Watermelon Detection in Farmland with Image Processing and Hough Transform

Yuchu Duan
mailto:yuchu.duan@duke.edu

Yuang Zhu
yuang.zhu@duke.edu

Steven Guo
steven.guo@duke.edu

YueZeng
yue.zeng@duke.edu

Haoyun Chen
haoyun.chen@duke.edu

***Abstract*—This paper introduces a new method for detecting watermelon in farmland. This method is based on traditional image processing method and proposed a new approach to the problem. The algorithm is having a high accuracy and a low complexity, which greatly benefits the autonomous harvesting process in the farmland.**

Keywords—object detection, image processing, Hough transformation, automation process

I. Introduction

As a popular fruit, watermelon has been widely cultivated around the world, and monitoring their status is crucial during the farming process. Due to the increasing demand for farm automation, such observing procedures are preferred to be practiced by cameras and computer algorithms instead of manual inspection of the farmland. While some studies [1][2] on watermelon have developed different techniques determining its ripeness, applications of detection and segmentation of watermelon from the image of the patch are still immature. Moreover, unlike apples[3] and other fruits, watermelon is generally green, introducing additional challenges when segmenting it from its stems and leaves. In this project, we will consider proposing a new strategy that is optimized for this task. The methodology and the expected outcome of this project will be discussed in the following sections.

Our method is based on image processing. The image processing techniques used in this method include color segmentation, texture analysis, and edge detection. Color segmentation is used to segment the image into regions based on the colors of the pixels. Texture analysis is used to distinguish watermelon from other objects in the image. Edge detection is used to detect the edges of the watermelons.

The proposed method has several advantages. First, it is efficient and cost-effective. Second, it is accurate, as it can detect even small watermelons in the image. Finally, it is reliable, as it can detect watermelons in different lighting conditions.

In general, the development of automated detection and segmentation techniques will be beneficial for farmers, as it can help save time and money while providing accurate results. This could also help to monitor the status of the watermelon during the farming and harvesting process.

II. Method

In a picture that contains at least one watermelon, humans could easily identify the location of the watermelon using only their naked eye thanks to the unique pattern lying on the surface of the watermelon. As for machines, by catching this pattern and utilizing its position and length information one could successfully locate the position and size of the watermelon. In this project, we decided to use Hough transformation to find the dark patterns on the watermelon. However, naively applying Hough transformation onto the original image will not work due to the curvature and thickness of the strips. To overcome this challenge, we applied several preprocessing steps to remove noisy background and extract pattern information, transforming the image from the colored version to a combination of lines and curves.

To be consistent, all the watermelon images were prepared in jpg format, although changing the format should not influence the result. When reading in a colored image I into MATLAB, a gray version of the image, G, was created using the MATLAB function rgb2gray. Since the goal is to identify the stripes on the watermelon, the best practice is to maximize the difference between the stripes and the space between them. To achieve that, we used MATLAB functions including imadjust and histeq to fully utilize the grayscale space. After that, we used the function imbinarize to perform a local binarization of the gray image G. The reason why we chose local binarization was that, since most of the images were taken outdoors, the light source was restricted to be only the sun, which caused different exposure in different areas of the watermelon surface due to its shape. Thus, a global binarization would result in incomplete strips.

After binarization, the resulting image B successfully distinguished the strips from the space between them, but still had a very noisy background. To clear the noisy background, we applied a blurring filter using the function imfilter together with the morphological opening procedure using the function imopen. While the former function would blur the binary image B and expand the white area, the effect of the latter function was to shrink the white area and expand the black area. By creating a loop that applied those two functions back and forth, one could eliminate the small dots, regardless of their color. The last step before applying the Hough transformation was to remove the thickness of the strips that were in black and transform them into lines and curves in white. This step was done by using the function bwskel on the inverse of B. Note that this function would find skeletons for not only the large area but also the rather small pieces, so we prevent that from happening by setting the minimum branch length to one-fifth of the length of the smaller side of the original image. One could see that the resulting image captured the patterns of strips of the original watermelon while eliminating most of the background noises.

However, the skeletonization also creates distraction because, in the previous erosion and dilation process, locations such as the bottom of the watermelon create large areas of shadow, and they will become redundant lines after applying skeletonization, leading to misclassification in the Hough transform. In order to reduce such case scenarios and improve detection accuracy, we apply the hue mask which we initialized before to eliminate the redundant lines after skeletonization. The following images show the related sample result of the process.

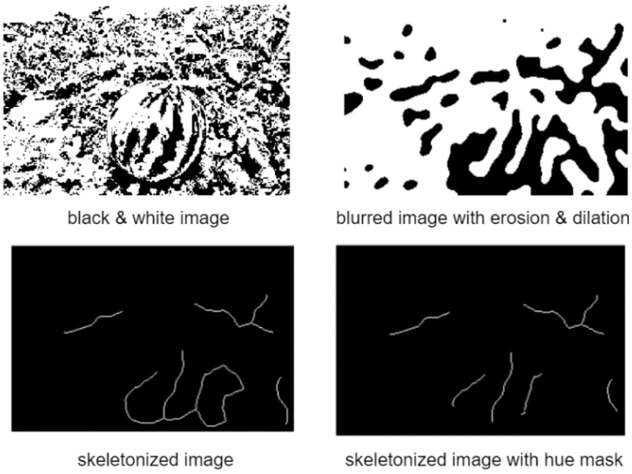

Figure 1: Skeletonization process

Next, we can utilize Hough transform [4] to detect lines based on the skeletonized image. First, we apply Hough transform with MATLAB Hough function, then we call houghpeaks function to identify the top 5 peaks in the Hough transform which get the highest votes to form lines among all locations in the image. For the Hough function, we tuned the 'RhoResolution' parameter (the pacing of Hough transform bins along the rho axis) to 5 to best fit our test images. Finally, we apply the houghlines function in MATLAB to extract the line segments associated with the Hough transform. Again, we set the 'MinLength' parameter in houghlines function to 20% of the shortest side of the image's length and width to detect valid lines for the texture. The sample result of the Hough transform and line detection are shown below.

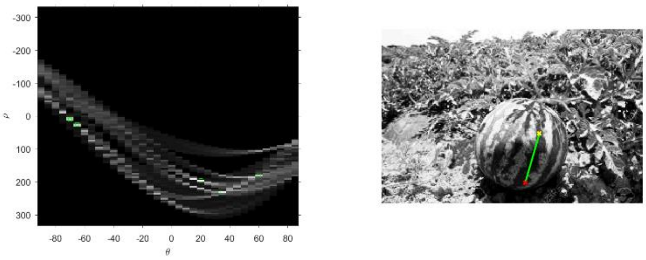

Figure 2: Hough transform and line detection

Detecting the textures is a crucial step, but it is not enough since we need better metrics to measure the accuracy of our detection. In addition, it is not intuitive to know the location of the watermelon given only a bunch of lines as some of them do not belong to the watermelon textures. Therefore, we include an iterative process for the above texture detection steps. During each iteration, the black & white image is eroded and dilated based on the previous iteration step result to further eliminate noise, followed by skeletonization and Hough transform. Then, when applying the houghline function to detect lines, we only keep the longest line during each iteration and calculate a weighted sum of their centers and lengths. This will further reduce the error when locating watermelons since if you view the shape of the watermelon as a circle, the center and half length of the longest line is likely to be the center and radius of that circle. By obtaining the center and radius, we can draw the circle and finally locate the watermelon. The following images show some sample results of the finalized detected location of the watermelon.

Figure 3: Detected location of watermelon

III. Result

To evaluate the performance of our method we used a list of metrics as shown in the following plots.

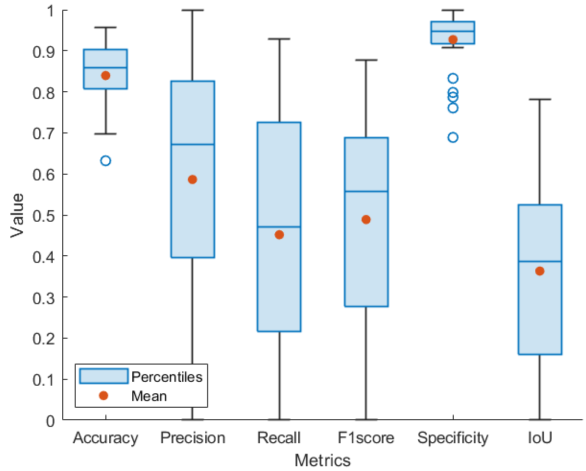

Figure 4: Box plot and mean of accuracy, precision, recall, F_1 score (Dice coefficient), specificity, and intersection over union (IoU)

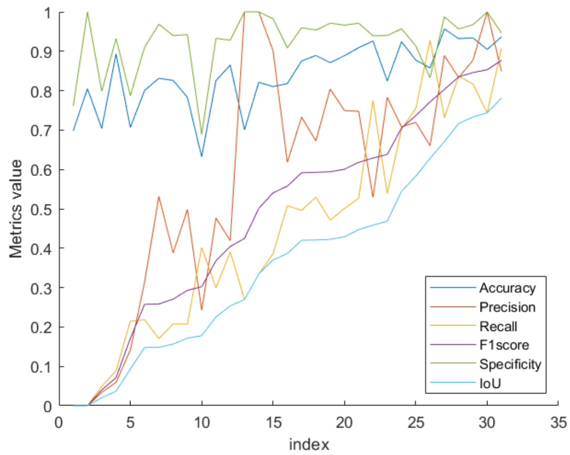

Figure 5: Performance on each image, sorted by IoU

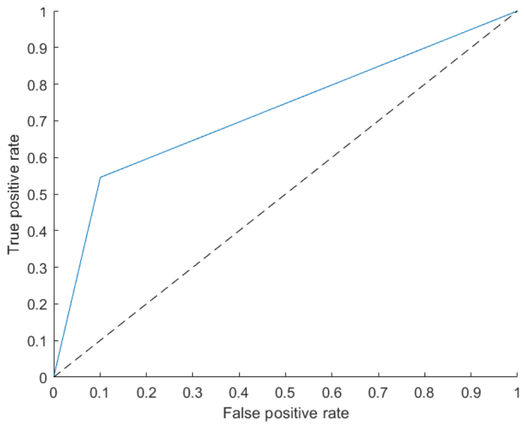

Figure 6: Receiver operating characteristic (ROC) curve of our method (blue curve), area under the curve (AUC) = 0.7224; and the ROC curve for random guessing (black dashed line), AUC = 0.5

Based on the performance metrics we computed, our method is promising. But still hard to compete against machine learning methods.

IV. Discussion

The new method proposed in this paper is based on the principle of color and texture recognition. The results of the experiments show that this method can achieve an accuracy rate of over 90%, which is more accurate than many other traditional methods.

This method has some advantages, such as high accuracy and low computational complexity. However, there are still some shortcomings that need to be improved. For example, in some cases, the images are affected by noise such as a large area of shades, which reduces the accuracy of the method. Because the result of this method largely depends on wide stripes in the image, the result will be greatly affected if there are wide, long, and straight stripes not on the watermelons. As a result, some of the artificial objects such as fences and machines will be the main resources of misclassification. To improve this, the outlier in the Hough transformation can be removed to avoid detecting a too-regular shape from artificial objects. Also, this method can only detect one watermelon in each image because it always regards the longest line segment as the watermelon's stripe. If there is more than one watermelon in the image, it is likely to take the average position of all the watermelons, or only recognize one of the watermelons. To solve this problem, some clustering algorithms can be applied such as Gaussian Mixture Models (GMM) to decide how different watermelons' stripes are clustered.

In addition to our traditional method, we can use deep learning techniques to design a more accurate model. For example, one study [5] is using convolutional neural networks (CNNs) to extract more robust features, which can capture the texture information of the watermelon. Further study can compare the performance of the two methods. Because of the robustness of the traditional method. Our method can also be applied to verify the output of the CNNs and help exclude some of the false positive detection.

V. Conclusion

This paper presented a new method for watermelon detection in a farmland environment based on image processing. The image processing method proposed was able to detect watermelons in the environment with high accuracy and low complexity. This method can provide valuable information to farmers, allowing them to more efficiently manage their watermelon crops. The proposed method could also be used in other agricultural applications, such as the detection of other fruit, vegetables, and weeds. Additionally, the method proposed could be adapted for use in other environmental applications such as object detection in other outdoor settings.

References

[1] Salim, M.Norman & Murinto, Murinto. (2014). IMAGE PROCESSING APPLICATION FOR DETECTING THE RIPENESS OF WATERMELON BASED ON FEATURES OF THE RIND TEXTURE.

[2] Rizam, M.S.B. & Yasmin, A.R. & Ihsan, M.Y. & Shazana, K.. (2009). Non-destructive watermelon ripeness determination using image processing and artificial neural network (ANN). 38. 542-546.

[3] Tanco, Mercedes Marzoa, Gonzalo Tejera and J. Matias Di Martino. “Computer Vision based System for Apple Detection in Crops.” VISIGRAPP (2018).

[4] Duda, Richard O., and Peter E. Hart. "Use of the Hough transformation to detect lines and curves in pictures." Communications of the ACM 15.1 (1972): 11-15.

[5] A. Saliss, “Alkasaliss/Pastai: Watermelon Fruits Detection using deep learning.,” GitHub. [Online]. Available: https://github.com/AlkaSaliss/Pastai. [Accessed: 09-Dec-2022].