Analog Clock Reader

In my project, I have used grayscaling, erosion, canny detection and hough transform algorithms to detect lines on the clock at first. Applying these four methods were not enough to extract the necessary data. The only prior information is that we are assuming that the clock images fill at least 90% of the image and are close to the center. Therefore, I have used this information to extract hour and minute hands. Since we expect them to intersect or be close to the center, I have put a point to the center and calculated point-to-line distance and extracted only the lines that are close to the center.

The next problem is that hough lines only have beginning and ending points, also we don't know about their orientation. For example the lines would point to “10” while they are actually pointing to “4”. I have fixed this by ensuring that the end points are the farthest points to the center. If a hough line's distance between beginning and the center is more than the distance between the ending point and the center, I simply swap them.

After that, there was another problem where the hough transform would fit more than one line into one hand. K-means was a solution to this but OpenCV's k-means needed a specific format so instead of converting my data, I have implemented my own simple clustering algorithm for this project which eventually does the same thing as k-means.

In the next step, I had to discriminate hour and minute hands. The most important problem in this was that the hough transform could fit fragmented small lines into one hand that constitute a whole line which prevents me to determine it's length. So I have modified my clustering algorithm so that it also kept the farthest point (from the center) on those fragmented lines. I have also used C++11 lambdas to sort those points assuming that the hour hand is the shortest and the minute hand is the longest because there may be cases where hough transform can fit interesting lines that would

break my clustering algorithm's threshold and create another cluster.
So far, I have always talked about the hour and minute hands but whenever a second hand is involved, it is hard to destroy it even with erosion. Not all of the analog clocks have standard hand length and widths so sometimes erosion could delete the second hand, sometimes it can't and since I assumed that the longest line is the minute hand, it can substitue the minute hand and lead to wrong results. I could not find a way to effectively destroy the second hand because there is no standard. If I were provided a certain test set, I would make up something according to the set but I could not generalize the algorithm so that it could destroy all kinds of second hands.

For the fine-tuning part, I have decided to resize every image that I loaded to 500x500, no matter what it's original resolution was. Whenever there is a really small image, erosion would destroy most of the useful data or whenever there is a big image, hough transform would fit lots of misleading lines into the big hands. Also, generalizing a formula for point-to-line distance was tricky. Therefore, I have fixed the resolution and tune my parameters accordingly.

Currently, the only problem is the second hand and the shapes of hands that mislead hough transform. The program can tell the time if it can fit the lines correctly since it can fix their orientation and lengths. I could not figure out a solution for destroying the second hand. If I were given the information about what kind of analog clock I am reading, I could have handled it. I think we need more information about the clocks to solve more problems.