Study and Analysis of Dual-Energy X-Ray Data

Chris Robert Kammerud
University of Tennessee - Knoxville

Follow this and additional works at: https://trace.tennessee.edu/utk_chanhonoproj

Recommended Citation
https://trace.tennessee.edu/utk_chanhonoproj/661

This is brought to you for free and open access by the Supervised Undergraduate Student Research and Creative Work at TRACE: Tennessee Research and Creative Exchange. It has been accepted for inclusion in Chancellor's Honors Program Projects by an authorized administrator of TRACE: Tennessee Research and Creative Exchange. For more information, please contact trace@utk.edu.
UNIVERSITY HONORS PROGRAM

SENIOR PROJECT - APPROVAL

Name: Chris Kammerud

College: COE

Department: ECE

Faculty Mentor: Dr. M. A. Abidi

PROJECT TITLE: Study and Analysis of Dual-Energy X-Ray Data

I have reviewed this completed senior honors thesis with this student and certify that it is a project commensurate with honors level undergraduate research in this field.

Signed: ____________________________, Faculty Mentor

Date: May 8, 2003

Comments (Optional): I am very pleased with the effort and achievement of Mr. Kammerud during the course of this class. Mr. Kammerud demonstrated a very creative mind and a great ability to address and solve new problems. I believe that Mr. Kammerud is able to tackle complex engineering issues, which makes him a very good candidate for graduate school. I do encourage him to pursue his graduate studies.
Study and Analysis of
Dual Energy X-Ray Data

Chris Kammerud
University of Tennessee, Knoxville
IRIS Labs, ECE Dept
Table of Contents

Abstract .......................................................................................................................... 2
1. Introduction ........................................................................................................... 2
2. Summary of the Algorithm .................................................................................. 3
   2.1 $K_{tb}$ ............................................................................................................... 3
   2.2 $K_{mat}$ ........................................................................................................... 4
3. Implementation of the Algorithm ........................................................................ 4
   3.1 H and L Values ............................................................................................... 4
   3.2 Background Areas and Calculation of $K_{tb}$ ............................................... 5
   3.3 Count Matrix ................................................................................................... 7
   3.4 $K_{mat}$ .......................................................................................................... 8
   3.5 Marking Areas ................................................................................................. 8
4. Colorization of the Image .................................................................................... 9
   4.1 Coloring Based Solely on $K_{mat}$ and Count ............................................... 9
   4.2 Using Original Gray Level In Color Equations .............................................. 11
   4.3 Keeping Track of $K_{mat}$ Corresponding to Maximum Vote Count ............ 12
   4.4 Normalization and Using Original Gray Level In Equations ....................... 12
5. Conclusions and Thoughts On Future Work ..................................................... 15
Flowchart of General Algorithm .............................................................................. 16
Colorization Flowchart ............................................................................................. 17
REFERENCES ............................................................................................................. 18
Dual Energy X-Ray Analysis
Chris Kammerud
University of Tennessee, Knoxville

Abstract
This project implements an algorithm for the analysis of dual energy x-ray data taken from airport security stations. The algorithm takes data from two images, created at two different x-ray levels, and through the relationship of the data identifies different types of material within the image. The results of the comparison are used in the creation of a colorized image that differentiates between types of materials by coloring different materials differently. The algorithm can be divided into two major parts: (1) the calculation of a parameter, $K_{ib}$, relating a pixel to the corresponding pixel in the other image; (2) colorizing the image based on the parameters calculated for each pixel. The algorithm proved successful in identifying parts of images using the $K_{ib}$ values calculated for each pixel. The colorization aids in distinguishing between objects based on the atomic composition of the elements.

1. Introduction

The events of the past years in New York, Washington, D.C, Iraq, and other areas around the world, have increased the attention paid to the luggage screening systems in airports. Focus has been placed on developing ways to improve the security systems in airports to automatically detect threat objects in luggage as well as produce images that facilitate an easier recognition of suspicious objects by the screener.

Traditional x-ray systems produce gray-level images using a single x-ray energy. The human eye only has the ability to distinguish between dozens of shades of gray, whereas it can distinguish between thousands of variations in color shading. Therefore if at all possible colorized images should be used in place of gray-level images when object recognition is the key issue. It is also true that using only a single x-ray energy can lead to overlaying objects hindering the system’s ability to detect or show a threat object that is hidden behind other materials[1]. These two factors point to the usage of a dual energy x-ray system that produces colorized images and can detect objects of certain atomic composition automatically.

This paper will summarize an algorithm discussed in a patent[2] and then describe the steps involved in its implementation. The data used in the project consisted of dual energy x-ray images of luggage, as well as colorized images created using an unknown algorithm. The algorithm discussed and implemented here was successful in its ability to approach the same coloring as those color images provided in the data.
2. Summary of the Algorithm

The algorithm implemented for this task comes from one described within patent 5,490,218. This patent, held by Vivid Technologies, followed the idea that the differences between high and low energy x-ray images could point to the composition of an object. A parameter, $K_{tb}$, is calculated for an area based on the attenuation of the x-rays at that location in the high-energy image versus the attenuation at that location in the low energy image. The algorithm also reduces noise by not simply looking at one area at a time, but by looking at an area and those regions that are nearby when calculating $K_{tb}$. This parameter is calculated several different times for the same area subtracting off different nearby areas each time. Each of these $K_{tb}$’s is compared to another parameter the algorithm makes use of, $K_{mat}$. The $K_{mat}$ parameter corresponds to the value a threat object would produce from the ratio of the high and low energy image values. A vote is kept regarding each area and the closeness of $K_{tb}$ to a specific $K_{mat}$ that the system is looking for. If a region receives a certain number of votes corresponding to the number of $K_{tb}$’s that fall within a range of a specific $K_{mat}$, then that region becomes marked as a possible threat region. This process is repeated for each area of the image that is being analyzed. After each area has a vote assigned to it then these votes are analyzed to determine the probability that a threat object is present; if the probability is high enough that area may be colored a specific way and an alarm may be sounded.

2.1 $K_{tb}$

$K_{tb}$ is determined using an equation relating the attenuation of a high to a low energy image area, as well as a nearby area. The attenuation of a high-energy area is denoted as $H$, for low it is denoted as $L$. In the course of the algorithm an area will be chosen as the test area for which a $K_{tb}$ is to be calculated. The attenuation of this area is denoted for the high and low energy images as $H_t$ and $L_t$ respectively.

$$H_t = \log(\text{image test area}) \quad (1)$$

$$L_t = \log(\text{image test area}) \quad (2)$$

Also included in the calculation of the parameter for this test area is the attenuation at a nearby area, or background area. The attenuation for this background area is referred to as $H_b$ and $L_b$.

$$H_b = \log(\text{image background area}) \quad (3)$$

$$L_b = \log(\text{image background area}) \quad (4)$$
$K_{tb}$ is found using the equation:

$$K_{tb} = \frac{(H_t - H_b)}{(L_t - L_b)}$$ (5)

As described in the summary of the algorithm the subtraction of the background area is used to reduce the noise from overlying areas on top of the area for which $K_{tb}$ is being calculated from. A number of $K_{tb}$'s are calculated using a determined number of background areas, this process creates an averaging affect that causes the $K_{tb}$ for an area to be almost solely related to the atomic composition of the area being tested.

2.2 $K_{mat}$

The value $K_{mat}$ as described in the algorithm can be calculated in two different ways. The first way involved using values calculated from doing direct experiments and measuring the ratio of high to low energy attenuations of a type of threat object. The second way was to take the ratio of $U_h$ to $U_l$. These two parameters are based on the attenuation at the test area and the nearby area in both the high and low energy images. In equation form:

$$U_h = f(H_t, L_t, H_b, L_b)$$ (6)

$$U_l = g(H_t, L_t, H_b, L_b)$$ (7)

The $K_{mat}$, again, refers to the attenuation characteristic of a specific material. It corresponds to the ratio $H / L$ of a specific material.

3. Implementation of the Algorithm

The algorithm was implemented in incremental steps leading up to a piece of code that could successfully identify areas that had an attenuation characteristic similar to that of a threat object. There were several questions that presented themselves after reading the patent and deciding to attempt its use. A few of these questions included the determining of $K_{mat}$, the patent does not go into specifics of its calculation, the determining of how to evaluate the number of votes an area receives and decide on the probability of a threat object, and finally the question of what exactly constituted an area.

3.1 $H$ and $L$ Values

In the processing of the images it was decided based on the way the algorithm was described in the patent to treat individual pixels as if they were areas. Therefore, the $H$ and $L$ values are determined using Eqs. 1-4 where the areas referred to, test and background, correspond to a specific test or background pixel. Rather than calculate the $H$ and $L$ values for each pixel each time it was needed for a $K_{tb}$ value, the $H$ and $L$ values
were calculated beforehand and stored in matrices, H_val and L_val, with dimensions equal to that of the input images.

\[
L_{\text{val}} = \log(L_{\text{img}}) \quad (8)
\]

\[
H_{\text{val}} = \log(H_{\text{img}}) \quad (9)
\]

When using Eq. 5 then, the values were taken from the appropriate matrix, H_val for H_t and H_b and L_val for L_t and L_b, at the coordinates coinciding with the test pixel or background pixel.

3.2 Background Areas and Calculation of K_{tb}

Choosing background pixels was based on choosing pixels falling on concentric squares expanding outward from the test pixel. Arbitrarily the number of pixels per side was chosen to be two, making the total number of background pixels per square eight. This number of pixels per square was maintained throughout the project, adding more caused the running time to increase disproportionately to any improvement in the results. The next decision was how many squares, and what should be the width of each square. The width of each square, with the test pixel at the center, would determine how far away a background pixel would be from the test pixel. In the first designs of the algorithm, the number of squares was chosen to be 3. The distances from the test pixel were 50, 75, and 100 pixels respectively to the sides. Below is a figure illustrating this concept.

The method to calculate K_{tb} involved looping through the three background distances and calculating eight different K_{tb}'s for the test pixel. At each background distance and each background pixel, Eq. 5 was used to calculate a K_{tb}.
In the early files only the top and bottom points on the square were calculated because of time issues. Figure 3.1 shows a preliminary result with pixels that had a vote (count) greater than seven colored white and those with votes above ten were colored a brighter white. The method described above can be described in this way:

\[
\begin{align*}
\text{Count}(\text{pixel}) \geq 7 & \Rightarrow \text{img}(\text{pixel}) = 220 \\
\text{Count}(\text{pixel}) \geq 10 & \Rightarrow \text{img}(\text{pixel}) = 255
\end{align*}
\]

Again, the number of votes refers to the number of times that the $K_{tb}$ calculated for a test pixel, using different background pixels, falls within the range of a specific $K_{mat}$. Initially $K_{mat}$ was chosen by hand calculations using data from the images. A particular location was looked at, and using Eq. 5, the $K_{tb}$ from the area was calculated. This $K_{tb}$ was treated as the $K_{mat}$ to search for.

Figure 3.1 $K_{mat}$ Range 0.85 to 0.95 ; Top and Bottom of Squares used
To illustrate the difference that choosing a different $K_{mat}$ range can make, Figure 3.2 is an image using a range of 0.55 to 0.7.

![Figure 3.2](image.png)

**Figure 3.2** $K_{mat}$ Range = 0.55 - 0.70 ; Top and Bottom of Squares used

### 3.3 Count Matrix

In order to keep track of the counts for each pixel, a matrix called Count was created with the same dimensions as the high and low energy images. A for loop runs through each pixel of the image, each of these pixels is treated as a test pixel. For each test pixel a $K_{tb}$ is calculated for each of the background pixels. The background pixels are selected as described and diagramed above in section 3.2. These $K_{tb}$ values are each compared to the $K_{mat}$ that the algorithm is currently looking for; if it falls within the range set then the Count matrix at the test pixel’s coordinates is incremented by one.
3.4 \textbf{Kmat}

In figures 3.1 and 3.2 you can see that different areas of the image were colored white, showing that the \textit{Kmat} does differentiate between different objects.

A problem though, was the fact that there was no listing about what types of objects fell under which types of \textit{Kmat}s. In order to better understand how different \textit{Kmat}s affected the colorization of an image that would be attempted later, a set of pictures were generated using the algorithm as described at different \textit{Kmat}s ranging from 0.1 to 1.2. A Count matrix is created for each \textit{Kmat} corresponding to the vote levels of each pixel at that \textit{Kmat}. Figure 3.3 illustrates this result.

You can see the variations in what pixels are colored white as \textit{Kmat} changes. The differences are quite distinct in places. Looking at \textit{Kmat}s 0.6 and 0.7 the dark canister object has none to very little pixels that are white. However, in the image using a \textit{Kmat} of 0.8, it is completely filled in. The algorithm does in fact show success at differentiating between objects that have different properties.

3.5 \textbf{Marking Areas}

The marking of areas was done using a simple loop that looked at each pixel in the image, and used the coordinates of that pixel to index the Count matrix and see what vote that pixel received for the current \textit{Kmat}. The vote threshold, which is set earlier, is looked at as the baseline for whitening a pixel. Pixels with subsequently higher votes are colored brighter and brighter. The method is the same as in Eqs. 10 and 11, but higher vote counts are checked.
4. Colorization of the Image

The goal of colorization was to approximate the scheme used to colorize images already obtained. Colorizing the image to produce this approximation proved to be the more difficult of the two parts of the project. This was due to several reasons, one of which was the lack of experience in the field of colorization, as well as the amount of noise inherit in the high and low energy images. Several methods were attempted in the process, with the results becoming progressively better with the last images created actually showing signs of matching the detail the project was aiming for from the start.

4.1 Coloring Based Solely on $K_{\text{mat}}$ and Count

The first experiments with colorization made use of the previous results from the methods that whitened certain pixels. The first image created only looked at three $K_{\text{mat}}$ values. These values were chosen as, 0.6, 0.7, and 0.8 due to montage results such as Figure 3.3. The assigning of the colors was done based on three things, the $K_{\text{mat}}$, the vote corresponded to, the amount of votes, and the colorization the image that this method was to approximate. From this given image, and the montage results, it was determined that $K_{\text{mat}}$s in the 0.8 range were orange, 0.7 should be green, and 0.6 should be blue with a mixture of green. The colors were assigned directly, in RGB format, according to which $K_{\text{mat}}$ the loop was on, $K_{\text{mat}}$ was looped in coloring as in montage, and the number of votes

\[
K_{\text{mat}} = 0.6 \text{ AND } \text{Count(pixel)} \geq \text{vote\_threshold} \Rightarrow \text{new\_img(pixel,1)} = 0 \\
\text{new\_img(pixel,2)} = 0.3 \\
\text{new\_img(pixel,3)} = 0.4 \\
\]

\[
K_{\text{mat}} = 0.7 \text{ AND } \text{Count(pixel)} \geq \text{vote\_threshold} \Rightarrow \text{new\_img(pixel,1)} = 0 \\
\text{new\_img(pixel,2)} = 0.5 \\
\text{new\_img(pixel,3)} = 0 \\
\]

\[
K_{\text{mat}} = 0.8 \text{ AND } \text{Count(pixel)} \geq \text{vote\_threshold} \Rightarrow \text{new\_img(pixel,1)} = 0.5 \\
\text{new\_img(pixel,2)} = 0.3 \\
\text{new\_img(pixel,3)} = 0 \\
\]

New\_img is the colorized image that is displayed. The three values associated with each pixel in new\_img represent the red, green, and blue channels respectively.

Displayed in Figure 4.1, is the image created using this file, and in Figure 4.2 is the image with the colorization scheme that the project is attempting to reproduce.
kmat: 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0, 1.1, 1.2

Figure 4.1 Image Created using Kctbmb134bcoloring.m

Figure 4.2 Target Colorization Image
The image created did not come close, using just the $K_{mat}$ and the vote, to the detail in the given image. However, some correlation could be seen which did show the process was on the right track. Notice two of the darkest orange areas in the given image, the canister in the upper right and the other rectangular shape in the bottom left. Looking at the created image these same two objects are colored bright orange. There is some miss coloring as the green object in the upper left of the luggage in the given image, appears to be colored more blue in the created image. Further attempts lowered the noise and colored more according to the goal being aimed at. A description of the major adjustments follows with images where appropriate.

4.2 Using Original Gray Level In Color Equations

In the coloring equations the result was not just based on $K_{mat}$ and the vote, it also included the original gray level image's value at that pixel. However, the image's background was originally set to white and not to the grey level. This in fact caused some of the loss of detail and resulted in the image below in figure 4.3. Also, the normalization process used caused detail to be lost. Instead of normalizing the entire color channel, the gray level intensity was normalized and then added to the RGB channels to produce a certain color. A later method corrected this problem.
4.3 Keeping Track of $K_{mat}$ Corresponding to Maximum Vote Count

One of the errors in the algorithm was that a pixel could be colored twice, the second coloring overwriting the first, if its vote for two or more $K_{mat}$s was above the vote threshold. This was due to the fact that a different Count matrix was used for each $K_{mat}$, and therefore a pixel could be colored based on a vote for one $K_{mat}$, and then a later $K_{mat}$ also with a vote greater than the threshold. In order to correct this, a new matrix was created called Max_ktmp. As the different $K_{mat}$s were looped through, a variable cnt was incremented where the Count matrix at the pixel coordinates was previously incremented. In this new method, the value in cnt, after progressing through all the background pixels, is compared to the previous value in Count at the current test pixel coordinates. If the cnt variable is greater then it is stored into the Count matrix and the Max_Ktmp matrix has stored in it, at the pixel’s coordinates, the $K_{mat}$ at which this maximum vote occurred. This not only solved the problem of coloring a pixel more than once, it eliminated the need for a Count matrix corresponding to each $K_{mat}$ value tested since each pixel now had only one count value associated with it.

4.4 Normalization and Using Gray Level as a Background

Finally a method was used that began to show results, and ultimately led to a routine that came very close to the goal being aimed at. First, it was decided to ignore any pixel with intensity greater than 230, as this corresponds to dead space, in the calculation of votes. The color image then had all of its values, each color channel at each pixel, set to the value of the original gray level intensity of the high band image, normalized so the values are between zero and one.

\[
\text{new_img(pixel,1)} = \frac{\text{H_img(pixel)}}{\text{H_max}} \quad (15)
\]

\[
\text{new_img(pixel,2)} = \frac{\text{H_img(pixel)}}{\text{H_max}} \quad (16)
\]

\[
\text{new_img(pixel,3)} = \frac{\text{H_img(pixel)}}{\text{H_max}} \quad (17)
\]

Initializing the image in this way meant that any pixel uncolored would simply look as it did in the original image. This kept the detail of the image, simply making the background black or white would lose this original detail.

As well, the coloring equations were changed in the way that they did normalization. In section 4.2 the equations normalized the gray-level values and then added certain fractions to that normalized gray level value based on what color was desired. This made it difficult to maintain a color channel value between zero and one. That factor caused checks to be needed to set the channel to zero or one if the value fell below zero or above one. The loss of detail because of this actually was a major culprit, as it was discovered this was happening on around half of the pixels. The equations were then reworked to normalize the color channels after their values had been assigned. Displayed below is a
set of coloring equations from when $K_{mat}$ is equal to 0.8 (a reddish-orange color) and then the normalization equations.

\[
\text{new\_img(pixel,1)} = \text{color} + H_{\text{img}}(\text{pixel}) \tag{18}
\]
\[
\text{new\_img(pixel,2)} = \text{new\_img(pixel,1)} * 4 / 10 \tag{19}
\]
\[
\text{new\_img(pixel,3)} = \text{new\_img(pixel,1)} / 10 \tag{20}
\]
\[
\text{new\_img(pixel,1)} = \text{new\_img(pixel,1)} / \text{redchannel\_max} \tag{21}
\]
\[
\text{new\_img(pixel,1)} = \text{new\_img(pixel,1)} / \text{greenchannel\_max} \tag{22}
\]
\[
\text{new\_img(pixel,1)} = \text{new\_img(pixel,1)} / \text{bluechannel\_max} \tag{23}
\]

The gray level image’s value ($H_{\text{img}}$) was used in Eqs. 13-15, but it was un-normalized when used in these equations since the entire color channel would be normalized later on with Eqs. 16-18. Color is simply an offset that is set equal to the number of votes that pixel received. Since a reddish-orange color is the desired shade, the first color channel, red, is set based on the offset and the original gray level in the high-energy image. The other two channels are set equal to fractions of the dominant color; this is how variations on a red, green, or blue color can be created. In this instance, the red is added to fractions of blue and green to create a kind of orange. If the $K_{mat}$ was lower and the target color was a type of green then the green channel would be set based on the offset and the gray level, and the red and blue channels would be fractions of the value assigned to the green channel. This method is followed in all coloring equations.

Finally, a wiener filter was applied to the color channels in order to reduce noise associated with the colorization. Wiener was first invoked to determine the noise of the channel. Wiener was then run again using the noise calculated as a parameter into the function to help the filter more accurately remove the noise. This method produced the image in Figure 4.4 on the next page, with Figure 4.2 below it.
Figure 5.2: Image Provided with Dual Energy Data, Coloring Project was aiming for

Figure 5.4: Colorized Image Using $K_{\text{max}} 0.1 - 1.0$ produced by Kctbmb134coloring43greynorm_wiener.m
5. Conclusions and Thoughts On Future Work

After several attempts and misfires, the images created by this project began to very closely resemble those images that were used as a template. It is clear there is still noise in the project’s outputted images, but it is also clear that the two images, the one given and the one created, share the same colorization scheme and color similarly.

In order to remove some of the noise there are several ideas that could be put into use. A method of looking at the surrounding colors of each pixel and partly basing the coloring on their average could be used to average out the noise present and create a smoother looking image. The wiener filter could also perhaps be tweaked to produce better smoothing, however with this, and any other averaging attempt including the first one described, there is the possibility of losing detail if the window of surrounding pixels used for noise reduction is too large. As well, a good place to start would be to attempt and reduce the noise inherent in the dual band images before coloring.

The created image does appear to have a redder tint in its orange, and the blue and green are also not exactly the same tint. Adjusting the equations used by the code can solve problems such as these. The objects are distinguished from each other in the same way, just colored slightly different. The algorithm using $K_{tb}$ and $K_{mat}$ worked, as it should, though there are almost assuredly ways to improve upon the accuracy of its detection of specific $K_{mat}$s. The current project could not determine the accuracy as no dual energy images were provided that did not include a threat object.

This method does show a great ability to distinguish between objects consistently and effectively. With more time spent on the processing of the final image the result can look as smooth as the image provided with the data. The fact that method does distinguish effectively and can color effectively makes it a powerful tool in the detection of threat objects in airport security systems. By gathering more dual energy images from a larger pool, containing non-threatening luggage, work can be done to determine and improve the accuracy of threat detection, creating a more secure and hassle-free airport environment.
Read in High and Low image data

Compute H and L values

Loop 1 to C K_{max}

Loop 1 to J Test Pixels

For 1 to N Background Distances

For 1 to M Background Pixels at Background Distance

Calculate K_{tb}

If K_{tb} within specified range of K_{max}
increment cnt variable

Next M

Next N

If cnt > Count(j)
Count(j) = cnt
Max_k_{tmp}(j) = C

Next J

Next K

Flowchart of General Algorithm
Colorization Flowchart

FOR I = 1 to N pixels

Check to see which K had the most number of votes for this pixel.

If the \#votes > vote_threshold

Color new_img accordingly

Next I

For J = 1 to 3 Color Channels of new_img

Find Max of color channel

Next J

For K = 1 to N Pixels

Divide each color channel for pixel that is > 1 by appropriate max

Next K
REFERENCES

1. **Wang, Ta Wee.** "Automated Threat Recognition Using Stereoscopic Dual-energy X-ray Images", 
   [Http://www.eee.ntu.ac.uk/research/vision/projects.html](http://www.eee.ntu.ac.uk/research/vision/projects.html)