21º Congresso Latino-americano de Software Livre e Tecnologias Abertas

LATINOWARE 2024

Latin.Science 2024

Proposed Approach for Creating Soybean Grain Image Dataset

Gesmar de Paula Santos Júnior*, Alexandre Cardoso*, Leonardo G. Marques†, Igor S. Peretta* and Pedro Carvalho‡
* Faculty of Electrical Engineering
Federal University of Uberlândia, Uberlândia, Minas Gerais 38408–902
Email: {gesmarjunior, alexandre, iperetta}@ufu.br
†Federal Institute of Goiás, Itumbiara, Goiás 75524–245
Email: leonardo.garcia@ifg.edu.br
‡Grider, 10 Dr. Duarte Square, Room 1, Uberlândia, Minas Gerais 38400–156
Email: pedro.carvalho@grider.com.br

***Abstract*—The integration of digital technologies and artificial intelligence in agriculture has the potential to significantly improve the accuracy and efficiency of grain classification. This study focuses on the development of a comprehensive methodology for soybean grain classification, utilizing a custom-built image acquisition system and advanced image processing techniques. High-resolution images of soybean grains were captured using a Nikon D3100 DSLR camera, with the setup optimized to ensure consistent lighting and contrast for precise image analysis. Various segmentation methods, including RGB and CMYK color channel separation, Otsu thresholding, and edge detection using the Canny algorithm, were employed to isolate and classify key features of the grains. Classical image processing techniques were used to create a robust and labeled dataset, providing essential training data for machine learning models. The results demonstrate the potential of combining classical image segmentation with machine learning to automate grain classification processes, enhancing reliability and ensuring compliance with industry standards.**

***Keywords*—Soybean Classification; Image Processing; Dataset Labeling; Machine Learning.**

I. Introduction

The use of digital technologies and artificial intelligence in agriculture offers important opportunities for the solution of classification problems. According to [1], with the use of digital technologies, it becomes possible to optimize grain selection, leading to an increase in agricultural productivity and directly enhancing income from the same unit of cultivated area. These applications of artificial intelligence in line with scientific studies can improve grain quality and classification, integrate tools and systems to precisely monitor and automate critical processes, and optimize the analysis of physical and chemical characteristics of grains. Additionally, these technologies allow for the prediction and mitigation of potential issues throughout production, ensuring that grains meet the strict quality standards required by the food quality.

The development of a methodology for grain image acquisition that can be used as input for an artificial neural network aims to enhance the transparency and reliability of the grain classification process, ensuring compliance with standards and repeatability of results. This approach is further supported by [2], who highlights the use of advanced technologies, such as artificial intelligence and image processing, has the potential to revolutionize the current methods of grain classification.

Along with artificial intelligence algorithms, research has been conducted using computerized image processing. In this context, studies on crops such as corn, rice, wheat, soybeans, and barley – the five most produced grain in the world – have been carried out to assess the applicability of computer vision in precision agriculture. Over the past five years, numerous articles have presented different approaches to disease detection, grain quality assessment, and phenotyping. [3] also supports the integration of digital technologies in agriculture, emphasizing their role in improving the accuracy and efficiency of such processes.

For this study, we have chosen to focus specifically on color images of soybean grains, given their significant economic and industrial relevance. The development of a specialized dataset aims to enhance the accuracy and effectiveness of soybean classification processes. A similar approach was followed by [4], whose work also aligns with the goals of improving classification through data collection. From the labeled images, a robust and annotated dataset is being constructed. This dataset will serve as a fundamental training set for artificial intelligence algorithms, enabling them to learn the distinctive patterns and characteristics of soybean grains.

Realização: itaipu parquetec · ITAIPU BINACIONAL · MAIS QUE ENERGIA · GOVERNO FEDERAL BRASIL UNIÃO E RECONSTRUÇÃO

27 a 29 de novembro de 2024
Foz do Iguaçu | Paraná | Brasil

II. Development

The first stage involves the construction of a box designed to house the cameras, lights and possible sensors, aimed to provide essential protection and stability during image capture in the field environment. After the image collection, we proceed to process the segmentation stage of the samples. This phase involves analyzing and dividing the images into distinct regions to isolate and identify key features, which is essential for accurate data for model training. Each captured image is labeled with characteristic information for analysis, such as grain type, foreign matter or impurities, damage type, and other relevant features. This labeling process is crucial for the development of the superrvised machine learning models, providing a labeled dataset for the classification artificial neural network.

A. Image Acquisition System

The image acquisition system consists of a professional camera Nikon D3100 [5], proper lens, built-in background blue bed, grain counting board, and light box as shown in Fig. 1.

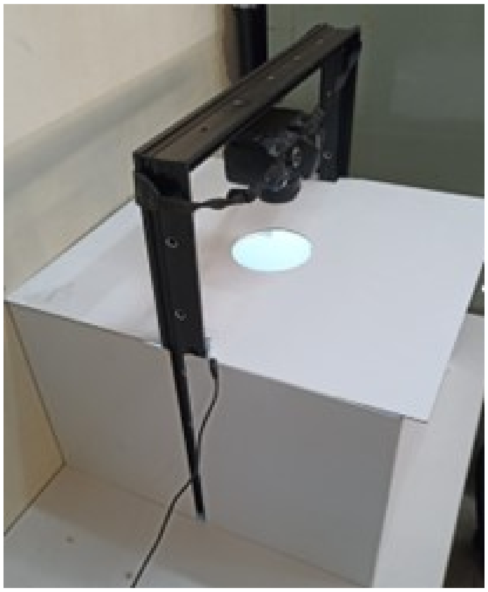

Figure 1. Image acquisition system.

1) Professional câmera: The professional camera was then placed in the center of the light source about 420mm from the plane of soybeans. The soybean images (4608$\times$3072 pixels) were saved in JPG (compressed) and NEF (raw, uncompressed) format.

The camera model Nikon D3100 [6] has Digital Single-Lens Reflex (DSLR) with 14.2 megapixels, and a 12-bit CMOS sensor resolution. Its features are automatic chromatic aberration correction, ISO range of 100-3200, 23.1 mm $\times$ 15.4 mm Nikon DX RGB CMOS sensor, $1.5\times$ crop factor, and 4.94 μm pixel size. The captured images are available both in JPEG and NEF file formats (compressed and raw, respectively).

2) Lens: We used a fixed 50mm Nikon F-mount lens with a 420 mm distance from the samples. This lens was chosen to minimize image distortion, bringing the captured image as close as possible as perceived in the real world.

3) Light source: The light source was positioned circularly around the camera in the inner top of the light box, up to the soybean seeds at approximately 420 mm, providing a light intensity of around 500 Lux (Fig. 2). For photographing grains in a dataset, we empirically detected that 500 lux provides a balance between intensity and softness, allowing the capture of precise details and textures of the grains. The luminosity was measured using a method also employed by [7], using a luxmeter. This level of illumination helps avoid harsh shadows and excessive highlights, allowing for a clear and accurate representation of the grains' features. According to tests, 500 lux was bright enough to highlight subtle nuances on the surfaces of the grains, which is crucial for analysis and classification in a dataset.

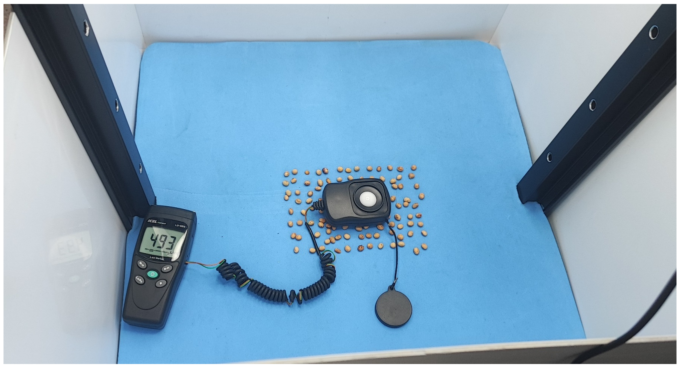

Figure 2. Measurement of light source luminosity.

4) Background blue base: The primary purpose of the RGB (Red, Green, and Blue) system is color reproduction in electronic devices such as TV monitors and digital cameras, as well as in traditional photography, as noted by [8]. In contrast, printers use the CMYK (Cyan, Magenta, Yellow, and Black) color model to produce colored printouts. The RGB color model is based on the sensitive cones of human eyes, leading to the trichromatic color vision theory of Young-Helmholtz and Maxwell's color triangle. CMYK, also known as four-color printing, works through the absorption of light, making the colors we see coming from the part of the light that is not absorbed. This system is employed by the printing industry to reproduce most colors in the visible light spectrum. It is a subtractive color system, as opposed to the additive RGB system. Cyan is the opposite of red, meaning it acts as a filter that absorbs red light (−R +G +B). Similarly, magenta is the opposite of green (+R −G +B), and yellow is the opposite of blue (+R +G −B). Therefore, magenta plus yellow will produce red, magenta plus cyan will produce blue, and cyan plus yellow will produce green. As noted by [9], CMYK pattern is primarily used for printing and other reflective medias, whereas monitors and televisions – light-producing medias – use the RGB pattern through three-colored LEDs (light emmiter diodes).

In this study, the background will be set to a shade of blue that contrasts with yellow in the CMYK color model, as soybean grains color are mainly shades of yellow tones. This choice is made to ensure a clear distinction between the grains and the background, enhancing the accuracy of image processing and segmentation, also motivated by the fact that in the CMYK color model yellow and blue are complementary colors, making them easy to distinguish during the image segmentation stage. By selecting a background color that is opposite to yellow on the CMYK spectrum, we improve the visibility and differentiation of the soybean grains in the images.

The background material used is synthetic EVA foam, embedded into the base bed of the light box. This material ensures that the background maintains its color and texture, offering a reliable contrast with the soybean grains and contributing to the overall accuracy of the image analysis process.

5) Light box: The built light box consists of a white PVC frame case around the entire environment in the frame case. According to [1], the closed box prevent the formation of shadows on the acquired images and preventing reverse light from outside. Also employed by [10], the objective of this box is also to maintain constant illumination conditions, and ensure sufficient contrast between the foreground and background, facilitating effective segmentation. This image acquisition system (Fig. 1) ensures the quality and consistency needed to create a useful and reliable database for future training and developing of machine learning models.

6) Grain counting board: The image area of the camera with this lens and focal distance measures 19 × 13 cm (Fig. 3). As highlighted by [11], manual counting is laborious, electronic automatic seed counter devices are expensive and their counting speed is very slow.

Classical image processing and segmentation techniques are used in this work and are based on analytical and mathematical methods, such as thresholding, edge detection, and color channel segmentation. These techniques are faster compared to deep learning-based methods but are limited in complexity. For the segmentation of the collected grain samples, where we used the grain counting board, classical techniques are sufficient.

Figure 3. Grain counting board.

B. Image Processing

Image segmentation is the process of dividing an image into parts or regions, typically to identify specific objects or areas within the image. Also as [12], the grain colors should be highlighted and the background of the images removed. For this, a processing algorithm was implemented that enhances pixel contrast by applying a filter, subsequently, the image was split into different channels.

There are two main approaches to image segmentation: classical techniques and deep learning-based techniques. Classical image segmentation techniques do not rely on machine learning or large volumes of labeled data. Instead, they use methods based on low-level image features such as color, texture, brightness, or contours. Among these techniques, we used thresholding-based methods, where the image is segmented into regions based on a chosen threshold, and edge-based methods, which detect edges or contours within the image to define regions. These techniques were executed quickly and were computationally lightweight, but they have limitations when it comes to segmenting complex images or those with significant variability in lighting, noise, or object shapes. To address these challenges, proper lighting was essential.

The image-processing algorithms was constructed with Python (Version 3.10.12) and OpenCV Library (Version 4.10.0). Processing was performed using the Google Compute Engine back-end in Python 3 via the Google Colab service, which provides 12.7GB of RAM and 107.7GB of storage for free [13].

1) Color channel: As noted by [10], color histograms are versatile constructs that can be generated from images in different color spaces and channels.

RGB channel segmentation:

Was employed the split function from the OpenCV library to decompose the RGB image into its individual color channels.

This function effectively separates the image into three distinct matrices, corresponding to the blue, green, and red channels. By isolating these channels (Fig. 4), it becomes possible to perform a more granular analysis of each color component, which can be crucial for tasks such as feature extraction and image processing. The specific function used is as follows:

```
(b_channel, g_channel, r_channel) = cv.split(image)
```

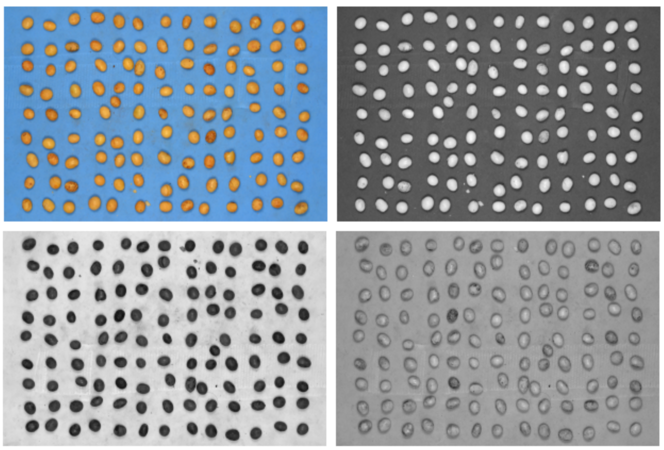

Figure 4. RGB Channels.

CMYK channel segmentation:

Upon splitting the RGB channels and visualizing them in grayscale, it was observed that the grains on the surface did not exhibit significant visual differentiation to the naked eye. To address this issue, we separated the image into CMYK channels, which provided much higher contrast between the grains and the background, as show on Fig. 5. To achieve this, the following algorithm was used:

```
#convert an image to CMYK and then split the channel
rgbdash = rgb.astype(np.float32)/255.
K = 1 -np.max(rgbdash, axis=2)
C = (1-rgbdash [...,2] - K)/(1-K)
M = (1-rgbdash [...,1] - K)/(1-K)
Y = (1-rgbdash [...,0] - K)/(1-K)
#Convert the input BGR image to CMYK colorspace
CMYK = (np.dstack((C,M,Y,K)) * 255).astype(np.uint8)
#Split CMYK channels
Y, M, C, K = cv.split(CMYK)
```

This method converts the RGB image to the CMYK color space and then splits the channels, yielding significantly improved contrast for more effective image analysis.

2) Thresholding: OTSU [14] adaptive thresholding is a classic binarization method that automatically selects the optimal threshold to divide the image into two distinct classes. [4] and [1] also use the method.

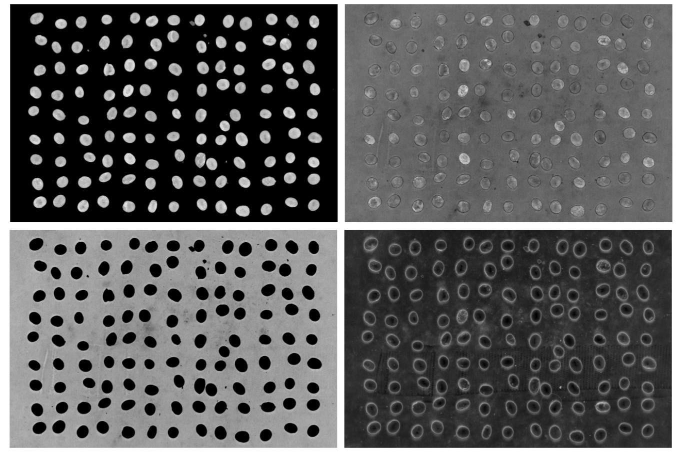

Figure 5. CMYK Channels.

```
#Convert the channel in a 3 channels image
Y_3channel = cv2.cvtColor(Y, cv2.COLOR_GRAY2BGR)
#Filter image
filtro = cv2.pyrMeanShiftFiltering(Y_3channel, 30,
    90)
gray = cv2.cvtColor(filtro, cv2.COLOR_BGR2GRAY)
#OTSU threshold create a binary mask
binaryImg = cv2.threshold(gray, 0, 255, cv2.
    THRESH_BINARY_INV | cv2.THRESH_OTSU)[1]
# remove background and return color image
withouBackground = cv2.bitwise_and(rgb, rgb, mask =
    cv2.bitwise_not(binaryImg))
withoutBackground = cv2.cvtColor(withouBackground,
    cv2.COLOR_BGR2RGBA)
mask = cv2.cvtColor(withouBackground, cv2.
    COLOR_RGBA2GRAY)
_, mask = cv2.threshold(mask, 0, 255, cv.
    THRESH_BINARY)
withouBackground [:,:,3] = mask
withouBackground = cv2.cvtColor(withouBackground, cv.
    COLOR_BGRA2RGBA)
cv2.imwrite("withouBackground.png", withouBackground)
plt.subplot(121);plt.imshow(binaryImg, cmap="gray");
plt.subplot(122);plt.imshow(withoutBackground);
```

In the code above, in addition to the threshold function, we have the pyrMeanShiftFiltering and bitwise functions. The pyrMeanShiftFiltering function, which is often used before segmentation, especially when the image contains noise or when it is necessary to clearly distinguish different regions, was used in our case to remove background noise that could interfere with the image. This function can also be combined with morphological techniques [15].

The bitwise function refers to a series of bitwise logical operations applied to the image and is an essential tool for image processing tasks [15]. The bitwise operations were used to create a mask, isolating the grains from the background. This

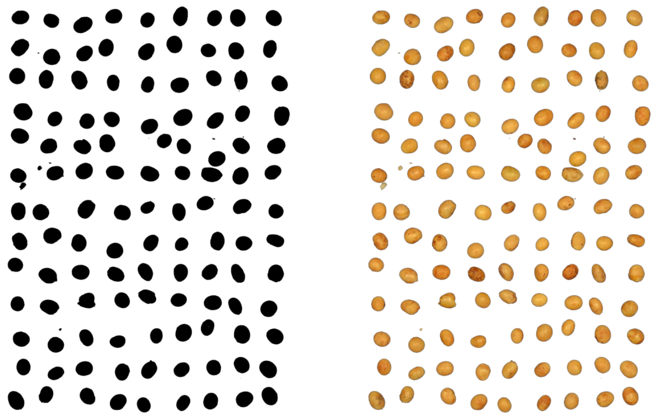

Figure 6. Image with mask and without background Channels.

logical operation works at the pixel level, manipulating pixel values in a binary manner. After applying the filtered image, bitwise was utilized to segment the image by creating the mask, As shown in Fig. 6 on the left.

The algorithm then uses the binary mask from the RGBA image (which includes an alpha channel for transparency) and applies this mask to the alpha channel of the original image to remove the background. This technique was useful for removing the background from an image, preserving only the objects of interest, as shown in Fig. 6 on the right.

3) Edge-based: Edge enhancement algorithms specifically target the edges to increase their prominence. These algorithms often use edge detection filters followed by image blending techniques to make the edges more pronounced, useful for enhancing feature extraction, as it improves the clarity of the images critical details.

As noted by [16], the edge or contour of the segmented seed was extracted using the Canny edge detector. The Canny edge detector, as described by [17], was also employed in this image processing (Fig. 7). Is an algorithm consisting of 4 major steps:

1. Reduce Noise using Gaussian Smoothing: Denoising technique are used to reduce noise in the image without significantly affecting the sharpness of image features. We applied Gaussian smothing, a classical image processing technique, that applies a Gaussian kernel to smooth the image, effectively reducing high-frequency noise.
2. Compute image gradient using Sobel filter: The Sobel filter in OpenCV calculates image derivatives, combining Gaussian smoothing with differentiation, helps in accurately identifying the gradient of the image, enhancing the edge detection process by reducing the impact of noise and ensuring that only significant edges are highlighted.
3. Non-maxima suppression: Applying non-maxima suppression to keep only the local maxima of gradient magnitude pixels that are pointing in the direction of the gradient.
4. Hysteresis thresholding: Finally, apply hysteresis thresholding with upper and lower threshold values, which helps to remove regions of an image that are not technically edges but still respond as edges after computing the gradient magnitude and applying non-maximum suppression.

```
#Canny() function with L2Gradient
img = withoutBackground
t_lower = 100 # Lower Threshold
t_upper = 230 # Upper threshold
aperture_size = 5 # Aperture size of the sobel filter
L2Gradient = True # Boolean
# Applying the Canny Edge filter
edge = cv.Canny(img, t_lower, t_upper, L2gradient =
    L2Gradient )
```

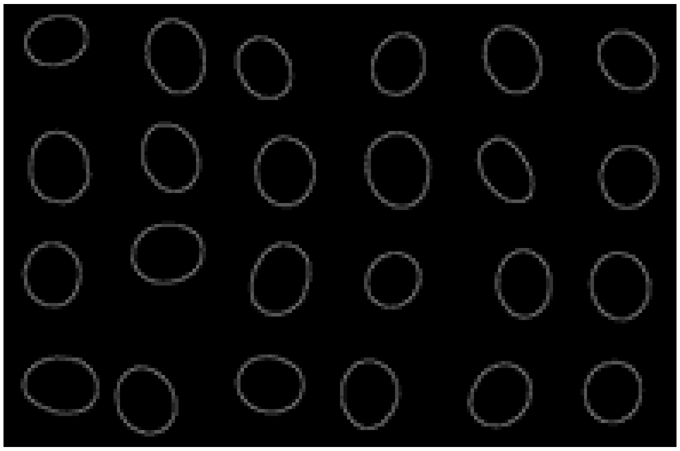

Figure 7. CannyEdge image processing.

4) Find Countours selection: Contours are defined as the lines that connect all points along the boundary of an image that share the same intensity. Contours are valuable for shape analysis, determining the size of the object of interest, and object detection. The `findContours` function retrieves all the contours from the binary image, including those nested within other contours. This makes it easier to identify the specific contours of interest and understand the hierarchy and relationships between contours, such as parent-child associations. The author [18] also employs this method to identify the objects of interest.

There are three essential arguments in the `findContours` function. The first is the input image, where we use the image containing the edges of the grains. The second is the contour retrieval mode, where we select only the external contours of the grains. The third is the contour approximation method, where we choose `cv2.CHAIN_APPROX_NONE` to ensure that all boundary points are stored. Each individual contour is a Numpy array containing the (x, y) coordinates of the object's boundary points.

The grain contours are saved using the drawContours function to facilitate potential future analyses related to grain morphology.

```
contours,_ = cv.findContours(edge, cv.RETR_EXTERNAL,
    cv.CHAIN_APPROX_NONE )
grainContours = np.zeros(edge.shape, dtype='uint8')
cv.drawContours(grainContours, contours, -1,
    (255,0,0), 3)
```

The minEnclosingCircle function was used to find the smallest circle that can enclose each grain. After identifying these circles, they were drawn on the image containing the grain contours (Fig. 8). This approach allows for future comparative analyses between the morphology of the grain and a perfect circular shape, which could provide valuable insights in subsequent studies. For example, [16] uses the Ellipse Fourier Descriptor method for extracting shape descriptors for quantitative evaluation of various biological organs.

```
idx = 0

while idx >= 0:
    center = (0, 0)
    radius = 0
    (center, radius) = cv.minEnclosingCircle(cont[idx
        ])
    cv.circle(grainContours, (int(center[0]), int(
        center[1])), int(radius), (0, 255, 255), 2)
    idx = _[0][idx][0]
```

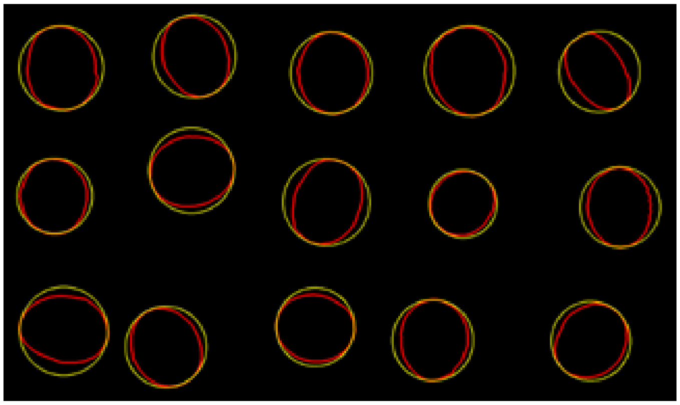

Figure 8. Contour of grains (red) and minimum circle around the grain (yellow).

To conclude the process, two images are saved for each grain: one without the background and another with a colored background. A rectangle is defined around each grain for demonstrate the cropping (Fig. 9), using the variable X to set the spacing from the center, which is detected using the findContours function. These separated images of individual grains are then used to create the dataset, providing a structured basis for future analysis and model training.

```
x = 30

cv.rectangle(whitoutBg, (int(center[0]-radius -x),
    int(center[1]-radius - x)), (int(center[0]+radius
     +x), int(center[1]+radius +x)), (0, 255, 0), 2)

cv.rectangle(whittBg, (int(center[0]-radius -x), int(
    center[1]-radius - x)), (int(center[0]+radius +x)
    , int(center[1]+radius +x)), (0, 255, 0), 2)
```

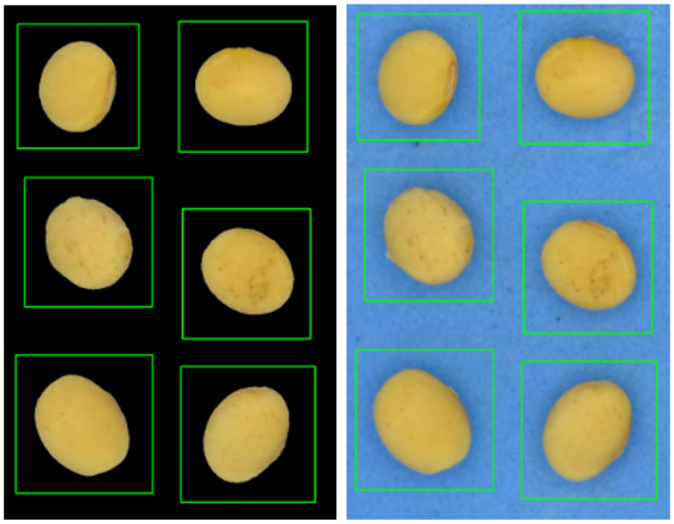

Figure 9. Grains separated with and without background.

III. RESULTS

In the algorithm, the CMYK color was selected to enhance the contrast of the soybean image, the yellow channels have the grains and the cyan have the background. Otsu adaptive thresholding was applied to segment the foreground and background of the enhanced image. The function `findContours` was used to locate those individual seeds on the binary image. Soybeans were marked according to the location of the variable `countours`. If the seeds were non-physically touching, those individual seed images were cropped out from the enhanced image and then resized as $300{\times}300$ pixels. If the seeds were physically touching, the sobel combined and canny edge function was applied to eliminate some tiny contact between seeds on the binary images after masking. Then, those seeds were relocated by selection. Finally, those individual seed images were cropped out in three formats of the grains were saved in the dataset, the individual image of the grain in color and with the background, the image of the grain without the background and the outline of the grain.

IV. Conclusion

In conclusion, this work successfully developed a comprehensive process for capturing and segmenting soybean grains using advanced image processing techniques. By utilizing functions such as `findContours`, `minEnclosingCircle`, and `thresholding`, we were able to isolate and enhance the morphological features of each grain. The creation of a structured dataset from these segmented images lays a solid foundation of input data for training machine learning models for grain classification.

Additionally, the use of both background removal and colored backgrounds provides flexibility for comparative analysis and further morphological studies, enhancing the potential for more precise grain quality assessments. This method not only contributes to automating grain classification but also improves the accuracy and reliability of the results.

This study presents promising results in the application of image processing and segmentation techniques for soybean grain classification. Although the approach demonstrated effectiveness, further studies are important to assess its efficiency compared to other available equipment and software configurations, which will be valuable in refining and optimizing the method. Additionally, a dedicated dataset is currently under development as part of this ongoing research. This dataset will be available for future testing and validation, aiming to support advancements in this field and to encourage reproducibility and broader adoption of these methodologies.

Acknowledgments

The authors would like to express their gratitude to the Foundation for Research Support of the State of Minas Gerais (FAPEMIG) for providing financial support through the COMPETE MINAS – LINHA TRÍPLICE HÉLICE APQ-03343-23 research grant, which was essential to the realization of this work. We also extend our thanks to the Virtual and Augmented Reality Group (GRVA) at the Federal University of Uberlândia, the Federal Institute of Goiás, and Grider Automação e Tecnologia LTDA for their significant contributions in providing the technological and structural support needed for this research.

Additionally, we acknowledge the assistance of AI tools in the drafting, organization, and translation of various sections of this paper into English. This integration of AI into our workflow highlights the growing potential of AI to enhance the research and academic production process, enabling us to achieve greater clarity, precision, and efficiency in our work.

References

[1] E. Avuçlu, S. Tasdemir, and M. Koklu, "A new hybrid model for classification of corn using morphological properties," *European Food Research and Technology*, vol. 249, 12 2022.

[2] F. J. Rodríguez-Pulido, D. F. Barbin, D. W. Sun, B. Gordillo, M. L. González-Miret, and F. J. Heredia, "Grape seed characterization by NIR hyperspectral imaging," *Postharvest Biology and Technology*, vol. 76, pp. 74–82, 2013. [Online]. Available: http://dx.doi.org/10.1016/j.postharvbio.2012.09.007

[3] K. G. Liakos, P. Busato, D. Moshou, S. Pearson, and D. Bochtis, "Machine Learning in Agriculture: A Review." [Online]. Available: www.mdpi.com/journal/sensors

[4] W. Lin, Y. Fu, P. Xu, S. Liu, D. Ma, Z. Jiang, S. Zang, H. Yao, and Q. Su, "Soybean image dataset for classification," *Data in Brief*, vol. 48, jun 2023.

[5] Nikon, "History of nikon products: 2010s," https://imaging.nikon.com/imaging/information/products_history/2010/, 2010, acessado em: Setembro 12, 2024.

[6] G. Laing, "Nikon d3100 review," 2024, accessed: 2024-09-20. [Online]. Available: https://www.cameralabs.com/nikon_d3100/

[7] W. Lin, Y. Lin, J. Phys, J. Chen, Z. Gao, and C. Huang, "Soybean image segmentation based on multi-scale Retinex with color restoration You may also like Underwater image enhancement algorithm based on Retinex and wavelet fusion Soybean image segmentation based on multi-scale Retinex with color restoration," *Journal of Physics: Conference Series*, vol. 2284, p. 12010, 2022.

[8] R. Andrade and C. A. Schneider, *A cor: teoria e prática*, Universidade Federal do Paraná, 2014, material da aula 2 do curso de Desenho Industrial. [Online]. Available: https://exatas.ufpr.br/degraf_rossano/wp-content/uploads/sites/16/2014/10/cor_aula_2.pdf

[9] U. F. do Ceará, "Padrão de cor rgb e cmyk," 2024, accessed: 2024-09-20. [Online]. Available: https://imprensa.ufc.br/pt/duvidas-frequentes/padrao-de-cor-rgb-e-cmyk/

[10] K. Kiratiratanapruk and W. Sinthupinyo, "Color and texture for corn seed classification by machine vision," *2011 International Symposium on Intelligent Signal Processing and Communications Systems: "The Decade of Intelligent and Green Signal Processing and Communications", ISPACS 2011*, pp. 7–11, 2011.

[11] Y. Li, J. Jia, L. Zhang, A. M. Khattak, S. Sun, W. Gao, and M. Wang, "Soybean seed counting based on pod image using two-column convolution neural network," *IEEE Access*, vol. 7, pp. 64 177–64 185, 2019.

[12] Y. T. Tovar, A. F. Calvo, and A. Bejarano, "Desarrollo de un sistema de clasificación de imágenes digitales para medir la humedad en granos de café," *Información tecnológica*, vol. 33, no. 3, pp. 117–128, 2022.

[13] G. Research, "Google colaboratory faq," 2024, accessed: 2024-09-20. [Online]. Available: https://research.google.com/colaboratory/faq.html

[14] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 9, no. 1, pp. 62–66, 1979.

[15] O. Team, "Opencv documentation," 2024, accessed: 2024-09-21. [Online]. Available: https://docs.opencv.org

[16] H. Vu, V. N. Duong, and T. T. Nguyen, "Inspecting rice seed species purity on a large dataset using geometrical and morphological features," *ACM International Conference Proceeding Series*, pp. 321–328, 2018.

[17] J. Canny, "A computational approach to edge detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-8, no. 6, pp. 679–698, 1986.

[18] T. Matos Maruyama, M. Hosoya Name, J. Rissa Franco, and R. Falate, "Development and Validation of a Method for Measurement of Root Length in 2D Images," *IEEE Latin America Transactions*, vol. 16, no. 3, pp. 940–947, mar 2018.