Ikhlas Attarwala / 10-08-2019
EECS 332 – Introduction to Computer Vision

I apologize, I was in a rush to finish so I didn't make the program very neat, but everything works!

When you open my ***main.py*** file in the command line, type in ***python main.py*** and it will run a simulation for every image that I wrote into the bottom the ***main.py*** file. If you would like to test other images, you will need to add the image to the folder and edit the ***main.py*** code to include the image name at the bottom of the python file. The function ***graph_normal('gun.bmp')*** will detect all objects, and the function ***size_filter(400, 'gun.bmp')*** will detect objects by filtering away pixel areas of size 400 or less, thus we can filter the noise away.

Connect Component Labeling is one of the various computer vision techniques to detect connected regions in an image. There are various methods and algorithms that can be used to achieve CCL, such as recursive tracking, parallel growing or row-by-row (aka sequential labeling). This homework assignment utilizes a **sequential algorithm** to scan across the image after it has been processed into a 2d image. In the code, I first import the image and transform it into an array of int type. Because this image is black and white, the values per pixel were binary, either 0 or 1. If I had color images, I would have used a different technique to get the sensitivity or RGB values. After I create a 2-dimensional binary array, I scan across the rows from left to right, and from up to down. When I reach a 1, I assign it a label to group it. If there is no labeled number to the left or above this 1, I will assign it to a new group, such as group 15, for example. If there was a labeled 1 to the left or above it, I would have assigned that group to our 1. If both the left and top have different labels, I give priority to the group above. Now we continue to move to the right, and if this number is a 1 and has not been assigned to a group yet, this number will also be assigned to group 15 as long as there is no group above it (priority). Eventually I will hit a 0, but once I hit a new 1 that is not connected to another number from the left or above, I will assign it to a new group, such as group 16. This pattern continues to the end of the row, and then repeats from the next row until every 1 in the 2d array has been labeled into a group. This was the first pass.

Now we'll conduct a second pass that again begins from the top-left going down to the bottom-right, but this time, we observe the groups to the left, right, above, and below the immediate target coordinate. If there is another labeled group that is adjacent to this label and is different from the target coordinate's group, we assign them to the same group (using an equivalence table), because they are **connected**. It is important to note that observing 4 adjacent positions is called 4-connectivity, while observing the labels of coordinates that are in 8 adjacent positions (top, top-left, left, bottom-left, bottom, bottom-right, etc.) is called 8-connectivity. In order to connect in the program, I developed Nodes, such that every coordinate was connected to the coordinate that initially helped labeled it in the first pass. As we venture along the parent nodes, we can find the root. Once this second pass was complete, each number was assigned to a root-label group. For example, if group 1 and group 2 and group 3 were connected, all of groups 1, 2, and 3 were assigned to the root group, which in this example was group 1. This allows us to connect individual components within an image.

One such example of how this technique can be useful can be seen with the use of radiology. If you have a CT-scan of the human brain and are observing the separated regions within the gray matter, we can label the different elements and organs within the brain, and then identify them with greater ease. The results showed what I expected to see, the ***test.bmp*** ran without error. ***face_old.bmp*** had additional labeled groups, but I realized that was because the picture had holes in it (observe the gaps in the smile and eyebrow). This wouldn't have been a problem if I used 8-connectivity. Without the program's results I would not have noticed otherwise. Finally, the ***gun.bmp*** works great both by normal sequential labeling methods and also by filtering the size of objects, so that I can reduce the noise in an image. Sorry the background is purple. Results are in the folder.