Exposing Digital Forgeries in Scientific Images

Hany Farid
Department of Computer Science
Dartmouth College
Hanover, NH 03755
farid@cs.dartmouth.edu

ABSTRACT

A recent case of scientific fraud, involving manipulated images in a high-profile scientific publication, has sent shockwaves through the scientific community. By some measures, however, this case is not isolated – in at least one journal, it is estimated that as many as 20% of accepted manuscripts contain figures with inappropriate manipulations, and 1% with fraudulent manipulations. Several scientific editors are considering putting safeguards in place to help reduce these numbers. While sensible policy and awareness are certainly important, there is likely to be a need for computational techniques that automatically detect common forms of tampering. We describe three such techniques for detecting traces of tampering in scientific images. Specifically, image segmentation techniques are employed to detect image deletion, "healing", and duplication.

Categories and Subject Descriptors

I.4 [**Image Processing**]: Miscellaneous

General Terms

Security

Keywords

Digital Tampering, Digital Forensics

1. INTRODUCTION

In 2004, Professor Hwang Woo-Suk and colleagues published what appeared to be ground-breaking advances in stem cell research [5]. Their paper appeared in *Science*, one of the most prestigious scientific journals. In late 2005, evidence slowly emerged that these results were manipulated and/or fabricated. After months of controversy, Hwang retracted the *Science* paper [7] and resigned his position at Seoul National University. An independent panel investigating the accusations of fraud found, in part, that at least

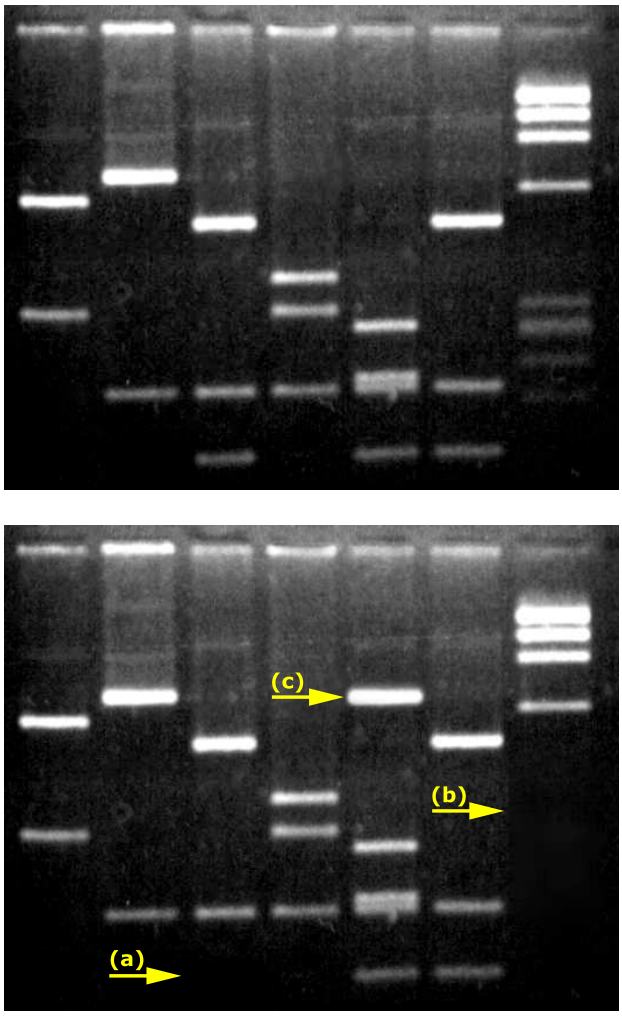

Figure 1: A sample gel that separates DNA fragments (top), and the result of several digital manipulations: (a) a band is erased; (b) several bands are removed using Photoshop's healing brush; and (c) a band from the left is copied and pasted into a new location.

MM&Sec'06, September 26–27, 2006, Geneva, Switzerland.

nine of the eleven customized stem cell colonies that Hwang had claimed to have made were fakes. Much of the evidence for those nine colonies, the panel said, involved doctored photographs of two other, authentic, colonies.

While this case garnered international coverage and outrage, it is by no means unique. In an increasingly competitive field, scientists are succumbing to the temptation to exaggerate or fabricate their results. Mike Rossner, the managing editor of the *Journal of Cell Biology* estimates that as many as 20% of accepted manuscripts to his journal contain at least one figure that has to be remade because of inappropriate image manipulation, and roughly 1% of figures are simply fraudulent [10].

While the *Journal of Cell Biology* has strict guidelines on which types of digital manipulations are acceptable and which are not [14], many journals do not. In the aftermath of the Hwang case, many scientific journals have begun to consider introducing policy and technological safeguards to reduce the occurrence of misleading and fraudulent images. To this end, we describe three techniques for detecting traces of digital tampering specifically in the context of scientific images (e.g., gels, micrographs, etc.). Unlike previous work (e.g., [2, 11, 9, 13, 12, 6, 8]), these techniques are specifically designed for scientific images and for common manipulations that may be applied to them.

Consider, for example, the gel shown in the top panel of Figure 1. Three simple manipulations were performed on this gel: (a) a band was erased; (b) several bands were removed using Photoshop's "healing brush"; and (c) a band was copied and pasted into a new location. In each case, the manipulation disturbs certain image statistics that are often imperceptible to the eye [1]. The erasing of a band removes small amounts of noise that, although not always visually salient, are present throughout the dark background of the image. The "healing" of a region disturbs the underlying spatial frequency (texture) [3]. The duplication of a band leaves behind an obvious statistical pattern – two regions in the image are identical (see [2, 11] for other approaches for detecting duplication). We formulate the problem of detecting each of these statistical patterns as an image segmentation problem. Detecting an erased or healed region, for example, is formulated as a segmentation problem where the gel background is segmented based on intensity or spatial frequency. A description of the segmentation algorithm is first provided, followed by several examples of detecting tampering in real scientific images.

2. METHODS

Consider a weighted graph $G = (V, E)$ with vertices V and edges E. The weight between vertices u and v is denoted as $w(u, v)$. A graph can be partitioned into two disjoint groups A and B such that $A \cap B = \emptyset$ and $A \cup B = V$. The "cost" associated with splitting the graph G into two subgraphs, A and B, is the sum of the weights between all of the vertices in A and B, termed the cut:

$$\text{cut}(A, B) = \sum_{u \in A} \sum_{v \in B} w(u, v). \quad (1)$$

[1]While many image manipulations are imperceptible to the eye, they may sometimes be revealed by contrast enhancing the image (erasure/healing), or by careful inspection of small image features (duplication).

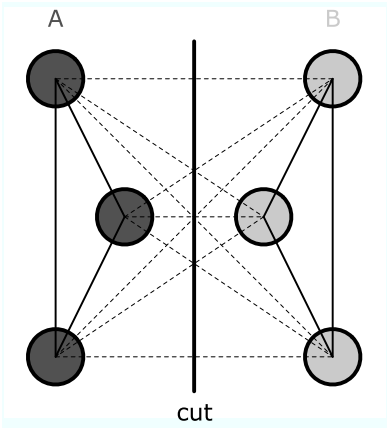

Figure 2: A six vertex graph. The weights between vertices colored with the same gray value are large (solid line), while the other edges have a small weight (dashed line). A bipartioning into sets A and B cuts this graph along low-cost edges.

The optimal bipartioning of a graph is that which minimizes the cut. Shown in Figure 2, for example, is a graph with six vertices. The weights between vertices colored with the same gray value are large (solid line), while the other edges have a small weight (dashed line). The optimal bipartioning, therefore, is one that partitions the graph along the line labeled "cut", which severs only low-cost edges, leaving three vertices in each of A and B.

Computing the optimal cut is NP-complete, as there are an exponential number of partitions to consider. There are, however, efficient approximation algorithms. We review one such technique, normalized cuts [15], and then discuss a number of computational issues necessary to make this algorithm efficient for medium- to large-sized graphs.

2.1 Normalized Cuts

When minimizing the graph cut, Equation (1), there is a natural tendency to simply cut a small number of low-cost edges. The normalized cut [15], introduced to remove this bias, is defined as:

$$\text{Ncut}(A, B) = \frac{\text{cut}(A, B)}{\text{assoc}(A, V)} + \frac{\text{cut}(A, B)}{\text{assoc}(B, V)}, \quad (2)$$

where,

$$\text{assoc}(A, V) = \sum_{u \in A} \sum_{v \in V} w(u, v) \quad (3)$$

and

$$\text{assoc}(B, V) = \sum_{u \in B} \sum_{v \in V} w(u, v). \quad (4)$$

This metric normalizes the cut by the total cost of all edges in the entire graph, V. As a result, small partitions are penalized. Solving for the optimal normalized cut is still NP-complete. Formulation as a real-valued problem, however, yields an efficient and approximate discrete-valued solution. A brief overview of this technique is given below – see [15] for complete details.

Let $G = (V, E)$ be a weighted graph with n vertices, and define W to be a $n \times n$ weighting matrix such that

$W_{i,j} = w(i,j)$ is the weight between vertices i and j. Define D to be a $n \times n$ diagonal matrix whose i^{th} element on the diagonal is $d_i = \sum_j w(i,j)$. Solve the generalized eigenvector problem $(D - W)\vec{e} = \lambda D\vec{e}$, for the eigenvector $\vec{e}$ with the second smallest eigenvalue λ. Let the sign of each component of $\vec{e}$ (corresponding to each vertex of G) define the membership of that vertex into one of two sets, A or B – for example, vertices with corresponding negative components are assigned to A and vertices with corresponding positive components are assigned to B.

2.2 Implementation Details

In our applications, we will be considering the partitioning of a graph with possibly as many vertices as pixels in an image. Contending with even a modest-sized image of 256×256 pixels is computationally prohibitive, as it requires us to solve a $65,536$-D eigenvector problem. Note that most of this computation is unnecessary, as only the second smallest eigenvalue eigenvector is required to partition a graph. To this end we employ Lanczos' method [4] for estimating large, sparse and symmetric eigenproblems, such as ours. This technique is particularly efficient when only a few of the extremal (maximum or minimum) eigenvalue eigenvectors are needed. Since our graphs will typically be sparse, sparse matrix calculations may be employed to further reduce the memory and computational costs. Complete details of these issues are given in Appendix A.

2.3 A Toy Example

In order to demonstrate how graph bipartioning may be used to segment an intensity image, consider the simple 11×11 pixel grayscale image shown in Figure 3. If we desire to segment this image based on gray value, then the pixels should be portioned along the vertical mid-line, where there is a clear change in brightness.

A graph $G = (V, E)$ is constructed where each pixel is a vertex in V, and there exists an edge in E between all pairs of vertices. The edge between vertices of similar gray values are given a large weight, while the edge between vertices with different gray values are given a small weight. Superimposed on the image of Figure 3 is such a graph, on a subset of the pixels. The solid edges represent a large weight, while the dashed edges represent a small weight. Given such a graph, a cut that separates the pixels along the vertical mid-line will be small, while any cut that separates pixel with similar gray value will be costly. Each edge may be assigned a weight given by the following Gaussian:

$$w(i,j) = e^{\frac{-(I(i)-I(j))^2}{\sigma_I^2}}, \quad (5)$$

where, $I(\cdot)$ denotes the gray value at a given pixel. When segmenting an image, it is natural to prefer to group pixels that are in spatial proximity to one another. This constraint may be added by augmenting the above weight with an addition term:

$$w(i,j) = e^{\frac{-(I(i)-I(j))^2}{\sigma_I^2}} \cdot e^{\frac{-\Delta_{i,j}^2}{\sigma_\Delta^2}}, \quad (6)$$

where $\Delta_{i,j}$ is the Euclidean distance between pixels i and j:

$$\Delta_{i,j} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}. \quad (7)$$

Note that the addition of this Euclidean distance constraint gives rise to a sparse graph, since only nearby pixels will have a non-zero weight. In practice, of course, this Euclidean metric needs to be truncated to zero for all values below a specified threshold.

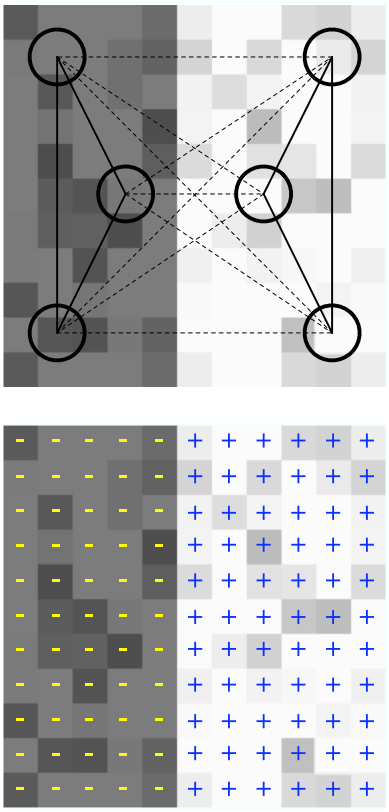

Figure 3: Shown in the top panel is a graph on a subset of the 11×11 pixel image. The edges joining pixels of similar gray value are large (solid lines), while the edges joining pixels of different gray value are small (dashed line). The normalized cut of a complete graph, constructed on all pixels, segments the image into regions of similar intensity. Shown in the bottom panel is the resulting segmentation where each pixel is annotated with $-$ or $+$ denoting its group membership.

The 11×11 pixel image of Figure 3 yields a 121×121 weighting matrix, constructed according to Equation (6). In this example $\sigma_I^2 = 0.01$ (gray values are scaled into the range $[0, 1]$) and $\sigma_\Delta^2 = 0.1$ (the sampling lattice is scaled into the range $[0, 1]$). The $n \times n$ diagonal matrix D is constructed so that the i^{th} element on the diagonal is $d_i = \sum_j w(i,j)$.

In this small toy example, the second smallest eigenvalue eigenvector can be solved for directly. Specifically, the generalized eigenvector problem $(D - W)\vec{e} = \lambda D\vec{e}$ is cast into a standard eigenvector problem, $D^{-1/2}(D - W)D^{-1/2}\vec{e} = \lambda\vec{e}$. The eigenvector, $\vec{e}$, with the second smallest eigenvalue, λ, of this system is a 121-D vector. The sign of each component of this vector corresponds to the group assignment of each vertex (i.e., image pixel). Shown in the bottom panel of Figure 3 is the resulting segmentation of the dark ($-$) and light ($+$) pixels. As described above, larger eigenvector problems require a more memory and computationally

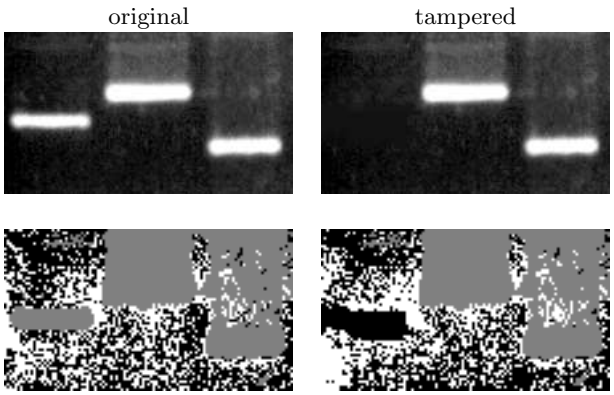

Figure 4: In the tampered image on the right, a band is removed by replacing a rectangular region with the mean intensity of the background. Shown below are the results of intensity-based segmentation. The gray pixels correspond to the first iteration where the bright bands are separated from the dark background. The black/white pixels correspond to grouping of only the background. This grouping clearly reveals the deleted band.

efficient approach, as described in Appendix A.

3. IMAGE MANIPULATIONS

Shown in Figure 1 are several manipulations on a common scientific image, a gel. The detection of each of these manipulations is framed as a segmentation problem. The graph cut algorithm described above is then used to detect traces of these forms of tampering. In the following three sections, the basic formulations are described and their efficacy shown on simple examples. The final section describes a technique for automatically detecting tampering based on results of segmentation.

3.1 Intensity

In order to detect differences in background intensity, a weighting function based on differences in intensity and distance between pixels is employed:

$$w(i,j) \quad = \quad e^{\frac{-(I(i)-I(j))^2}{\sigma_I^2}} \cdot e^{\frac{-\Delta_{i,j}^2}{\sigma_\Delta^2}}, \qquad (8)$$

where, $I(\cdot)$ denotes the gray value at a given pixel, and $\Delta_{i,j}$ is the Euclidean distance, Equation (7), between pixels i and j.

Shown in the top row of Figure 4 is a portion of a gel (left) and a tampered version (right), where a band has been erased by replacing a rectangular region around it with the mean of the background intensity. Shown in the bottom row of this figure are the results of graph cut segmentation. These three-tone images correspond to two iterations of segmentation. In the first iteration, the image is grouped into regions corresponding to the bands (gray pixels) and the background. In the second iteration, the background region is further grouped into two regions (black and white pixels). Note that in the original image, the grouping of the background does not yield any large cohesive areas, while the grouping of the tampered region clearly yields the rectangular region that was erased. See Section 3.4 for an automatic approach to detecting tampering from these segmentation results.

While the above formulation and example pertains to grayscale images, the extension to color is straight-forward. In order to detect differences in a RGB color image, the weighting function takes the form:

$$w(i,j) \quad = \quad e^{\frac{-\|\vec{I}(i)-\vec{I}(j)\|^2}{\sigma_I^2}} \cdot e^{\frac{-\Delta_{i,j}^2}{\sigma_\Delta^2}}, \qquad (9)$$

where $\vec{I}(\cdot)$ is a 3-D vector consisting of the RGB values at a given pixel, and $\|\cdot\|$ is vector norm. This approach may, of course, be applied to an image representation of choice, for example, RGB, HSV, or CMYK.

3.2 Texture

In order to detect differences in background texture, a weighting function based on differences in the magnitude of the image gradient [2] and distance between pixels is employed:

$$w(i,j) \quad = \quad e^{\frac{-(I_g(i)-I_g(j))^2}{\sigma_g^2}} \cdot e^{\frac{-\Delta_{i,j}^2}{\sigma_\Delta^2}}, \qquad (10)$$

where, $I_g(\cdot)$ denotes the magnitude of the image gradient at a given pixel, and $\Delta_{i,j}$ is the Euclidean distance, Equation (7), between pixels i and j. The image gradient is computed as follows. The partial derivative of the intensity image is computed via linear convolutions with a pair of 1-D filters [3]:

$$I_x(x,y) \quad = \quad (I(x,y) \star d(x)) \star p(y) \qquad (11)$$

$$I_y(x,y) \quad = \quad (I(x,y) \star p(x)) \star d(y). \qquad (12)$$

The magnitude of the gradient, $\nabla I(x,y) = [I_x(x,y)\ \ I_y(x,y)]$, is given by:

$$I_g(x,y) \quad = \quad \sqrt{I_x^2(x,y) + I_y^2(x,y)}. \qquad (13)$$

Shown in the top row of Figure 5 is a portion of a gel (left) and a tampered version (right), where a band has been erased using the Photoshop "healing brush" tool. The healing brush removes small blemishes, but also has the effect of reducing overall spatial frequency [3]. Shown in the bottom row of this figure are the results of graph cut segmentation. These three-tone images correspond to two iterations of segmentation. In the first iteration, the image is grouped, using intensity-based segmentation (Section 3.1) into regions corresponding to the bands (gray pixels) and the background. In the second iteration, the background region is further grouped into two regions (black and white pixels) using the texture-based segmentation described in this section. Note that in the original image, the grouping of the background does not yield any large cohesive areas, while the grouping of the tampered region clearly yields the region that was "healed". See Section 3.4 for an automatic approach to detecting tampering from these segmentation results.

[2] The Laplacian, $I_l(x,y) = I_{xx}^2(x,y) + I_{yy}^2(x,y)$ may also be considered as a measure of background texture. We find, however, that the image gradient provides slightly better results.

[3] A pair of 7-tap derivative filters [1] are used to compute image derivatives:
$p(\cdot) = [0.0047\ 0.0693\ 0.2454\ 0.3611\ 0.2454\ 0.0693\ 0.0047]$
and
$d(\cdot) = [0.0187\ 0.1253\ 0.1930\ 0.0\ -0.1930\ -0.1253\ -0.0187]$.

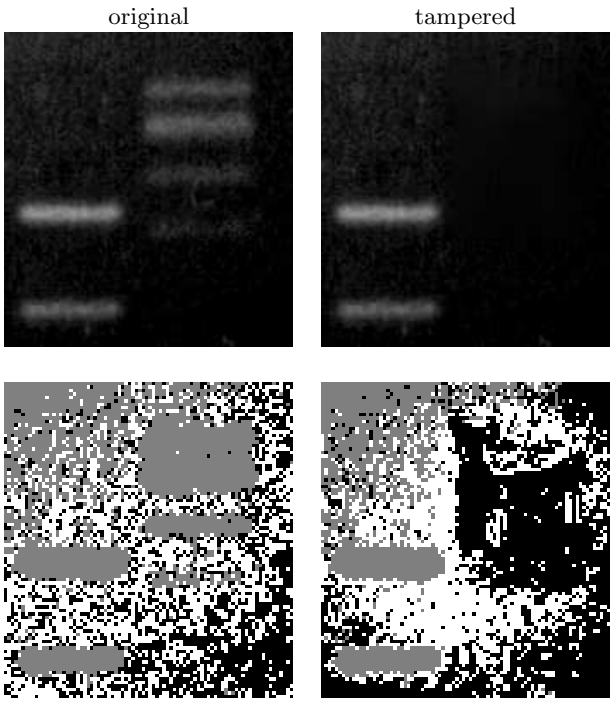

Figure 5: In the tampered image on the right, several bands are removed using the Photoshop healing brush. Shown below are the results of texture-based segmentation. The gray pixels correspond to the first iteration of intensity-based segmentation where the bright bands are separated from the dark background. The black/white pixels correspond to grouping of only the background. This grouping clearly reveals the deleted bands.

3.3 Duplication

Shown in top row of Figure 6 is a portion of a gel (left) and a tampered version (right), where two bands were copied and pasted into a new location. In order to detect this sort of duplication, a "minimum difference" image is first computed as follows. For a 5×5 neighborhood centered at each pixel, x, y, in the original image, a similarly-sized neighborhood is found with minimum average intensity difference:

$$I_d(x,y) = \text{argmin}_{u,v} \left[\sum_{c_x,c_y=-2}^{2} (I(x-c_x, y-c_y) - I(u-c_x, v-c_y))^2 \right] \quad (14)$$

Duplicated regions are then detected by defining the following weighting function:

$$w(i,j) = e^{\frac{-(I_d(i)-I_d(j))^2}{\sigma_d^2}} \cdot e^{\frac{-\Delta_{i,j}^2}{\sigma_\Delta^2}}, \quad (15)$$

where, $I_d(\cdot)$ denotes the value of the difference image at a given pixel, and $\Delta_{i,j}$ is the Euclidean distance, Equation (7), between pixels i and j. Note that this similarity measure does not contain the location of the duplicated region. This could, however, be easily incorporated by considering a similarity metric on the vector between regions of minimal intensity difference.

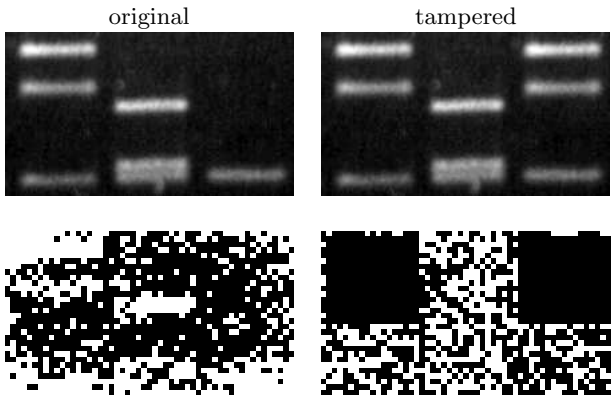

Figure 6: In the tampered image on the right, several bands are copied and pasted into a new location. Shown below are the results of duplication-based segmentation. The black/white pixels correspond to grouping results. This grouping clearly reveals the duplicated bands.

Shown in the bottom row of Figure 6 are the results of graph cut segmentation. These two-tone images correspond to one iteration of segmentation. The black and white pixels correspond to the resulting grouping of the pixels. Note that in the original image, the grouping does not yield any large cohesive areas, while the grouping of the tampered region clearly yields the region that was duplicated. See Section 3.4 for an automatic approach to detecting tampering from these segmentation results

3.4 Automatic Detection

In the previous three sections, we saw the ability of graph cut segmentation to reveal traces of three types of tampering. While the traces of tampering are fairly obvious by visual inspection of the resulting segmentation results (Figures 4, 5, and 6), it would be beneficial to have a quantitative metric for automatic detection. Such a metric is introduced below, thus making the detection of tampering fully automatic.

Recall that the segmentation partitions an image into one of two groups, with a group value of 0 or 1. Denote the segmentation map as $S(x,y)$. Consider all pixels x, y with value $S(x,y) = 0$ such that all 8 spatial neighbors, $S(x-c_x, y-c_y)$, $c_x, c_y \in [-1, 1]$, also have value 0. The mean of all of the edge weights between such vertices is computed across the entire segmentation map. This process is repeated for all pixels x, y with value $S(x,y) = 1$. These average weights, s_0 and s_1 are used to quantify the likelihood that the underlying image was manipulated. The intuition behind these metrics is that values near 1 are indicative of tampering because the edges weights in contiguous areas are high indicating significant similarity in the underlying measures of intensity, texture, or duplication.

For the results of Figure 4, the values of s_0 and s_1 are 0.10 and 0.00 for the original image, and 0.91 and 0.16 for the tampered image. For the results of Figure 5, the values of s_0 and s_1 are 0.48 and 0.00 for the original image, and 0.75 and 0.12 for the tampered image. For the results of Figure 6, the values of s_0 and s_1 are 0.00 and 0.00 for the

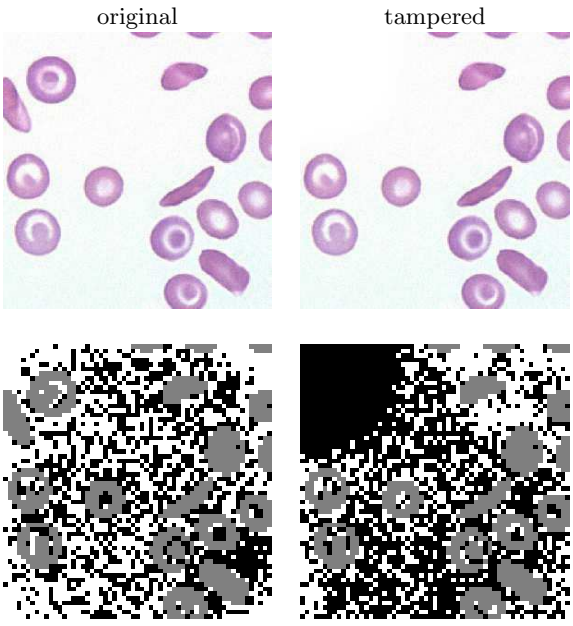

Figure 7: In the tampered image, two cells are removed by replacing a region with the mean intensity of the background. Shown below are the segmentation results, revealing the tampered region.

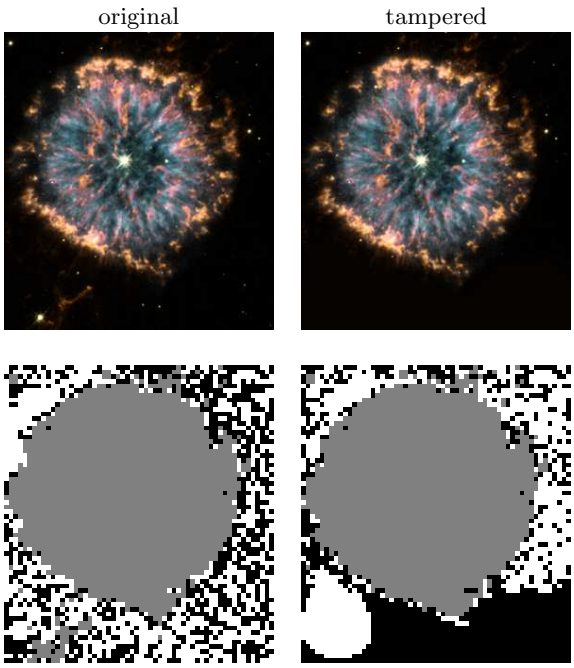

Figure 8: In the tampered image, stars along the bottom are removed by replacing a region with the mean intensity of the background. Shown below are the segmentation results, revealing the tampered region.

original image, and 0.99 and 0.00 for the tampered image. In each case, one group from the tampered images has a high value, while both groups from the original images have a relatively low value. The $\max(s_0, s_1)$ can, therefore, be used as a metric to quantify the likelihood that the image has been doctored – with values closer to 1 denoting a high likelihood, and values closer to 0 denoting a low likelihood.

Shown in Figure 7 is an example of tampering of a microscope image. In the tampered image, two cells are removed by painting the corner with the average intensity of the background. The segmentation clearly reveals the area of tampering. The likelihood of the original image being tampered is 0.19 and the tampered image is 0.99.

Shown in Figure 8 is an example of tampering of an image from the Hubble telescope. In the tampered image, stars along the bottom of the image are removed by painting with the average intensity of the background. The segmentation clearly reveals the area of tampering. The likelihood of the original image being tampered is 0.30 and the tampered image is 0.98. Note that the tampered region is broken into two parts (the left-most corner is grouped differently than the remaining bottom portion of the image). The reason for this is that the similarity metric favors regions that are in close spatial proximity to one another. This constraint can be relaxed by increasing the value of σ_Δ in Equation (8).

Shown in Figure 9 is another example of tampering in an image from the Hubble telescope. In the tampered image a star from the lower-left corner is copied and pasted into each corner of the image. Shown in the bottom row of Figure 9 are the segmentation results. The likelihood of the original image being tampered is 0.50 and the tampered image is 0.97.

Shown in Figure 10 is another example of tampering in a chest x-ray. In the tampered image the tumor in the right lung is removed by duplicating a flipped copy (about the vertical axis) of the left lung, and removing some distinguishing features to make the duplication less obvious. Since the duplicated regions are related by a transformation, the previously described approach has to be adapted. The most straight-forward way to do this is to compute the difference between a local neighborhood and a similarly transformed region elsewhere in the image. Specifically, the difference image, Equation (14) becomes:

$$I_d(x,y) = \text{argmin}_{u,v} \left[\sum_{c_x,c_y=-2}^{2} \left(I(x-c_x, y-c_y) - I(u+c_x, v-c_y)\right)^2 \right] \quad (16)$$

Other transformations such as rotations can be similarly detected by adjusting the difference image appropriately. Shown in the bottom row of Figure 10 are the segmentation results. The likelihood of the original image being tampered is 0.05 and the tampered image is 0.98.

4. DISCUSSION

The ease with which digital media can be altered and manipulated is affecting nearly every corner of our world: media, politics, law, business, and science. As we continue to grapple with the ethical and technological implications of this, it is important that we develop techniques for detecting tampering in digital media. To this end, we, and others, have previously developed general-purpose detection algorithms. Here, we describe detection algorithms specific to

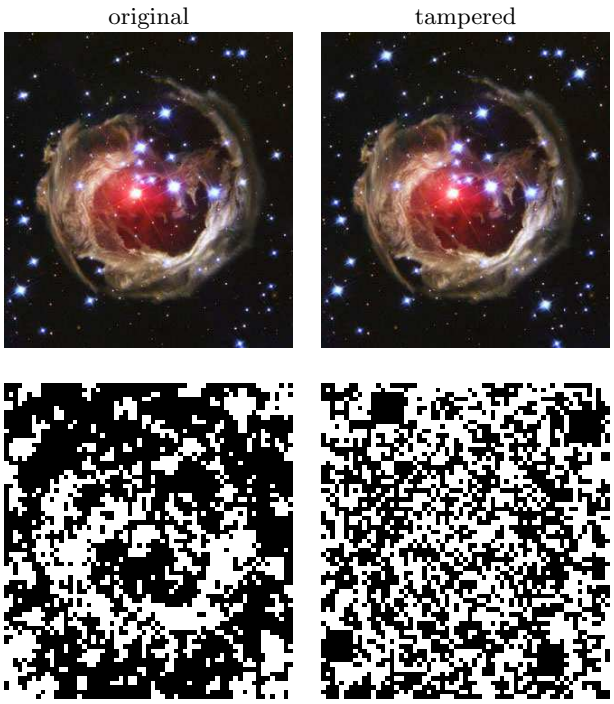

Figure 9: In the tampered image on the right a star from the lower-left corner is duplicated into each corner of the image. Shown below are the segmentation results, revealing the tampered regions.

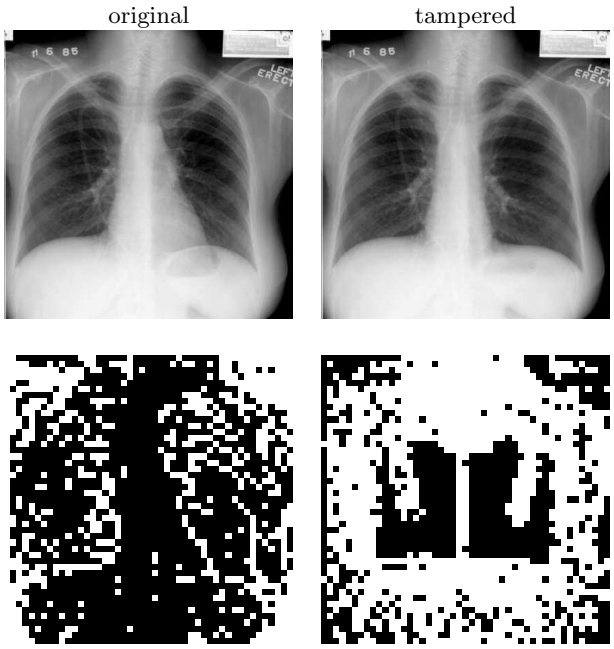

Figure 10: In the tampered image on the right, the tumor is removed from the right lung by duplicating a flipped copy of the left lung, and removing some distinguishing features to make the duplication less obvious. Shown below are the segmentation results, revealing the tampered region.

scientific images. These techniques are applicable to common scientific images such as gels and micrographs. The three detection techniques described here employ a segmentation algorithm for detecting regions in an image with a common intensity pattern, texture pattern, or duplicated pixels. Grouping based on similar intensity and texture detect tampering caused by erasing using a uniform intensity or a tool such as Photoshop's healing brush that affects the underlying spatial frequency. Grouping based on duplicated pixels detects tampering caused by a simple copy/paste. As usual, these techniques are vulnerable to a host of countermeasures that can hide traces of tampering. As we continue to develop new detection techniques, however, it will become increasingly difficult to evade all such approaches.

5. ACKNOWLEDGMENTS

This work was supported by a gift from Adobe Systems, Inc., a gift from Microsoft, Inc., a grant from the United States Air Force (FA8750-06-C-0011), and under a grant (2000-DT-CX-K001) from the U.S. Department of Homeland Security, Science and Technology Directorate (points of view in this document are those of the author and do not necessarily represent the official position of the U.S. Department of Homeland Security or the Science and Technology Directorate).

Appendix A

This section gives a detailed description of how Lanczos' method is used to solve for the second smallest eigenvalue eigenvector of the generalized eigenvector problem $(D - W)\vec{e} = \lambda D\vec{e}$. To begin, this problem is cast into a standard eigenvector problem, $D^{-1/2}(D-W)D^{-1/2}\vec{e} = \lambda\vec{e}$. The second smallest eigenvalue eigenvector is then estimated as follows:

LANCZOS(W,D,M)

$\triangleright$ W is a $n \times n$ weighting matrix
$\triangleright$ D is a $n \times n$ diagonal matrix
$\triangleright$ m = number of iterations
$A = D^{-1/2}(D - W)D^{-1/2}$
$k = 0$
$\vec{v} = \vec{0}$ $\quad\triangleright$ n-D column vector
$\vec{w} = \vec{1}$ $\quad\triangleright$ n-D column vector
$\vec{w} = \vec{w}/\sqrt{w^t w}$ $\quad\triangleright$ unit length
$b_0 = 1$
while $k < m$
    do if $k > 0$
        do for $i = 1$ to n
            do $t = w_i$
                $w_i = v_i/b_k$
                $v_i = -b_k t$
            $E_{\cdot,k} = \vec{w}$ $\quad\triangleright$ k^{th} column of matrix E
        $\vec{v} = \vec{v} + A\vec{w}$
        $k = k + 1$
        $a_k = \vec{w}^t\vec{v}$
        $\vec{v} = \vec{v} - a_k\vec{w}$
        $b_k = \sqrt{\vec{v}^t\vec{v}}$

$$T = \begin{pmatrix} a_1 & b_1 & 0 & 0 & 0 & \dots & 0 \\ 0 & b_1 & a_2 & b_2 & 0 & \dots & 0 \\ \vdots & & & & \vdots & & \vdots \\ 0 & 0 & 0 & 0 & b(m-2) & a(m-1) & b(m-1) \\ 0 & 0 & 0 & 0 & 0 & b(m-1) & a(m) \end{pmatrix}$$

$T\vec{e}_j = \lambda_j\vec{e}_j$ $\quad\triangleright$ solve eigenvector problem, $j = 1...m$
$\vec{x} = \vec{0}$ $\quad\triangleright$ n-D column vector
for $k = 1$ to m
    do $\vec{x} = \vec{x} + \lambda_2 E_{\cdot,k}$
$\triangleright$ $\vec{x}$ is an approximation to the second smallest
$\triangleright$ eigenvalue eigenvector of the matrix A

The number of iterations m is typically chosen to balance accuracy and run-time efficiency. In all of our results, we consider all values of $m \in [5, 50]$ and use the result from the value of m that yields the minimum cost of the resulting graph cut (i.e., the graph cut with the minimum normalized cut, Equation (2)).

6. REFERENCES

[1] H. Farid and E. Simoncelli. Differentiation of multi-dimensional signals. *IEEE Transactions on Image Processing*, 13(4):496–508, 2004.

[2] J. Fridrich, D. Soukal, and J. Lukáš. Detection of copy-move forgery in digital images. In *Proceedings of DFRWS*, 2003.

[3] T. Georgiev. Vision, healing brush, and fiber bundles. In *Proceedings of SPIE*, San Jose, CA, 2005.

[4] G. Golub and C. F. V. Loan. *Matrix Computations*. John Hopkins, Baltimore, MD, 1996.

[5] W. Hwang, S. I. Roh, B. C. Lee, S. K. Kang, D. K. Kwon, S. Kim, S. J. Kim, S. W. Park, H. S. Kwon, C. K. Lee, J. B. Lee, J. M. Kim, C. Ahn, S. H. Paek, S. S. Chang, J. J. Koo, H. S. Yoon, J. H. Hwang, Y. Y. Hwang, Y. S. Park, S. K. Oh, H. S. Kim, J. H. Park, S. Y. Moon, and G. Schatten. Evidence of a pluripotent human embryonic stem cell line derived from a cloned blastocyst. *Science*, 303(5664):1669–1674, 2004.

[6] M. Johnson and H. Farid. Exposing digital forgeries by detecting inconsistencies in lighting. In *ACM Multimedia and Security Workshop*, New York, NY, 2005.

[7] D. Kennedy. Editorial retraction. *Science*, 211(5759):335, 2006.

[8] J. Lukáš, J. Fridrich, and M. Goljan. Detecting digital image forgeries using sensor pattern noise. In *Proceedings of the SPIE*, volume 6072, 2006.

[9] T. Ng and S. Chang. A model for image splicing. In *IEEE International Conference on Image Processing*, Singapore, 2004.

[10] H. Pearson. Image manipulation: CSI: Cell biology. *Nature*, 434:952–953, 2005.

[11] A. Popescu and H. Farid. Exposing digital forgeries by detecting duplicated image regions. Technical Report TR2004-515, Department of Computer Science, Dartmouth College, 2004.

[12] A. Popescu and H. Farid. Exposing digital forgeries by detecting traces of re-sampling. *IEEE Transactions on Signal Processing*, 53(2):758–767, 2005.

[13] A. Popescu and H. Farid. Exposing digital forgeries in color filter array interpolated images. *IEEE Transactions on Signal Processing*, 53(10):3948–3959, 2005.

[14] M. Rossner and K. Yamada. What's in a picture? the temptation of image manipulation. *The Journal of Cell Biology*, 166(1):11–15, 2004.

[15] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.