Table of Contents

Whit Boland - jwb10028
Professor Gerig
CS6643 Computer Vision
22 February, 2024

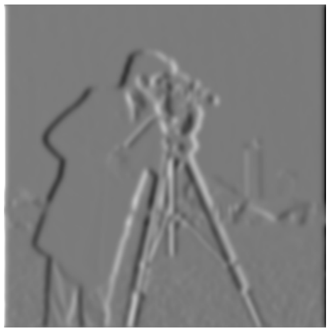
Lab 2 Report

A.1.1

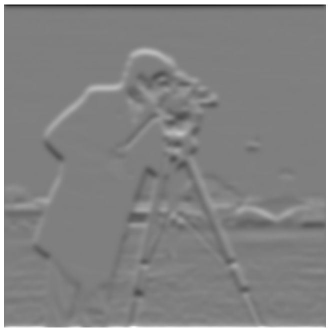
(a) Results:

original - ‘cameraman.png’

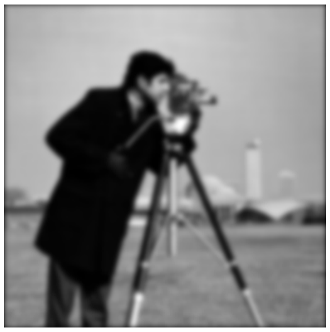
x-derivative img gradient



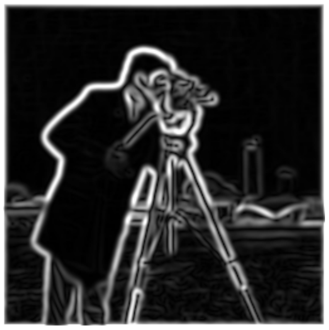
y-derivative img gradient



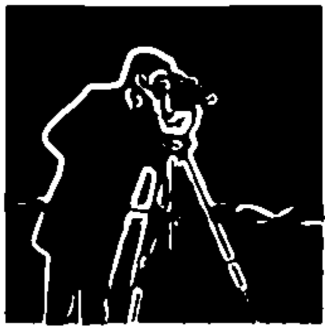
gaussian blur (σ = 2.0)



gradient magnitude

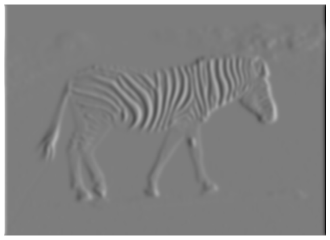


threshold img (t = 128)

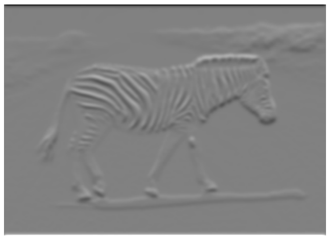


original - ‘zebra.jpg’

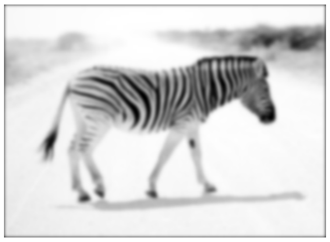
x-derivative img gradient



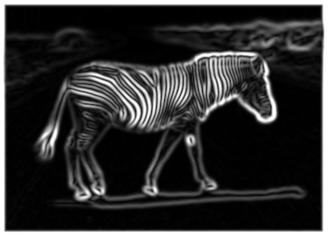
y-derivative img gradient



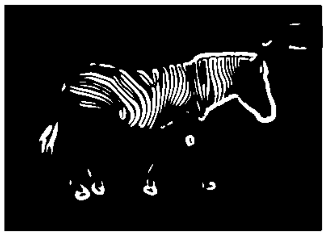
gaussian blur (σ = 1.0)



gradient magnitude



threshold img (t = 128)



A.1.1

(b) Discussion:

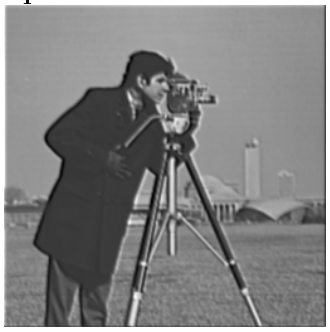
The above results represent both original images 'cameraman.png', and 'zebra.png' throughout the binary edge, and zero-crossing detection process. In order to perform the above tasks it is necessary to perform convolution filtering on the source image in several different divide and conquer steps. First, in order to distinguish the tradeoff between edge smoothing and localization the source image is pre-processed with a Gaussian kernel. Based on several different trial and error procedure's it can be determined that an optimal sigma value for the Gaussian kernel was right around σ = 2.0, and filter widths as instructed at 6*σ+1. It is important to note while discussing the gaussian pre-processing step that a smaller sigma will result in edges that represent the finer features of the source image, while a larger sigma value allows the edge detection model to hone in on the broad edges of the source image. Therefore, as stated the optimal sigma value is chosen in order to define the tradeoff between these fine/broad edges of the source image. After the source image has been convolved with the gaussian kernel, the resulting gaussian blurred image is convolved with a separable derivative filter. This process plays a role in resulting two important images, the x & y derivatives of the source image (image gradients) . In the report implementation, using separable hardcoded sobel filters seemed to provide optimal results for both the image gradient results. These results were then adjusted using an 'imageAdjust' routine, which stretches the min/max pixel intensity values of the image to the range [0-255]. Once both the x & y image gradients are stretched, the gradient magnitude is computed using the equation $Gmag = \sqrt{(xderiv)^2 + (yderiv)^2}$. To perform the final step, the resulting gradient magnitude image is thresholded. In the above report, the 'cameraman.png' is manually thresholded with a threshold value of 128 (right at the midpoint) which seemed to provide an appealing binary_edges image. At this threshold value the edge detection model seems to do an efficient job of outlining the photographer from the backdrop. The model seems to pick up some edges present in the horizon. Of course, if the end goal is to isolate the subject from the backdrop, it would always be possible to tinker with the gaussian blur/threshold value in order to refine the edge detection results.

A.1.2

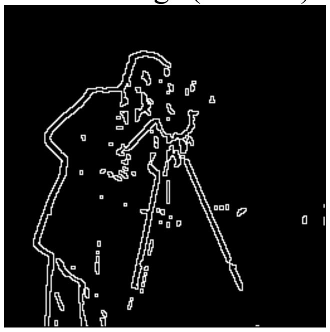
(a) Results:

original - 'cameraman.png'

laplacian filter



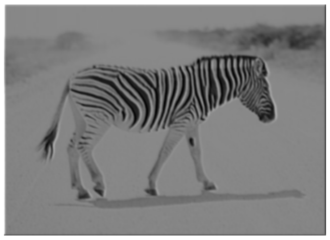
zero crossings (δ = -1.0)



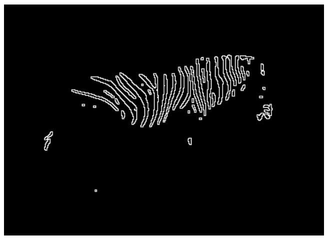
(a) Results:

original - ‘cameraman.png’

laplacian filter



zero crossings (δ = -1.0)

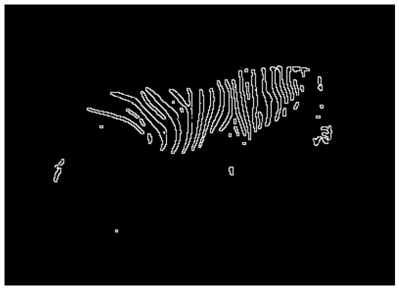


(b) Discussion:

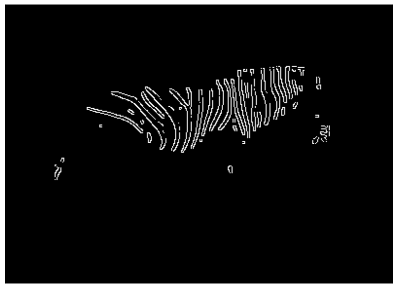
Another method displayed above on image edge detection uses the laplacian filter to compute the second derivative of an input source image. The major difference here when compared with the gradient magnitude implementation is that the second derivative computation will result in the detected zero crossings of the source image. On the other hand, in the prior implementation both the x & y derivatives allow the detection of peaks/maxima in the input image signal, which are used to compute a resulting gradient magnitude image, and then thresholded. In the second case, after the image is convolved with the laplacian filter, the detect_zero_crossings function is used to iterate over each pixel in the laplacian filtered image and perform several computations. If the current pixel’s intensity value is greater than 0, and the neighbors of the pixel are less than a heuristically chosen delta value (in the above case δ = -1.0), then the current pixel’s intensity value is assigned to 255. The delta value provides a method for filtering “strong” vs. “weak” edges detected in the input source image. It is important to note that when we compare the resulting detected edges using the gradient magnitude method, the smoothing + thresholding as expected results in a binary mask with very thick edges. On the other hand, the detections using the second derivative image result in edges that are one pixel in width, and have connected contours. Overall, when performing edge detection there are several ideas that need to be considered for an optimal result. For example, a useful edge detection model needs to have good detection accuracy, good localization, and the model needs to be single response (minimize the number of local maxima around the true edge). If the two models above are compared for edge detection optimization it would reason that the laplacian filter might result in stronger localization, however, the result of the gradient magnitude image could be further optimized using non-maximal suppression in order to thin detected edge lines.

TRIAL AND ERROR (VARIABLE DELTA):

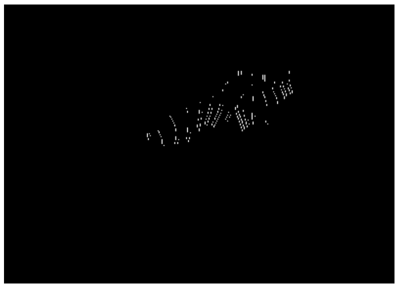
δ = -1.0



δ = -10.0



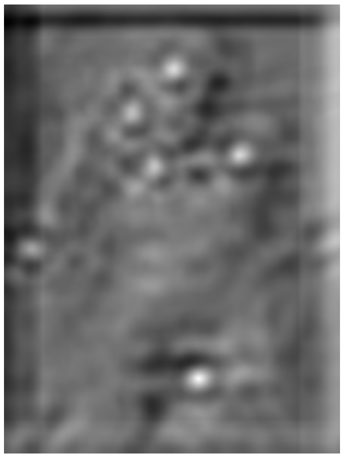
δ = -60.0

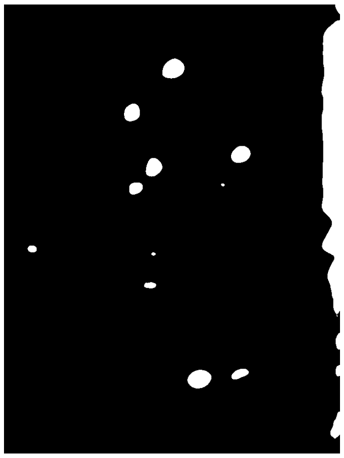


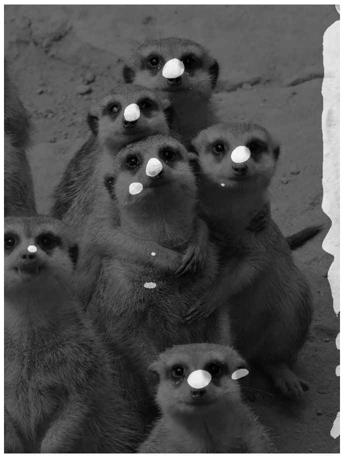
B.1

(a) Results:

original


correlation_result

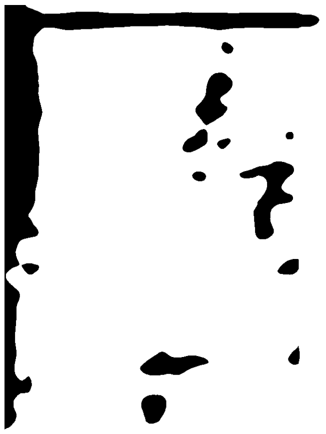

binarized_peaks (t=150)

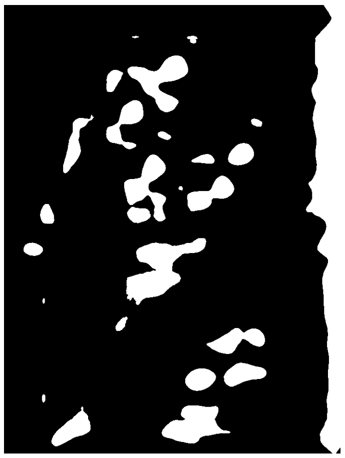

overlay_image

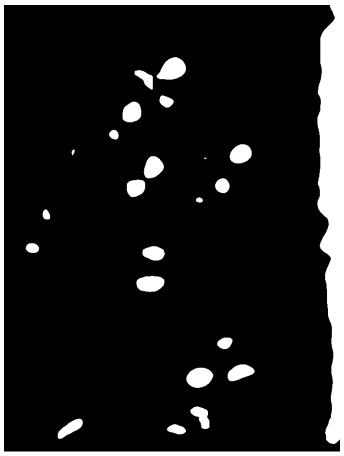
(b) Discussion:

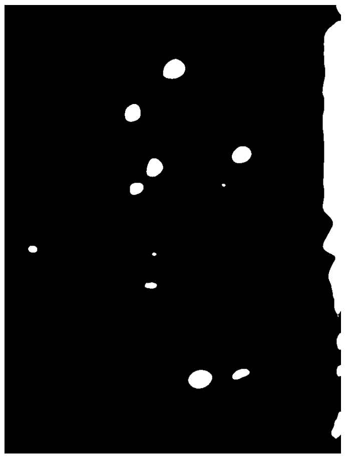
The above images represent results from each step in the template matching procedure. In the first step, the source image is cross correlated with the template mask in order to detect peaks/maxima in the source (regions of high correlation). In order to perform this step there were several preparatory measures that needed to be accounted for. To be specific, in preparation for cross correlation the template image needed to be normalized such that the mean value of the image should equate to 0.0. The result is labeled the "zero-mean" template. After the zero-mean template is calculated, cross correlation is performed using the source image as the correlation image, and the zero-mean template as the template mask. Of course, similarly to the convolution implementation the cross-correlation function required that the boundaries of the image are padded in order to ensure a centered correlation result. After correlation, the results are processed using the manual thresholding implementation, the output threshold image represents the result detected peaks of the template matching model. Looking at the above results, the peaks within the binarized_peaks image represent a peak for each detected animal within the source image i.e. there is a high correlation between the template and the current region. It is important to note that as the threshold value increases, the detection peaks seem to fade as the threshold implementation starts to only include pixel neighbors closer to local maxima in the correlation_result. On the other hand, as the threshold value is decreased it can be noted that areas of peak detection would grow in size until the overall definition of peaks is ultimately lost. Looking at the overlay image, the peaks are accurately placed over areas in the image that have the greatest similarity to the template image which seems to be a sure sign that the cross-correlation model is performing peak detection as intended. Below are some images of the peak detection model thresholded at various threshold values. These results provide a demonstration of how detected peaks could be lost if not properly thresholded.

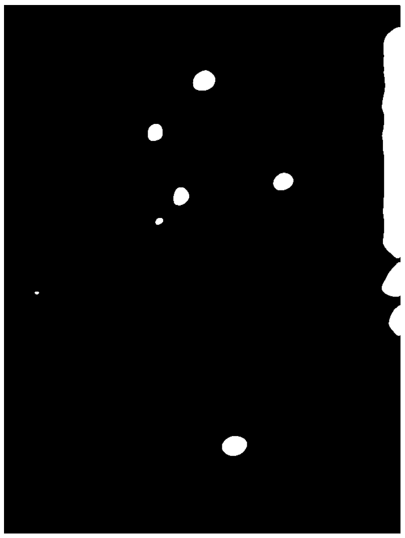
TRIAL AND ERROR BINARIZED PEAK DETECTIONS:

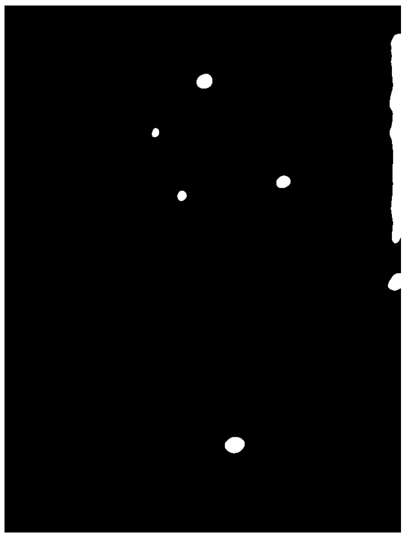

t = 75

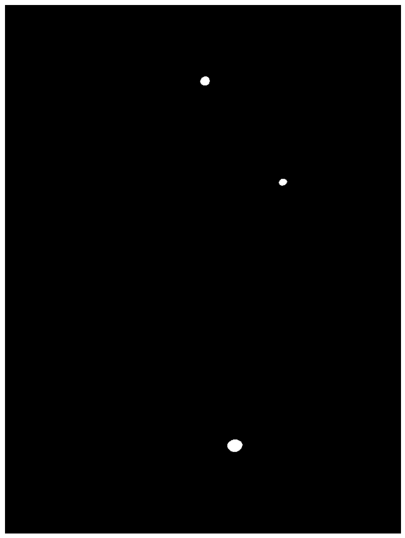

t = 128

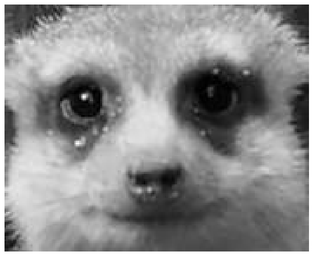

t = 140

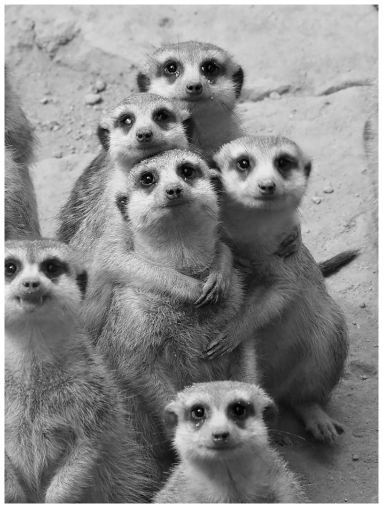

t = 150


t = 160


t = 180


t = 200


template


original

(a) Discussion (continued):

When comparing the trial and error binarized peak detections in comparison to the above discussion, it can be noted that these results accurately depict the initial hypothesis. As the threshold value is decreased the size of the peak detections increases as more neighbors surrounding local maxima/peaks are included in the thresholded binary image. On the other hand, as the threshold value increases the size of the peak detection decreases. Knowing that this holds true, the threshold value has a huge effect on what is to be considered part of the binarized peak. To further explain the process, within the cross correlation function the zero-mean template normalizes the template image to shield against varying light intensities, and emphasize the edge pattern of the template. After the zero-mean template is calculated the cross correlation process is performed to find areas of high correlation between the source and template image. The last step of the process is to threshold, and output the binarized image.