**Introduction**

The following report summarizes 2 major efforts. The first is an image processing task in which OpenCV is used to mask out an image of a motherboard and separate it from its background. The second involves training a YOLOv8 model to detect and classify various hardware components from a PCB. These techniques have the potential to greatly enhance the manufacturing efficiency of these components, as they can replace time-consuming manual inspections.

**Object Masking**

A black and white photo of a computer motherboard

Description automatically generatedIn order to generate the image mask, OpenCV tools were extensively utilized. After loading the image and converting it to grayscale, the first major step was to apply a gaussian blur to the image. This has the effect of reducing sharp details in the image, ensuring only the large “structural” features remain. The next step was to threshold the image, creating distinct regions based on pixel intensity. For this step, OTSU thresholding was used, meaning the threshold pixel value was automatically generated. Presented below is the thresholded image.

Figure 1: Thresholded image

From this, it can be seen that certain regions of the mother board, like the heatsinks on the left-hand side, are not sufficiently distinct from the background and as such appear as white. This adversely affects the edge detection. Additionally, as the image was taken at a slight angle, there is a shadow casted on the lower left-hand side. This creates a region that is interpreted as part of the motherboard. It is recommended that future images be taken on a distinct background, from an overhead angle, and with sufficient lighting. This would ensure that a cleaner mask can be created, to better extract the motherboard.

After thresholding the image, Canny edge detection was used, generating the following.

A black and white drawing of a computer motherboard

Description automatically generatedFigure 2: Detected Edges

Once the edges were detected, a morphological operation was applied to close edges. A large kernel size was used, as it had the effect of connecting the jagged edges around the motherboard, leading to a more complete mask.

A black rectangular frame with a white background

Description automatically generatedAfter closing the edges, contour detection was performed, and the largest contour was filtered out in order to draw the mask presented next.

Figure 3: Mask Image

A close-up of a computer motherboard

Description automatically generatedLastly, the mask was used to extract the final image of the motherboard.

Figure 4: Final extracted image

As seen in figure 4, the mask obviously isn’t exact. The two main areas of improve as mentioned above would be using a distinct background to better detect components at the edges and using lighting and angles to eliminate shadows.

**Model Training – Setup**

**Model Training - Results**

**Model Evaluation**

**Conclusion**