Assignment 3: Making your own object detector!

As we saw in class, binary image processing begins to give us a handle on some core vision tasks like detecting, localizing, and identifying simple shapes. In this part of the assignment, you will put this into practice by building a tool that can respond to queries like “how many small green rectangles are there in this image, and where are they?”

(And you don't need machine learning for this!)

During development, you’ll use the shapes.png image given to you here

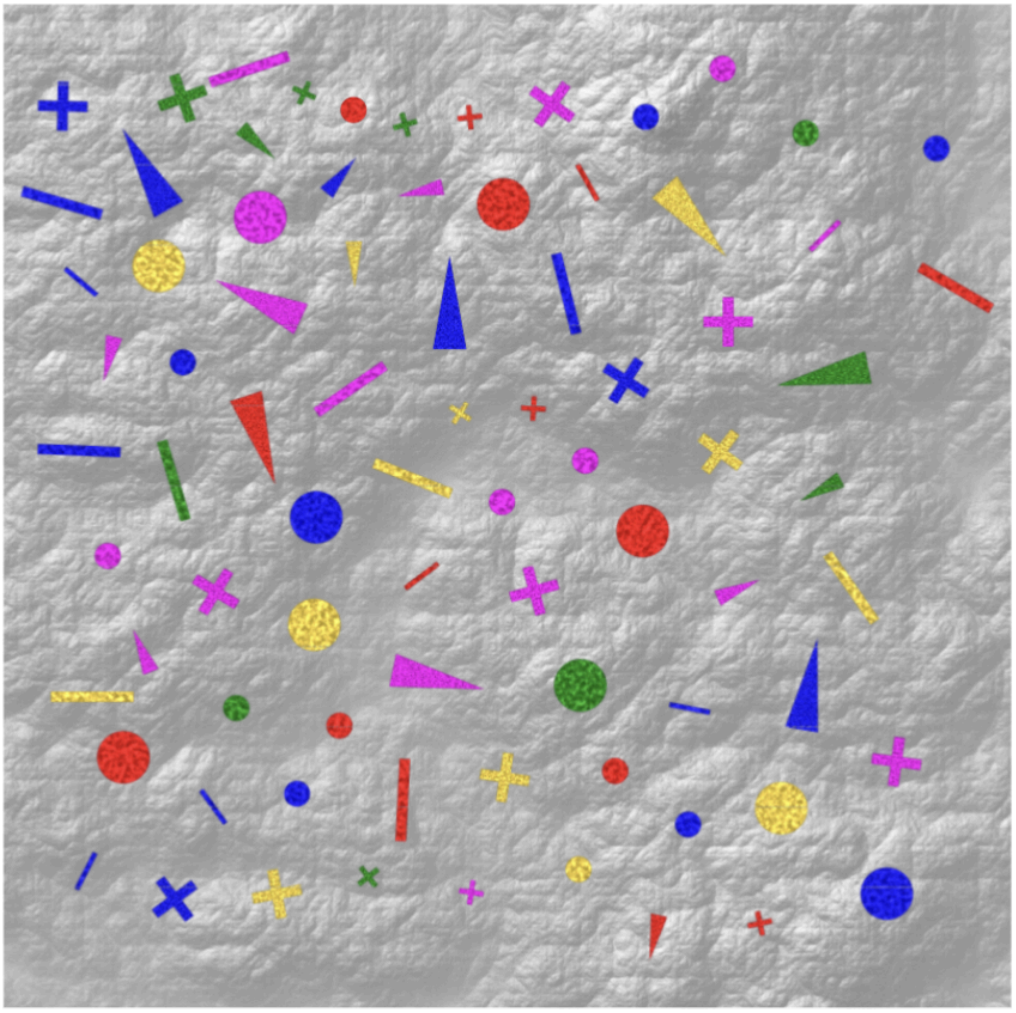

Figure 1: shapes.png

Some important notes to consider:

- The shapes will not overlap with each other or occlude each other
- The shapes will always be one of “rectangle”, “circle”, “wedge”, or “cross”, and they will always have the same relative proportions as those you’re seeing here
- Each shape will come in roughly one of two sizes - small or large. However, the exact threshold for what counts as ‘small’ or ‘large’ will vary and has to be determined
- In general, they will be one of the 6 primary hues (red, yellow, green, cyan, blue, magenta), plus or minus some slight hue variation and texture-like noise (as in the example)

You are given code that runs and is already commented and factored in a way that suggests how it should work, but contains a number of subtle bugs and omissions. Your job is to **find and fix all the bugs**.

There are no bugs in annotate_locations or in the if __name__ == "__main__": block. Focus on the other functions in the shape_finder.py file.

Collaboration and Generative AI disclosure

As always, did you collaborate with anyone? Did you use any Generative AI tools? Briefly explain what you did in the collaboration-disclosure.txt file

Examples of correct outputs

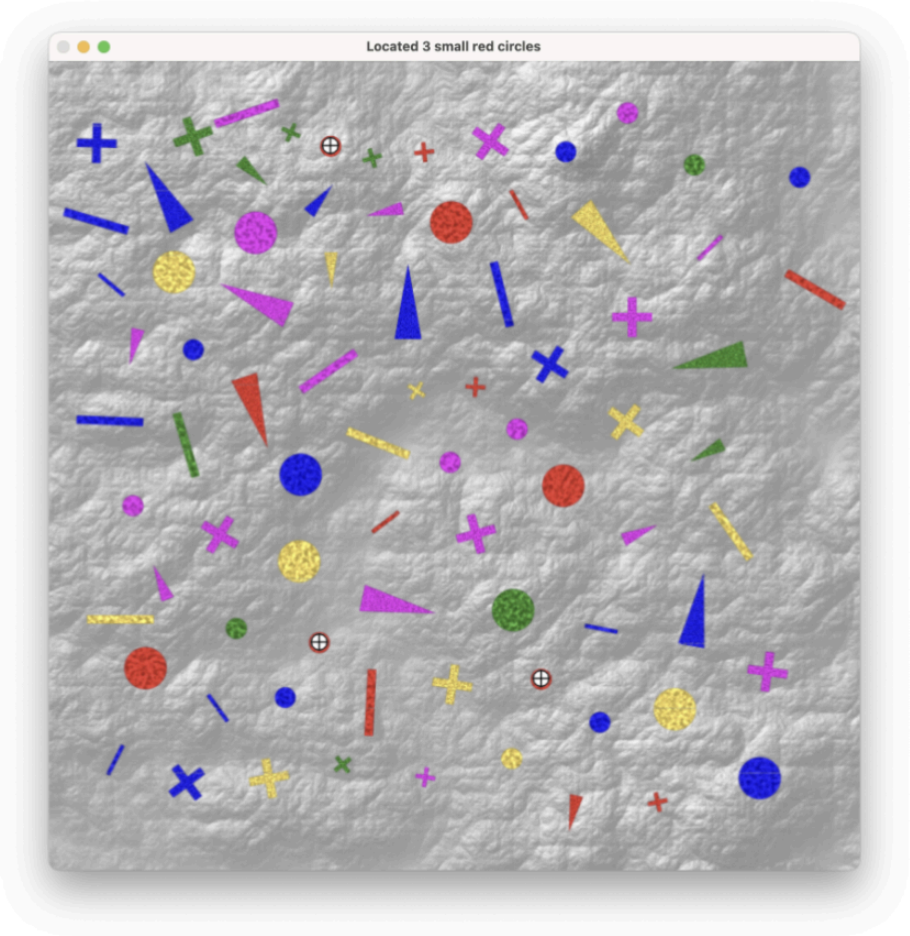

Figure 2: Example output of `python shape_finder.py shapes.png small red circle`

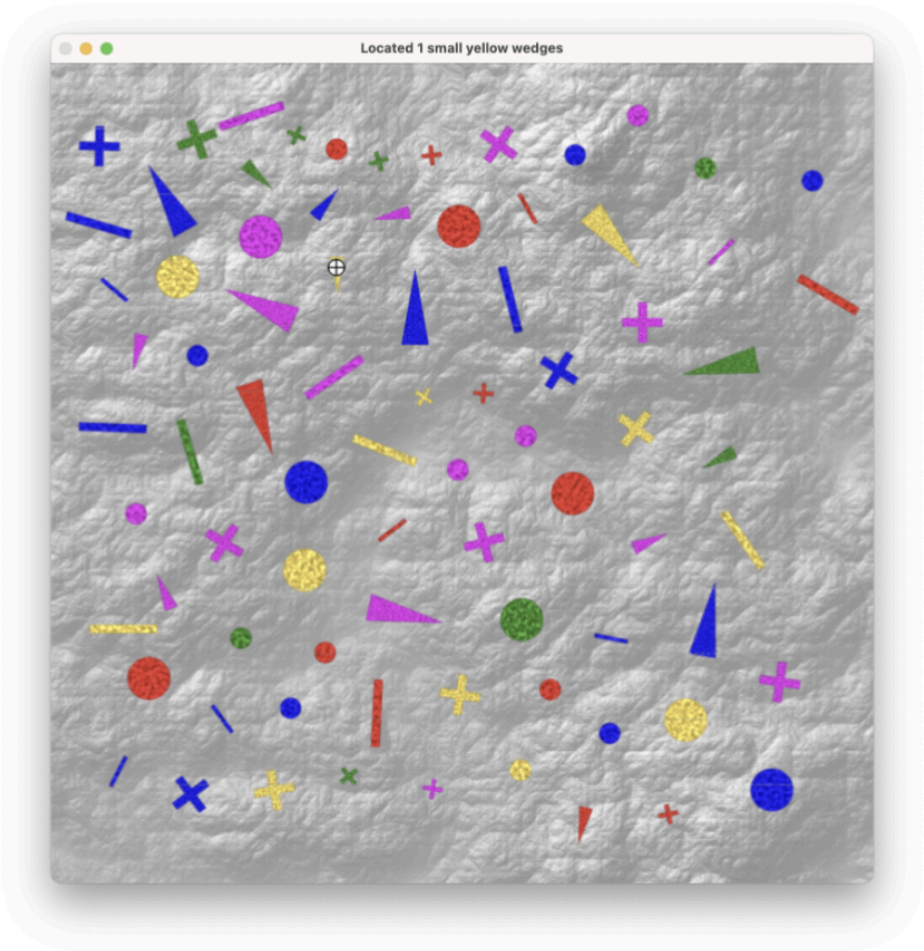

Figure 3: Example output of `python shape_finder.py shapes.png small yellow wedge`

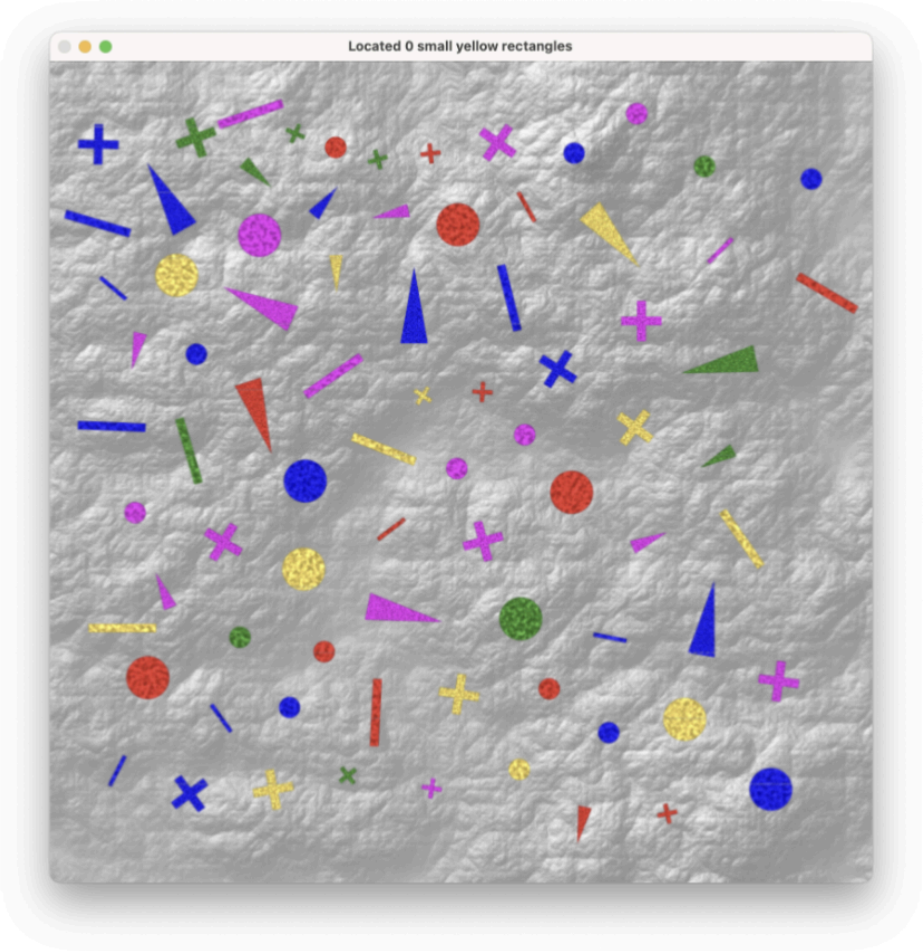

Figure 4: Example output of `python shape_finder.py shapes.png small yellow rectangle`

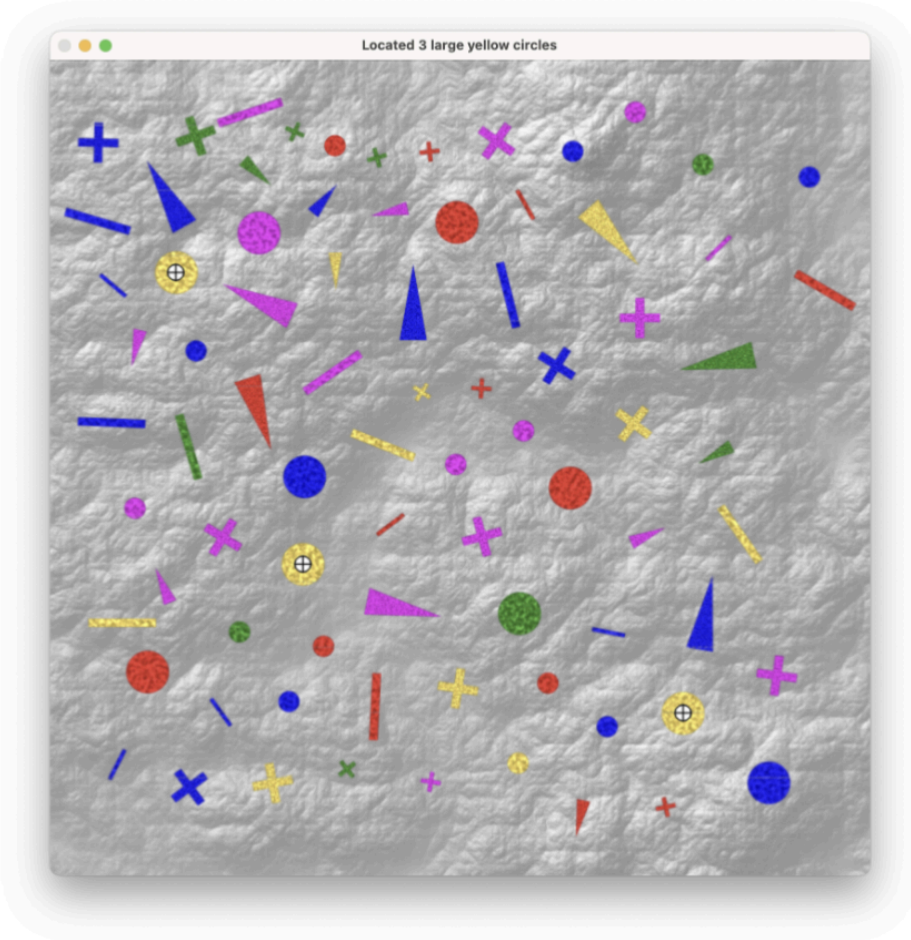

Figure 5: Example output of `python shape_finder.py shapes.png large yellow circle`