VISÃO COMPUTACIONAL NO ENSINO DA DERIVADA: UM ESTUDO SOBRE A MEDIÇÃO DE OBJETOS UTILIZANDO IMAGENS

COMPUTER VISION IN THE DERIVATIVE TEACHING: A STUDY ON THE OBJECTS MEASUREMENT

Alana Cavalcante,[1] Vinícius Fernandes,[2] Thiago Silva Figueiredo,[3] Jamisson Jader Moraes Pereira Júnior,[4] Gustavo Tupini Silveira,[5] Rangel Henrique Miranda Trindade[6]

RESUMO

Cálculo Diferencial e Integral é uma das principais disciplinas do ciclo básico em um curso de Engenharia, uma vez que desenvolve o raciocínio lógico, possibilitando a resolução rápida de problemas do cotidiano. O ensino da derivada é geralmente abordado de maneira teórica. Entretanto, em áreas de cunho tecnológico – como a Engenharia de Computação –, a aplicação dos conceitos de derivadas não é algo trivial. Os sistemas computacionais funcionam com base em dados discretos, o que impossibilita utilizar as técnicas como é visto dentro das salas de aula nos primeiros anos do curso. A falta de aplicabilidade provoca uma desmotivação no aluno, visto que ele não consegue entender o porquê estudar aquele determinado conteúdo. Pensando nisso, utilizando a aprendizagem baseada em projetos, foi proposto um trabalho em grupo para elaboração de um projeto que envolvesse algum tema da Engenharia de Computação e a utilização da derivada. Esse artigo visa a estudar o processamento de imagens, mais especificamente a detecção de bordas proposta por John Canny, buscando demonstrar aos alunos de Engenharia de Computação formas de se aplicar o cálculo diferencial como uma ferramenta nos algoritmos. A fim de apresentar o potencial dessa ferramenta vinculada ao processamento de imagem, foi desenvolvido um algoritmo capaz de utilizar as bordas de uma imagem para realizar medições de objetos em tempo real.

Palavras-chave: Ensino de Derivada; Detecção de Borda; Engenharia de Computação; Metodologia Ativa de Ensino.

ABSTRACT

Differential and Integral Calculus is a logical reasoning approach that enables a rapid modeling of everyday problems. The teaching of the derivative is usually approached theoretically. However, in technology areas such as Computer Engineering, the application of these concepts is not trivial. Computer systems operate on the basis of discrete data, which makes it difficult to use some techniques as seen in the classroom during the first years. The lack of applicability decreases the motivation in the students, since they can not understand the reason to study that particular content. Thinking about this, using project-based learning, a group work was proposed to elaborate a project that involved some subject of Computer Engineering and the use of the derivative. This article aims to study the image processing, more specifically the edge detection proposed by John Canny, trying to demonstrate to Computer Engineering students ways of applying differential calculus as a tool in algorithms. In order to present the potential of this tool linked to image processing, an algorithm that uses the edges of an image was developed to perform objects measurements in real time.

Keyword: Active learning; Edge detection; Computational engineer; Derivative Teaching.

1 Professora Doutora na UFOP; alana.decea@ufop.edu.br

2 Discente UFOP; vinicius.fernandes1@aluno.ufop.edu.br

3 Discente UFOP; thiago.figueiredo@aluno.ufop.edu.br

4 Discente UFOP; jamisson.junior@aluno.ufop.edu.br

5 Discente UFOP; gustavo.tupini@aluno.ufop.edu.br

6 Discente UFOP; rangel.trindade@aluno.ufop.edu.br

INTRODUÇÃO

A disciplina de Cálculo Diferencial e Integral está presente no ciclo básico dos cursos de Engenharia, sendo uma das ferramentas matemáticas mais importantes para esses cursos, uma vez que possibilita a solução de inúmeros problemas das mais diversas áreas de atuação do engenheiro.

Devido à sua importância, Cury (2009) aponta que cerca de 42% dos artigos publicados nos anais do COBENGE (Congresso Brasileiro de Ensino de Engenharia), entre 1992 e 2001, tinham como foco o ensino e a aprendizagem de Cálculo. Mais recentemente, Wrobel, Zeferino e Carneiro (2013) apresentaram resultados de uma investigação bibliográfica sobre os artigos relacionados ao ensino de Cálculo I publicados no mesmo congresso entre 2003 e 2012.

Segundo De Jesus Brito e Cardoso (2018), esta disciplina é ministrada sempre da mesma forma, com a mesma metodologia, os mesmos exemplos, sem se levar em consideração a natureza do curso. O método expositivo faz com que alguns alunos utilizem grande parte de sua capacidade de abstrair para compreender determinados conceitos, mas outros têm dificuldade em construir seu próprio conhecimento e compreender a importância da matemática na prática de engenharia.

Como forma de verificar a experiência dos alunos com relação ao Cálculo I, foi realizada uma pesquisa por meio de um formulário eletrônico em seis universidades na região de João Monlevade, totalizando 214 respostas. Com essa pesquisa concluímos que a percepção dos alunos em relação à aplicabilidade da derivada era maior à medida que o curso avançava. Entretanto, uma quantidade significativa dos alunos conhecia apenas os fundamentos teóricos do uso da derivada, não sabendo a aplicabilidade dos conceitos em seu curso.

É importante ressaltar que os altos índices de reprovação e evasão no ciclo básico dos cursos de Engenharia estão relacionados às disciplinas de Cálculo. Rafael (2017) demonstra preocupação em relação a esses elevados índices, uma vez que esse fracasso pode levar ao abandono do curso e pode influenciar o estudante na decisão de não ingressar em um curso de graduação no qual a disciplina seja obrigatória.

Pode-se facilitar o processo de aprendizagem, alternando a exposição do conteúdo com aplicações concretas através de recursos visuais, viabilizando um estímulo para a construção do conhecimento. De acordo com Silva e Ferreira (2009), o uso de *softwares* computacionais se apresenta como uma ferramenta auxiliar na construção de conceitos e aplicações relacionados ao ensino de matemática.

Outra estratégia para tornar o processo de aprendizagem mais efetivo é a utilização da metodologia ativa, na qual o aluno é o protagonista do seu aprendizado. Essa metodologia é baseada no pensamento de Confúcio: "o que eu ouço, eu esqueço; o que eu vejo, eu lembro; o que eu faço; eu compreendo". Essa metodologia desenvolve no aluno o trabalho em equipe, a comunicação, a criatividade e a autonomia. De acordo com Lima, Andersson e Saalman (2017), a aprendizagem baseada em problemas é uma das estratégias de metodologia ativa mais utilizadas no contexto da Engenharia.

Este presente artigo tem como objetivo descrever e avaliar uma experiência de uso da Aprendizagem Baseada em Projetos (PBL) para ensinar o Cálculo I para estudantes de Engenharia, que aconteceu no primeiro semestre de 2018. Esta experiência é baseada no trabalho de Cuzzuol et al. (2018).

Especificamente, iremos tratar da utilização da derivada na Engenharia de Computação. Existem subáreas que possibilitam uma aplicação visual desse conceito, permitindo uma maior compreensão por parte dos estudantes. Uma delas é um recurso computacional que realça fatores de interesse em vídeos ou fotos visando à obtenção de dados, conhecido como processamento de imagens. Neste trabalho, ainda iremos enfatizar a detecção de bordas, uma vez que se utiliza em uma de suas etapas o cálculo da derivada.

Um dos métodos para a detecção de bordas, conhecido como Método Canny, é um conceito consolidado. Isso se justifica pelo fato de *softwares* que trabalham com o processamento de imagens já possuírem funções próprias para a sua aplicação. Utilizando uma biblioteca do tipo *open source* (código aberto com permissão para modificações) chamada OpenCV, é pos-

sível implementar algoritmos sem que haja a necessidade de técnicas avançadas na programação.

Para a obtenção dos resultados e escrita do texto adotamos os seguintes critérios: 1) Não exigir conhecimento que ultrapasse a grade curricular do primeiro e segundo períodos de Engenharia de Computação; 2) Ter aplicação visível. Além disso, os conceitos aqui abordados englobam conhecimentos computacionais simples para auxiliar a compreensão da derivada pelo aluno.

O artigo é organizado da seguinte maneira: na primeira seção abordamos conceitos iniciais sobre processamento de imagem; descrevemos o método Canny para detecção de bordas, explicando cada etapa de seu procedimento com a utilização do Cálculo na segunda seção; na terceira seção desenvolvemos um algoritmo capaz de detectar bordas e fazer medições de objetos; os resultados são apresentados na última seção.

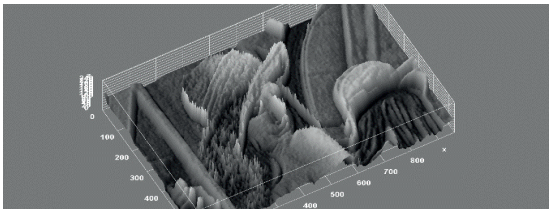
PROCESSAMENTO DE IMAGEM

O processamento de imagem é um conjunto de operações que separam pontos de interesse e resultam em um conjunto de valores que podem ou não serem representados em uma imagem de saída (MARENGONI; STRINGHINI, 2009). Com a utilização de imagens é possível extrair informações sobre as características físicas e geométricas básicas dos objetos, tais como dimensão, área, perímetro e a forma.

Para que um sistema computacional reconheça os elementos de um ambiente ou objeto fotografado é necessária a digitalização de cada ponto. Ou seja, transforma-se o contexto contínuo de "infinitos" átomos em um sistema discreto, isto é, em finitos pontos, denominado *Picture Element,* pixel formando uma imagem. Para isso são necessários dois procedimentos: primeiramente a amostragem e em seguida a quantização. Segundo Gonzalez e Woods (2010), a amostragem consiste na captura de pontos igualmente espaçados no ambiente fotografado, de forma que esses pontos sejam coletados gerando um conjunto de localizações, ou seja, pares ordenados. Já a quantização transforma os valores contínuos da intensidade de cores em valores inteiros contidos em um intervalo, por exemplo, 0 (zero) representando o preto e 255 representando o branco.

Desta forma, pode-se representar a imagem como uma função bidimensional denotada por $F(x,y)$, tal que x e y são coordenadas de um plano, obtidas pelo processo de amostragem, e F expressa a intensidade de cinza, proveniente da quantização (GONZALEZ E WOODS, 2010), como pode ser visto na Figura 1. Com isso, torna-se possível a alocação das intensidades de cada ponto em uma matriz, permitindo aplicação de operações matriciais sobre a fotografia.

Figura 1: Representação das dimensões da imagem e a intensidade de cada pixel



Fonte: Acervo dos autores.

As imagens são, normalmente, representadas no sistema RGB (do inglês *Red*, *Green and Blue*). Isto significa que cada pixel é uma aproximação resultante das intensidades de vermelho, azul e verde. A união das três cores em cada pixel forma uma imagem de três bandas (BIASI et al., 2002). Segundo de De Queiroz e Gomes (2006), a imagem de três bandas pode ser representada pela equação:

$$F(x,y) = F_r(x,y) + F_g(x,y) + F_b(x,y), \quad (1)$$

onde F_r, F_g, F_b representam respectivamente a intensidade das bandas vermelha, verde e azul.

Para a maioria das imagens, adotou-se que a intensidade do pixel igual a 0, em todas as bandas, representa o preto, enquanto 255 representa o branco. Todas as outras cores são variações nesse intervalo.

Assim, o processamento se inicia transformando a imagem formada pelas bandas RGB em tons de cinza, conforme verifica-se na Figura 2.

Figura 2: Representação de uma imagem nas bandas RGB e em escala de cinza

Fonte: Acervo dos autores.

Essa transformação é necessária para analisar a imagem apenas como uma variação de cinza. Segundo a documentação oficial dos desenvolvedores da biblioteca OpenCV, a transformação aplicada é baseada em uma média ponderada das componentes RGB de cada pixel, dada pela fórmula:

$$F(x,y) = 0.299 * F_r(x,y) + 0.587 * F_g(x,y) + 0.114 * F_b(x,y), \quad (2)$$

Com essas definições, torna-se possível aplicar o método para a detecção de borda desenvolvido por John Canny. A imagem de saída resultante da aplicação desse método é do tipo imagem binária, pois os elementos variam entre 0 ou 1, de forma que 1 representa a presença de borda em um determinado pixel. Por meio desse método, a Figura 3 (a) é processada, resultando em uma imagem representante das bordas – Figura 3 (b).

Figura 3: Representação da borda de uma imagem

Fonte: Acervo dos autores.

Sendo assim, os tópicos seguintes terão como objetivo explicar as técnicas para se realizar o processamento do Método de Canny. Além disso, iremos demonstrar a aplicação de derivada nesse método.

MÉTODO CANNY PARA DETECÇÃO DE BORDAS

Canny apresentou a sua primeira ideia sobre detecção de borda em 1983, em seu artigo denominado *A variational approach to edge detection* (CANNY, 1983). Nesse artigo são apresentados três parâmetros que Canny considera como sendo um filtro ideal para a detecção de borda: 1) o filtro deveria ter a menor taxa de erro possível, de forma a retornar uma maior quantidade de bordas verdadeiras de uma imagem. Essas bordas são provenientes da variação de cor entre um objeto e um segundo plano; além disso, 2) a posição dos pixels marcados como borda deveria estar ao centro da borda; e, por fim, 3) a borda deveria ter como espessura o tamanho de um pixel.

É importante ressaltar que existem vantagens desse método para medições. Isso se justifica uma vez que as imagens de borda dos objetos devem ter o mínimo de bordas falsas, pois essas podem provocar dimensões que não são reais no corpo. Além disso, a espessura da borda e a sua localização em relação ao corpo também podem provocar erros nas medições. Portanto, uma borda com espessura de um pixel e localizada próxima ao centro da variação de cor é o ideal para o nosso sistema.

Para retornar uma imagem binária, representando as bordas de objetos, o método Canny realiza quatro tipos de operações sobre a fotografia original: 1) a suavização de ruídos; 2) aplicação da derivada sobre a imagem; 3) seleção das derivadas com intensidade máxima; 4) filtragem dos valores considerados bordas.

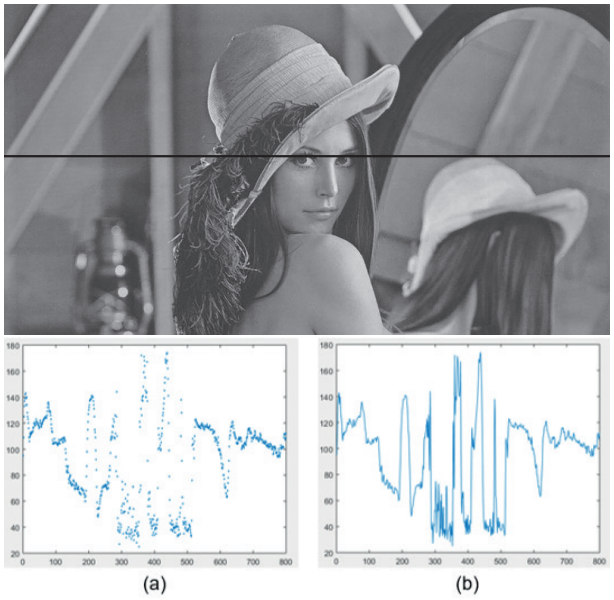
Suavização Gaussiana

A primeira operação aplicada sobre a imagem é conhecida como Suavização Gaussiana, que tem como objetivo diminuir os ruídos sobre a imagem e facilitar a detecção de bordas. Por ruídos, entende-se as variações bruscas da intensidade de cor que não correspondem à realidade. Nas imagens, o ruído aparece como pontos em superfícies que deveriam ser lisas.

No contexto de detecção de borda o ruído é um problema, pois sua ocorrência dificulta na determinação do valor da derivada sobre a imagem. Ou seja, uma figura com ruídos possui diversas variações bruscas de cores, o que implica em diversas bordas falsas. A Figura 4 exemplifica a atuação dos ruídos. Selecionan-

do-se os valores das intensidades dos pixels na direção da linha destacada, plotou-se um gráfico da posição em relação à intensidade desses pixels, obtendo-se o gráfico representado na Figura 4(a). Para facilitar a visualização, o gráfico da função foi tratado como contínuo, resultando em Figura 4(b).

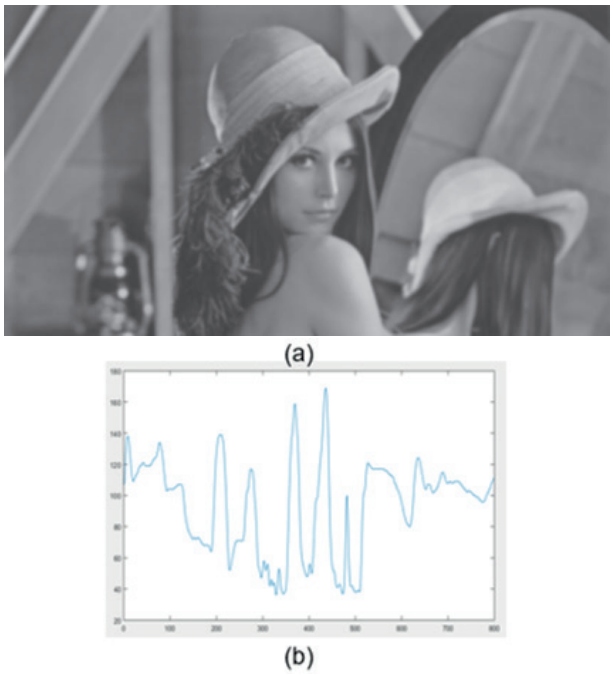
Figura 4: Atuação do ruído na função de uma imagem



Fonte: Acervo dos autores.

Conforme é possível visualizar, existem diversas variações bruscas, impossibilitando a aplicação da derivada nesse sistema. Por isso, torna-se necessário fazer um tratamento na imagem. A suavização Gaussiana se baseia em tirar a média ponderada da intensidade de cada pixel com os seus vizinhos, de forma que os vizinhos mais próximos possuem maior peso. Essa medida provocará um borramento na imagem e no contexto de função, o gráfico, portanto, ficará mais suave, como pode ser visto na Figura 5 (b).

Figura 5: Aplicação da suavização Gaussiana



Fonte: Acervo dos autores.

Detecção do Gradiente

Após a suavização de ruídos na imagem, os métodos de detecção das bordas precisam de ferramentas que permitam localizar variações bruscas de cores. É nesse momento que o conceito de derivada é utilizado por meio do gradiente.

O gradiente é um vetor bidimensional que representa a taxa de variação da função *F* na posição (x, y). Ele é dado por:

$$\nabla \mathrm{F} = \left(G_x, G_y\right) = \left(\frac{\partial \mathrm{F}}{\partial \mathrm{x}}, \frac{\partial \mathrm{F}}{\partial \mathrm{y}}\right) \qquad (3)$$

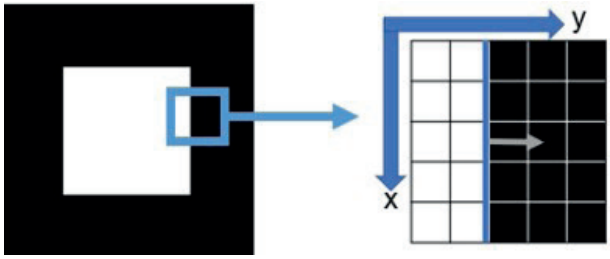
É necessário utilizar esse tipo de ferramenta uma vez que a imagem é representada por uma função de duas dimensões, podendo ocorrer variações de intensidade de cores tanto em relação ao eixo x quanto ao eixo y.

No contexto do processamento de imagem, o gradiente fornecerá a direção e o sentido da variação de intensidades de cores entre um pixel e seus vizinhos por meio do ângulo α obtido pela equação:

$$\alpha(x, y) = \arctan\left(\frac{G_y}{G_x}\right). \qquad (4)$$

A direção do gradiente é importante uma vez que ele é ortogonal à borda, como é possível visualizar na Figura 6, na qual se vê, do lado esquerdo, uma figura constituída por duas cores: branco (255) e preto (0); e, do lado direito, uma ampliação dessa imagem, em que há a borda (representada por uma reta azul). A direção do vetor gradiente é representada pela seta na cor cinza.

Figura 6: Vetor gradiente representado sobre a imagem



Fonte: Acervo dos autores.

Além disso, a magnitude, que indica intensidade da variação, é dada por:

$$M = M(x,y) = \sqrt{(G_x)^2 + \left(G_y\right)^2} \qquad (5)$$

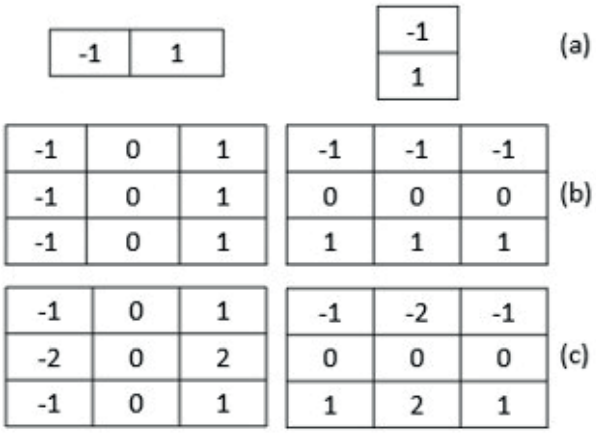
O gradiente é uma importante ferramenta para encontrar as bordas, entretanto, como aplicar a derivada em um sistema com valores discretos com duas dimensões? Para isso se utiliza uma aproximação da derivada parcial, que é aplicada pixel-a-pixel. Conforme já visto, a derivada é conceituada como a variação instantânea de uma dada função. Generalizando esse conceito, pode-se chegar às seguintes equações que representam a diferença da intensidade dos pixels a serem analisados $f(x, y)$ e seus vizinhos em y ($f(x, y+1)$) e em x ($f(x+1, y)$).

$$G_x = \frac{\partial f(x,y)}{\partial x} = f(x+1,y) - f(x,y)$$
$$G_y = \frac{\partial f(x,y)}{\partial y} = f(x,y+1) - f(x,y) \qquad (6)$$

Isso significa que a diferenciação em um elemento da imagem pode ser vista como a diferença das intensidades dos elementos vizinhos. Com essa definição, o resultado da aproximação é igual a zero em locais onde a intensidade é constante e diferente de zero nos pontos em que ocorrem variações de cores, satisfazendo o conceito de derivada.

É importante ressaltar que as formas apresentadas na equação (6) para a obtenção do gradiente não são suficientes quando se trata de bordas diagonais, uma vez que a intensidade de cor dos pixels vizinhos na diagonal não são considerados. A Figura 7(a) demonstra a equação na forma matricial. Uma forma de solucionar esse problema é construir uma máscara de duas dimensões – matriz 3x3 – que detecte variações de todos os oito vizinhos. Segundo Gonzalez e Woods (2010), essas máscaras recebem o nome de *Operadores de Prewitt* e podem ser vistas na Figura 7(b). O método de Canny utiliza o *Operador de Sobel*, o qual valoriza os pixels mais próximos por meio de um peso, representado na Figura 7(c).

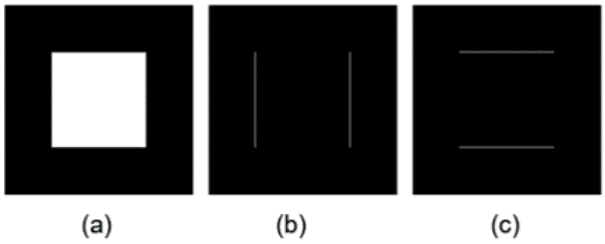
Figura 7: Matrizes representantes dos operadores diferenciais



Fonte: Acervo dos autores.

Um exemplo da aplicação do *Operador de Sobel* pode ser visualizado na Figura 8. A Figura 8(a) é constituída apenas por duas tonalidades de cinza, 0 (preto) e 255 (branco). Dessa forma, a variação é detectada exatamente quando há a troca de cores, (Figura 8(b) e (c)), sendo representada por uma linha branca.

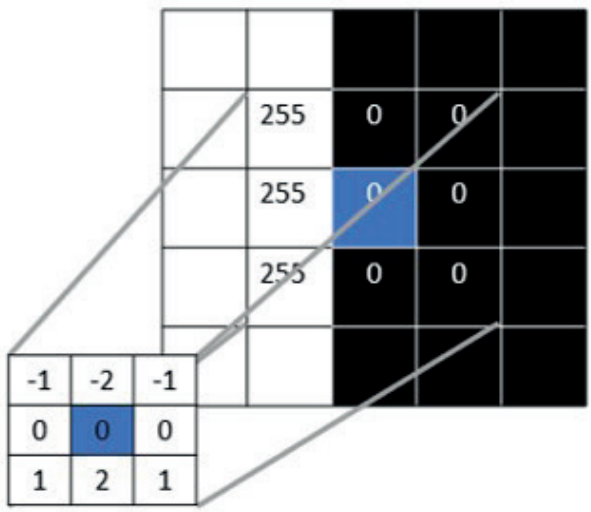
Figura 8: Representação da aplicação do gradiente na direção de x (b) e y (c)



Fonte: Acervo dos autores.

Como forma de demonstrar o método de detecção da variação de cores, suponha que se deseja detectar a existência de uma borda no pixel destacado (na cor azul) – Figura 9. Suponha ainda que a cor escura seja igual a 0 e a cor clara seja 255. O primeiro passo é centralizar o pixel com o centro do *Operador de Sobel*.

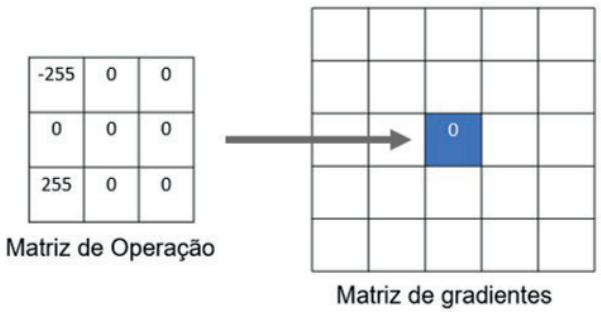
Figura 9: Centralizando pixel com o Operador de Sobel



Fonte: Acervo dos autores.

Multiplica-se, assim, o pixel analisado e seus oito vizinhos pelo valor correspondente à sua posição no operador. Esse processo resultará em uma matriz com as mesmas dimensões do operador. O próximo passo é fazer a somatória dos elementos desse *array* resultante, detectando dessa forma a intensidade do gradiente em uma das direções. Por fim, o valor obtido é colocado, na mesma posição do pixel que foi estudado, em uma nova matriz, denominada matriz de gradientes. Esse procedimento pode ser observado na Figura 10, na qual $G_x = 0$, o que faz sentido uma vez que não há variação de cores na direção do eixo x.

Figura 10: Encontrando a intensidade do gradiente



Fonte: Acervo dos autores.

Realizando-se os mesmos procedimentos com o operador em y, obtém-se $G_y = -1020$. Utilizando as operações por meio da equação (4), constata-se que o ângulo α tende a 90° em relação ao eixo x. Por fim, pela equação (5) é possível deduzir um valor representante da variação nas duas dimensões. Assim obtemos a magnitude $M = 1020$. Esse valor é alocado em uma nova matriz, na posição correspondente ao pixel estudado.

Após obter o gradiente em todos os pixels em relação aos seus vizinhos, inicia-se o processo de filtragem dos pontos que serão considerados como bordas. Isso significa que os próximos passos do método de Canny serão responsáveis por encontrar os valores máximos da função gradiente, uma vez que esses serão as bordas.

Supressão do não máximo e Limiarização

O primeiro passo para a filtragem dos valores obtidos por meio do gradiente é a supressão do não máximo. É nesse momento que se utiliza a orientação, ou seja, o ângulo do gradiente, de forma que será anulado todos os valores não máximos nessa direção.

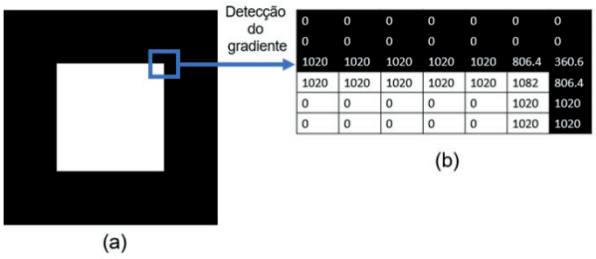
Inicialmente, é necessário fazer um arredondamento do ângulo obtido por meio do gradiente. Isso em razão de a imagem ser armazenada como matriz, logo não é possível obter um ângulo de 43°, por exemplo, entre dois pixels. Os possíveis valores de vizinhança para um ponto sempre serão múltiplos de 45° em relação ao eixo x, ou seja, 0° e 180° para os vizinhos localizados na direção vertical ao ponto; 90° e 270° para os vizinhos na direção horizontal; 45°, 135°, 225° e 315° para os vizinhos na diagonal. Todos os valores diferentes desses são arredondados para o mais próximo. Esse arredondamento é tratado automaticamente pela biblioteca utilizada no algoritmo desenvolvido neste artigo.

Após esse processo, o método de Canny segue todas as direções dos gradientes, anulando os valores não máximos nessas direções. Como visto, a direção do gradiente é ortogonal à borda. Assim, a supressão do não máximo não afetará a borda, mas os valores ao redor dela, uma vez que é na borda que há a maior intensi-

dade do gradiente. Dessa forma, a supressão do não máximo possibilitará a detecção de possíveis pixels representantes de bordas.

Realizando-se o gradiente em todos os pixels da Figura 11 obtém-se como matriz resultante a Figura 11(b). É possível constatar que alguns dos valores de gradientes não são máximos em sua direção. Levando em consideração o método de Canny esses valores são anulados.

Figura 11: Detecção de gradiente



0	0	0	0	0	0	0
0	0	0	0	0	0	0
1020	1020	1020	1020	1020	806.4	360.6
1020	1020	1020	1020	1020	1082	806.4
0	0	0	0	0	1020	1020
0	0	0	0	0	1020	1020

Fonte: Acervo dos autores.

Aplicando a supressão do não máximo na direção do gradiente obtemos uma redução dos valores possíveis para a borda – Figura 12. Isso porque foram suprimidos os valores não máximos.

Figura 12: Aplicando supressão do não máximo na matriz resultante do gradiente

0	0	0	0	0	0	0
0	0	0	0	0	0	0
1020	1020	1020	1020	1020	0	0
1020	1020	1020	1020	1020	1082	0
0	0	0	0	0	1020	1020
0	0	0	0	0	1020	1020

Fonte: Acervo dos autores.

O processo final da detecção de bordas elaborado por Canny é a limiarização. Essa etapa consiste em separar a imagem em duas partes: a primeira representa a região de interesse (borda) e a segunda representa áreas não desejadas como ruídos, bordas falsas, fundos ou preenchimentos dos objetos. Isso significa que esse processo se resume em categorizar a intensidade cinza por meio de um valor limiar, de forma que os valores abaixo e acima desse limite são tratados de maneira diferente.

Desse modo o processo de limiarização produz uma imagem binária bem definida, ou seja, os valores maiores que o limiar recebem a intensidade um, valor máximo. Por outro lado, todas as outras intensidades são trocadas por zero. Segundo Gonzalez e Woods (2010), esse processo é denominado binarização da imagem, que pode ser definido matematicamente como a equação:

$$G(x,y) = \begin{cases} 1, & se\ F(x,y) > T \\ 0, & se\ F(x,y) \leq T \end{cases} \qquad (7)$$

onde $F(x,y)$ é a imagem de entrada, $G(x,y)$ é o valor de limiar e $G(x,y)$ é a imagem de saída ou limiarizada. Podemos observar que na equação (7) os valores maiores que o limiar são considerados como borda e os valores abaixo desse limiar são considerados como áreas não desejadas.

Esse valor de limiar é a segunda aproximação que o sistema de Canny utiliza, ou seja, não há um resultado concreto e único. Cada ambiente possui variáveis que influenciam nesse limiar como iluminação, quantidade de ruídos etc. No contexto do algoritmo desenvolvido, é função do usuário fazer uma aproximação de limiar com base nas suas necessidades.

DESENVOLVIMENTO DO ALGORITMO

Segundo Canny (1983), a detecção da variação da intensidade dos pixels de uma imagem é algo canônico na visão computacional. Existem diversas áreas que utilizam a detecção de bordas como ferramenta para analisar o ambiente. Neste trabalho, o método de Canny foi utilizado no desenvolvimento de um algoritmo capaz de fazer retornar a medição de um objeto.

Alguns critérios foram levados em consideração para o desenvolvimento do algoritmo: 1) o programa deveria envolver lógica de programação apresentada no ciclo básico; 2) o algoritmo não deveria envolver conceitos avançados de processamento de imagem, mas demonstrar a aplicação da derivada nessa área.

A primeira etapa do desenvolvimento do algoritmo foi dedicada à manipulação da matriz representante de uma imagem, visando a transformá-la em uma matriz apenas com elementos binários. Utilizando a biblioteca OpenCV, foi desenvolvido um algoritmo que captura uma imagem da câmera de um computador e a salva na forma matricial. O comando responsável por

isso pode ser visto na Figura 13, linha 2. Para isso, utilizamos uma variável do tipo *Mat*, ou seja, uma variável que consegue armazenar as três *arrays* representantes das bandas RGB.

Na sequência, transformam-se os valores dos canais RGB em um único representante na escala de cinza. Por fim, é aplicado o método Canny, que é capaz de identificar mudanças na intensidade dos pixels da imagem para detectar a presença de bordas, retornando uma matriz do tipo binária. A Figura 13 contém os comandos necessários para as primeiras transformações.

Figura 13: Comandos para transformação inicial da imagem

```
1. Mat img, gray, borda;
2. cam.read(img);
3. cvtColor(img,gray,COLOR_BGR2GRAY);
4. Canny(gray, borda, a, b);
```

Fonte: Elaborada pelos autores.

Interpretando a matriz resultante como um sistema de coordenadas, em que as colunas representam o eixo *y* e as linhas representam o eixo *x*, foram criadas duas funções. A primeira função, demonstrada na Figura 14, realiza a varredura da matriz utilizando laços encaixados do comando *for,* buscando pelo primeiro e pelo último elemento (*x,y*) cujo valor seja diferente de 0, ou seja, presença de borda. Subtraindo as coordenadas do último elemento considerado borda (salvo na variável tam[1]), pela coordenada do primeiro elemento (salvo na variável tam[0]), obtém-se a altura em pixels do objeto. Essa função limita-se a medir objetos sem que haja aplicação de rotações.

Figura 14: Método de medição 1

```
1.  boolean primeiro_ponto = false;
2.  for (int x = 0; x < borda.rows; x++) {
3.  for (int y = 0; y < borda.cols; y++) {
4.  if (borda.at<uchar>(x,y)!=0 && !primeiro
    _Ponto ) {
5.  tam[0] = x;
6.  primeiro_Ponto = true;
7.  } else if (borda.at<uchar>(x, y) != 0) {
8.  tam[1] = x;
9.  }
10. }
11. }
12. tam[2] = tam[1] - tam[0];
```

Fonte: Elaborada pelos autores.

A segunda função, representada na Figura 15, realiza duas varreduras na matriz em sentidos opostos utilizando o mesmo conceito de laços encaixados do método anteriormente citado. Isso possibilita encontrar duas quinas do objeto. Dessa maneira, com a informação da coordenada (*x,y*) das quinas, utilizou-se o conceito de distância entre dois pontos para encontrar o valor em pixels de uma quina à outra.

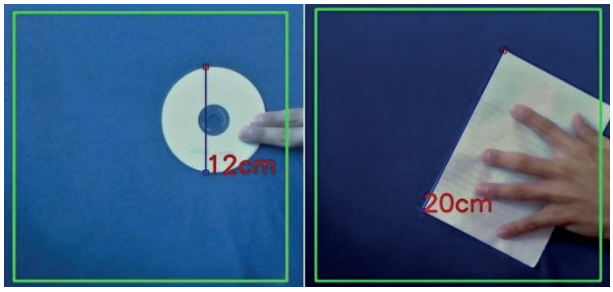
Figura 15: Método de medição 2

```
1.tam[0] = A.x - B.x;
2.tam[1] = A.y - B.y;
3.tam[2] = sqrt(pow(tam[0], 2) + pow(tam[1],
  2));
```

Fonte: Elaborada pelos autores.

Essa função limita-se a medir objetos com quinas bem definidas. Quando a medida é o diâmetro, por exemplo, utiliza-se o primeiro método citado. Rotações que não envolvam o eixo de profundidade são medidas sem que haja danos ao resultado. Os dois métodos são exemplificados na Figura 16.

Figura 16: Demonstração dos métodos de medição



Fonte: Acervo dos autores.

Buscando-se realizar medições em tempo real, foi necessário criar uma função (Figura 17) para estabelecer uma relação inicial entre pixels e centímetros. Essa relação deve ser mantida para estabelecer medições de objetos que estejam na mesma distância da câmera. Esse processo consiste na utilização de um objeto com dimensões conhecidas para, a partir da medição desse objeto, dada em pixels, ser possível computar a relação entre pixel e centímetro.

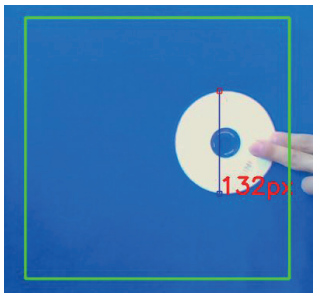
Figura 17: Função para estabelecer relação entre pixel e centímetro

```
1. void relacaoTamanho()
2. {
3. float objcm, pxl;
4. cout << "Insira o Tamanho do Objeto em
   cm" << endl;
5. cin >> objcm;
6. pxl = (float)tam[2];
7. rel = (float)objcm / pxl;
8. }
```

Fonte: Elaborada pelos autores.

Na Figura 18 verifica-se que a medição do diâmetro do CD foi equivalente a 132 pixels. Como um CD padrão tem diâmetro de 12,0 centímetros, o *software* computou a relação pixel/centímetro equivalente à aproximadamente 0,091 centímetros por pixel. Essa relação foi estabelecida com a câmera a uma distância de 750 milímetros do objeto e mantida para as demais medições.

Figura 18: Computando a altura em pixels para ser utilizado como referência



Fonte: Acervo dos autores.

RESULTADOS DO ALGORITMO

Foram realizadas diversas experimentações para constatar a correlação entre a medida em pixels e a medida real do objeto em milímetros. Nesse processo algumas variáveis foram controladas, como: a distância do objeto à câmera; a rotação do objeto; e o processo de limiarização do método Canny.

Tabela 1: Resultados das medições realizadas a 500mm de distância

Tamanho (mm)	Medida em pixels	Relação pixel/mm
210	280	0,75
294	394	0,75
140	185	0,76
94	124	0,76
66	88	0,75

Fonte: Elaborada pelos autores.

Tabela 2: Resultados das medições realizadas a 1000mm de distância

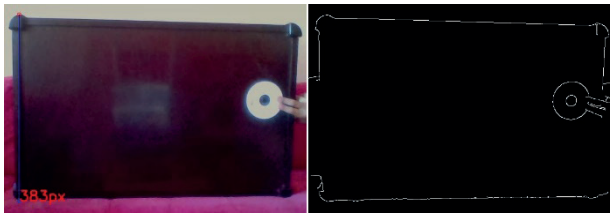
Tamanho (mm)	Medida em pixels	Relação pixel/mm
210	142	1,48
294	188	1,56
140	91	1,54
94	64	1,47
66	45	1,47

Fonte: Elaborada pelos autores.

Como se observou na Tabela 1, o desvio padrão da relação pixel/milímetros é de 0,0005mm, sendo inferior ao 0,0450mm constatada na Tabela 2. Isso indica que quanto menor a distância entre a câmera e o objeto medido, melhor será a precisão. Esse resultado está relacionado com a perda de detalhes do objeto ao afastá-lo da câmera.

Como as bordas que não representam o objeto e as bordas do objeto são armazenadas da mesma maneira, ao se analisar a imagem completa a presença de detalhes indesejados impedia o processo de medição. Isso ocorre porque o algoritmo procura o primeiro e o último elemento da matriz que representam bordas. Ao encontrar falsas bordas, ou bordas que não representam o objeto, erros de medições eram gerados. Sendo assim, foi desenvolvida uma nova função no programa com o objetivo de delimitar a região de interesse. Essa função tem como objetivo reduzir a área de busca por bordas, transformando a matriz da imagem em uma matriz menor.

Figura 19: Detecção de borda sem a região de interesse



Fonte: Acervo dos autores.

Figura 20: Detecção de borda dentro da região de interesse

Fonte: Acervo dos autores.

Conforme se observou na Figura 19, em que não houve a delimitação da área de busca, o algoritmo retornou uma altura em pixels equivalente a 383px, o que não correspondia à realidade. Já na Figura 20, a região de busca por bordas correspondeu à área interna do retângulo de contorno verde. Isso significa que somente o que estiver compreendido na região de interesse será analisado.

Além do problema com a região de interesse, houve também a necessidade de permitir que o usuário pudesse modificar, quando necessário, a limiarização desejada para o processo. Isso em razão de a aplicação da limiarização ser difícil e envolver experimentação. Na Figura 20 observou-se que as bordas ficam restritas ao objeto, já na Figura 21 ocorre a presença de falsas bordas, sendo possível perceber o impacto de uma limiarização mal aplicada, o que interfere na medição.

Figura 21: Detecção de borda com limiarização mal aplicada

Fonte: Acervo dos autores.

Por fim, reduzindo a região de interesse e permitindo a limiarização durante o processo de medição, o algoritmo foi capaz de retornar medições dos objetos com uma precisão de 0,0962 centímetros, ou seja, inferior a 1 milímetro (vide Tabela 3). Esse é um resultado aceitável, visto que se buscava a demonstração de uma utilização dos dados de saída do método de detecção de borda. Porém, o fato de trabalharmos com uma câmera 2D implicou na perda de percepção de profundidade. Logo, quando há rotação do objeto em profundidade a medição passa a se tornar inválida. Aprofundar os cálculos para encontrar medições do objeto independente da rotação é uma das possíveis vertentes que este trabalho poderá seguir, portanto, não será abordada essa variável no processo.

Tabela 3: Medições dos objetos com precisão de cm

Medida esperada (cm)	Medida encontrada (cm)	Erro (cm)
12,00	11,91	0,09
8,00	7,83	0,17
5,40	5,55	0,15
8,56	8,65	0,09
21,00	21,06	0,06
29,70	29,63	0,07
20,00	19,91	0,09
14,00	13,95	0,05

Fonte: Elaborada pelos autores.

CONCLUSÃO

Os altos índices de reprovação sugerem que a forma de apresentar a disciplina de Cálculo Diferencial e Integral necessita de inovação para se adequar às necessidades dos cursos na área de tecnologia. Aplicações visuais e concretas do conteúdo teórico podem facilitar no aprendizado e rendimento dos alunos. Dessa forma, o processamento de imagem, atrelado à visão computacional, é uma área que possui potencial para apresentar, de modo prático, os conceitos do Cálculo 1, sobretudo os conceitos de derivada.

Uma das vantagens mais relevantes para a utilização dos sistemas de visão computacional foi a sua visualização em tempo real, ou seja, a possibilidade de capturar uma imagem do ambiente em que o aluno se localizava, modificando, interpretando e retornando fatores de interesse. Características sobre a forma e as dimensões dos objetos capturados na fotografia foram possíveis utilizando essa área. Além disso, a biblioteca OpenCV possibilitou o desenvolvimento do algoritmo sem a necessidade de conhecimentos avançados na área.

É importante salientar que a detecção de bordas não se limita aos conceitos aqui apresentados e o aprofundamento do estudo nessa área exige conhecimentos complexos, o que inviabilizaria sua demonstração para alunos que estão iniciando o curso de Engenharia de Computação. Fatores como otimização do algoritmo e calibração da câmera (para detecção de profundidade) não foram levados em consideração para facilitar o entendimento.

Portanto, o objetivo deste trabalho foi conquistado uma vez que foi possível encontrar uma área que aplicasse os conceitos da derivada para alunos de Engenharia de Computação. Espera-se que este artigo possa motivar profissionais da área acadêmica a utilizarem formas de alinhar a teoria com problemas reais, transformando os alunos em construtores do seu próprio conhecimento.

REFERÊNCIAS

BIASI, H. H. D. *et al.* **Desenvolvimento de uma metodologia de visão computacional para o auxílio no planejamento cirúrgico no implante de próteses endoluminais.** 2002. Dissertação (Mestrado em Ciência da Computação) – Centro Tecnológico, Universidade Federal de Santa Catarina, 2002.

CANNY, J. F. **A variational approach to edge detection,** AAAI, vol. 1983, p. 54-58, 1983.

CURY, H. N. **Pesquisas em análises de erros no ensino superior:** retrospectiva e resultados. Educação Matemática no Ensino Superior: pesquisas e debates. Recife: SBEM, 2009.

CUZZUOL, G. D. et al. Engineering students can use the words “calculus” and “Love” in the same sentence: using active learning the impossible can happen. **Zeteike**, 5(1), p. 272-281, 2018.

DE JESUS BRITO, A.; CARDOSO, V. C. Uma abordagem histórico-pedagógica dos fundamentos do cálculo diferencial: reflexões metodológicas, **Zeteike**, 5(1), p. 129-144, 2018.

DE QUEIROZ, J. E. R.; GOMES, H. M. Introdução ao processamento digital de imagens. **Rita**, 13(2), p. 11-42, 2006.

GONZALES, R. C; WOODS, R. C. **Processamento digital de imagens,** 3ed., Gabriela Trevisan, 2010.

LIMA, R. M.; ANDERSSON, P. H.; SAALMAN, E. **Active learning in engineering education:** a (re)introduction, 2017.

MARENGONI, M.; STRINGHINI, S. Tutorial: Introdução à visão computacional usando Opencv, **Revista de informática Teórica e Aplicada**, 16(1), p. 125-160, 2009.

RAFAEL, R. C. **Cálculo Diferencial e Integral**: um estudo sobre estratégias para redução do percentual de não aprovação. 2017. Dissertação (Mestrado em Educação Matemática) – Instituto de Ciências Exatas, Universidade Federal de Juiz de Fora, 2017.

SILVA, J. I. G.; FERREIRA, D. H. L. O uso de tecnologias na disciplina de cálculo diferencial e integral I. **Anais...** XIV Encontro de Iniciação Científica da PUC, Campinas, 2009.

WROBEL, J. S.; ZEFERINO, M. V.; CARNEIRO, T. C. Um mapa do ensino de Cálculo nos últimos 10 anos do COBENGE**. Anais...** CONGRESSO BRASILEIRO DE EDUCAÇÃO EM ENGENHARIA, Gramado, Rio Grande do Sul: ABENGE, 2013.

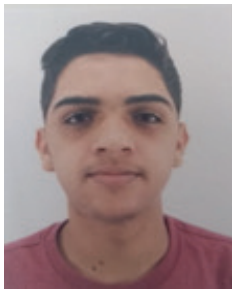
DADOS BIOGRÁFICOS DOS AUTORES

Alana Cavalcante – Graduada (Licenciatura) em Matemática pela Universidade Federal de Ouro Preto (2010); Mestrado em Matemática Pura na área de Topologia/Teoria de Singularidades pela Universidade Federal de Viçosa (2013); Doutorado em Matemática Pura na área de GeometriaAlgébrica/Folheações Holomorfas (2018). Professora do Departamento de Ciências Exatas e Aplicadas da Universidade Federal de Ouro Preto, *campus* João Monlevade desde 2013.

Jamisson Jader Moraes Pereira Junior – Graduando em Engenharia de Computação pela Universidade Federal de Ouro Preto, *campus* João Monlevade, 2018. Área de Interesse: Automação residencial utilizando microcontroladores, Internet das Coisas (IOT), sistemas robóticos de baixa potência, processamento de imagem e visão computacional.

Rangel Henrique Miranda Trindade – Graduando em Engenharia de Computação pela Universidade Federal de Ouro Preto, *campus* João Monlevade, 2018. Área de Interesse: Constituição e processos de máquinas e recursos computacionais associando a comunicação *hardware*/*software* ao ser humano.

Thiago Silva de Figueiredo – Graduando em Engenharia da Computação pela Universidade Federal de Ouro Preto, *campus* João Monlevade (2018); Técnico em Mecatrônica pela Escola Politécnica de Minas Gerais (2010); Auxiliar de programação pela Sustenta Perfis Metálicos (2010~2012); Programador de robôs pela COMAU (2012~2018).



Vinicius Fernandes Silva – Graduando em Engenharia Elétrica pela Universidade Federal de Ouro Preto, *campus* João Monlevade, 2018. Área de Interesse: Automação residencial utilizando microcontroladores, integração entre a visão computacional e sistemas elétricos.

Gustavo Tupini Silveira – Graduando em Engenharia da Computação pela Universidade Federal de Ouro Preto, *campus* João Monlevade, 2018; Técnico em Química pelo Instituto Federal Fluminense, *campus* Bom Jesus do Itabapoana, 2017.