24-678: Computer Vision for Engineers

Ryan Wu

ID: weihuanw

PS3 Report

Due: Sep 29 2023

This file contains the following:

PS3-1 Image improvement via area-to-pixel filters

* pcb-improved.png
* golf-improved.png
* pots-improved.png
* rainbow-improved.png
* readme.txt
* source code file(s) (attached to the end)

PS3-1 Edge detection

* cheerios-sobel.png, cheerios-canny.png
* professor-sobel.png, professor-canny.png
* gear-sobel.png, gear-canny.png
* circuit-sobel.png, circuit-canny.png
* readme.txt
* source code file(s) (attached to the end)

**Using 1 late day for this assignment**

**PS3-1 Information on filter combinations used**

Median filter:

* kernel size: 5

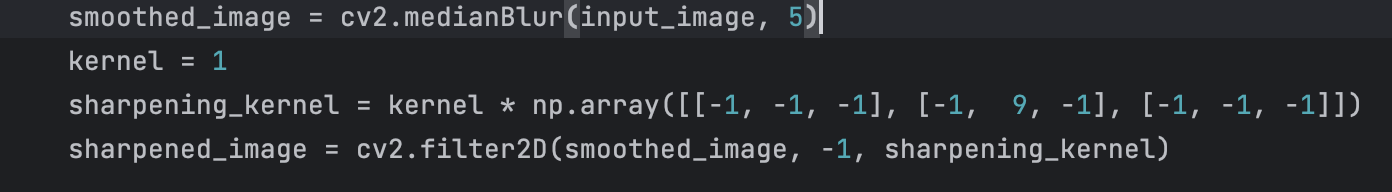


Figure 1. Code used for median filtering.

Bilateral filter:

* pixel value: 9 by 9 neighborhood
* : 75

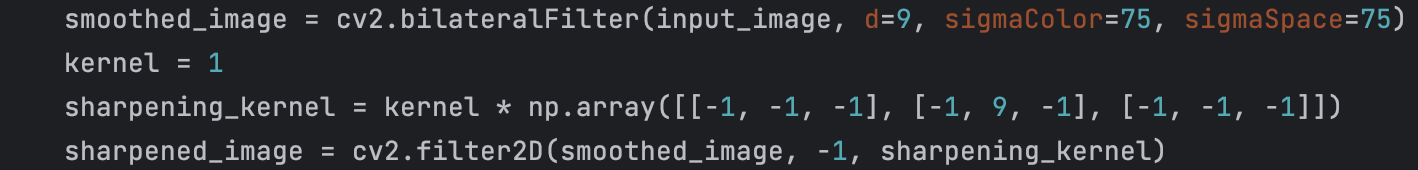


Figure 2. Code used for bilateral filtering.

Sharpening filter:

* kernel size 1

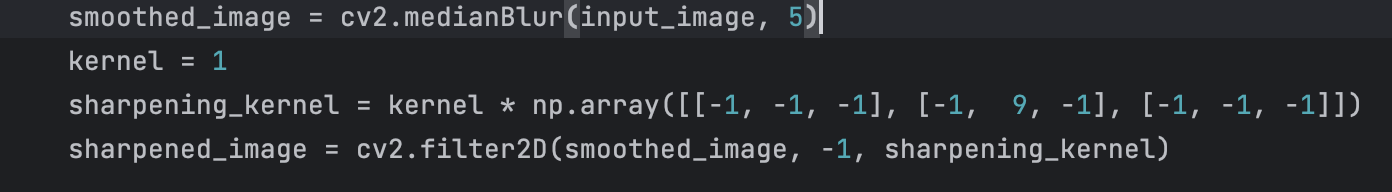


Figure 3. Code used for sharpening filtering.

**PS3-1 PCB image (filter combination in order: median & sharpening)**

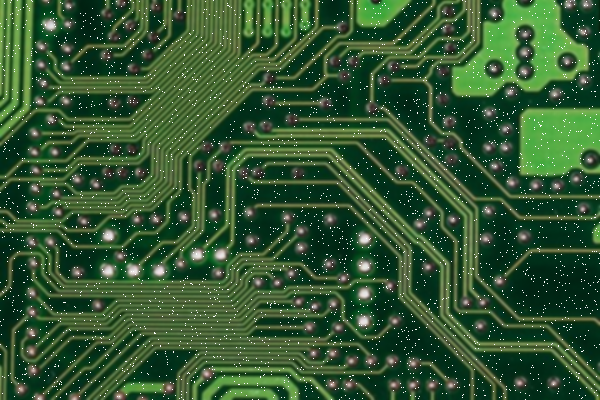
****

Figure 4. The given PCB image without filtering.



Figure 5. The improved PCB image with filters applied.

**PS3-1 Golf image (filter combination in order: median & sharpening)**



Figure 6. The given golf image without filtering.



Figure 7. The improved golf image with filters applied.

**PS3-1 Pots image (Filter combination in order: median & sharpening)**



Figure 8. The given pots image without filtering.



Figure 9. The improved pots image with filters applied.

**PS3-1 Rainbow image (filter combination in order: bilateral & sharpening)**



Figure 10. The given rainbow image without filtering.



Figure 11. The improved rainbow image with filters applied.

**PS3-1 readme.txt**

24-678: Computer Vision for Engineers

Ryan Wu

ID: weihuanw

PS3-1 Image improvement via area-to-pixel filters

Operating system: macOS Ventura 13.5.2

IDE you used to write and run your code: PyCharm 2023.1.4 (Community Edition)

The number of hours you spent to finish this problem: 6 hours.

**PS3-2 Information on edge detection method used**

**Sobel:**

* Horizontal Sobel matrix:
* Vertical Sobel matrix:

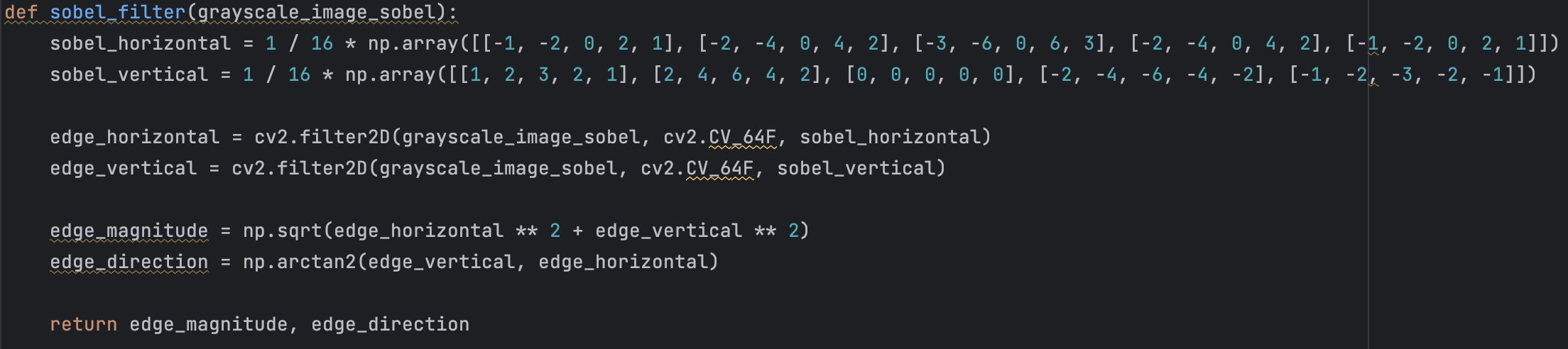
****

Figure 12. Code used for Sobel edge detection.

**Canny edge:**



Figure 13. Code used for Canny edge detection.

**Findings and discussion:**

In all the below comparisons, both Sobel and Canny edge detection method were used on each given image. The result shows that Sobel gives a slightly thicker edge border and also provides gradient information on both the horizontal and the vertical directions. However, Canny edge detection offers high-quality, well-localized edges and also reduces noise.

I would recommend using Sobel method when tasked with simpler image inputs that requires fast results. On the other hand, I would recommend using Canny edge method for more professional edge detection task.

**PS3-2 Cheerios (Sobel and canny edge detection)**

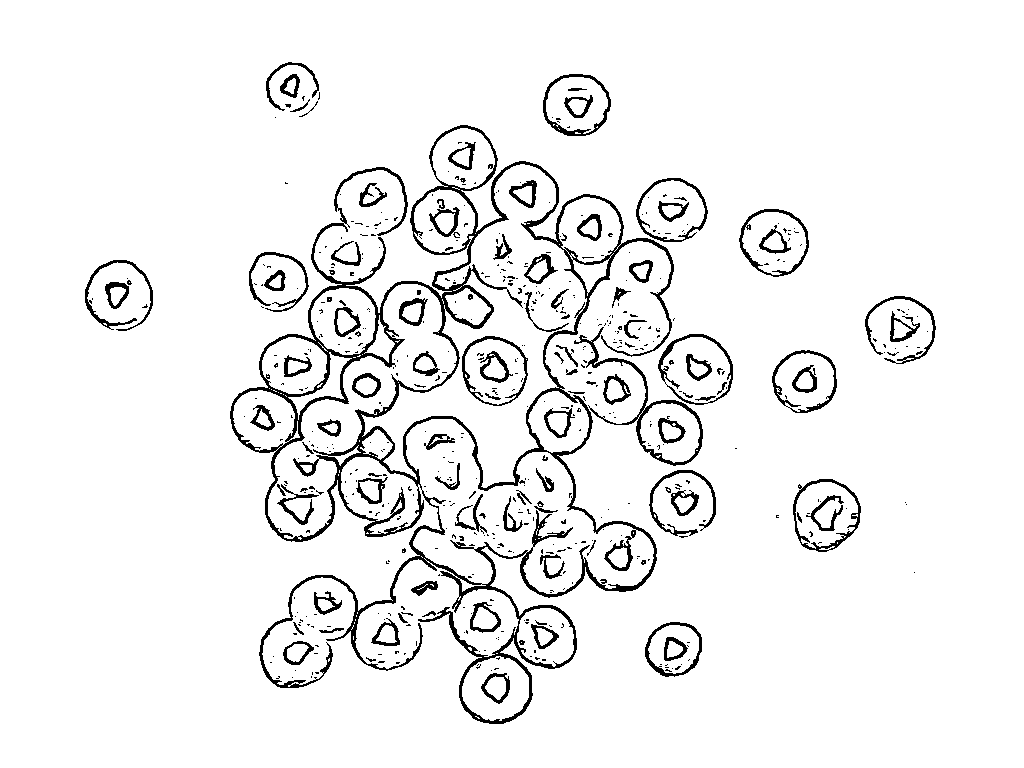


Figure 14. Cheerios binary image (Threshold: 195) with Sobel filter applied.

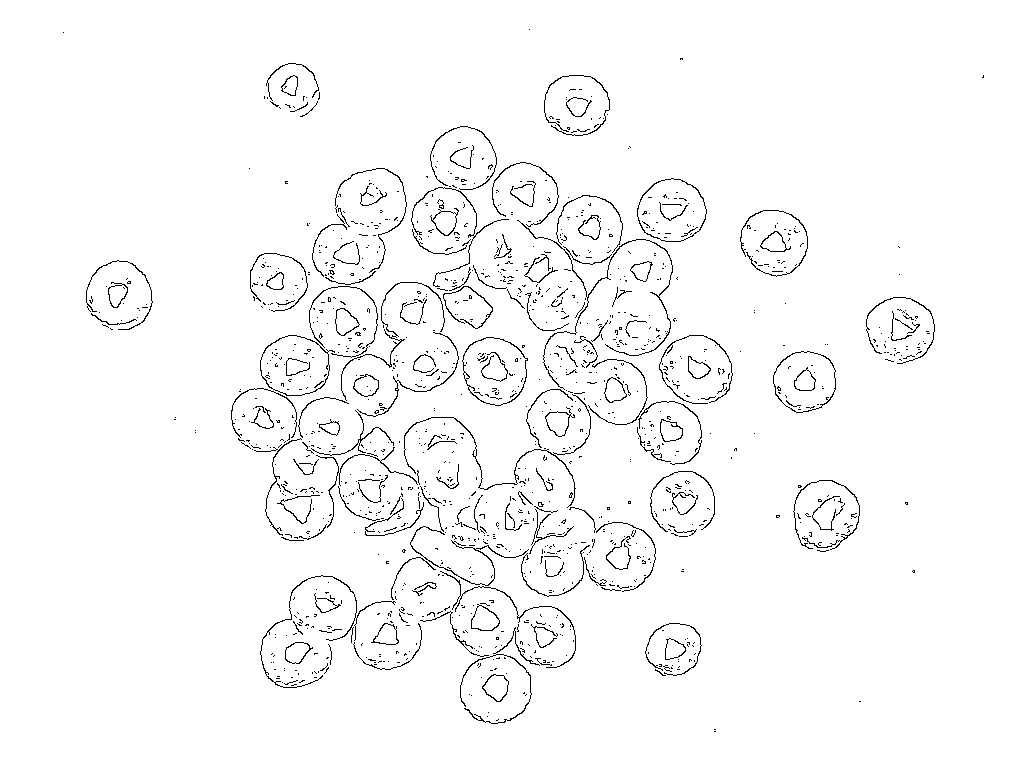


Figure 15. Cheerios image with canny edge detection applied. (Threshold 1: 255, Threshold2: 255, Aperture: 3, and using L2)

In Figure 14 and Figure 15, the given cheerios image is a mixed of simple and complex image structure with several edges. In this case, it will depend on your computational power and availability in using Sobel or Canny edge detection.

**PS3-2 Professor image (Sobel and canny edge detection)**



Figure 16. Professor binary image (Threshold: 220) with Sobel filter applied.



Figure 17. Professor image with canny edge detection applied. (Threshold 1: 150, Threshold 2: 50, Aperture: 3, and not using L2)

In Figure 16 and Figure 17, the given professor image can be considered a more complex image with many edges. In this case, the Canny edge detection would be the recommend method.

**PS3-2 Gear image (Sobel and canny edge detection)**

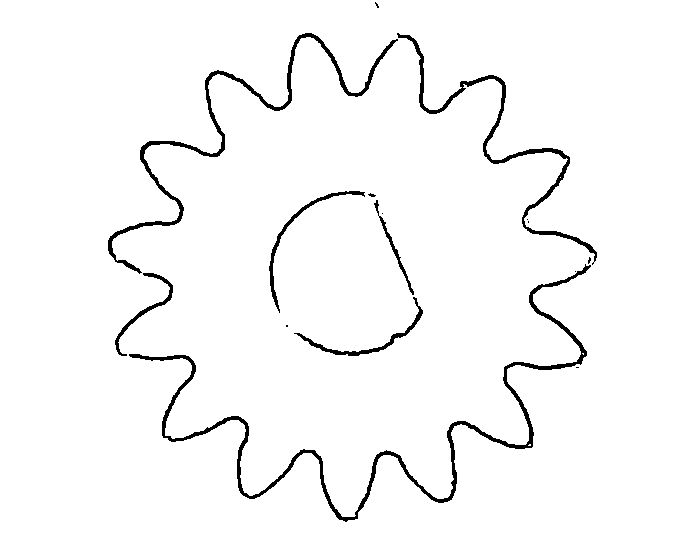


Figure 18. Gear binary image (Threshold: 155) with Sobel edge detection applied.

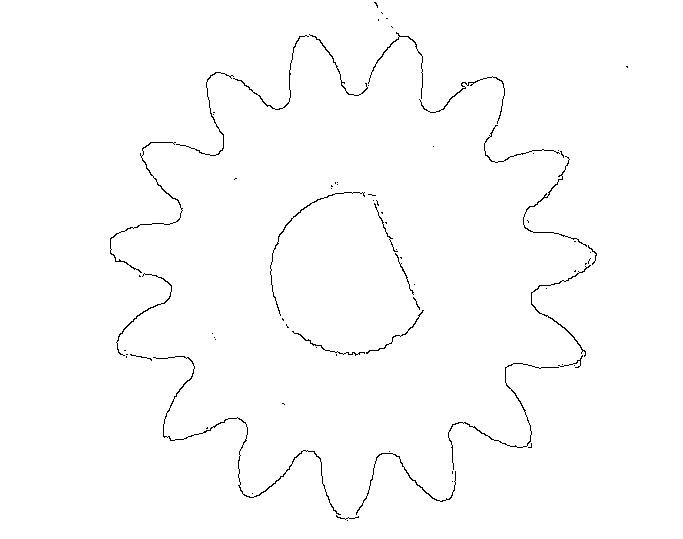
****

Figure 19. Gear image with canny edge detection applied. (Threshold 1: 255, Threshold 1: 255, Aperture: 3, and not using L2)

In Figure 18 and Figure 19, the given gear image can be considered a simpler image with less edges. In this case, the Sobel edge detection method would be sufficient.

**PS3-2 Circuit image (Sobel and canny edge detection)**

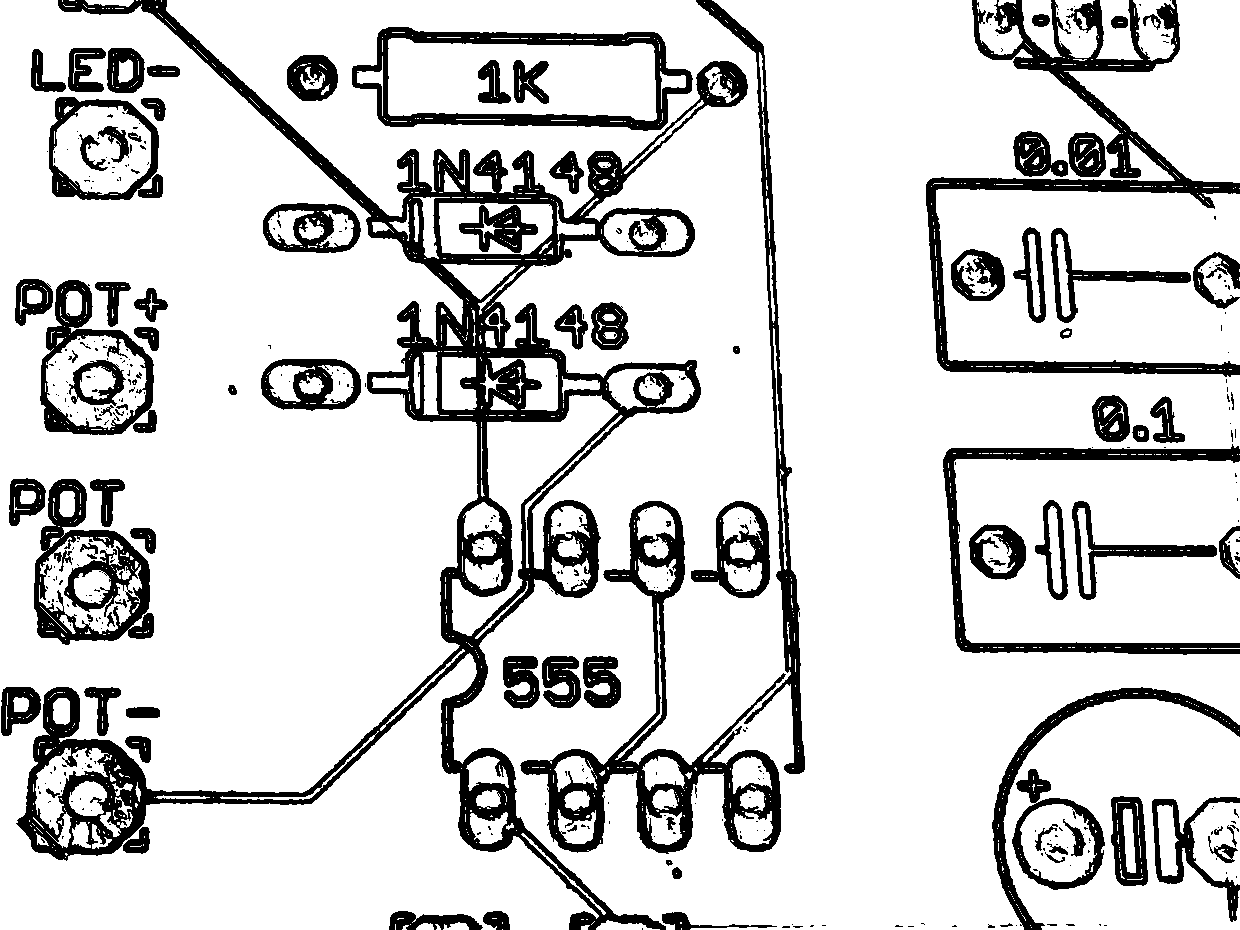


Figure 20. Circuit binary image (Threshold: 240) with Sobel filter applied.

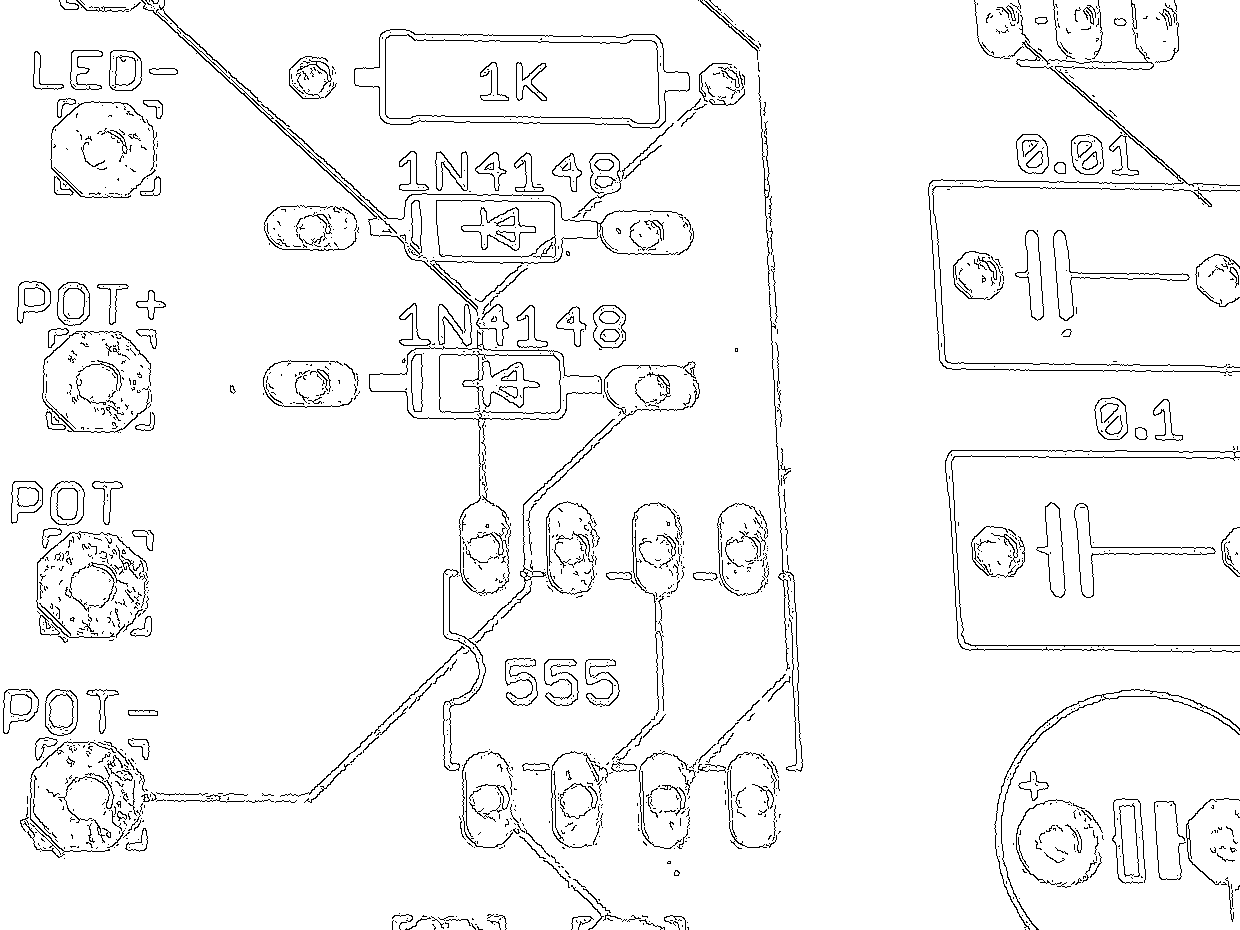
****

Figure 21. Circuit image with canny edge detection applied. (Threshold 1: 90, Threshold 2: 60, Aperture: 3, and not using L2)

In Figure 20 and Figure 21, the given circuit image can be considered a more complex image with many edges. In this case, the Canny edge detection would be the recommend method.

**PS3-2 readme.txt**

24-678: Computer Vision for Engineers

Ryan Wu

ID: weihuanw

PS3-2 Edge detection

Operating system: macOS Ventura 13.5.2

IDE you used to write and run your code: PyCharm 2023.1.4 (Community Edition)

The number of hours you spent to finish this problem: 6 hours.