Hough Transforms

Daniel Perry

- Code is written in matlab
- I've included a file main.m that creates all the graphs and derived images shown below.

Hough Transform for Straight Lines

CODE:

Hough Transform

- hough.m - creates the accumulator image for straight lines.
- hough_decrement.m - same, but also runs a "decrement" phase as described in [1].
- hough_orientation.m - same, but uses the edge orientation to limit parameter space search.
- findmaxima.m - locates maxima in the accumulator image.
- drawlines.m - draws the lines from parameters on original image.
- drawpoints.m - draws the parameters on accumulator image.

Smoothing and Edge Detection (reused from previous project)

- makeDiff1D.m - creates a 1D derivative estimation filter.
- makeGauss1D.m - creates a 1D Gaussian smoothing filter.
- convolve.m - convolves a separable 1D filter and an image in both directions.
- convolve1D.m - convolves a 1D filter and an image in one direction.
- edgeMap.m - generates the edge map (ie gradient magnitude) of an image. This function makes use of the derivative code above as well.
- orientationMap.m - generates the orientation map (ie gradient orientation). Also makes use of the derivative code above.
- rescaleDiffImage.m - helper function that shifts and scales the intensities in a derivative image so they are in the range [0,255]. Only used for display purposes.

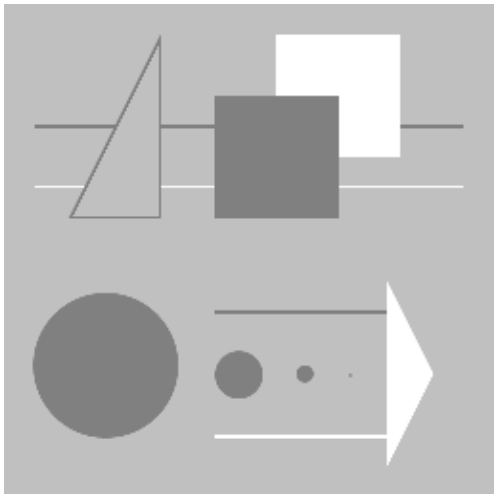
Code Description

I've created a hough function that takes an edge image and the range of θ values, from the theta values it determines how large of an accumulator image is needed, for simplicity the resolution of the ρ values are set to the same (accumulator is a square image). The accumulator is generated by iterating over all θ, and calculating $\rho = x\cos\theta + y\sin\theta$ for each x,y edge point.

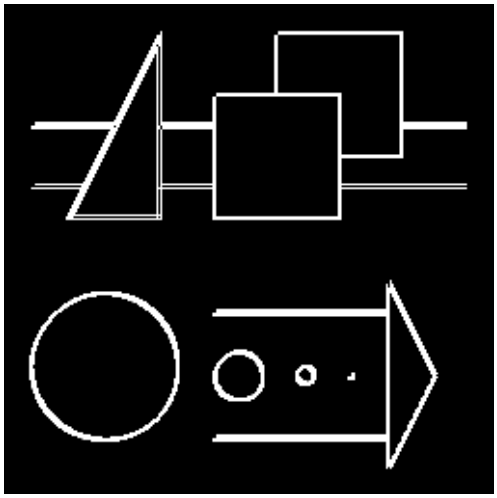
A second hough_decrement function was created to run a second "decrement" phase, as described in [1].

I've made use of my previous edge detection and image smoothing code, as indicated above. I've copied the edge detection and rescaling (for viewing) functions from the previous edge detection project.

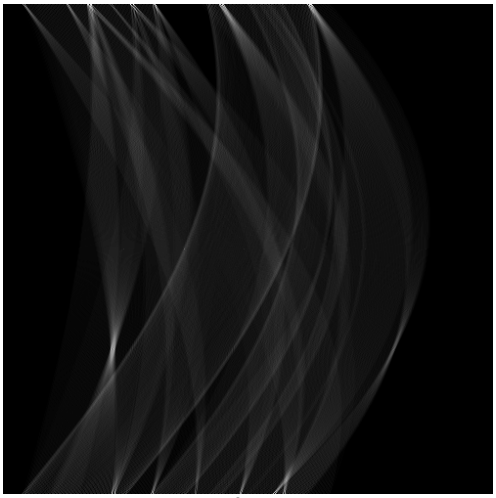
EXAMPLES:



Original



Edge map

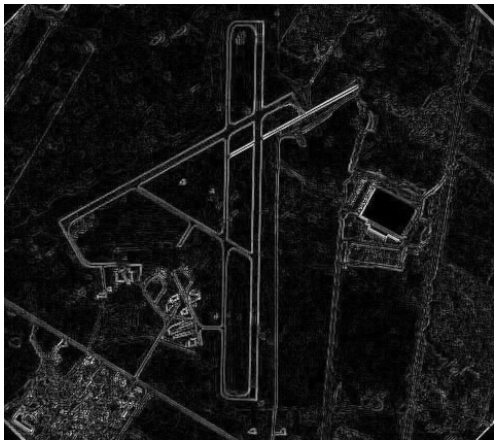


Accumulation image (θ on x-axis, ρ on y-axis)

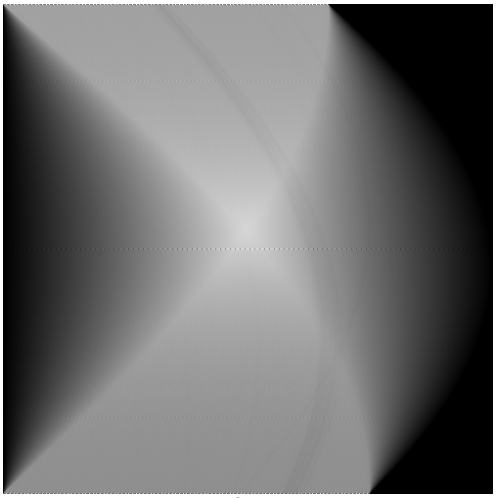
The above example shows the hough transform for a straight line applied to the edge map of the image with shapes above. As you can see the accumulator image shows the curves in parameter space representing each of the points on the edge map.

We haven't done any thresholding to the edge map because it is already so clean.

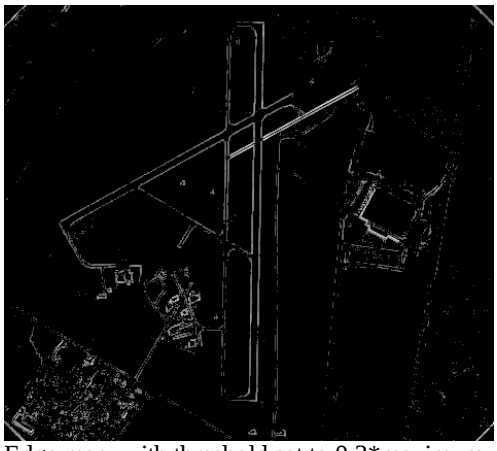
Original



Edge map



Accumulation image (θ on x-axis, ρ on y-axis)



Edge map, with threshold set to 0.3*maximum.

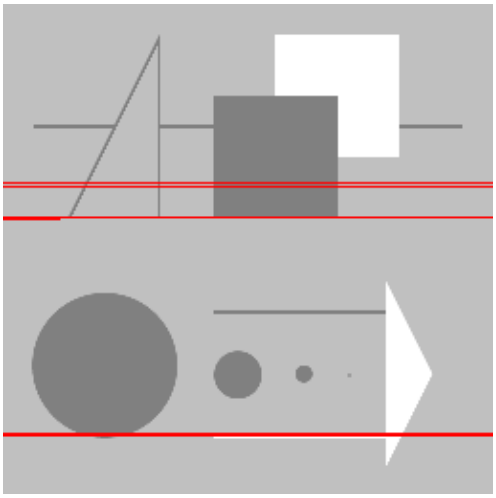
Accumulation image (θ on x-axis, ρ on y-axis)

Here we see the hough transform for a straight line applied to the edge map of the image with a runway shown above. Again, the accumulator image shows the curves in parameter space representing each of the points on the edge map. Because this is a real photograph, the edge map has a lot more noise, which causes the accumulator curves to be much less distinct, the process also takes much longer to complete.

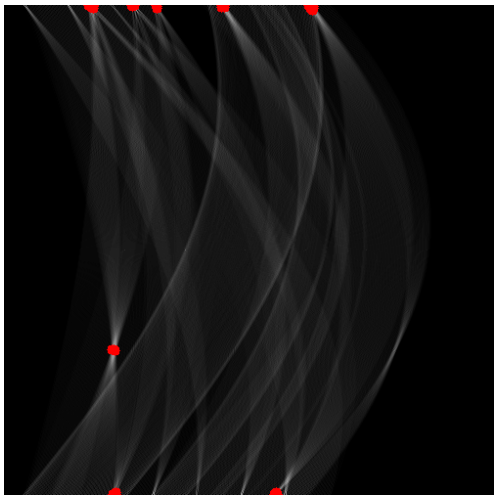
To address this problem, we threshold the edge map, to only get the more strong edges, and ignore the weak edges (which will also includes most of the noise). Here we threshold at 0.3*maximum edge value. The result is shown above, as you can see the majority of the runway edge data is still there, but most of the less useful edge and noise data have been removed. The resulting accumulator image is also much more clean - the individual curves are now distinguishable.

Raw Accumulator

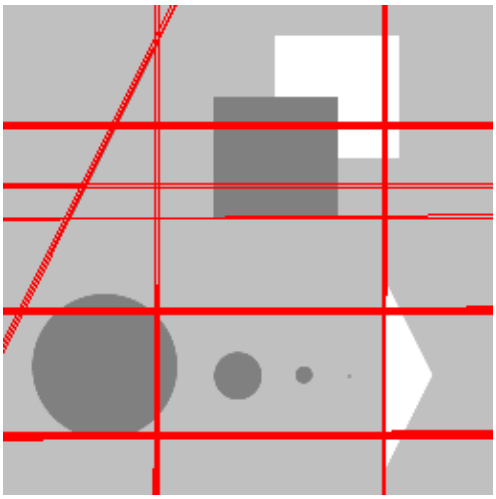
Maxima (red dots) detected using 0.9*maximum value(dots have been enlarged). (6 maxima)



Lines corresponding to the maxima shown the left (threshold of 0.9*maximum).

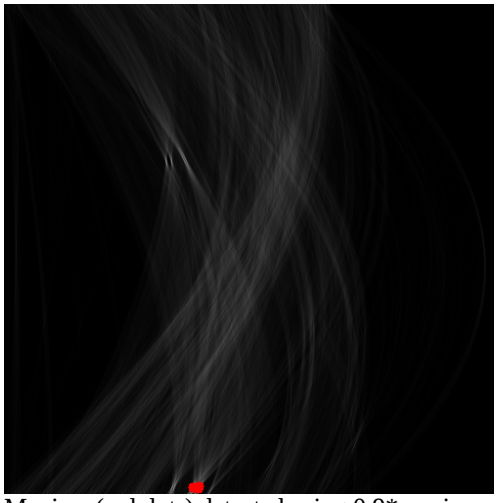


Maxima (red dots) detected using 0.7*maximum value (47 maxima).

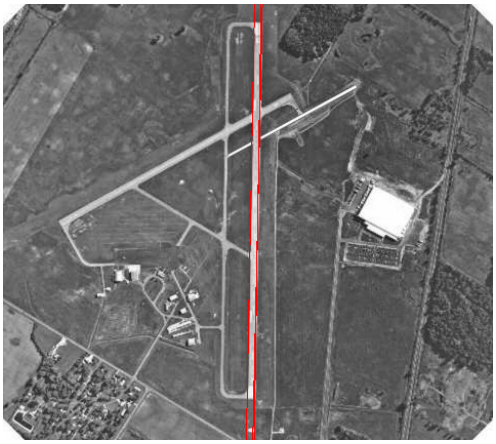


Lines corresponding to the maxima shown the left (threshold of 0.7*maximum)

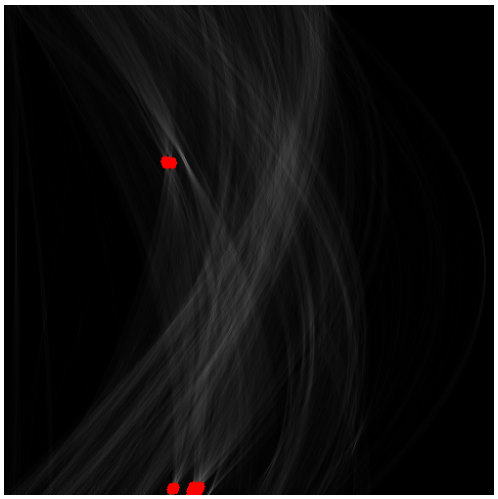
The above maxima were found without any gaussian blurring of the accumulator, or decrementation. These are just the maxima of the simple hough transform. As you can see the process seems to capture the lines and edges pretty well. If we turn the threshold on the accumulation image down to 0.7*maximum value even more edges are captured.



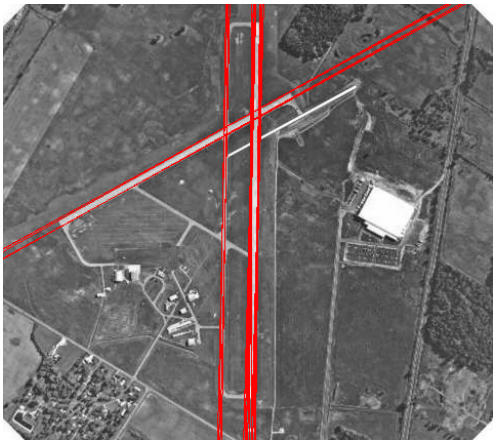
Maxima (red dots) detected using 0.9*maximum value(dots have been enlarged). (3 maxima)



Lines corresponding to the maxima shown the left (threshold of 0.9*maximum).



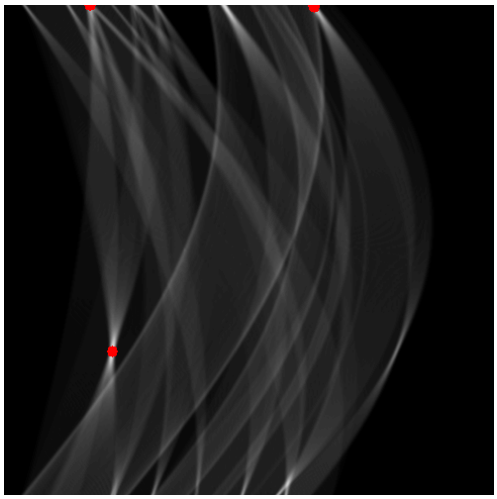
Maxima (red dots) detected using 0.5*maximum value (32 maxima).



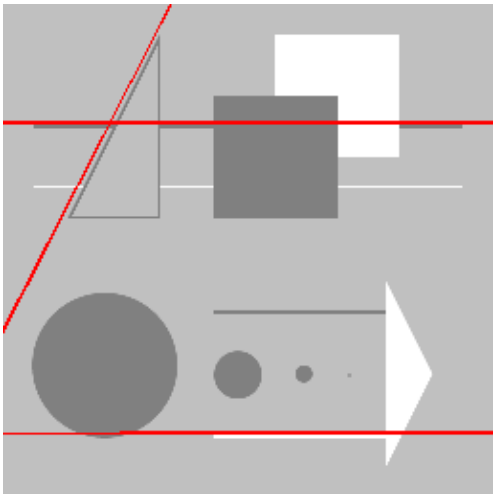
Lines corresponding to the maxima shown the left (threshold of 0.5*maximum)

Similarly, here we are finding the maximum on the raw accumulator without any smoothing or decrementing. As the image shows we are able to capture the main verticle lines of the runway (which is encouraging, since those are the same lines found in the text book on p.737). If we lower the threshold on the accumulator we are able to find more lines, including the diagonal runway edges, which is cool.

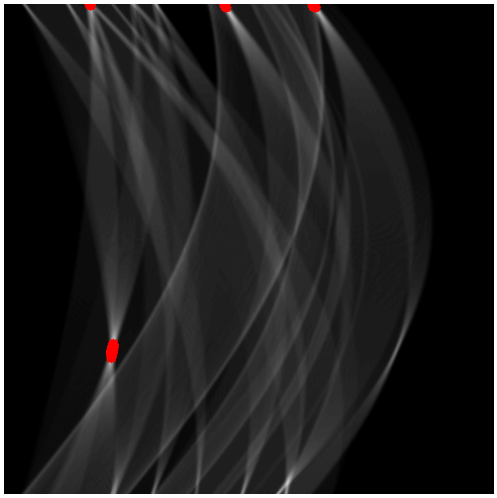
Gaussian Blurring on Accumulator



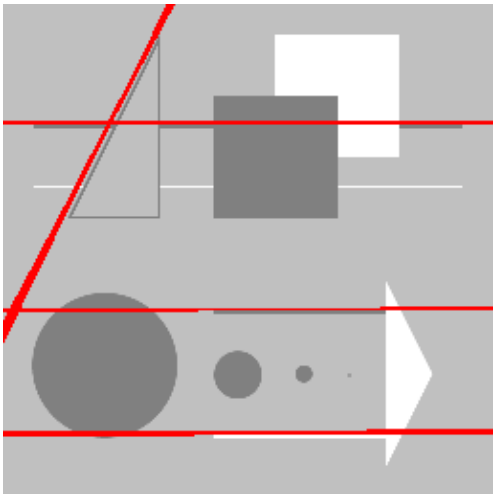
Maxima (red dots) detected using 0.95*maximum value(dots have been enlarged). (4 maxima)



Lines corresponding to the maxima shown the left (threshold of 0.95*maximum)

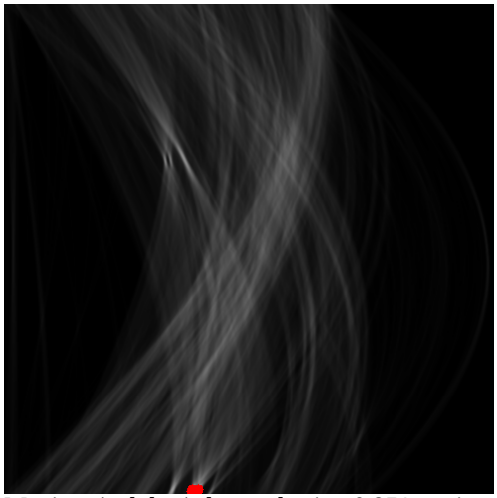


Maxima (red dots) detected using 0.9*maximum value (26 maxima).

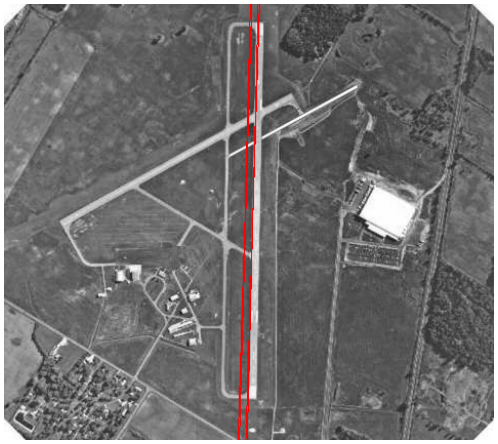


Lines corresponding to the maxima shown the left (threshold of 0.9*maximum)

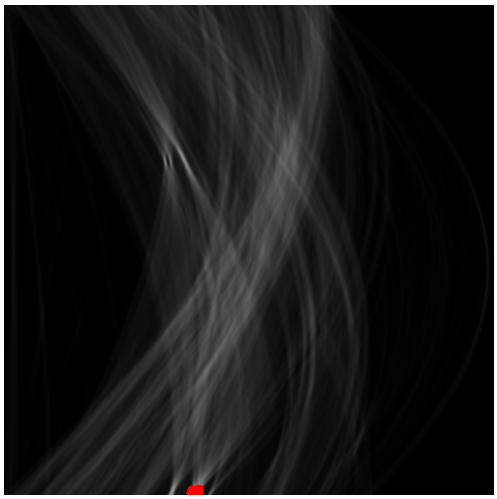
The above maxima were found using gaussian blurring with $\sigma=1$ of the accumulator. With the blurring, more points were closer the the global maximum value (using .9*maximum as threshold still had 26 matches). This makes sense, as the maximum has probably been lowered by the blurring, and other values may have been increased.



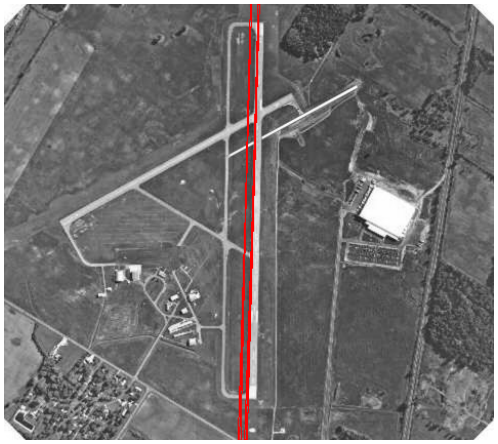
Maxima (red dots) detected using 0.95*maximum value(dots have been enlarged). (5 maxima)



Lines corresponding to the maxima shown the left (threshold of 0.95*maximum)



Maxima (red dots) detected using 0.9*maximum value (9 maxima).

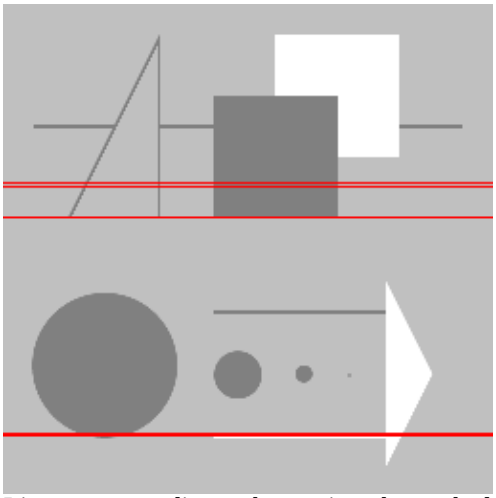


Lines corresponding to the maxima shown the left (threshold of 0.9*maximum)

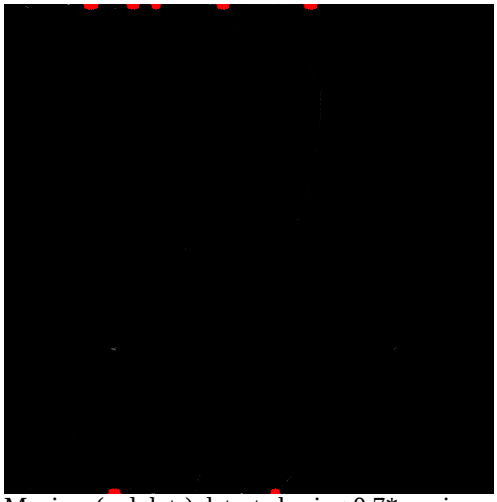
Similarly, the above maxima were found using gaussian blurring with $\sigma=1$ of the accumulator. However one interesting difference is that the blurring doesn't seem to have as drastic an effect on the number of extrema found at the threshold. In this case at 0.9*maximum found 5 instead of 3 (for the shapes image it changed from 6 to 26).

Decrementor on Accumulator

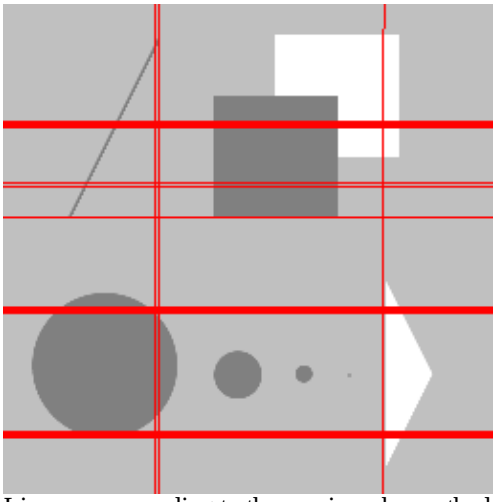
Maxima (red dots) detected using 0.9*maximum value(dots have been enlarged). (5 maxima)



Lines corresponding to the maxima shown the left (threshold of 0.9*maximum)



Maxima (red dots) detected using 0.7*maximum value (18 maxima).



Lines corresponding to the maxima shown the left (threshold of 0.7*maximum)

These maxima were found using the decrementation approach described in the [1]. Decremenation does a great job of cleaning up the accumulator, now very few (maxima) points are left, which makes the maxima discovery much more straightforward. The curves are no longer visible (decremented away), but the maxima are still available.

The result above show taht we are able to again find more lines. Some lines that weren't discovered in the raw or Gaussian approaches can now be discovered more easily.

Maxima (red dots) detected using 0.9*maximum value(dots have been enlarged). (5 maxima)

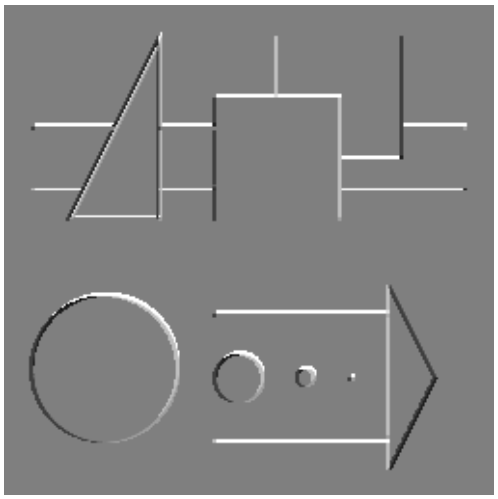
Lines corresponding to the maxima shown the left (threshold of 0.9*maximum)

Maxima (red dots) detected using 0.7*maximum value (18 maxima).

Lines corresponding to the maxima shown the left (threshold of 0.7*maximum)

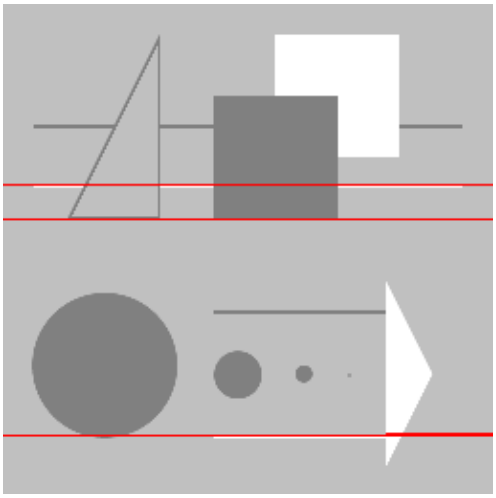
The story is very similar for this image. The decrementation strategy really cleans up the accumulator and we are able to find the same vertical lines in the image.

Limit Accumulation using Edge Orientation

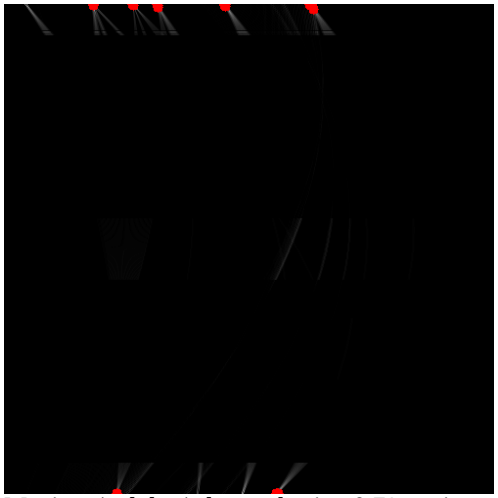


Orientation image.

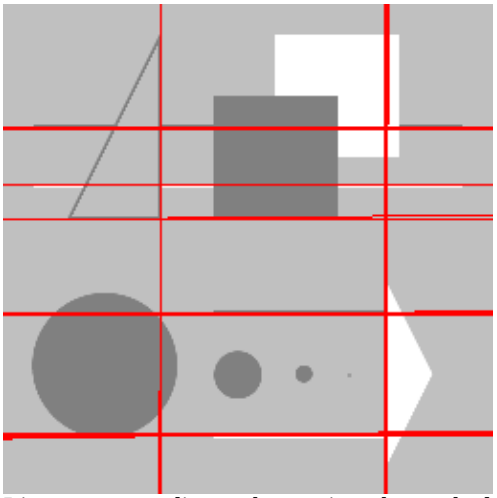
Maxima (red dots) detected using 0.9*maximum value(dots have been enlarged). (5 maxima)



Lines corresponding to the maxima shown the left (threshold of 0.9*maximum)



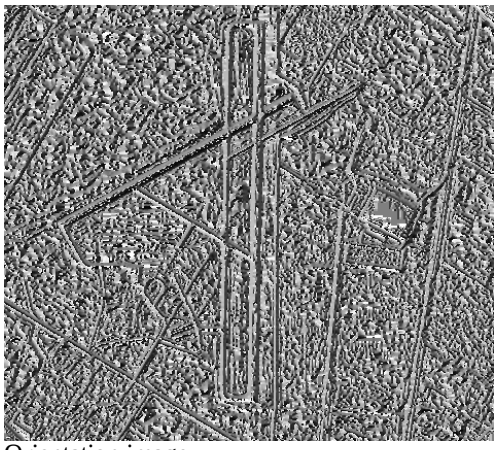
Maxima (red dots) detected using 0.7*maximum value (23 maxima).



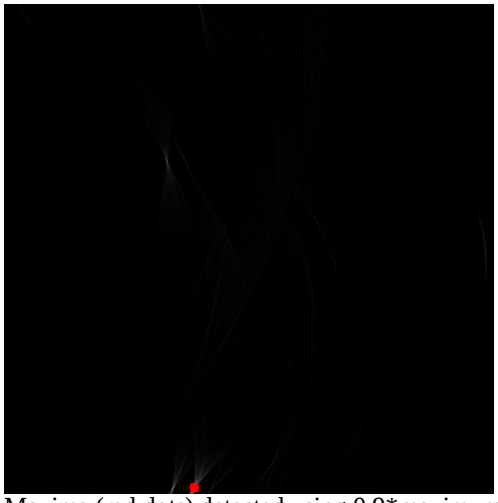
Lines corresponding to the maxima shown the left (threshold of 0.7*maximum)

Another approach to speeding up the accumulation step is to limit the θ parameter using the edge orientation of the edge points. The resulting accumulation image is shown above. It's interesting to note that while we are limiting θ (x-axis) the most drastic/noticeable breaks in the curves is along the ρ parameter (y-axis).

The approach appears successful, as you can see above the appropriate lines are detected again, but with much less computation. If we lower the threshold we find additional lines like before.

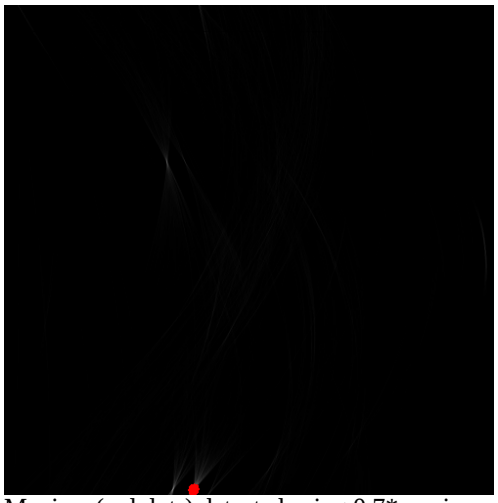


Orientation image.

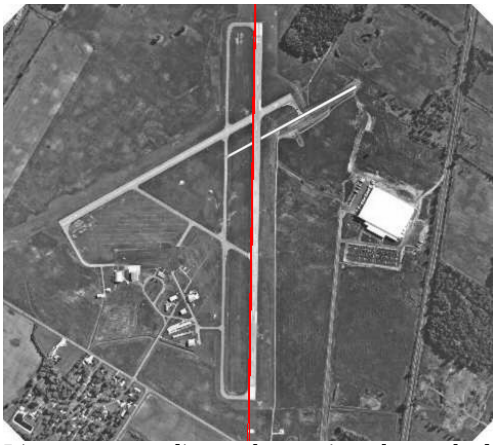


Maxima (red dots) detected using 0.9*maximum value(dots have been enlarged). (1 maxima)

Lines corresponding to the maxima shown the left (threshold of 0.9*maximum)



Maxima (red dots) detected using 0.7*maximum value (4 maxima).



Lines corresponding to the maxima shown the left (threshold of 0.7*maximum)

The results for this image are very similar. We are able to compute the accumulator image very quickly, but the results are largely successful. In this particular instance we only get one of the vertical edges.

References:

1. Gerig, G. "Linking Image-Space and Accumulator-Space: A New Approach for Object-Recognition". ICCV 1987.