1 Abstract

The increase in sophistication of image manipulation software such as Adobe Photoshop, coupled with improvements in imaging algorithms, has given rise to digital forgeries that are very hard to detect. My project here is based on the paper 'Exposing Digital Forgeries in Complex Lighting Environments' [4], wherein the authors detail a technique to identify digital forgeries via inconsistencies in the lighting model.

2 Introduction

In the recent past we have seen many cases of images being tampered. A few that I showed in my presentation are displayed here in Figures 1(a)–(c). The images are quite compelling composites at first glance. Only on close inspection of, say the Brad Pitt and Angelina Jolie image, can you quantify that the sand is perhaps slightly different. Prof. Hany Farid’s page on 'Photo Tampering Throughout History' contains many more such images. The field of digital forensics has grown substantially in the past few years to counter this menace. Many techniques have been developed to detect forms of digital tampering; statistical techniques for detecting cloning [2,8], splicing [7], resampling artifacts [1,9], color filter aberrations [9], and disturbances of a camera’s sensor noise pattern [6].

This paper uses the property that images usually have a complex lighting environment and thus matching the environment exactly might not be possible. In previous papers, the authors have shown techniques to estimate the direction of the light source, and detecting tampering via inconsistencies in lighting direction [3,5]. Here, the approach is slightly modified to allow for the effect of multiple light sources. The paper displays how computing lighting coefficients for different parts of an image can help in distinguishing the original image from the forgeries.

3 Methods

The authors make a few simplifying assumptions to model the complex lighting environment. The first assumption is that the lighting is distant. Thus we’ll see constant behavior from different parts of the object. The second assumption is that the surface is convex and Lambertian. The convex surface assumption negates the possibility of cast shadows and interreflections. For a Lambertian surface, the apparent brightness of the surface to an observer is independent of the observer’s angle of view. Two further minor
(a) Star magazine cover of April 2005.  
(b) Doctored image of Vice-Presidential candidate Sarah Palin.  
(c) Image exaggerating Iran's arsenal. 

Figure 1: Digital forgeries.
assumptions are also necessary. The first one follows from the surface being Lambertian. The surface reflectance is constant, i.e. the radiation is reflected in all directions nearly equally. Also, the camera response is linear, implying that the lighting coefficients can only be estimated to within an unknown scale factor.

3.1 Representing lighting environments

Under the assumption of distant lighting, an arbitrary lighting environment can be expressed as a function on the sphere, \( L(\vec{V}) \), where \( \vec{V} \) is a unit vector in Cartesian coordinates and the value of \( L(\vec{V}) \) is the intensity of the incident light along direction of \( \vec{V} \). As a result of the convex surface assumption, we can parametrize the irradiance, \( E(\vec{N}) \), by the unit length surface normal \( \vec{N} \). The irradiance can be written as the convolution of the lighting environment \( L(\vec{V}) \), and the reflectance function of the surface, \( R(\vec{V}, \vec{N}) \):

\[
E(\vec{N}) = \int_\Omega L(\vec{V}) R(\vec{V}, \vec{N}) d\Omega,
\]

where \( \Omega \) represents the surface of the sphere and \( d\Omega \) is an area differential on the sphere. We can expand the integral above in terms of spherical harmonics. With some further clever math as detailed in the paper, we can relate irradiance to intensity via the following equation:

\[
I(\vec{x}) = f(\rho t E(\vec{N}(\vec{x}))),
\]

where \( \vec{N}(\vec{x}) \) is the surface normal at point \( \vec{x} \), and \( t \) is the exposure time. Since we assume linear camera response, we can further simplify this relationship.

\[
I(\vec{x}) = E(\vec{N}(\vec{x})).
\]

It is important to note that we can only estimate the lighting coefficients to an unknown multiplicative factor as a consequence of our linear camera response.

3.2 Estimating lighting environments

Since we are considering intensity equal to irradiance, we can write the intensity in terms of spherical harmonics. But we need to compute 3D surface normals at \( p \geq 9 \) points on the surface of the object, to estimate a least-squares solution to the system of linear equations (refer equation (11) in paper). Without any idea about the geometry of the arbitrary image or without multiple images, this requirement can be fairly intractable. Thus we observe that under the assumption of orthographic projection, the z-component of the surface normal is zero along the occluding contour of an object. So we can write the intensity at any given point along the occluding contour as:

\[
I(\vec{x}) = l_{0,0} \frac{\pi}{2\sqrt{\pi}} - l_{2,0} \frac{\pi}{16} \sqrt{\frac{5}{\pi}} + l_{1,-1} \frac{2\pi}{3} Y_{1,-1}(\vec{N}) + l_{1,1} \frac{2\pi}{3} Y_{1,1}(\vec{N}) + l_{2,-2} \frac{\pi}{4} Y_{2,-2}(\vec{N}) + l_{2,2} \frac{\pi}{4} Y_{2,2}(\vec{N}).
\]
The $Y_{ij}(\cdot)$ functions depend only on the $x$ and $y$ components of the surface normals, and are computed as:

$$Y_{1,-1}(\vec{N}) = \sqrt{\frac{3}{4\pi}} y \quad Y_{1,1}(\vec{N}) = \sqrt{\frac{3}{4\pi}} x$$

(5)

$$Y_{2,-2}(\vec{N}) = 3\sqrt{\frac{5}{12\pi}} xy \quad Y_{2,2}(\vec{N}) = \frac{3}{2}\sqrt{\frac{5}{12\pi}} (x^2 - y^2)$$

(6)

The least-squares solution to the linear system now becomes:

$$M \vec{v} = \vec{b}$$

(8)

This least-squares solution gives us the five lighting coefficients as:

$$\vec{v} = (M^T M)^{-1} M^T \vec{b}.$$ 

(9)

The computation of the surface normals as well as the calculation of the intensities is bound to be noisy, since we are using a reduced model. Thus to better condition the lighting environment vector $\vec{v}$ we consider an error function $E(\vec{v})$, that combines the least-squares solution of the linear system with a regularization term. In this case, we use the Tikhonov regularizer. The error term is defined as:

$$E(\vec{v}) = \|M \vec{v} - \vec{b}\|^2 + \lambda \|C \vec{v}\|^2,$$

(10)

where $\lambda$ is a scalar, and the matrix $C$ is diagonal with $(1\ 2\ 2\ 3\ 3)$ on the diagonal. Matrix $C$ is meant to dampen the effects of higher order spherical harmonics. Taking the derivative of this error term, setting the result to zero, and solving for $\vec{v}$, we get the following expression for $\vec{v}$:

$$\vec{v} = (M^T M + \lambda C^T C)^{-1} M^T \vec{b}.$$ 

(11)

### 3.3 Comparing lighting environments

Now that we have seen how to compute the lighting coefficients, we present a method to compare these coefficients. The intuition is that distinct lighting environments would have different $\vec{v}$ vectors. We would then use these different $\vec{v}$ vectors to detect forgeries in the image. We compute a correlation between two lighting coefficient vectors $\vec{v}_1$ and $\vec{v}_2$ as

$$corr(\vec{v}_1, \vec{v}_2) = \frac{\vec{v}_1^T Q \vec{v}_2}{\sqrt{\vec{v}_1^T Q \vec{v}_1} \sqrt{\vec{v}_2^T Q \vec{v}_2}},$$

(12)
where matrix $Q$ is diagonal with $(0 \frac{\pi}{6} \frac{\pi}{6} \frac{15\pi}{12} \frac{15\pi}{12})$. These lighting coefficient vectors are estimated to within an unknown multiplicative factor. Also, since we assumed a linear camera response, we can ignore any additive bias because of different exposure times. The final error is then given by:

$$D(v_1, v_2) = \frac{1}{2} (1 - corr(v_1, v_2)),$$

with values in the range $[0, 1]$.

### 4 The Dataset

My dataset consists of 19 color images and 1 grayscale image. The solitary grayscale image was simply to test the implementation on a non-color image. Out of the 19 color images, 15 color images were taken from Worth1000. Two images were created by adding forgeries to personal images. Two further images were taken from Google Images.

### 5 Implementation

The steps in the implementation are as follows:

1. Draw a few coarse contours along the occluding boundaries of objects which were in the image, and the forgeries that are being tested. In the paper, the authors chose 2-4 contours, I have typically taken around 5-7. I used Adobe Photoshop to draw these contours, although since these are coarse contours even Paintbrush, IrfanView, or any other basic imaging software would work. An example is given in Figure 2.

![Figure 2: Coarse contours marked out in white only.](image-url)
2. The next step is very important. I fit a quadratic to the previously drawn contours along the occluding boundary, and compute 2D surface normals at each point along the quadratic curve. There are multiple ways of fitting a quadratic curve. For this implementation, I use a basic quadratic function with 3 control points. I also tried using quadratic splines, and the code is attached. But I had issues with the spline when the occluding contour was either horizontal or vertical. In communication with the original authors, it was also found that methods such as snakes and active contours could also be used. For the 3 control points, I rely on user input. The authors of the paper revealed to me that automatic boundary detection is a well-researched yet unsolved problem. Once I fit the quadratic function, I compute the surface normals at each point. This is done by taking the derivative of the function at each point, thereby giving the slope of the tangent. Since the normal is perpendicular to the tangent, I take the negative inverse of the slope and having the point on the quadratic curve, I have the equation for the normal. The surface normal is then normalized so as to get a unit surface normal. Figure 3 shows surface normals along an occluding boundary.

3. The next step is the least-squares estimate of the lighting coefficients, which was defined in detail in the Methods 3 section. I take all the surface normals along the occluding contour and express them in spherical harmonics, as in Equation [7]. I then calculate the intensities along the curve at each point that I computed the surface normal. I am only considering the green channel here; it is possible to consider the red or blue channels too, but only one particular channel should be used. Comparison between color channels is not advisable since lighting coefficients across color chan-
nels are different. The other reason for using the green channel is its preponderance in nature as compared to the other two color channels. Typically one can consider one surface normal per pixel length of the quadratic curve. Using Equation 9, I get the least-squares estimate for the lighting environment.

4. As described in the ‘Estimating Lighting Environments’ subsection 3.2, noise gets added due to the computations in step 2 of the implementation. Thus I combine the least-squares estimate with a regularization term 10, and get the final estimate for the lighting environment 11.

5. Now I have the lighting coefficients along occluding contours of the original image, as well as forgeries added to the image. Figure 4 displays occluding contours along the original image, and the forgery, i.e. all contours along Megan Fox. Using Equation 12, I now compute the correlation between lighting coefficient vectors of the original image and the forgeries added to the image. Lastly, I compute the final error using Equation 13.

6 Results

As described in the dataset section 4, there are 20 digitally manipulated images on which I now present the results of the algorithm described in section 5. The algorithm returned correct results for 17 images. A correct result is when the error distance (Equation 13) between lighting environment vectors computed from the original image is lesser than the
Table 1: Images and lighting environment coefficients.

<table>
<thead>
<tr>
<th>Number</th>
<th>Image name</th>
<th>Lighting coefficients</th>
<th>Success</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Original image only</td>
<td>Original image + forgery</td>
</tr>
<tr>
<td>1</td>
<td>aniston.png</td>
<td>0.0308</td>
<td>0.0642</td>
</tr>
<tr>
<td>2</td>
<td>brangelina.png</td>
<td>0.0504</td>
<td>0.0647</td>
</tr>
<tr>
<td>3</td>
<td>bushkatrina.png</td>
<td>0.0349</td>
<td>0.0640</td>
</tr>
<tr>
<td>4</td>
<td>cigarguyww2.png</td>
<td>0.1768</td>
<td>0.1257</td>
</tr>
<tr>
<td>5</td>
<td>cigarguytodayshow.png</td>
<td>0.0485</td>
<td>0.0598</td>
</tr>
<tr>
<td>6</td>
<td>drew.png</td>
<td>0.0173</td>
<td>0.0493</td>
</tr>
<tr>
<td>7</td>
<td>gscnobama.png</td>
<td>0.0481</td>
<td>0.0462</td>
</tr>
<tr>
<td>8</td>
<td>mandela.png</td>
<td>0.0414</td>
<td>0.0595</td>
</tr>
<tr>
<td>9</td>
<td>marcjlo.png</td>
<td>0.0634</td>
<td>0.0923</td>
</tr>
<tr>
<td>10</td>
<td>mfoxhome.png</td>
<td>0.0180</td>
<td>0.0217</td>
</tr>
<tr>
<td>11</td>
<td>obamabeyonce.png</td>
<td>0.0197</td>
<td>0.0485</td>
</tr>
<tr>
<td>12</td>
<td>obamamaramaradona.png</td>
<td>0.0315</td>
<td>0.0549</td>
</tr>
<tr>
<td>13</td>
<td>olsen.png</td>
<td>0.0498</td>
<td>0.0664</td>
</tr>
<tr>
<td>14</td>
<td>rihanna.png</td>
<td>0.0453</td>
<td>0.0686</td>
</tr>
<tr>
<td>15</td>
<td>sandra.png</td>
<td>0.0762</td>
<td>0.0502</td>
</tr>
<tr>
<td>16</td>
<td>scarletlatifah.png</td>
<td>0.0441</td>
<td>0.0512</td>
</tr>
<tr>
<td>17</td>
<td>scarlet.png</td>
<td>0.0122</td>
<td>0.0347</td>
</tr>
<tr>
<td>18</td>
<td>tigercigarguy.png</td>
<td>0.0390</td>
<td>0.0447</td>
</tr>
<tr>
<td>19</td>
<td>view.png</td>
<td>0.0060</td>
<td>0.0129</td>
</tr>
<tr>
<td>20</td>
<td>pope.png</td>
<td>0.0289</td>
<td>0.0441</td>
</tr>
</tbody>
</table>

As we can see the algorithm does not work for all the images. Let us consider the images where the algorithm fails. Image 4 was the grayscale image. The algorithm clearly does not work in grayscale. It requires a particular color channel only. Grayscale is an unequal combination of all colors. Since every color channel has a different lighting coefficient set, grayscale images are bound to fail with this method. The lighting coefficients for Image 7 for the original image and the forgery are fairly close. This particular image is an exception and it is possible that with the reduced 2D model, the lighting coefficients cannot be distinguished clearly. I must admit though that I was surprised that this image did not produce a positive result. From first glances at the image, it looks like a good image to test the algorithm on. Image 15 produced a negative result because it violated our basic assumption that the lighting is distant. When you look at the image, you see a very bright light on Miss Bullock, whereas the two gentlemen in the background have more diffuse lighting. The brightness of the light led me to believe that the light is fairly close. Let us also look at some of the positive results to understand the working of the algorithm. Let’s consider image 5 with the pure white background. Even though the background is white, there is a clear difference in the light falling on the faces of The Today Show hosts, and the light on the ‘Cigar Guy’s’ face. Also, I tried images
where paintings/portraits have been digitally altered. Examples of such images include image numbers 6, 13, and 17. These examples really help in quantifying the worth of the algorithm. First glances at the images do not betray any indication of digital forgery. But clearly paintings are complex lighting environments and it would be expected that reproducing the environment in its entirety would be quite a complicated task.

The direction to a light source can also be estimated using $\tan^{-1}(l_{-1,1}, l_{1,1})$. Refer Figure 5 for an example. The direction to the light source for Pope Benedict XVI is $48^\circ$, whereas it is $22.23^\circ$ for the crashing airplane. Finally I show a plot of the errors between the contours selected for the original image, and contours selected for the forgery. As is evident, the original image error estimate is lesser than the error estimate of the original image with the forgery.
7 Conclusions

The algorithm had a 89.5% success rate, only returning incorrect results for two of the images. I'm discounting the grayscale image since grayscale images are beyond the scope of the algorithm. The most significant concern that I had with the algorithm was the utter dependence on the user. The user has to define the control points which are an integral part of the algorithm. If the user is not careful in this step, the algorithm can produce very wrong results. Thus I feel it would be productive to invest time in coming up with an automated approach to boundary detection or a better segmentation algorithm. In my implementation I only use a very basic quadratic function decided by the 2 control points. Using quadratic splines, active contours, snakes etc. would lead to more reliable results. Another significant point is that the distant lighting and Lambertian surface assumptions are not always true. Although I did not encounter the Lambertian surface problem, the authors give example images of this case. Also the algorithm requires the presence of enough surface normals. This is a hard task too. In some of the images, the forgeries are very close to the objects in the original image, thereby making it difficult to come up with enough occluding contours. Finally, since I use the 2D model, my results are not as robust as the 3D model. But all in all, I think this is a particularly useful first order approximation. Also I had fun researching the literature on digital forensics and lighting environments. I share the original authors' opinion that with increasing sophistication of forensic tools, it would become increasingly more difficult to create compelling forgeries.

References


