(U) Automated Analysis of Digital Radiographs to Support an Image Audit

Walter Rose
Michael Cairo

US Army CCDC
Armaments Center

Building 908
Picatinny Arsenal, NJ 07806

Walter.s.rose5.civ@mail.mil
macairo@email.wm.edu

(U) Abstract: In this paper, we detail how we used MATLAB® to analyze digital radiographs. Our dataset consisted of 153,516 images of M1 shells. We created a MATLAB® program that converted each image into a matrix of gray values corresponding to each pixel. We identified regions of interest by pixel location and performed our analysis using statistics associated with each region. The program successfully distinguished three different categories of defects, sorted the images according to which case they manifested, and had a category for ambiguous cases that required human attention. This methodology flags the most obvious cases, leaving the more contentious work for human eyes. It also allows for potential critical escapes to be quickly identified and brought to the attention of Level 3 radiographers and program managers.

(U) This paper will introduce the theory behind our method, explain the method used, and detail the development of the program. We will describe the performance of the program referring to the M1 dataset. We will then assess our results and speculate about future developments and applications.

(U) Research Innovation and Objective(s): Our research allowed us to partially automate a production facility audit. By running the submitted images through our program, we were able to expedite the preliminary portion of the image audit considerably. Although our program is custom tailored to recognize known defects for M1, the method developed here can be expanded upon to identify different defects on different items. This paper will detail how such changes can be made.

(U) Impacts on Warfighter Mission: We were able to identify M1 rounds that were missing a drill hole for the subcharge. Being able to catch these means warfighters will not encounter deficient product on the battlefield when they find their round is critically flawed. More generally, this research makes quality control more efficient and error-proof, meaning more high quality products reach the warfighter.

(U) Keywords: Automatic Defect Recognition (ADR), Supplementary Charge, Ghosting, Image Processing, MATLAB®

1. (U) Introduction

(U) 1.1 Digital Radiography and Image Analysis

(U) Digital radiographs are taken by exposing an array of small detectors to X-rays. Each detector on the array is represented as a pixel in a digital representation of the data. These images are in 16 bit black-white meaning each pixel can assume a value of 0-65,535 spanning between pure white and black (see figure 1). When a pixel is overexposed, or directly bathed in X-ray light, it will assume a high value close to white(numeric value of 65,535.) When an object stands between the X-ray source and the pixel, the light is attenuated and the pixel receives a lower, grayer, value.

(U) In this way, objects are X-rayed and represented digitally. Every image produced by this method is essenatially a large array of gray values that allows itself to be digitally manipulated. Digital manipution of images incudes applying filters, where mathematical operations are applied to the whole array, effecting aspects of the image. The array can also be fed to a program for automated analysis. Computer programs analyzing digital arrays to identify defects is known as automatic defect recognition (ADR).

(U) Figure 1: Shades of black-white correspond to numeric values on the 16 bit representation. The left panel has a value of 0, the center 40,000, and the right 65,535.

(U) Our analysis was done with MATLAB®, using various image processing fucntions available in the program. The function `dicomread(_)`[2] converts an image file into a matrix where each pixel with its corresponding value is represented by an element in the matrix. So, if our image was 3x3 pixels, (see figure 2) we would produce a 3x3 matix where the top left pixel occupies the (1,1) matrix element. (see figure 3)

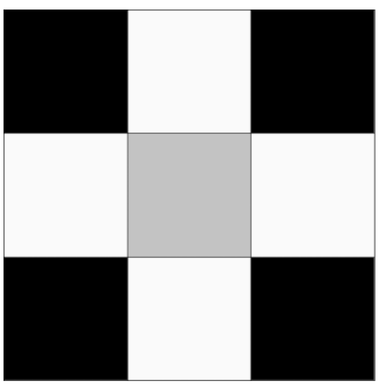

Figure 2*:* A 3x3 pixel image.

0	65,535	0
65,535	40,000	65,535
0	65,535	0

Figure 3: The result of applying the MATLAB® function, dicomread(_) on figure 2.

(U) As `dicomread(_)` produces a matrix, we can define new matrices within it to isolate a region of interest. If we were interested in seeing only the top third of the image, we could declare a 1x3 matrix. This becomes useful for larger, more elaborate images where we can crop out arbitrary sections for further analysis.

(U) With the image converted into a matrix, and the ability to crop matrices into smaller matrices, we can then perform simple operations on them. MATLAB® allows us to find the mean value and standard deviation of the gray values in each matrix. We can also display the mean value as a color swatch by multiplying a matrix of ones by the mean value. With a numeric value of the average, and with a visual representation of it, we can perform mathematical and visual comparisons of general regions in our radiographs.

(U) MATLAB® allows us to create logic gates and loops for values associated with the matrices we derive. For example, one can envision a flowchart that passes all images whose top half is brighter than the bottom half (on average), but separates all those within a 1000 value threshold for a closer inspection of the middle region. Or, we can fail all images who are (on average) lower than a certain threshold, and inspect the center four pixels of all remaining images. The simple operations of cropping, statistics, and logic loops allow for a wide variety of analysis capability for a dataset of standardized images. We applied these concepts to build our method for automated inspection.

(U) 1.2 Relevant Phenomena

(U) The first relevant phenomenon we were interested in detecting was ghosting. Ghosting occurs when a digital radiography system's detectors are burned through repeated exposure or over exposure. As the radiographic image acquisition was automated and repeated, we expected ghosting to occur.

(U) The second relevant phenomenon had to do with the stitching of the images. Occasionally, the automated image production would fail and produce an incomplete or corrupted image. We know there were several of these images in the data set and were interested in isolating them.

(U) The third relevant phenomenon concerns the build of the item. M1 rounds have a drilled region containing a subcharge. The cylindrical area can be clearly seen in the upper quarter of the round in radiographs. The three possible conditions of the round are:

1. not drilled with subcharge absent (see figure 4)
2. drilled with subcharge absent (see figure 5)
3. drilled with subcharge present. (see figure 6)

(U) The only acceptable condition is if the round is drilled and the subcharge is present, case 3. A missing subcharge is considered a critical failure.[1]

(U) We created a program that allowed us to identify these cases from a large dataset.

Figure 4: not drilled with subcharge absent.

Figure 5: drilled with subcharge absent.

Figure 6: drilled with subcharge present.

2. (U) Method

2.1 Statistics and Regions

(U) We were able to identify the desired cases by calculating the mean pixel value in different regions of the images. Because the images were standardized, we were able to count on the selected regions to be consistent throughout the dataset. We began the program with the `dicomread(_)` function to convert the image into a matrix.[2]

(U) 2.2 Ghosting

(U) The first case we were interested in is identifying ghost images. These images were visually identifiable by their excessive darkness. (see figure 9) The first step in our program was to check each image in our dataset for this excessive darkness. After identifying which images contained a lack of information, we wanted to produce a list of barcodes that were ghosts in our dataset.

(U) To demostrate the phenomenon more clearly, we had MATLAB® produce a histogram of gray values for these two cases. As expected, the darker, ghosted image has a histogram that tends much closer to black. (see figure 7) Compare this to the histogram of an image with an acceptable range of information. (see figure 8)

(U) We were able to separate these two cases by calculating the mean gray value of the entire image. By sampling several known ghost images, we determined a gray value threshold of 1000 for the images that were ghosts. That is, any image whose mean gray value was less than 1000 was labeled as a ghost. Accordingly, the program calculated the average of the whole matrix produced by `dicomread(_)` and compared it to 1000. If the average was less than 1000, the image would be labeled “GHOST”. Otherwise, it would be labeled “NO GHOST.”

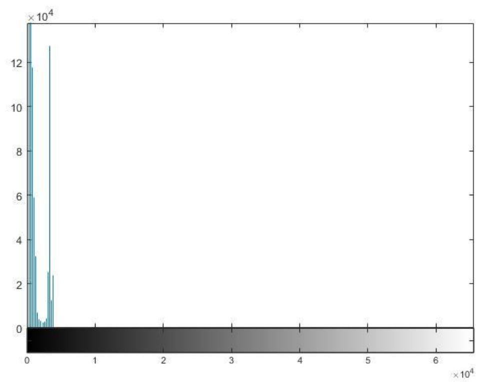

Figure 7: Histrogram of a Ghost.

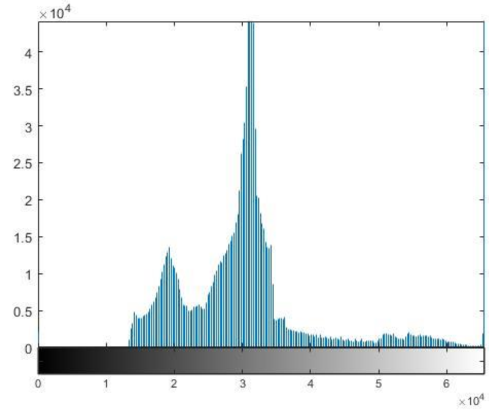

Figure 8: Histogram of a typical Radiograph.

Figure 9: Radiograph of a Ghost.

(U) A flow chart is shown in figure 12 of the appendix that represents the logic that our program used for detecting these images.

(U) In a similar manner, we could also detect images that were saturated. That is, we calculated if an image’s average gray values exceeded a certain threshold. For this dataset, we determined the threshold to be 38,000.

(U) 2.3 Missing Subcharges

(U) For the remaining images, we then needed to identify the three cases introduced earlier concerning the subcharge. First, we specified a rectangular region in the image that is representative of the homogeneity of the subcharge region. We achieved this by declaring submatrices that isolated the area where the subcharge should be present. (see figure 10)

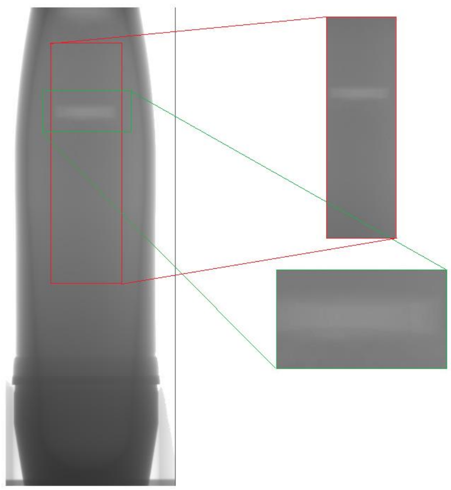

Figure 10: Submatrices in a Radiograph.

(U) When the subcharge is missing, there is less material between the X-ray source and the detector, leading to less attenuation. As figure 5 shows, this makes the region much brighter than the case where the subcharge is present. (see figure 6) We exploited this fact to distinguish between these two cases. Having identified the subcharge region, we can then isolate a region below it to perform a comparison between the two.

(U) If the mean gray value of the subcharge region is higher than the region below it, then the subcharge is missing. In the first implementation of the program, no thresholds were specified. The program simply compared the two regions. This allowed for greater confidence in the number of true negatives but increased the number of false positives. In order to decrease the number of false positives, we introduced a threshold. If the mean gray value delta was greater than 5000, we declared this to be a case of missing subcharge. If the mean gray value delta is between -100 and 5000 then the program identifies this to be indicative of a subcharge present. Lastly, if the mean delta gray value is less than -100, the program labels this as “Inspect”.

(U) As one could see from figure 4, the gray values where the drilled cavity should be and the region below it are very similar. When we were designing the MATLAB® program, we were unaware of the existence of these missing drill-holes. The creation of this third scenario allowed for mathematical completeness; that is, every calculation of delta will fall into exactly one of the three categories.

(U) Within the "inspect" category, further submatrices were defined. Referring to figures 10 and 11, the mean gray value was calculated for these new regions. Similarly, a threshold was declared to separate the inspect category into not drilled and drilled. If the region in figure 11 is of similar density to the region below it, then we can identify this round as not being drilled.

Figure 11: Edge of Subcharge region.

(U) 2.2 Program and Logic

(U) The output of the program produced two matrices. One matrix contained columns that stored the serial numbers, ghosting declaration, and the presence of the subcharge. This matrix also contained the mean gray value for the various submatrices defined in section 2.1. This allowed for statistical analysis and easy debugging.

(U) We refer the reader to the appendix for the flowcharts as well as the MATLAB® code.

(U) First, the average of the whole image was found. If the average was higher or lower than our thresholds, it would print “GHOST” or “Error”. If it was within our threshold, it would pass to the next step. This is diagrammed in figure 12.

(U) The next step was an “if” condition designed to detect the presence of a subcharge. If the difference in brightness between the subcharge region and the M1 body region is greater than 5000, it prints “Missing.” And, if the difference is greater than -100, it prints “PASS.” If this is not met, it prints “Inspect.” This is diagrammed in figure 13.

(U) The cases where the mean delta gray value was less than -100 were put through another "if"-loop to separate “Drilled” from “Not Drilled”. The loop selected new regions: one containing the band below a filled subcharge (figure 11) and another band containing the body of the round. If the average of the band region was brighter than the body region by 900, the program prints “Drilled”. Otherwise, it prints “Not Drilled.” This is diagrammed in figure 14.

3. (U) Results & Discussion

(U) The program took an hour to analyze all 153,516 images. The resultant matrices were moved to Microsoft Excel® where they could be sorted and commented on. We manually inspected the output of the program using Vi3® imaging software suite made by VJ technologies®.

(U) The program was successful in processing all the images and identifying critical defects. In fact, when beginning the image audit, we did not know there were rounds in the set that weren’t drilled. Manually inspecting the images labeled “Inspect” led us to notice cases where the rounds were not drilled. This brought us to the third iteration of our MATLAB® code where a second loop was constructed to identify the not drilled M1 rounds.

(U) The MATLAB® program supported the radiographic laboratory during the audit process. We were able to compare our program’s rejected rounds with those rejected by human inspectors at the facility. A cautionary note is in order here. While ADR is a very useful tool, it must be used with caution. The output of the program was checked alongside qualified radiographers per NAS410.

(U) The most common cause for erroneous results were cases that were closest to our thresholds. This is typical behavior of any threshold. The thresholds we chose attempted to minimize the amount of erroneous identifications, but could never eliminate them. As a diversity of cases populate the space where the averages are ambiguous, any choice will allow some false judgments into the wrong category. To mitigate this unavoidable source of error, we chose our thresholds on the side of caution. That is, our thresholds prefer a passable munition to be failed than a defective one passed.

4. (U) Conclusion

(U) In this paper, we described the development and execution of a program that automatically analyzed digital radiographs. We used it to identify several cases present in a dataset of 153,516 images. We successfully distinguished between ghost images, saturated images, rounds that were not drilled, and missing subcharges.

(U) This paper details a program developed specifically for M1 artillery. However, the method used here can be applied to other objects with broad repeating defects. MATLAB® was chosen due to its extensive image processing package.

5.(U) Future Work & Acknowledgements

(U) Lastly, this method is not limited to defect recognition. Future developments on this can help us automatically identify the presence of image quality indicators, or hunt for a region in the image with the best signal-to-noise ratio (SNR.) These tasks are essential to image quality assessment and can greatly speed up audits of production facilities. Lastly, we would like to acknowledge Rueben Favarro for many fruiful discussions.

References

1. MIL-DTL-45195F(AR)w/AMENDMENT 3
2. Gonzalez, Woods, and Eddins (2016). Digital Image Processing using MATLAB. Chennai [etc.]: McGraw Hill Education (India) Private Limited.

5. (U) Appendix

5.1 Flowcharts

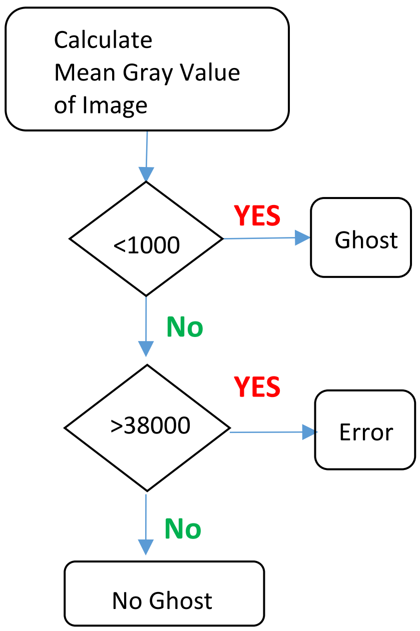

Figure 12: Flowchart for detection of a Ghost.

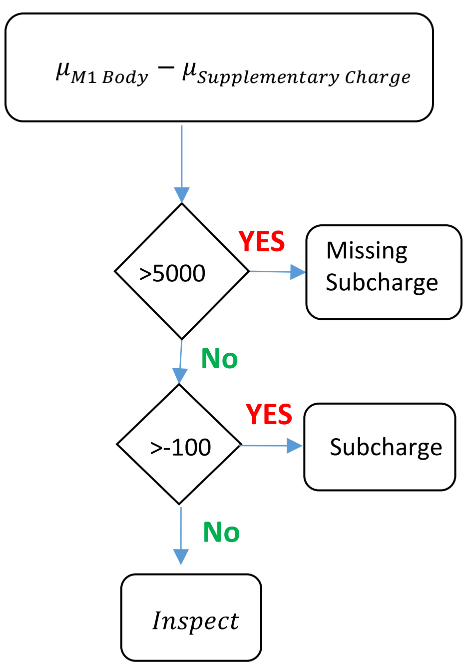

Figure 13: Flowchart for detection of a Missing Subcharge.

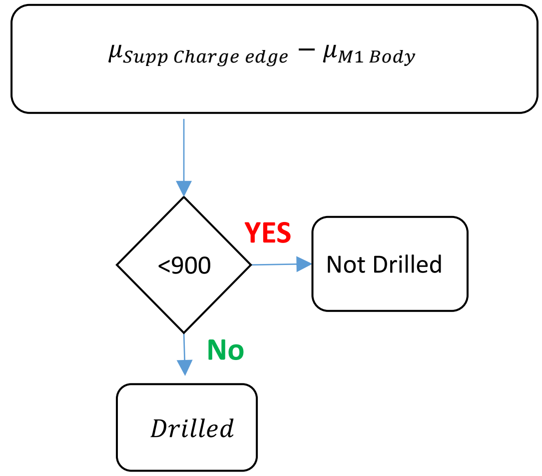

Figure 14: Flowchart for detection of a Missing Subcharge.

5.2 Matlab Code

```
for i = 1:nfiles

filename = strcat('C:\',I(i).name);

X = dicomread(I(i).name);

mean = mean2(X);

F(i,1) = filename;

F(i,2) = string(I(i).name);

if mean < 10^3

F(i,3) = "Ghost";

elseif mean > 38000

F(i,3) = "ERROR";

else

F(i,3) = "No Ghost";

end

M = X(640:1000,260:442);

Y = X(90:450,260:442);

meanM = mean2(M);

meanY = mean2(Y);
```

```
 F(i,5) = meanM;
 F(i,6) = meanY;
 F(i,7) = meanY-meanM;

if    meanY - meanM > 5000

F(i,4) = "FAIL";

elseif     meanY - meanM > -100

F(i,4) = "PASS";

else

F(i,4) = "Inspect";

W = X(380:520,240:480);

R = X(100:1100,200:500);

meanW = mean2(W);

meanR = mean2(R);

C(i,1) = filename;

C(i,2) = meanW;

C(i,3) = meanR;

C(i,4) = meanW-meanR;

if      meanW-meanR < 900

C(i,5) = "Missing";

else

C(i,5) = "Drilled";

end

end

end
```