1.2 Iluminação

A presença de sombra sobre os objetos de interesse, por exemplo, revela um sistema de iluminação mal configurado. Para tanto, existem técnicas que podem melhorar a iluminação de um sistema de visão computacional, como as iluminações *bright field* e *dark field* (MICROSCAN, 2018), apresentadas na Figura 3. Na iluminação *bright field* a fonte de luz é posicionada de forma a fazer com que a luz especular seja refletida na câmera, gerando reflexos visíveis na imagem e alto brilho. Já na iluminação *dark field*, a fonte de luz é posicionada lateralmente, destacando características da imagem como altura dos objetos e sombra.

Figura 3 – Configurações de iluminação *bright field* e *dark field*



Fonte: Reproduzido de MicroScan (2018)

Outra possibilidade é a de posicionar a fonte de iluminação diretamente contra a lente da câmera, fazendo com que a luz incidente seja capturada. Dessa forma, os objetos são iluminados por trás e a imagem capturada apresenta silhuetas bem definidas. As características das configurações de iluminação *bright field*, *dark field* e frontal são sumarizadas na Tabela 1.

Tabela 1 – Sumarização das configurações de iluminação *bright field*, *dark field* e frontal.

	Bright Field	**Dark Field**	**Frontal**
Característica de Luz	Luz especular é refletida na câmera	Luz difusa é refletida na câmera	Luz incide diretamente na câmera
Característica de imagem	Brilho alto	Destaque de bordas e contornos	Separação quase binária de objeto/fundo

Fonte: O autor (2019)

2.1.2.1 Efeito estroboscópico

Efeito estroboscópico é o efeito produzido por uma fonte de luz pulsante iluminando um objeto em movimento. Uma iluminação estroboscópica pode oferecer vantagens na área de visão de máquina, principalmente por remover o efeito *blur*[1] em aplicações de inspeção no qual os objetos devem ser examinados enquanto se movimentam (NOVINI, 1988).

O efeito *blur* nas imagens é causado durante o período em que os fótons de luz estão sendo acumulados pelo sensor da câmera. Este efetua uma espécie de "varredura" para produzir a imagem digital, e o deslocamento que o objeto sofre no tempo desta ação é o que produz o *blur*. O deslocamento D do objeto durante a varredura do sensor é dado pela equação 2.2, onde V é a velocidade do objeto e T é o tempo da varredura (NOVINI, 1988).

$$D = V * T \tag{2.2}$$

Por exemplo, digamos que um objeto move-se na velocidade de 5 metros por segundo e o tempo de varredura da câmera é de $5 * 10^{-3}$ segundos. Nesse caso, a distância percorrida pelo objeto nestes 5 milissegundos será de 2,5 centímetros. Ao sujeitar a câmera a apenas um pulso de luz de 1 milissegundo, o efeito de borrão já será diminuído em cinco vezes, ou seja, o deslocamento do objeto será de $0,5$ centímetros. Portanto, ao utilizar uma fonte de luz que permita pulsos na casa dos microssegundos, é possível produzir uma imagem de boa qualidade para a inspeção, na qual os objetos aparentemente estejam estáticos.

2.2 PRÉ-PROCESSAMENTO

Tendo capturada e à disposição a imagem da cena real desejada, é possível realçar atributos e remover ruídos desta imagem através de "filtros", de modo a facilitar a aplicação dos algoritmos de processamento de imagem em si. Essa etapa é conhecida como pré-processamento ou restauração e realce (CONCI; AZEVEDO; LETA, 2008).

Os ruídos podem ser oriundos do sensor utilizado, da iluminação do ambiente, de condições climáticas e da posição relativa do objeto de interesse e a câmera. Ressalta-se

[1] borrão que prejudica a forma e a nitidez dos objetos em uma imagem, geralmente causado pela rápida movimentação dos elementos capturados (NOVINI, 1988).

que quaisquer interferências que atrapalhem a análise da imagem são considerados ruídos (MARENGONI; STRINGHINI, 2009). A forma discretizada de representação das imagens digitais permite que ruídos na imagem sejam representados através de uma função matemática f. A correção destes ruídos é obtida com a aplicação da função inversa f^{-1} sobre a imagem.

Os filtros são transformações aplicadas em uma imagem, pixel a pixel. A equação 2.3 expressa uma função f aplicada sobre um pixel da coordenada (x, y) da imagem, onde $g(x, y)$ é a intensidade resultante no pixel e T é operador sobre f. Comumente os filtros espaciais são chamados de máscaras, por levarem em consideração pixels próximos à (x, y) para produzir o pixel da imagem resultante (CONCI; AZEVEDO; LETA, 2008).

$$g(x, y) = T[f(x, y)] \tag{2.3}$$

Em operações de convolução, é definida uma máscara $h[i, j]$ que contém os "pesos" de cada pixel vizinho, ou seja, descreve qual a importância de cada um na composição do pixel resultante. A equação 2.4 representa o processo de convolução, com cada pixel $g(x, y)$ resultante do somatório dos valores de seus pixels vizinhos multiplicados pelos respectivos pesos (CONCI; AZEVEDO; LETA, 2008).

$$g[x, y] = \sum_{i=1}^{n} \sum_{j=1}^{m} f[x - i, y - j] * h[i, j] \tag{2.4}$$

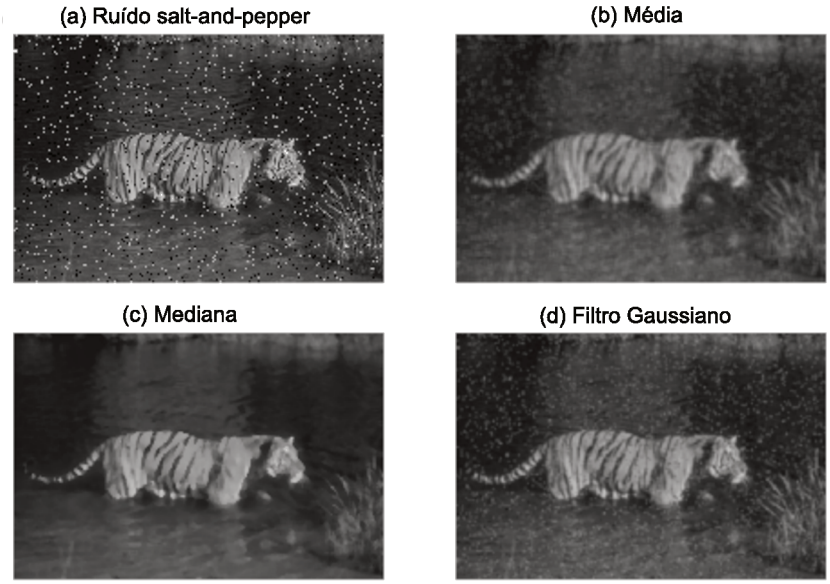
Entre os filtros que usam máscaras de convolução encontram-se os filtros *passa-baixa* e *passa-alta*. Filtros passa-baixa são usados para suavizar imagens e reduzir ruídos, porém fazem a imagem perder nitidez, uma vez que diminuem as variações de contornos e bordas na imagem. Já os filtros passa-alta são usados para realçar detalhes na imagem, destacando bordas, linhas, curvas e regiões que mudam subitamente de intensidade, mas acabam destacando ruídos (MARQUES FILHO; NETO, 1999).

As Figuras 4 e 5 apresentam, respectivamente, a aplicação dos filtros passa-baixa de média, mediana e Gaussiano e dos filtros passa-alta de *Sobel*, *Laplaciano* e *Canny*. Os filtros passa-baixa foram aplicados sobre uma imagem com ruído do tipo *salt-and-pepper*, ilustrando a suavização que esse tipo de filtro gera e mostrando maior recuperação da originalidade do tigre através do filtro Gaussiano. Já os filtros passa-alta foram aplicados sobre a imagem original, livre de ruídos, destacando os locais de maior diferença de intensidade (contornos do tigre e demais elementos do *foreground*).

2.2.1 Filtro Gaussiano

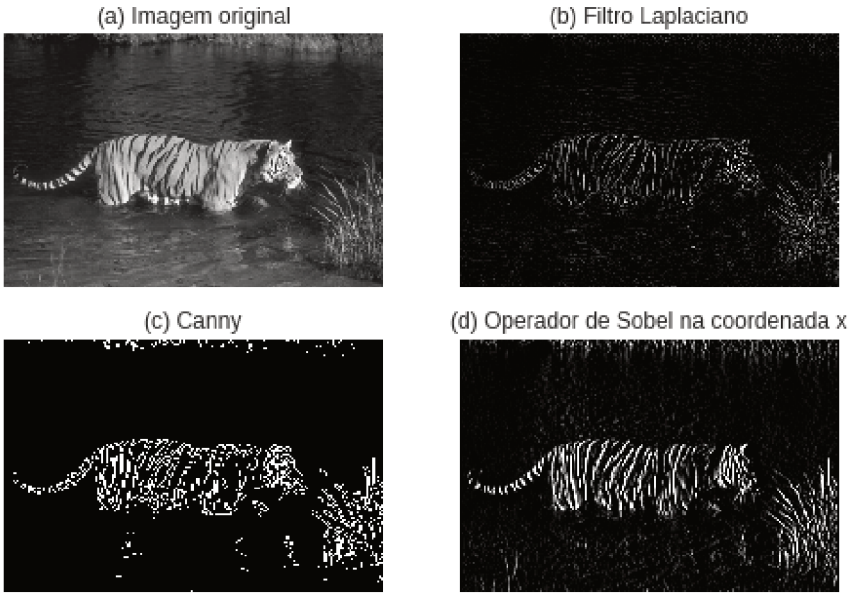
De acordo com Conci, Azevedo e Leta (2008), o filtro de suavização gaussiano é o mais importante filtro passa-baixa. É aplicado para minimizar ou mesmo retirar informações

Figura 4 – Filtros de suavização (passa-baixa).



Fonte: O autor (2019)

Figura 5 – Filtros de atenuação (passa-alta).



Fonte: O autor (2019)

indesejadas dos pixeis da imagem. Tem como referência a função gaussiana contínua (equação 2.5).

$$g(x, y) = \frac{1}{2\pi\sigma^2} e^{\frac{-(x^2+y^2)}{2\sigma^2}} \tag{2.5}$$

Na representação digital em 2D, o filtro gaussiano pode ser aproximado através de uma matriz quadrada ímpar (3x3, 5x5, etc.), com os pesos de convolução mais valorosos ficando

mais próximos do centro da matriz. A equação 2.6 apresenta uma matriz 3x3 que produz uma filtragem gaussiana com desvio padrão igual a 1. A aplicação desta máscara resulta em uma imagem com menor perda de foco se comparada às matrizes que representam curvas gaussianas com desvio padrão maior (CONCI; AZEVEDO; LETA, 2008).

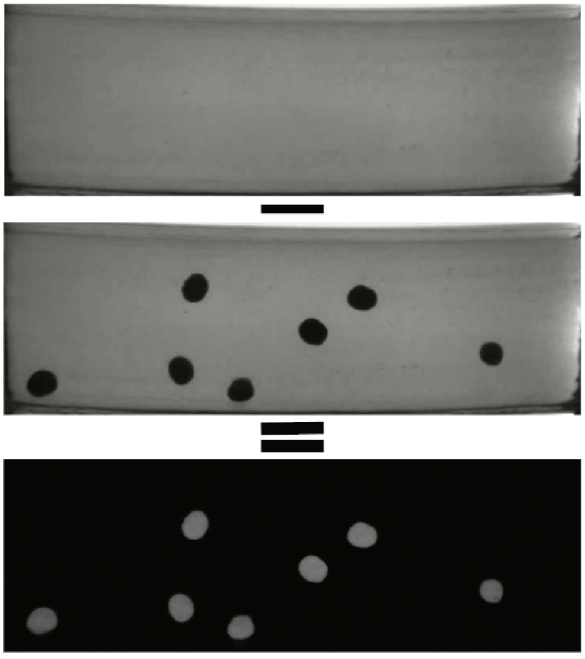
$$z = \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \tag{2.6}$$

2.2.2 Subtração de Fundo

Subtração de fundo é a operação de subtração entre duas imagens, pixel a pixel, utilizada para excluir elementos estáticos ou que fazem parte do fundo, mantendo apenas os objetos do plano de frente que se deseja analisar.

Para realizar subtrações de fundo é necessária a utilização de um "modelo" de plano de fundo, ou seja, um cenário em que sabe-se haver pouca ou nenhuma variação quadro a quadro. Para tal, comparam-se imagens subsequentes com o modelo, numa operação onde subtrai-se a parte comum entre as imagens (o fundo). Os objetos resultantes da operação potencialmente são os objetos do plano de frente, em regiões onde houve maior variação de pixels entre as imagens (BRADSKI; KAEHLER, 2008). A Figura 6 ilustra uma subtração de fundo obtida pela diferença absoluta de dois frames.

Figura 6 – Resultado da diferença absoluta entre imagem e fundo.



Fonte: O autor (2019)

2.3 SEGMENTAÇÃO

Para Marques Filho e Neto (1999) "a tarefa básica da etapa de segmentação é a de dividir uma imagem em suas unidades significativas, ou seja, nos objetos de interesse que a compõe". O objetivo é isolar regiões de pontos da imagem pertencentes a objetos para posterior extração de atributos e cálculo de parâmetros descritivos. Algumas classes de algoritmos são: segmentação por corte, segmentação por detecção de borda e segmentação baseada em crescimento de região (CONCI; AZEVEDO; LETA, 2008).

2.3.1 Limiarização de *Otsu*

A segmentação por limiarização, também conhecida como segmentação por corte ou binarização, é a técnica de eleger um valor de intensidade limiar, para o qual valores abaixo deste são considerados "0" e acima, "1". A dificuldade reside em encontrar um bom valor para o limiar, sendo que nem sempre é interessante que se mantenha na aplicação um valor estático encontrado empiricamente (CONCI; AZEVEDO; LETA, 2008).

O algoritmo de *Otsu* calcula um limiar de separação entre objetos e fundo em uma imagem. Para tal, itera sobre todos os valores de limiares possíveis e calcula a variância para os níveis de pixel de cada lado do limiar. Por fim, escolhe o valor de limiar no qual a variância inter-classe (do primeiro plano com o plano de fundo) é mínima (BARBEDO, 2012). A Figura 7(a) mostra uma imagem 6x6 com 6 tons de cinza e seu histograma. O algoritmo calcula as variâncias inter-classe dos valores de limiar de 0 a 5, elegendo o valor de limiar 3 que segmenta a imagem como apresentado na Figura 7(b).

Figura 7 – Binarização de uma imagem após a limiarização de *Otsu*.

Fonte: Adaptado de Greensted (2010)

2.3.2 Detecção de contornos

Contornos são listas de pontos que representam uma curva numa imagem digital. Podem representar regiões limítrofes de objetos, as bordas, característica de grande valor para aplicações

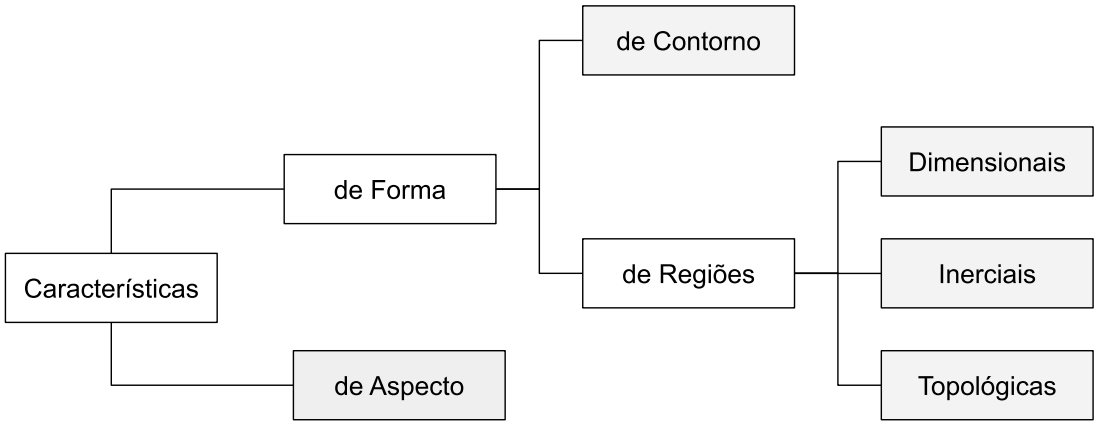
que efetuam reconhecimento de padrões. Segundo Conci, Azevedo e Leta (2008), "o contorno de um objeto é uma região onde a intensidade da imagem muda rapidamente. Se detectarmos esta região, denominada *edge*, conseguiremos discernir os objetos dentro de uma imagem". Existem métodos que baseiam-se na análise do gradiente da função que indica a variação de intensidade dos pixeis na imagem, como o detector de bordas de *Canny* (CANNY, 1987), e métodos de seguimento de borda, como a implementação que pode ser encontrada na OpenCV (SUZUKI et al., 1985).

2.4 EXTRAÇÃO DE CARACTERÍSTICAS

As saídas da etapa de segmentação geralmente deixam descritas características dos objetos, como quantidade de regiões, perímetro, área, luminosidade e textura (CONCI; AZEVEDO; LETA, 2008). A quantidade de características extraídas de uma imagem impacta no desempenho da aplicação, portanto o conjunto definido deve descrever bem o objeto, mas de forma mínima.

A Figura 8 mostra as classes de características que podem ser extraídas de uma imagem digital. Na classe de descritores de forma, por exemplo, podem ser exploradas características de contorno, dimensionais (geometria dos objetos), inerciais (propriedades dos corpos rígidos, e.g. centro de gravidade) e topológicas (número de furos, componentes conectados, etc.). Na outra parte do diagrama, a classe de aspecto representa características como textura e cor dos objetos.

Figura 8 – Classificação dos tipos de características.

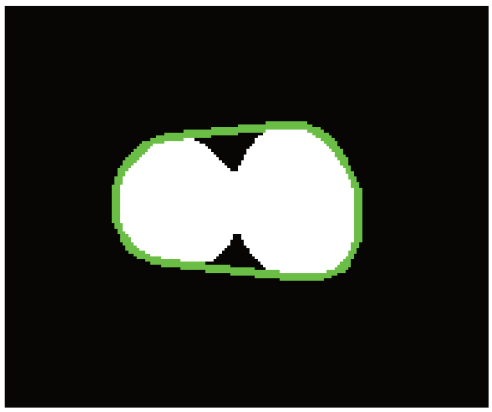


Fonte: Adaptado de Conci, Azevedo e Leta (2008)

2.4.1 Casco convexo

Em sua definição matemática, "casco convexo ou fecho convexo é a intersecção de todos os conjuntos convexos que contém um determinado conjunto" (VANDERBEI et al., 2015). Em uma imagem, o casco convexo pode ser representado como um contorno do objeto, no qual as curvas são sempre salientes ou no mínimo planas. A Figura 9 ilustra o casco convexo de um objeto processado na biblioteca OpenCV.

Figura 9 – Casco convexo (contorno verde) de uma região de pixels brancos.



Fonte: O autor (2019)

2.4.2 Solidez

Uma propriedade dos contornos derivada do casco convexo é a solidez. A solidez denota a razão entre as áreas do contorno e do casco convexo de um objeto (Equação 2.7) (WIRTH, 2001). Quanto mais próximo de 1.0 for esta razão, menor é a proporção de pixels pretos contidos na região do casco. Portanto, a propriedade acaba representando a intensidade dos defeitos de convexidade do objeto, ou, em outra perspectiva, a proporção das regiões côncavas do objeto.

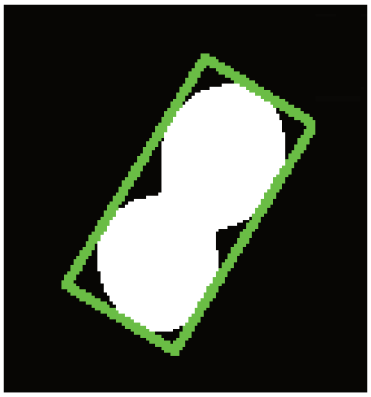
$$solidez = \frac{área\ do\ objeto}{área\ do\ casco\ convexo} \tag{2.7}$$

2.4.3 *Aspect Ratio*

Aspect ratio, para contornos, é a propriedade que denota a razão entre a largura e a altura do menor retângulo que contém o objeto (WIRTH, 2001). A Figura 10 ilustra, em destaque, o menor retângulo que contém um agrupamento de duas sementes.

A utilização do menor retângulo que contém o objeto, ao invés de um retângulo envolvente qualquer, tem por objetivo efetuar o cálculo correto do *aspect ratio* independente dos objetos de interesse serem mais alongados no eixo horizontal ou no eixo vertical.

Figura 10 – Menor retângulo que contém um contorno detectado.



Fonte: O autor (2019)

2.5 RECONHECIMENTO POR CLUSTERIZAÇÃO

O reconhecimento é o processo de atribuição de um rótulo a um objeto, baseado nas características obtidas da imagem na etapa de extração de características. Reconhecer é uma das principais funções da visão computacional. Está relacionado ao reconhecimento de padrões, pois geralmente buscam-se padrões individuais em um determinado objeto de uma imagem, e a presença de um conjunto destes culmina no reconhecimento do objeto como um todo (CONCI; AZEVEDO; LETA, 2008).

Na clusterização, diferentemente de um aprendizado por classificação, não é feita suposição sobre os grupos, não há classes predefinidas e tampouco classes rotuladas nos conjuntos de treinamento. De acordo com Cassiano (2014),

> a Clusterização de Dados é uma técnica de mineração de dados multivariados que através de métodos numéricos e a partir somente das informações das variáveis de cada caso, tem por objetivo agrupar automaticamente por aprendizado não supervisionado os n casos da base de dados em k grupos, geralmente disjuntos denominados clusters ou agrupamentos.

As técnicas de aprendizado não supervisionado são as preferidas em sistemas de contagem que exigem robustez e alta adaptabilidade. Nelas o algoritmo é capaz de se auto organizar, sem que haja fornecimento dos rótulos que identificam a classificação correta de cada entrada (COPPIN, 2010). O algoritmo *K-Means* e os Mapas de *Kohonen* são exemplos de clusterizador não supervisionado e mapa auto-organizável.

2.5.1 Algoritmo *K-Means*

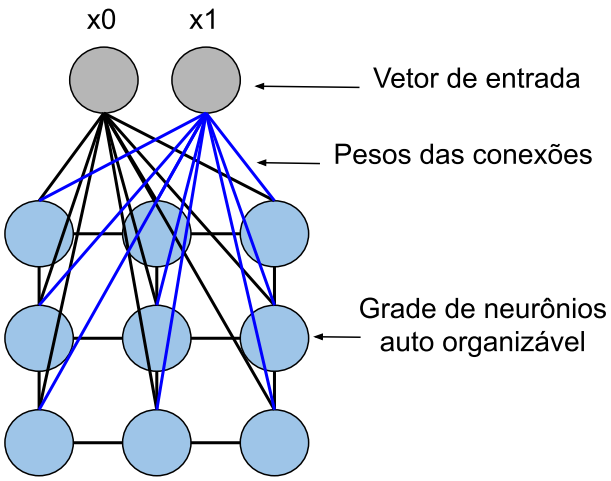
O *K-Means* é um algoritmo de clusterização que utiliza método particional. Divide os dados em k grupos distintos e cada dado deve estar contido em apenas um grupo. Inicialmente são escolhidos k objetos como centro dos grupos, e então os objetos seguintes são divididos entre os k *clusters* de acordo com uma medida de similaridade (CASSIANO, 2014).

A diferença do *K-Means* para outros métodos particionais de clusterização reside na escolha do centro do *cluster*. A similaridade dos dados, geralmente calculada por uma medida de distância como distância Euclidiana ou *Mahalanobis*, tem como referência o centro do *cluster*. No *K-Means*, esse centro é calculado como a média de todos os dados presentes no grupo, ou seja, representa o centro de gravidade do *cluster*. Uma pequena variação do *K-Means*, denominada *K-medoids*, calcula o centro de gravidade e então escolhe o elemento mais próximo desta coordenada para elegê-lo elemento central do cluster (conceito de mediana) (CASSIANO, 2014).

2.5.2 Mapas de *Kohonen*

Um mapa de *Kohonen* é um mapa auto-organizável, não supervisionado, constituído por uma rede neural de duas camadas e aprendizado competitivo do tipo "vencedor leva-tudo". Sua primeira camada é a camada de entrada, que recebe um vetor de dados de tamanho *n* a ser classificado. A segunda camada é organizada em forma de grade, na qual cada um dos seus neurônios estão conectados aos *n* dados da entrada. Cada uma destas conexões possui um peso, que é ajustado durante o treinamento da rede (COPPIN, 2010). A Figura 11 ilustra um mapa de *Kohonen* com vetor de entrada bidimensional totalmente conectado a uma grade de neurônios 3x3, cujas conexões são ponderadas através de pesos.

Figura 11 – Mapa de *Kohonen* com vetor de entrada 2D e grade de neurônios 3x3.



Fonte: O autor (2019)

Para determinar o neurônio vencedor, seus pesos são tratados como um vetor, que é comparado ao vetor de entrada. O neurônio cujos pesos sejam mais próximos do vetor de entrada - dada uma função de ativação como a distância Euclidiana - é declarado vencedor, fornece a classificação de saída do mapa e tem seus pesos atualizados (COPPIN, 2010). Na prática, os neurônios vizinhos até um determinado raio de distância também tem seus pesos atualizados, mas com menor intensidade ao passo que se distanciam do vencedor.

Após a fase de treinamento, que leva algum tempo até os parâmetros de aprendizado convergirem, apresenta rapidez no reconhecimento. Também pode continuar atualizando os pesos dos neurônios durante a tarefa, sem haver a necessidade de recomeçar o processo de treinamento como alguns modelos estáticos (COPPIN, 2010).

2.6 RASTREAMENTO

Segundo Marengoni e Stringhini (2009), "rastreamento é um processo de reconhecer um padrão em uma sequência de imagens". O objetivo do rastreamento é rotular cada objeto detectado, a fim de acompanhá-lo na sequência de imagens e evitar que seja contado mais de uma vez. Esse processo é relativamente lento, mas pode ser otimizado se o movimento do objeto for conhecido. Se são conhecidas a direção e amplitude de deslocamento do objeto na sequência de imagens, é possível aplicar a técnica de rastreamento por predição de localização.

2.6.1 Rastreamento por predição de localização

Ao contrário de abordagens que contam os objetos ao passarem por uma "linha de chegada" na imagem, o rastreamento por predição de localização acompanha o movimento de cada objeto ao longo de todo o seu trajeto pela área visível da imagem. A cada quadro do vídeo, os locais dos contornos detectados são comparados às predições de localização realizadas nos objetos já conhecidos. Então, assume-se que o contorno mais próximo do local predito para o objeto do quadro *n* representa o próprio objeto no quadro *n+1* (NEILSEN et al., 2017).

Podem ser efetuados ajustes no algoritmo de rastreamento para criar sistemas mais dinâmicos. Podem sofrer atualizações periódicas parâmetros como:

- a área mínima que um contorno deve possuir para ser processado, para evitar falsas contagens;
- o deslocamento médio dos objetos frame a frame, para melhorar a predição de localização;
- quantidade média das vezes em que um objeto é identificado, para analisar possíveis duplicatas ou objetos não contabilizados;
- operações sobre outras características topológicas ou de movimento dos objetos.

2.7 OPENCV

A OpenCV (*Open Source Computer Vision Library*) é uma biblioteca de *software* de visão computacional e aprendizado de máquina em código aberto. Seu foco são as aplicações em tempo real, pelo comprometimento com a otimização do desempenho e da memória (BRADSKI; KAEHLER, 2008). Tem tido um papel muito importante na expansão da visão computacional,

pois implementa mais de 2500 algoritmos otimizados, sejam clássicos ou o estado da arte, numa abordagem compreensível e de documentação ampla, contando com uma comunidade de cerca de 47 mil pessoas em todo mundo. Hoje conta com *APIs* nas linguagens *C++*, *Python*, *Java* e pode ser utilizada no *Android* (OPENCV, 2018).

As funcionalidades da OpenCV são divididas em módulos. Por exemplo, algoritmos de processamento de imagem são encontrados no módulo *imgproc*, técnicas de aprendizado de máquina são implementadas no módulo *ml*, aprendizado profundo no módulo *dnn*, entre vários outros existentes. O núcleo da OpenCV foi inicialmente desenvolvido pensando na *CPU*, mas a partir da concepção do módulo *gpu* passou a contar com algoritmos também adaptados para execução nas placas gráficas (PULLI et al., 2012).

A OpenCV viabiliza a construção de um sistema de visão computacional completo. Além de conter funções para leitura, escrita e exibição de imagens, algumas funções importantes para sistemas de visão disponíveis na OpenCV são:

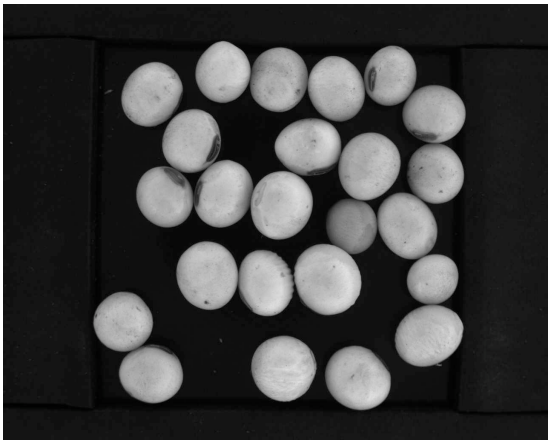
- *cv2.VideoCapture* - Classe para captura de vídeo em arquivos, sequência de imagens ou câmeras;
- *cv2.selectROI* - Permite a seleção da área de interesse na imagem;
- *cv2.absdiff* - Calcula a diferença absoluta entre *arrays* (útil para subtração de fundo em imagens);
- *cv2.GaussianBlur* - Implementação do filtro da Gaussiana;
- *cv2.threshold* - Limiarização. Parâmetros definem se os limiares são estáticos ou adaptativos, bem como o algoritmo a ser executado (*Otsu*, média, dentre outros);
- *cv2.SimpleBlobDetector* - Classe para extração de "bolhas" em uma imagem. Bolhas são grupos de pixeis conectados que possuem uma característica em comum (por exemplo, intensidade na escala de cinza) (**learn_opencv**);
- *cv2.findContours* - Encontra contornos numa imagem binária. Ferramenta útil para detecção e reconhecimento de objetos;
- *cv2.contourArea* - Retorna a área de um dado contorno;
- *cv2.convexHull* - Encontra o casco convexo de um conjunto de pontos;
- *cv2.convexityDefects* - Encontra os defeitos de convexidade de um contorno;
- *cv2.minAreaRect* - Encontra o menor retângulo que contém um contorno.

3 TRABALHOS RELACIONADOS

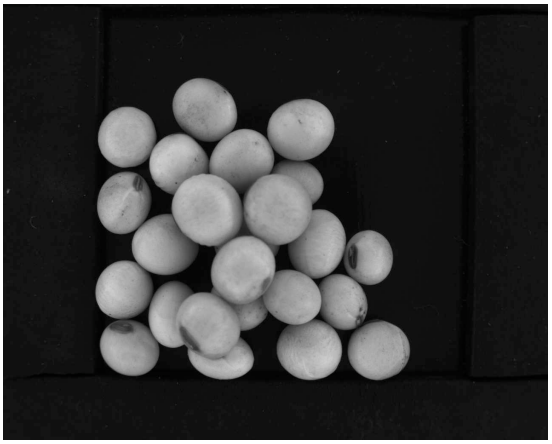
Este capítulo aborda trabalhos de contagem de objetos por visão computacional usando diversos procedimentos metodológicos. O primeiro, de Ni et al. (2016), visa obter um método eficaz de segmentação de objetos circulares, que contorne os problemas causados pela sobreposição parcial dos objetos a serem contados. Baygin et al. (2018) aborda procedimentos semelhantes, mas sem bons resultados perante objetos muito sobrepostos. Por último, Neilsen et al. (2017) propõe uma abordagem de contagem de sementes na qual a aquisição de imagens pode ser efetuada por câmeras mais modestas, encontradas em dispositivos móveis comuns, utilizando também um algoritmo simples mas eficaz para o rastreamento.

Figura 12 – Imagens para teste dos trabalhos relacionados

(a) Sementes de soja separadas



(b) Sementes de soja sobrepostas



Fonte: O autor (2019)

Os trabalhos das seções 3.1 e 3.2 foram reproduzidos sobre as Figuras 12(a) e 12(b). Extraídas as metodologias correspondentes, foram avaliadas sua acurácia e velocidade de execução. As especificações do computador e ambiente usados para a execução dos testes podem ser verificadas no apêndice A.

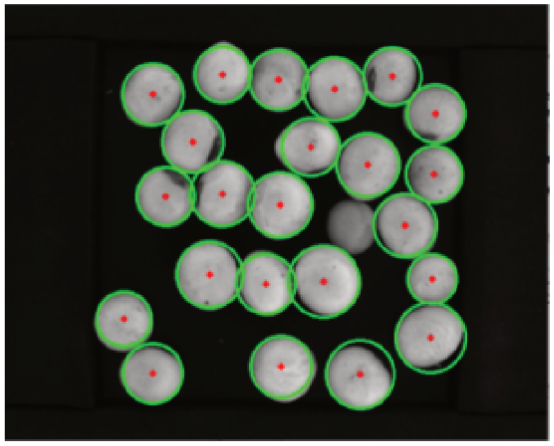
3.1 DETECÇÃO DE OBJETOS CIRCULARES SOBREPOSTOS

O primeiro trabalho relacionado é o "*Automatic Detection and Counting of Circular Shaped Overlapped Objects Using Circular Hough Transform and Contour Detection*", ou "Detecção e contagem automática de objetos circulares sobrepostos usando transformada circular de *Hough* e detecção de contornos" (NI et al., 2016). Esse trabalho busca um método mais eficaz para segmentação de objetos circulares, que resolva os problemas causados pela sobreposição parcial dos objetos a serem contados. Para isso, utiliza uma implementação modificada da Transformada de *Hough* para formas circulares e um algoritmo de detecção de contornos. Segue o passo-a-passo da metodologia empregada:

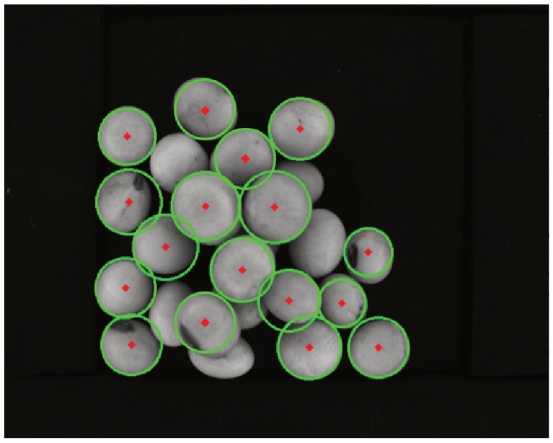
1. Obter a imagem em escala de cinza;
2. Aplicar a técnica *Canny* para detecção de bordas;
3. Destacar as bordas detectadas para contrastar melhor ante o *background*;
4. Aplicar a Transformada Circular de *Hough* para detectar os contornos dos objetos;
5. Desenhar circunferências preenchidas nos contornos encontrados;
6. Aplicar novamente a Transformada Circular de *Hough*, dessa vez para separar mais os objetos;
7. Contar os objetos e desenhar os resultados na imagem original.

Figura 13 – Aplicação da metodologia de Ni et al. (2016)

(a) Sementes de soja separadas.



(b) Sementes de soja sobrepostas.



Fonte: O autor (2019)

Foram reproduzidos testes do algoritmo em sementes separadas e sobrepostas (Figura 13) com parâmetro de raio máximo da circunferência igual à 50. No caso das separadas, obteve acurácia de 95,83% para um tempo de execução de aproximadamente 0,03 segundos. Nas sobrepostas, *Canny* mostrou-se ineficiente para a imagem suprida na entrada, conseguindo bordas ruins e mostrando que exige maior contraste.

3.2 CONTAGEM DE OBJETOS BASEADA EM PROCESSAMENTO DE IMAGEM

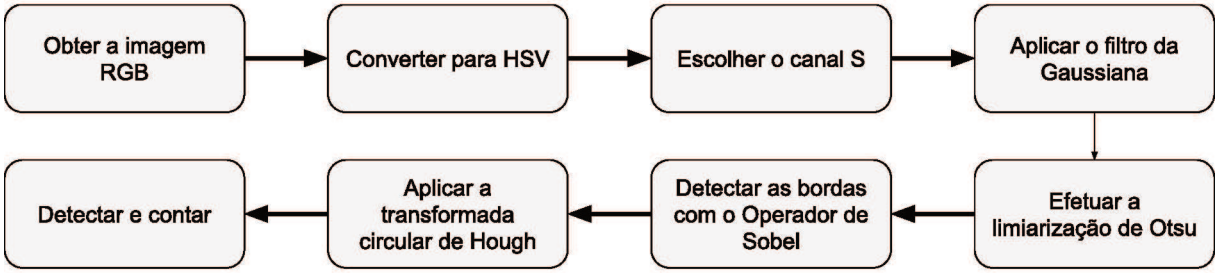
No artigo "*An Image Processing based Object Counting Approach for Machine Vision Application*" ("Uma abordagem de contagem de objetos baseada em processamento de imagem para aplicações de visão de máquina") é descrita a implementação de um sistema capaz de contar objetos independentemente do formato e da cor, de modo a inspecionar uma linha de produção e identificar possíveis faltas de produtos em embalagens (BAYGIN et al., 2018). Como

exemplos de aplicação, teríamos a contagem de garrafas em engradados e de ovos em bandejas. A abordagem baseia-se nas aplicações do limiar de *Otsu* e na transformada de *Hough*.

Os produtos passam por uma esteira e são capturadas imagens através de uma câmera posicionada superiormente, nas quais são aplicadas as técnicas de processamento e o resultado é obtido em tempo real.

A metodologia aplicada possui sequência ilustrada na Figura 14. As imagens são obtidas e convertidas em canais de cor *HSV* (matiz, saturação e intensidade), onde é selecionado o canal de cor corresponde à saturação. Nele, é aplicada a limiarização através do algoritmo de *Otsu*, efetuada uma detecção de bordas pelo Operador de *Sobel* e contados os objetos resultantes a partir da execução da transformada de *Hough*.

Figura 14 – Passo a passo da metodologia proposta por Baygin et al. (2018)

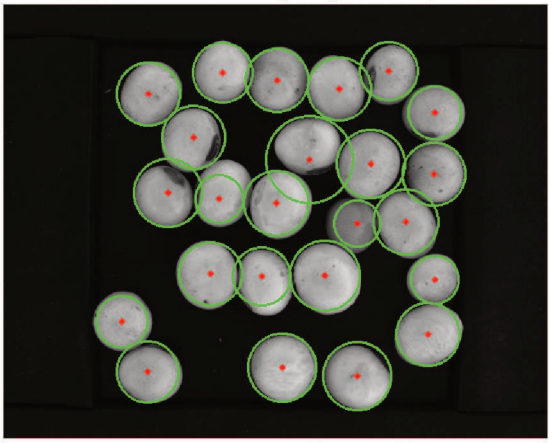


Fonte: Adaptado de Baygin et al. (2018)

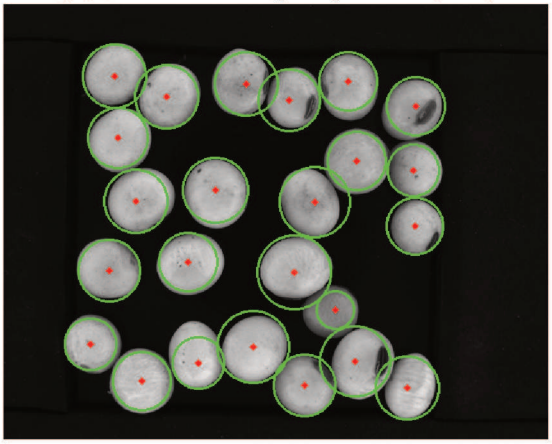
Foram reproduzidos testes do algoritmo em sementes separadas (Figura 15) e sobrepostas (Figura 16). Em ambas, o parâmetro de raio máximo das circunferências detectadas pela transformada de *Hough* foi definido empiricamente como 60. Pelo fato do método inspecionar produtos separados, pode-se verificar que nos testes com sementes separadas há confiabilidade ótima (acurácia de 100%), enquanto nas sobrepostas o desempenho é inaceitável para aplicações reais (acurácia de 73,9%). O tempo de execução médio foi de 0,038 segundos.

Figura 15 – Aplicação da metodologia de Baygin et al. (2018) em sementes separadas

(a) Sementes de soja separadas (nº 1)

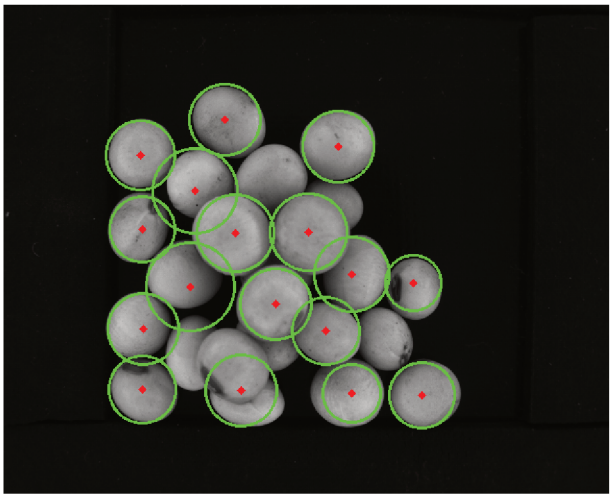


(b) Sementes de soja separadas (nº 2)



Fonte: O autor (2019)

Figura 16 – Aplicação da metodologia de Baygin et al. (2018) em sementes sobrepostas



Fonte: O autor (2019)

3.3 ALGORITMO DINÂMICO PARA CONTAGEM DE SEMENTES

O terceiro trabalho relacionado é "*A Dynamic, Real-time Algorithm for Seed Counting*", ou "Um algoritmo dinâmico e em tempo real para contagem de sementes" (NEILSEN et al., 2017). A aplicação é voltada para a contagem de sementes usando câmeras simples de dispositivos móveis. As sementes são despejadas manualmente em uma rampa improvisada por uma folha de papel, inclinada de 15 a 20 graus e retroiluminada (Figura 17). É possível executar o algoritmo para contar até 300 sementes em 5 segundos de vídeo. Nas melhores condições de resolução (1280x720) e taxa de aquisição (120 *FPS*) o sistema obteve confiabilidade acima dos 99%.

Para cada *frame* é executado o seguinte algoritmo:

1. Converter imagem para escala de cinza;
2. Reduzir ruído usando um filtro de Gaussiana 5x5;
3. Computar a diferença absoluta entre o *frame* atual e o primeiro *frame* (o plano de fundo);
4. Aplicar limiarização binária com limiar=50;
5. Reduzir ruído usando a operação morfológica de abertura;
6. Encontrar os contornos da imagem;
7. Predizer a localização atual de cada semente existente baseado em uma localização prévia e na **taxa de fluxo média**. Se a taxa de fluxo média ainda não é conhecida, usar o valor 0 ou 1, o que significa que a localização predita é a mesma ou próxima à anterior;

Figura 17 – Rampa retroiluminada para fluxo das sementes.

Fonte: Reproduzido de Neilsen et al. (2017)

8. Processar de baixo para cima todos os contornos de tamanho razoável (ao menos 1/4 do **tamanho médio dos contornos**). Se o tamanho médio dos contornos não é conhecido, é usado um valor conhecido de outras execuções do algoritmo. Para cada contorno:

    a) Determine qual local previsto da semente não marcada está mais próximo do centroide do contorno atual;

    b) Se não há semente no raio de quatro vezes a taxa de fluxo média, marque o contorno como uma nova semente e adicione à uma lista. Caso haja, atualize as coordenadas da semente mais próxima usando o centroide do contorno, marque a semente como **visitada** e incremente seu **número de atualizações**;

9. Para cada semente não marcada no passo anterior, incremente a **idade** da semente. Se a idade da semente for incrementada em mais da metade do seu percurso na tela e a idade for maior que o número de atualizações, remova a semente da contagem. A idade máxima da semente é a metade da quantidade de vezes que uma semente deve aparecer na tela [1] mais um;

10. Computar a taxa de fluxo média e o tamanho médio das sementes em pixels, considerando todas as sementes detectadas no *frame*. Continuar o processo indefinidamente, estando as sementes contadas disponíveis na lista.

[1] A quantidade de vezes que uma semente deve aparecer na tela é dada pela altura da tela dividida pela taxa de fluxo média (NEILSEN et al., 2017)

3.4 ANÁLISE

A abordagem visando segmentar objetos de diferentes formatos torna correlato o trabalho de Ni et al. (2016). O método proposto também possui a segmentação como desafio. Portanto, é um dos pontos a ser mais otimizado, de modo a separar os objetos da melhor forma possível com *overhead* mínimo. No entanto, em sua aplicação original efetua testes de contagem em objetos coloridos, que acabam causando diferenças de intensidade mais notáveis nas oclusões.

Embora tenha seu foco na detecção de objetos circulares sobrepostos, a transformada de *Hough* para detecção de objetos com intensidade semelhante de cor não apresenta uma boa solução para a segmentação. Como as capturas no método proposto destacam a silhueta das sementes, inclusive efetuando binarização, é perdida a informação das diferenças de intensidade de cor entre as sementes agrupadas, algo que o método de Ni et al. (2016) preserva.

O método de Baygin et al. (2018) tem outras aplicações como alvo: a de inspeção de objetos separados, como garrafas dispostas em engradados ou ovos em bandejas. Assim, a metodologia não encara problemas de sobreposição, dado que os ambientes de aquisição já efetuam segmentações físicas para os objetos. Dessa forma, verifica-se a efetividade da transformada de *Hough* na detecção dos círculos, mas em situações pouco desafiadoras.

Quanto ao método de Neilsen et al. (2017), há similaridades importantes com o presente trabalho: o fato da aplicação ser voltada para sementes e a elaboração do ambiente controlado para aquisição das imagens. Em comparação à descida das sementes pela rampa, no transporte por sucção de ar as sementes alcançam maior velocidade, havendo de se preocupar primeiramente na aquisição de imagens sem *blur*. Além disso, a *ROI* (região de interesse) é menor devido às proporções da turbina e da iluminação, o que obriga o sistema a ter que processar mais *frames* com as sementes aparecendo em sequência mais breve de imagens.

A ideia do "envelhecimento" das sementes que não são detectadas na sequência de *frames* tem incorporação mais delicada ao se usar transporte por sucção de ar. Sendo que uma semente aparece em cerca de 5 a 7 *frames*, há menos informações para a tomada da decisão de descarte ou não de uma contagem. Em Neilsen et al. (2017) cada semente aparece em dezenas de *frames* durante a descida na rampa, reduzindo a probabilidade de situações que possam acarretar a perda do rastreamento de uma semente.

Adaptando o algoritmo ao problema do transporte por sucção de ar, aproveitam-se de (NEILSEN et al., 2017) a configuração de iluminação, disposta atrás dos objetos contados, e a técnica de rastreamento baseada em predição pelo fluxo médio.

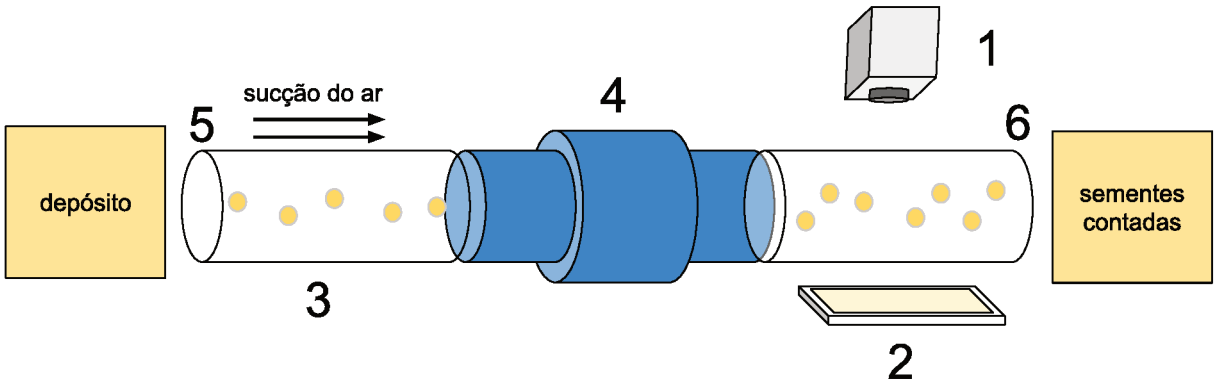
4 DESENVOLVIMENTO

As etapas desenvolvidas constituem dois módulos: a construção do protótipo físico, ou seja, o ambiente controlado de inspeção; e a implementação do software de contagem, formado pelos algoritmos de visão computacional e aprendizado de máquina.

4.1 IMPLEMENTAÇÃO DO AMBIENTE CONTROLADO

A construção do ambiente controlado destina-se a favorecer a aquisição de melhores imagens para o sistema de contagem. O sistema proposto é ilustrado na Figura 18 e é composto basicamente por (1) uma câmera *Basler acA1300-200um*, (2) uma luminária *LED* de sobrepor de 16W produzindo efeito estroboscópico, (3) tubos translúcidos para o transporte das sementes, (4) uma turbina de sucção de ar pneumática e terminais de (5) entrada e (6) saída das sementes. Também foram utilizadas ferramentas de auxílio, como um compressor de 8 bar para o suprimento de ar comprimido à turbina, uma estrutura de metal para impedir a interferência da iluminação externa ao protótipo, um sistema de vibração para dar mais constância ao fornecimento das entradas e uma conexão a um computador utilizando interface *USB* 3.0.

Figura 18 – Ambiente controlado de aquisição das imagens.



Fonte: O autor (2019)

O emprego da turbina de sucção pneumática para realizar o transporte das sementes faz com que o deslocamento destas seja extremamente rápido, tornando quase impossível até mesmo a visualização sua a olho nu. Consequentemente, deve se utilizar uma câmera com alta taxa de aquisição de *frames* associada a uma iluminação que não produza rastros dos objetos nas capturas.

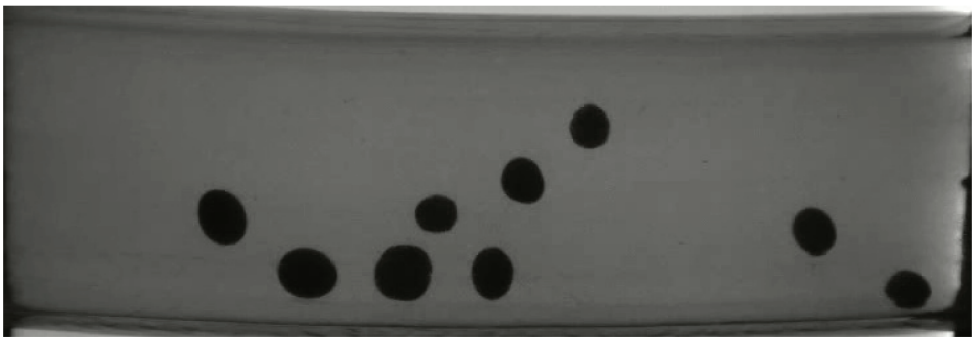
4.1.1 Iluminação e câmera

A câmera utilizada é monocromática, equipada com um sensor *CMOS* e foi configurada para capturar 200 *frames* por segundo em resolução 800x600. Ela monitora continuamente o

terminal de saída da turbina de sucção.

A iluminação é posicionada abaixo do tubo translúcido de saída das sementes, ou seja, a câmera é voltada diretamente contra a lâmpada. Essa configuração faz com que a área da imagem correspondente à lâmpada seja mais clara, tendo bem definida a silhueta das sementes que passam por ali. Esse contraste de sombra e luz, ilustrado na Figura 19, visa facilitar a segmentação de objeto e fundo no software de visão computacional.

Figura 19 – Silhuetas bem definidas pela iluminação projetada atrás dos objetos.

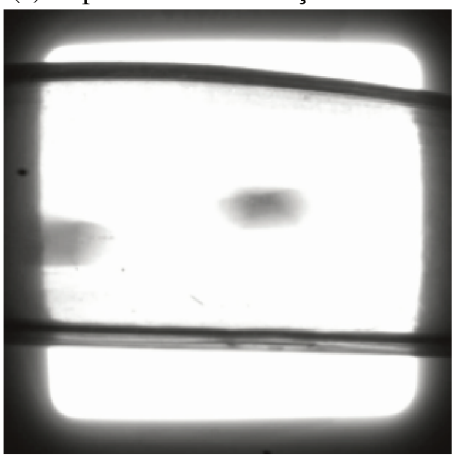


Fonte: O autor (2019)

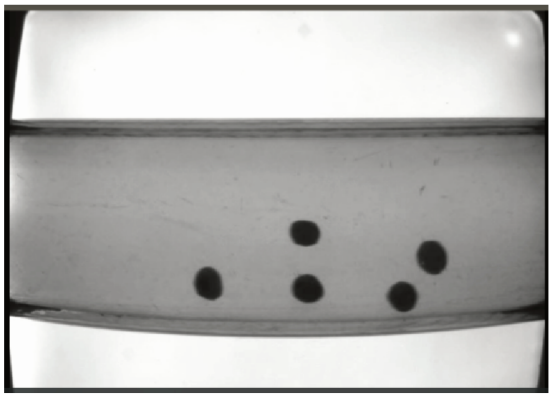
Para impedir que a câmera capture rastros das sementes a iluminação é sincronizada com a taxa de aquisição da câmera, simulando um efeito estroboscópico. Ou seja, a frequência da lâmpada é configurada em 200 Hz, fazendo com que ligue e desligue 200 vezes por segundo em pulsos que a mantém ligada somente 2,5% do tempo. A Figura 20(a) exibe uma captura efetuada com emissão contínua de luz (lâmpada 100% do tempo ligada), enquanto a Figura 20(b) exibe a melhoria obtida com a lâmpada em frequência de pulsos de 200 Hz.

Figura 20 – Demonstração do efeito estroboscópico da iluminação nas sementes.

(a) Captura com iluminação contínua



(b) Captura com emissão de luz na frequência de 200 Hz



Fonte: O autor (2019)

A iluminação externa ao protótipo, ou seja, a luz ambiente do laboratório em que foram realizadas as capturas, diminuiria o contraste entre objeto e fundo obtido com o posicionamento

estratégico da câmera e da lâmpada. Para evitar esse efeito indesejado, a região do protótipo contendo a câmera e a área inspecionada foi isolada com o uso de um invólucro de metal e material plástico de cor preta, mantendo-se apenas uma abertura superior para o posicionamento da câmera (Figura 21).

Figura 21 – Invólucro para amenizar a interferência da iluminação externa.

Fonte: O autor (2019)

4.1.2 Fornecimento de amostras e captura de vídeo

Foi utilizada uma amostra com aproximadamente 2000 sementes de soja posicionadas na bandeja de entrada do protótipo. O sistema de vibração é proprietário do equipamento ESC 2011 (Sanick Equipamentos de Precisão Ltda, Chapecó, Brasil) (Figura 22), o qual possui uma bandeja sob vibração e fazendo com que os objetos em sua superfície movimentem-se de forma circular.

Inicialmente, com as sementes apenas dispostas em uma caixa na região do terminal de entrada, elas eram sugadas praticamente todas de uma só vez. Logo, o uso do sistema de vibração melhorou a constância do fluxo de sementes na sequência de imagens capturadas, permitindo melhor distribuição ao longo do vídeo e a possibilidade de se aumentar ou diminuir o volume de sementes por segundo.

O processo empregado para a obtenção das imagens consistiu em:

a) Selecionar a amostra de sementes de soja e posicionar no terminal de entrada do protótipo;

b) Simultaneamente:

   – Acionar a vibração da bandeja com as sementes, no terminal de entrada;

Figura 22 – Equipamento ESC 2011 utilizado para fornecimento das sementes.

Fonte: Reproduzido de Sanick (2019)

  - Iniciar a execução do software de captura de vídeo;
  - Iniciar o fornecimento de ar pneumático do compressor para a turbina;

c) Após *n* segundos, interromper o fornecimento de ar pneumático e o sistema de vibração;

d) Ao não existirem mais sementes nos tubos de transporte por ar, terminar a execução do software de captura, gravando os *frames* capturados.

Em seguida, as sementes que passaram pela turbina e pela inspeção da câmera foram coletadas em uma caixa no terminal de saída do sistema para verificação. As sementes presentes nesta caixa foram contadas três vezes num equipamento Sanick ESC 2011 em velocidade lenta para se obter a quantidade exata da amostra. Assim, ao final da execução do algoritmo de contagem sobre o vídeo foi possível produzir a análise de resultados do capítulo 5.

4.2 IMPLEMENTAÇÃO DO *SOFTWARE* DE CONTAGEM

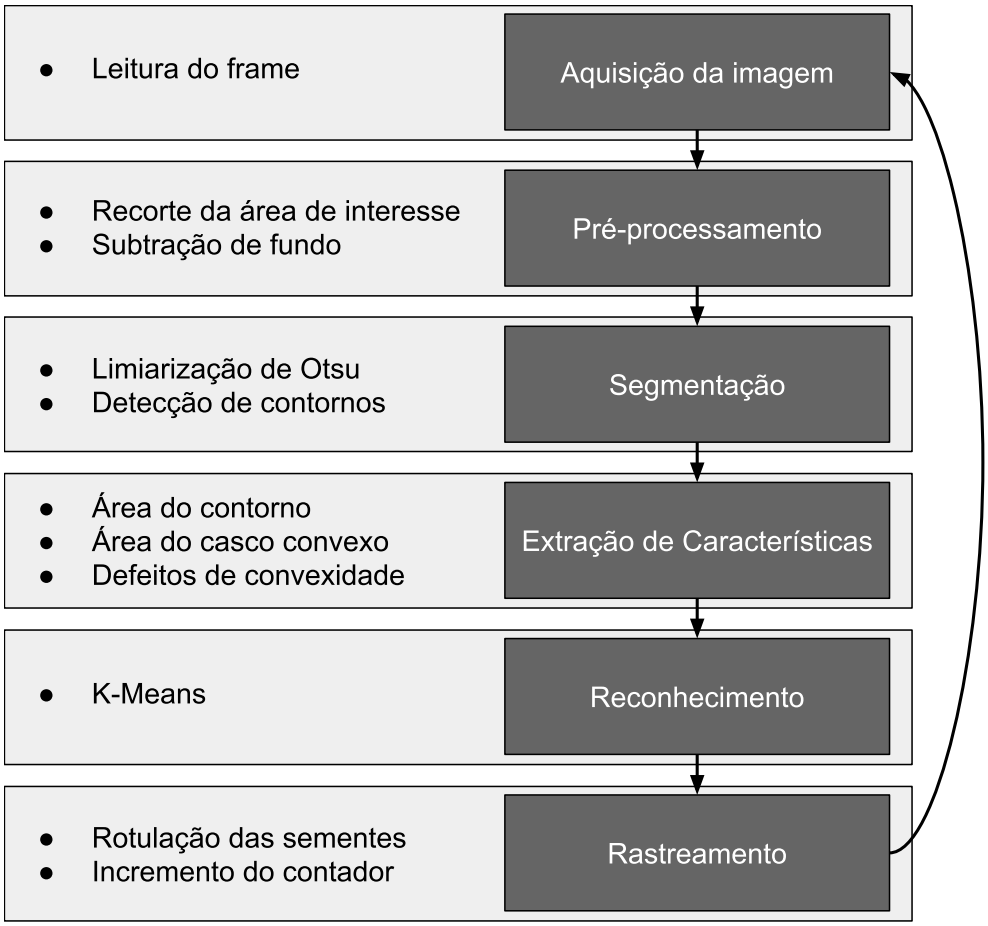
Tendo disponível o vídeo da captura ou uma câmera fornecendo imagens em tempo real, os *frames* são processados no *software* de contagem. Todo *frame* obtido pelo sistema de aquisição é submetido a seis etapas, conforme o diagrama de blocos da Figura 23: leitura da imagem, pré-processamento, segmentação, extração de características, reconhecimento e rastreamento.

As seções subsequentes descrevem as etapas do *software* de contagem.

4.2.1 Pré-processamento

Na etapa de pré-processamento são eliminados os ruídos provenientes da captura da imagem e efetuadas modificações que possibilitam o processamento do *frame* nas etapas posteriores. Para cada imagem carregada da entrada são aplicados os seguintes passos:

Figura 23 – Sequência de operações realizadas sobre cada *frame*.



Fonte: O autor (2019)

- Definição da área de interesse (*ROI*): É delimitada a área retangular que contém o tubo translúcido por onde as sementes são transportadas. Por padrão seleciona a segunda terça parte da imagem, mas também permite que o usuário selecione a área, caso desejar. Essa seleção define a área da imagem a ser cortada (*crop*);

- Subtração de *background*: Aplica-se a subtração de *background* para realçar as sementes presentes no *frame*. Inicialmente, o plano de fundo é um *frame* sem a presença de sementes. À medida em que é processado um novo *frame* sem contornos detectados, o plano de fundo é substituído por este último;

- Suavização das bordas: Efetuada a subtração, aplica-se um filtro de suavização da Gaussiana 7x7 para reduzir os ruídos. Uma máscara 7x7 tem efeito mais forte de suavização do que de uma máscara 5x5, por exemplo, e é suficiente para disfarçar razoavelmente os ruídos nas bordas das sementes. Então, a imagem será preparada para separar o que é objeto do que é fundo através da limiarização de *Otsu*, que adapta o limiar de forma a encontrar a melhor segmentação entre objeto (parte representada pela cor branca) e fundo (parte representada pela cor preta). A distinção binária de cores permite a execução do

algoritmo de detecção de contornos, descrito na próxima subseção.

4.2.2 Segmentação

Sobre a imagem binarizada é aplicado o algoritmo de detecção de contornos. Ele busca por regiões contíguas de pixels brancos, seguindo as regiões de borda - onde há presença de pixels pretos - e retornando numa lista todos os pontos que formam a região limítrofe ao objeto.

O algoritmo detector de bolhas implementado na OpenCV também foi testado. No entanto, apresentou o retorno de poucas características dos objetos detectados, limitando-se ao centroide e tamanho das áreas de pixeis brancos, além de sua execução ser mais demorada se comparada a da detecção de contornos. Dessa forma, optou-se pela utilização do algoritmo detector de contornos para prover as entradas à etapa de extração de características.

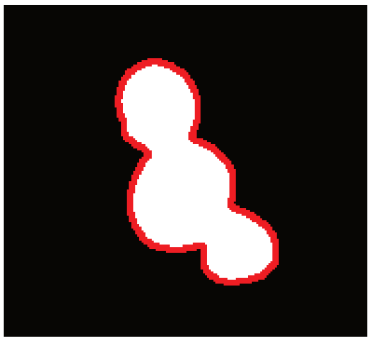
Na etapa de segmentação deste trabalho os objetos agrupados não são separados. A abordagem empregada é a utilização do próprio algoritmo de reconhecimento para identificar as sobreposições, fazendo uso das características obtidas com a detecção dos contornos.

4.2.3 Extração de Características

A extração de características reúne o que for necessário para o algoritmo de reconhecimento. No contexto deste sistema, foca no que é importante para se entender as sobreposições dos objetos.

Por inspeção humana é possível, em muitos casos, verificar que existe uma sobreposição de objetos circulares, como o caso da Figura 24. Essa capacidade vai além: pode inferir a quantidade de objetos agrupados. Isso ocorre porque identificamos as características dimensionais, inerciais e topológicas do objeto ou grupo em questão.

Figura 24 – Sobreposição entre três sementes, causando a detecção de apenas um contorno.



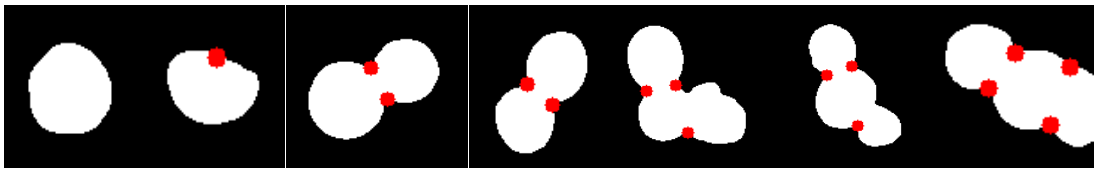
Fonte: O autor (2019)

Uma das características importantes encontrada nas sobreposições é a presença de pontos de defeitos de convexidade, que são obtidos após o cálculo do casco convexo do objeto. A Figura 25 apresenta detecções de sementes, onde são indicados os pontos de defeito de convexidade

significativos existentes nos contornos. Verifica-se visualmente que, quando há menos de dois pontos, geralmente não há sobreposição; quando há dois pontos, há sobreposição de duas sementes; quando há três ou mais pontos, há sobreposição de três ou mais sementes.

Para a configuração de altura da câmera e tamanho das sementes de soja neste projeto, foi definido que defeitos de convexidade significativos têm profundidade maior que 15 pixels. Este parâmetro depende da altura focal, uma combinação entre a altura em que a câmera é posicionada e a regulagem do foco da câmera. Somente os pontos de defeitos dotados desta característica são considerados e levados ao algoritmo de aprendizado.

Figura 25 – Pontos de defeitos de convexidade nos contornos detectados.



Fonte: O autor (2019)

Outras propriedades e características extraídas foram: a solidez, derivada das áreas do casco convexo e do contorno já calculadas; a própria área do contorno, ou seja, a quantidade de pixels que compõe o objeto; e o *aspect ratio*, a proporção entre largura e altura do menor retângulo que contém cada contorno detectado.

4.2.4 Modelo de aprendizado por clusterização

A etapa de reconhecimento utiliza um clusterizador, portanto, um método de aprendizado não supervisionado. A escolha deste tipo de aprendizado faz-se necessária devido ao caráter dinâmico que o algoritmo deve possuir, no qual o reconhecedor deve ser capaz de reagir razoavelmente quanto aos *outliers*[1] que surgirão. Optou-se pelo uso de um *K-Means* clássico, técnica baseada na divisão das amostras observadas em K grupos.

4.2.4.1 Classes

A observação do conjunto de imagens obtidas pôde inferir a necessidade do reconhecimento de ao menos quatro classes de objetos:

1. objetos que consistem em apenas uma semente;
2. objetos que apresentam a sobreposição ou contato de duas sementes;
3. objetos que apresentam a sobreposição ou contato de três sementes;

[1] um *outlier* é um objeto que desvia significativamente do resto dos objetos, ou seja, é um "ponto fora da curva".

4. objetos de morfologia atípica; por exemplo, sementes cuja casca desprende-se e descaracteriza a silhueta convencional (Figura 26).

Figura 26 – Morfologia anômala de uma semente com a casca solta.

Fonte: O autor (2019)

As quatro classes definidas constituem os grupos de ocorrência mais provável. Em casos de agrupamentos com mais sementes, a tendência é de que o aprendizado os associe à classe de 3 sementes devido possuírem características mais semelhantes. Em todo caso essa classificação está incorreta, e espera-se que as consequências sejam tratadas na sequência de imagens através do algoritmo de rastreamento.

Uma semente de soja pode vir a perder sua casca durante o trajeto, ainda mais devido ao movimento forçado da sucção pelo ar. Essas cascas não deveriam entrar na contagem porque, de fato, não representam o objeto alvo da contagem. Porém, a distinção entre casca e semente torna-se complicada através da análise da silhueta: as cascas podem se desprender mantendo sua forma praticamente intacta, podendo ser perfeitamente confundidas com as sementes. A diferença reside no comportamento numa sequência de *frames*: devido a massa das cascas ser bem inferior, elas se movimentam muito mais rápido através da turbina, e, portanto, cabe ao algoritmo de rastreamento detectar a anomalia e desconsiderar sua contagem fazendo a análise do deslocamento.

4.2.4.2 *Dataset* de treinamento

Com base nas implementações da biblioteca OpenCV, foram testados os algoritmos de detecção de bolhas (*cv2.SimpleBlobDetector*) e de detecção de contornos (*cv2.findContours()*) sobre as imagens pré-processadas. A detecção de bolhas é um algoritmo que consiste na verificação de pixels brancos contíguos. Retorna pontos-chave, cujas características revelam a posição do objeto identificado e a sua área. No entanto, em comparação ao algoritmo de detecção de contornos da OpenCV, este dispõe de mais funções a fim de caracterizar o objeto

identificado. Dentre elas, é possível calcular área, casco convexo, defeitos de convexidade, proporção do objeto, etc., além de possuir tempo de execução inferior.

Foram analisados modelos de aprendizado possuindo até quatro características no vetor de entrada: quantidade de pontos de defeito de convexidade, solidez, *aspect ratio* e área do contorno. Todos os contornos detectados nos *frames* têm essas características extraídas e fornecidas ao modelo.

O *dataset* de treinamento foi construído a partir da extração das características dos contornos de todos os *frames* de um vídeo contendo a passagem de sementes. O vídeo contém 1500 *frames* com contornos detectados, fornecendo um total de 4000 registros para o *dataset* de treinamento.

Os dados de cada coluna do *dataset* são normalizados para uniformizar a escala dos atributos. Caso contrário, o atributo 3 (área de contorno), por exemplo, seria muito mais significativo que o atributo 2 (solidez). Isso ocorreria porque a área de um contorno é representado pela quantidade de pixels, podendo valer centenas, enquanto a solidez é uma razão que produz sempre valores entre 0 e 1. Ao se utilizar a distância euclidiana como medida de proximidade dos centros dos *clusters* aos elementos, a solidez teria valor irrisório pois seus baixos valores seriam muito menos significativos ao cálculo do que os valores da área.

A normalização dos dados aplicada é a normalização por desvio padrão (equação 4.1), que considera o valor médio (X) e o desvio padrão para cada coluna do *dataset*. O resultado para cada atributo de registro será negativo caso o valor atual seja menor que a média dos valores, ou positivo caso esteja acima da média.

$$X_{\text{norm}} = \frac{X_i - \overline{X}}{\sigma} \tag{4.1}$$

4.2.4.3 *K-Means*

O *dataset* de treinamento foi fornecido para execuções do algoritmo *K-Means* com $K = 3$, $K = 4$ e $K = 5$, onde as classes apresentadas na seção 4.2.4.1 podem ser associadas a um ou mais *clusters*.

Para que seja tomada a decisão de quantas sementes um contorno representa, é necessário que seja do nosso conhecimento qual classe representa qual quantidade de sementes no resultado do *K-Means*. O *K-Means* simplesmente agrupa os elementos, não sabendo a semântica do que representam os grupos. Porém, ao efetuar uma análise estatística e tendo conhecimento sobre o *dataset*, considera-se haver maior frequência de sementes solitárias e menor frequência do maior número de sementes sobrepostas. Logo, infere-se que o *cluster* que agrupar mais objetos, será o *cluster* que representará os contornos com apenas uma semente.

4.2.5 Algoritmo de rastreamento e contagem

O objetivo do rastreamento é rotular cada contorno detectado com uma ou mais sementes, a fim de acompanhá-las em seu trajeto na área inspecionada pela câmera e evitar que cada uma seja contada mais de uma vez. Neste trabalho, a ideia é uma especialização do algoritmo presente em (NEILSEN et al., 2017), no qual assume-se um sentido único de fluxo das sementes e há conhecimento prévio do movimento.

O algoritmo assume uma distinção importante entre semente e contorno: semente é o objeto que se deseja rastrear, a partir dela fazem-se as predições de localização no *frame* subsequente; já o contorno é simplesmente o resultado da etapa de detecção, o objeto a ser processado para saber a qual(is) semente(s) ele deve ser associado.

A semente é uma classe representada pelos atributos:

a) *ID*: é o rótulo, o número atual da contagem de sementes para fins de visualização;

b) *previous_location*: coordenadas da semente no *frame* anterior;

c) *predicted_location*: coordenadas preditas para a semente, localização onde ela deve estar no *frame* atual;

d) *current_location*: coordenadas reais da semente no *frame* atual;

e) *marked*: *flag* que controla se a semente já foi associada a algum contorno no *frame* atual;

f) *times_associated*: quantidade de vezes que a semente foi identificada na sequência de *frames*;

g) *age*: quantidade de *frames* que uma semente existente não é encontrada no rastreio;

h) *average_flow_rate*: taxa de fluxo médio, denota quantos pixeis a semente deverá deslocar-se até o próximo *frame*.

4.2.5.1 Funcionamento

Tangente ao conceito de semente, o algoritmo de rastreamento mantém uma lista de sementes "ativas", que são as sementes identificadas em *frames* anteriores e que potencialmente devem ser encontradas no *frame* atual. Para cada semente ativa:

1. Faz-se uma predição da localização atual da semente: o novo local (*predicted_location*) será a posição da semente no *frame* anterior (*previous_location*) somada com a taxa de fluxo médio (*average_flow_rate*). Inicialmente o *average_flow_rate* é um valor padrão, empiricamente definido como 100, e só é somado na coordenada x devido ao deslocamento horizontal das sementes. Após cada *frame*, a taxa é substituída pela média dos deslocamentos em pixels de cada uma das sementes detectadas;

2. Associa-se a semente com o contorno mais próximo do local predito: à essa altura, um contorno já está rotulado com a quantidade de sementes que representa. Através de um contador mantém-se o número de sementes que foram associadas ao contorno no *frame* atual. Dessa forma, caso identificado que um contorno contenha *n* sementes sobrepostas, ele pode se associar a sementes no máximo *n* vezes.

Estando associadas as sementes ativas, deve-se verificar se não há sementes novas no *frame* através da análise dos contornos. Se o rótulo que identifica a quantidade de sementes no contorno é maior que o número de sementes associadas a ele no *frame* atual, cria-se uma nova semente. Isto serve tanto para criar as sementes na primeira vez em que aparecem no campo de visão da câmera, quanto para rastrear sementes que se agruparam numa sequência de imagens.

4.2.5.2 Desafios para o rastreamento

Existem várias situações que podem vir a ocorrer durante a trajetória da semente ao longo da largura da *imagem*:

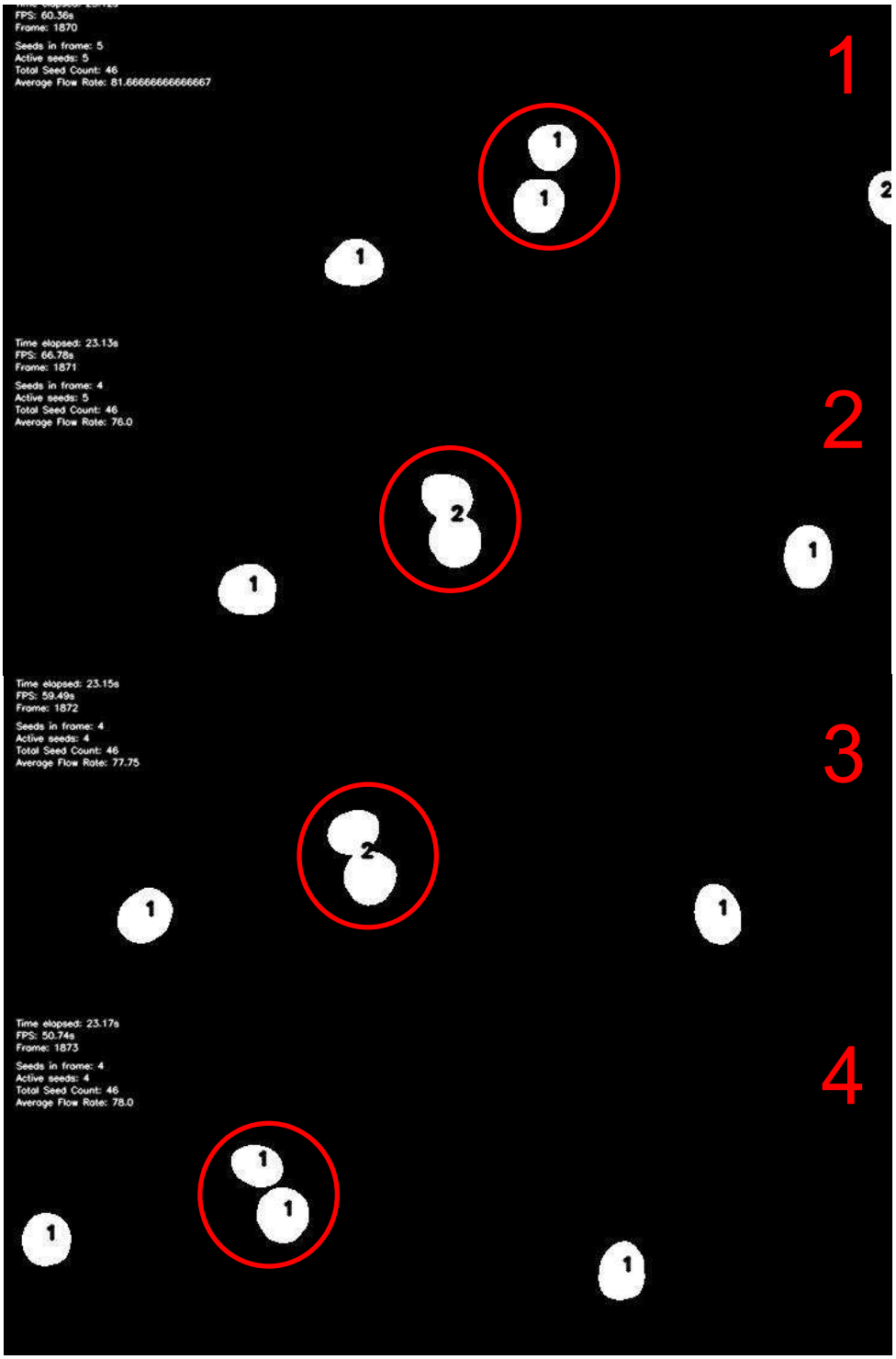
1. sementes que percorrem todo o trajeto sem sobreposições;
2. sementes separadas em um *frame n* que eventualmente se sobrepõe no *frame n* + 1;
3. sementes sobrepostas em um *frame n* que eventualmente se separam no *frame n* + 1.

A Figura 27 mostra, em destaque, a trajetória de duas sementes pela sequência de *frames*, na qual ocorre primeiramente o fluxo solitário de ambas; depois, a sobreposição das duas; no terceiro quadro, o rastreador continua rotulando as duas sementes sobrepostas; no último quadro, ocorre novamente a separação das sementes.

Nas situação 2 e 3, o algoritmo pode ter de tratar uma exceção gerada pelo clusterizador: reconhecimentos incorretos. Sementes extras podem ser geradas incorretamente caso o reconhecedor interprete que há uma quantidade s de sementes no contorno, mas exista uma quantidade real de sementes t onde $t < s$. Logo, também é possível que uma semente deixe de ser identificada no *frame*, caso o reconhecedor indique que uma quantidade s de sementes no contorno analisado, mas exista uma quantidade real de sementes t onde $t > s$.

Quando uma semente nova é gerada sem existir, o rastreador confia que no próximo *frame* o reconhecedor não cometerá o mesmo erro novamente, deixando a falsa semente ser esquecida ao não ser mais associada a nenhum contorno. Já quando uma semente é perdida ao ser agrupada num contorno com mais sementes, o algoritmo continua predizendo seu futuro local e a "envelhece", contabilizando em quantos *frames* a semente não foi identificada. Se a semente num *frame* próximo for associada novamente a um contorno, o rastreio continua normalmente. No entanto, se a semente for envelhecida mais vezes do que a quantidade de vezes que ela iria aparecer no trajeto (largura da tela dividido pelo *average_flow_rate*), ela deve ser descartada.

Figura 27 – Rastreamento numa sequência de imagens.



Fonte: O autor (2019)

5 RESULTADOS E ANÁLISES

Para análise das contagens, foram capturados vídeos em condições diferentes de fornecimento pelo ambiente controlado. Serão analisadas as características de:

- fluxo médio de sementes, na unidade de medida *sementes por segundo*. É obtida através da análise do número dos *frames* inicial e final de contagem, ou seja, primeiro e último *frame* em que há ocorrência de sementes. Assim, o fluxo médio (FM) é dado pela equação 5.1, onde PF é o número do primeiro *frame*, UF é o número do último *frame* e o valor 200 refere-se à taxa de amostragem de *frames* por segundo da câmera;

$$FM = \frac{UF - PF}{200} \tag{5.1}$$

- pico máximo de volume de sementes, que consiste na maior quantidade de sementes detectadas em um *frame* do vídeo.

A Tabela 2 contém as informações dos vídeos utilizados para análise. Cada um possui taxas de fluxo médio diferente, o que é um fator a ser avaliado tanto na análise de tempo de execução quanto pela análise da acurácia atingida. Destaca-se a dificuldade para se produzir vídeos com as características desejadas, principalmente por não haver controle eletrônico da pressurização do ar no ambiente controlado em questão.

Tabela 2 – Características dos vídeos utilizados na análise do método proposto.

ID	**Fluxo médio (sem/s)**	**Pico máximo de volume de sementes**	**Média da área total dos contornos por *frame* (pixels)**	**Quantidade real de sementes**
1	36,9	14	2810	718
2	42,9	6	2586	935
3	55,2	9	1680	1327
4	84,5	16	2496	547
5	97,8	17	1726	932
6	183,7	23	3129	997

Fonte: O autor (2019)

5.1 ANÁLISE DE TEMPO DE EXECUÇÃO

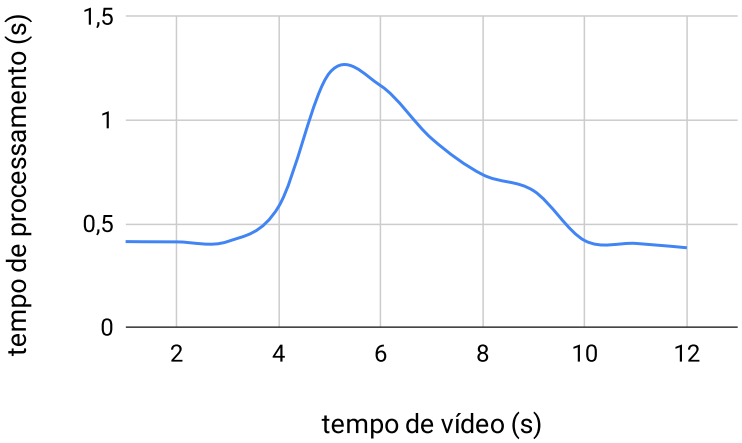
Os gráficos desta seção levam em consideração a quantidade de tempo de processamento que é despendido para se processar um segundo de vídeo da câmera, dado o volume de contornos que ocorre em cada segundo do vídeo, em uma execução nas especificações computacionais do apêndice A. Como o maior processamento é despendido na tarefa de reconhecimento, é utilizado nesta análise o vídeo 6, que possui o maior pico máximo de volume de sementes.

A série do gráfico da Figura 28(a) representa a quantidade de segundos necessárias para se processar cada segundo do vídeo. Para tal, o *software* foi executado 100 vezes e computou o tempo médio de processamento para cada segundo de vídeo. Em comparação ao gráfico da Figura 28(b), nota-se a existência de maior processamento nos mesmos segundos em que há maior ocorrência de contornos. A mesma execução do algoritmo sobre o vídeo ainda produziu os seguintes dados:

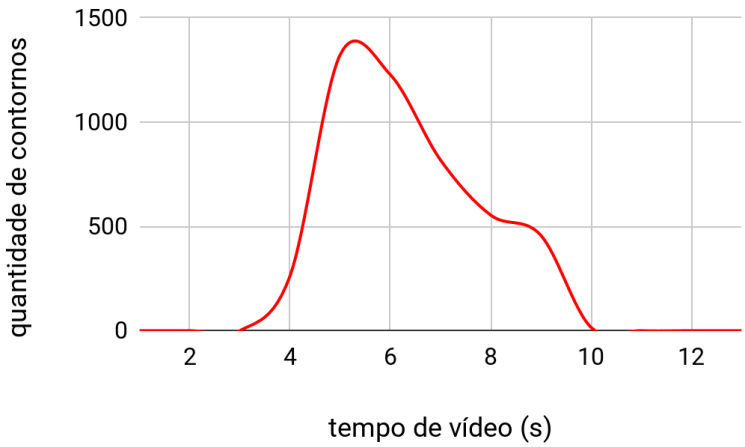
- maior tempo médio gasto para processar um *frame*: 16,34 ms
- menor tempo médio gasto para processar um *frame*: 1,3 ms
- tempo total de vídeo: 13 s
- tempo médio gasto para o processamento: 8,32 s

Figura 28 – Análise do tempo de execução para o vídeo 6.

(a) Segundos despendidos para o processamento



(b) Contornos processados por segundo



Fonte: O autor (2019)

Como a câmera obtém 200 *frames* por segundo, para executar o algoritmo em tempo real, sem considerar a utilização de uma estrutura de *buffer* para as imagens capturadas, cada *frame* deveria ser executado em no máximo 5 milissegundos. No entanto, o tempo máximo gasto para processar um *frame* encontrado na análise revela os gargalos nos pontos onde é atingido o pico máximo de volume de sementes, tornando a estrutura de *buffer* auxiliar necessária para aplicação em tempo real.

5.2 ANÁLISE DO MODELO DE RECONHECIMENTO

A avaliação de um método não supervisionado não é trivial como encontrar o número de erros ou a precisão de um método supervisionado, por não existirem conjuntos de teste e validação rotulados com as classes que cada tupla do *dataset* representa (CASSIANO, 2014). Geralmente, a avaliação é feita através de uma análise da aplicação do modelo ao problema que se quer resolver, e então se decide empírica e probabilisticamente o quanto o método está correto. Neste trabalho, a etapa de reconhecimento é a mais importante para a acurácia das contagens, porque o posterior rastreamento depende da precisa definição da quantidade de sementes em cada contorno para conseguir rastreá-los e, consequentemente, contá-los corretamente. Logo, a acurácia do modelo de reconhecimento é intimamente ligada à acurácia do sistema, avaliada pela análise desta seção.

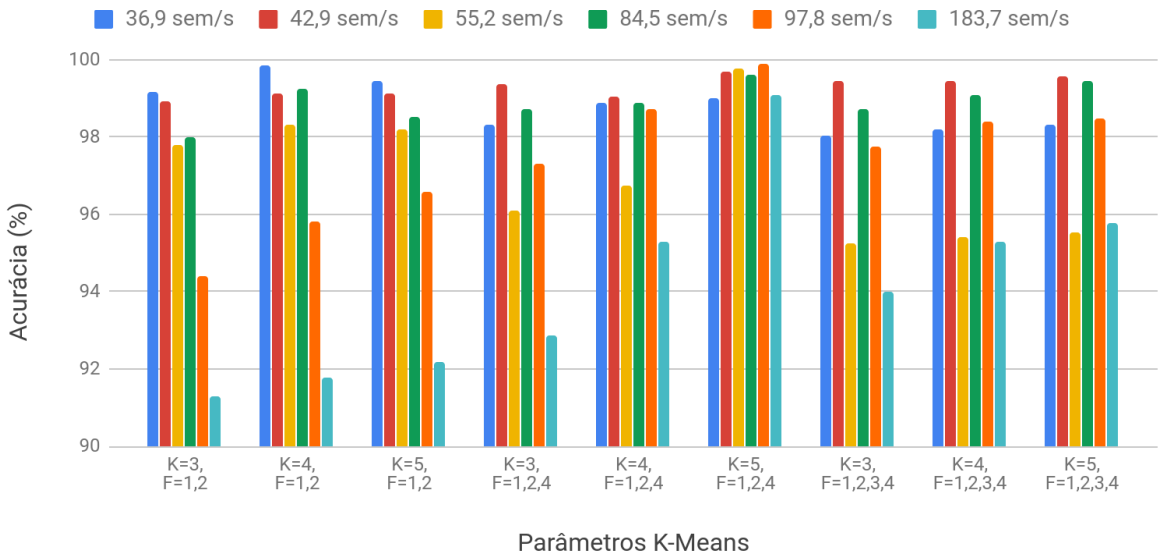
Os gráficos desta seção apresentam a análise efetuada para identificação das características e do número de *clusters* que constituem o modelo que produz a maior acurácia de contagem. Para devido entendimento dos parâmetros utilizados, explana-se que K representa o número de *clusters* formados pelo *K-Means* e F representa o vetor de *features* (características) supridas como entrada ao modelo. Os números indicam as características da forma que segue:

1. Quantidade de defeitos de convexidade;
2. Solidez;
3. Área do contorno;
4. *Aspect ratio*.

No gráfico da Figura 29 é apresentada a acurácia por vídeo de cada configuração de parâmetros para o *K-Means*. Os vídeos avaliados são os já elencados pela Tabela 2, os quais possuem fluxos distintos de sementes por segundo e de aglomeração. Neste contexto, busca-se mensurar a capacidade de manutenção da acurácia e a escalabilidade do método proposto.

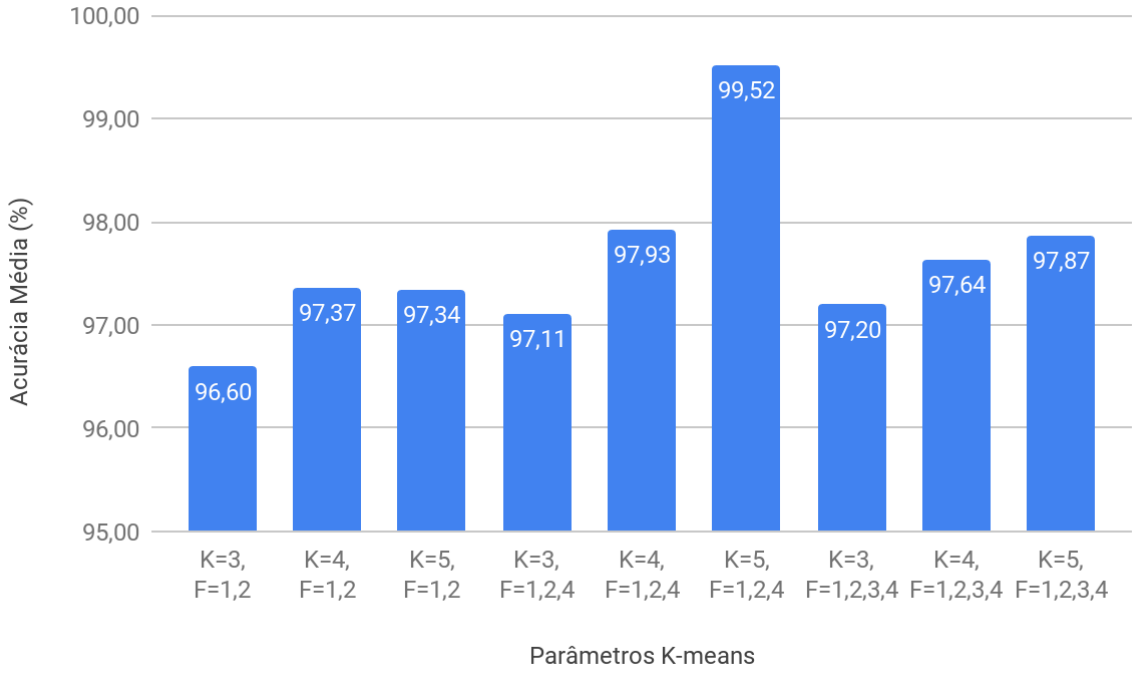
Foram obtidos bons resultados para $K = 5$ e uso das características 1, 2 e 4, onde houve menor prejuízo com o aumento do fluxo de sementes. O gráfico da Figura 30 destaca os 99,52% da acurácia média obtida a partir do conjunto de parâmetros supracitado, valor superior comparado às demais execuções no âmbito dos testes aplicados.

Figura 29 – Acurácia por vídeo dos conjuntos de parâmetros do *K-Means*.



Fonte: O autor (2019)

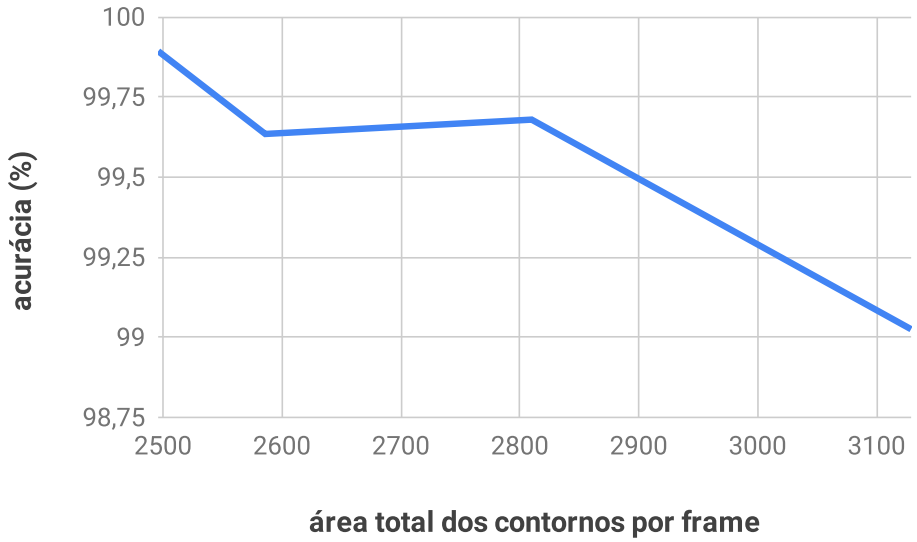
Figura 30 – Acurácia média por conjuntos de parâmetros do *K-Means*.



Fonte: O autor (2019)

Verifica-se que, de modo geral, as configurações tiveram a acurácia declinada com o aumento do fluxo de sementes e consequente aumento do número de aglomerações. Porém, o gráfico da Figura 31 indica que o erro está mais relacionado à intensidade de aglomeração de objetos, dado pela média da área dos contornos detectados nos *frames* dos vídeos.

Figura 31 – Relação acurácia do modelo vs. intensidade de aglomeração de objetos.



Fonte: O autor (2019)

No gráfico da Figura 32(a) é apresentada a clusterização resultante da execução do algoritmo *K-Means* com $K = 5$ e vetor de características [*número de defeitos de convexidade*, *solidez*, *aspect ratio*], que consiste na configuração de parâmetros com a melhor acurácia para o método proposto. É perceptível a maior aglomeração de elementos nas classes 0 e 3, que representam os contornos com apenas uma semente, seguidas pelas ocorrências das classes 1 e 2, que representam duas sementes e, por último, da classe 4 que representa três sementes em um contorno.

Para $K < 5$ a clusterização resultante gera regiões maiores, porém conhecendo as características do problema sabe-se que essas regiões na verdade misturam as chamadas "regiões de fronteira" (vide Figura 32(b)), aumentando o erro por não definir qual a quantidade de sementes que representa melhor os elementos mapeados nestas regiões.

Principais conclusões a respeito dos parâmetros após a avaliação do modelo:

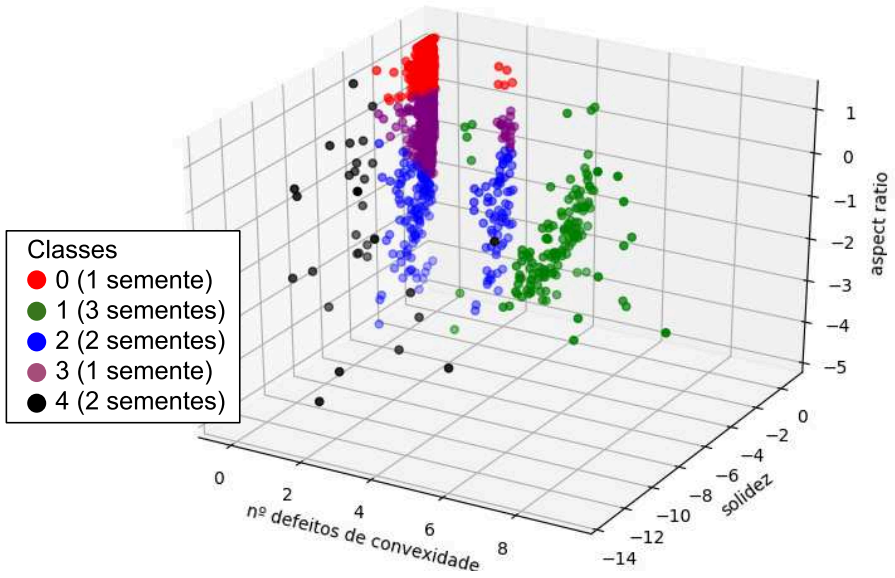
a) A característica 1, quantidade de defeitos de convexidade, é a mais significativa. Não há necessidade de aplicação de métodos como o *PCA* (*Principal Component Analysis*) para se obter esta informação, devido ao conjunto de características extraídas ser pequeno e de fácil análise gráfica (a característica 1 tem visível maior importância na divisão dos *clusters*);

b) Nota-se que a característica 3, área do contorno, torna o método menos dinâmico. Isso porque essa característica depende que as sementes possuam tamanho mais homogêneo, além de exigir rígida regulagem da altura e posicionamento da câmera no sistema de aquisição;

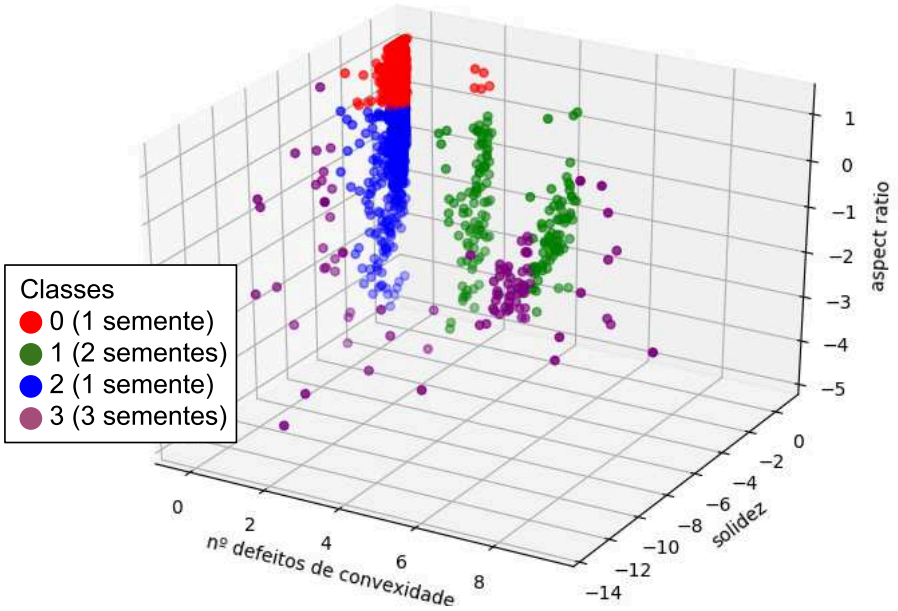
c) A utilização do par de características 1 e 2 somente torna-se insuficiente para enfrentar

Figura 32 – Clusterizações resultantes para K=4 e K=5 com as *features* 1, 2 e 4.

(a) Clusterização para K=5



(b) Clusterização para K=4



Fonte: O autor (2019)

casos mais intensos de sobreposição.

6 CONCLUSÃO

O algoritmo de contagem mostrou-se eficaz quando os objetos se locomoveram na maior proporção do tempo sozinhos. O clusterizador, atuando também como algoritmo de segmentação, não trouxe uma solução escalável para sobreposições, ou seja, apresentou boa performance quando os contornos possuem 1, 2 ou 3 sementes, mas para maiores volumes falha pela possibilidade de haver um número indeterminado de sementes que podem se tocar ou se sobrepor. Não há nada que impeça, numa aplicação dinâmica e real, que a imagem seja constituída por grupos de dezenas de sementes. Positivamente, o algoritmo de rastreamento efetua também o papel de corretor das falhas originadas no clusterizador, através do acompanhamento dos *frames* subsequentes.

A vantagem deste trabalho é a aplicação de algoritmos de baixo custo computacional para a tarefa de contagem e, portanto, que algoritmos mais especializados possam ser incorporados para melhorar a acurácia do método. Como apresentado, o maior problema enfrentado é a oclusão das sementes. Com a exploração das características morfológicas e utilização de um clusterizador, o reconhecimento por aprendizado não supervisionado deste método demonstrou bons resultados em condições menos desafiadoras, onde há grupos com menos sementes agrupadas. Em trabalhos futuros, as regiões de fronteira de *clusters* poderiam ser exploradas com novos clusterizadores e características, buscando melhorar as decisões do modelo.

Através do sistema de iluminação e transporte desenvolvido, o método proposto torna viável a inspeção das sementes em alta velocidade. Esse fator gera empolgação para a especialização do método, vislumbrando que diversas unidades do protótipo poderiam ser colocadas em paralelo para aproveitar o potencial de transporte de grandes volumes de semente, uma carência existente nas linhas de produção atuais.

Em uma comparação aos 99,9% de acurácia exigidos pela indústria, os resultados alcançados não seriam imediatamente viáveis. Entretanto, para a utilização deste algoritmo em trabalhos futuros, poderia ser otimizada a entrada das sementes nos tubos para controlar mais a sucção, de maneira a limitar a abertura do terminal de entrada. Por outro lado, para condicionar menos a acurácia do modelo à configuração do ambiente controlado, é mais indicado que sejam utilizados algoritmos especializados de segmentação e reconhecimento, combatendo as falhas não tratadas pelo método proposto.

REFERÊNCIAS

BARBEDO, Jayme Garcia Arnal. A review on methods for automatic counting of objects in digital images. **IEEE Latin America Transactions**, IEEE, v. 10, n. 5, p. 2112–2124, 2012.

BAYGIN, Mehmet et al. An Image Processing based Object Counting Approach for Machine Vision Application. **arXiv preprint arXiv:1802.05911**, 2018.

BRADSKI, Gary; KAEHLER, Adrian. **Learning OpenCV: Computer vision with the OpenCV library**. [S.l.]: "O'Reilly Media, Inc.", 2008.

BROSNAN, Tadhg; SUN, Da-Wen. Improving quality inspection of food products by computer vision—-a review. **Journal of food engineering**, Elsevier, v. 61, n. 1, p. 3–16, 2004.

CANNY, John. A computational approach to edge detection. In: READINGS in computer vision. [S.l.]: Elsevier, 1987. p. 184–203.

CASSIANO, Keila Mara. Análise de Séries Temporais Usando Análise Espectral Singular (SSA) e Clusterização de Suas Componentes Baseada em Densidade. **Pontifıcia Universidade Católica do Rio de Janeiro**, 2014.

CONCI, Aura; AZEVEDO, Eduardo; LETA, Fabiana R. **Computação gráfica, Teoria e prática**. [S.l.]: Elsevier Brasil, 2008. v. 2.

COPPIN, Ben. **Inteligência Artificial**. [S.l.]: LTC, 2010. v. 1.

GREENSTED, Andrew. Otsu thresholding. **The Lab Book Pages**, v. 17, 2010.

HONG, Hanmei et al. Visual quality detection of aquatic products using machine vision. **Aquacultural Engineering**, Elsevier, v. 63, p. 62–71, 2014.

JOTABASSO. **Embalagens big bags são tendência de mercado na comercialização de sementes**. [S.l.: s.n.], 2019. Disponível em: <https://www.jotabasso.com.br/noticia/embalagens-big-bags-sao-tendencia-de-mercado-na-comercializacao-de-sementes>. Acesso em: 10 jun. 2019.

KOBAYASHI, Takumi et al. HLAC approach to automatic object counting. In: IEEE. BIO-INSPIRED Learning and Intelligent Systems for Security, 2008. BLISS'08. ECSIS Symposium on. [S.l.: s.n.], 2008. p. 40–45.

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

MARQUES FILHO, Ogê; NETO, Hugo Vieira. **Processamento digital de imagens**. [S.l.]: Brasport, 1999.

MICROSCAN. **Bright Field and Dark Field lighting**. [S.l.: s.n.], 2018. Disponível em: <https://www.microscan.com/en-us/resources/know-your-tech/bright-field-and-dark-field-lighting>. Acesso em: 30 set. 2018.

NEILSEN, Mitchell L et al. A Dynamic, Real-time Algorithm for Seed Counting, 2017.

NI, Jianjun et al. Automatic detection and counting of circular shaped overlapped objects using circular hough transform and contour detection. In: IEEE. INTELLIGENT Control and Automation (WCICA), 2016 12th World Congress on. [S.l.: s.n.], 2016. p. 2902–2906.

NOVINI, Amir R. Fundamentals of Strobe Lighting for Machine Vision. In: INTERNATIONAL SOCIETY FOR OPTICS e PHOTONICS. AUTOMATED Inspection and High-Speed Vision Architectures. [S.l.: s.n.], 1988. v. 849, p. 149–158.

OPENCV. **About**. [S.l.: s.n.], 2018. Disponível em: <https://opencv.org/about.html>. Acesso em: 20 set. 2018.

PAIM, Paulo Fernando Escobar et al. Protótipo de contadora de sementes usando resistores dependentes de luz e arduino. **Anais do Salão Internacional de Ensino, Pesquisa e Extensão**, v. 7, n. 2, 2016.

PULLI, Kari et al. Real-time computer vision with OpenCV. **Communications of the ACM**, ACM, v. 55, n. 6, p. 61–69, 2012.

SANICK. **Contador Eletrônico de Sementes e Grãos ESC2011**. [S.l.: s.n.], 2019. Disponível em: <https://www.sanick.com.br/Produtos/Contador_Eletr%C3%B4nico_de_Sementes_e_Gr%C3%A3os_ESC_2011>. Acesso em: 30 mai. 2019.

SUZUKI, Satoshi et al. Topological structural analysis of digitized binary images by border following. **Computer vision, graphics, and image processing**, Elsevier, v. 30, n. 1, p. 32–46, 1985.

VANDERBEI, Robert J et al. **Linear programming**. [S.l.]: Springer, 2015.

WIRTH, Michael A. Shape analysis and measurement, 2001.

APÊNDICE A – ESPECIFICAÇÕES COMPUTACIONAIS PARA TESTES

As principais configurações do computador utilizado para a análise dos trabalhos relacionados e, posteriormente, do método proposto foram:

a) Memória: 2x8GB DDR4 2400MHz Hitachi;

b) Processador: Intel Core i7-7700T @ 2.90GHz x 8;

c) Gráficos: Intel HD Graphics 630 (*Kaby Lake* GT2);

d) Armazenamento SSD LiteOn CV3-8D256 256GB;

e) Sistema Operacional: Ubuntu 17.10;

f) OpenCV versão 3.4.5;

g) *python* versão 3.6.7;

h) *numpy* versão 1.16.2;

i) *scikit-learn* versão 0.20.3;

j) *pypylon* versão 5.2.0.