Digital Image Forensics

farid@cs.dartmouth.edu
www.cs.dartmouth.edu/farid

0. History of Photo Tampering

1. Format-Based Forensics

2. Camera-Based Forensics

3. Pixel-Based Forensics

4. Statistical-Based Forensics

5. Geometric-Based Forensics

6. Physics-Based Forensics

7. Video Forensics

8. Printer Forensics

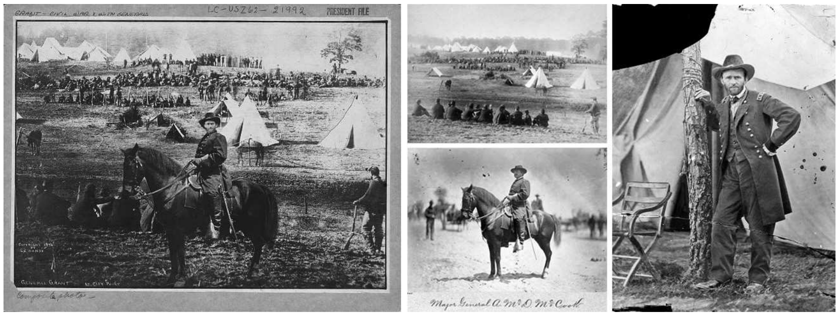
9. MatLab Code

Sections denoted with† cover basic background material.

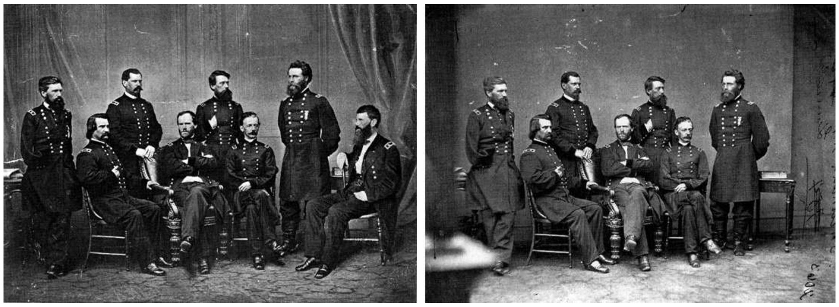
0. History of Photo Tampering

0.1 History

Photography lost its innocence many years ago. Only a few decades after Niepce created the first photograph in 1814, photographs were already being manipulated. With the advent of high-resolution digital cameras, powerful personal computers and sophisticated photo-editing software, the manipulation of photos is becoming more common. Here we briefly provide examples of photo tampering throughout history, starting in the mid 1800s. In each case, the original photo is shown on the right and the altered photo is shown on the left.

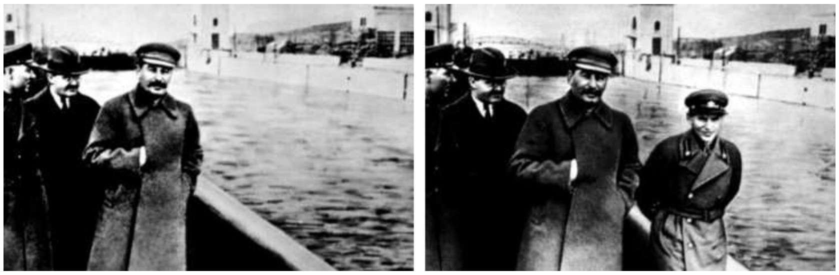


circa 1864: This print purports to be of General Ulysses S. Grant in front of his troops at City Point, Virginia, during the American Civil War. Some very nice detective work by researchers at the Library of Congress revealed that this print is a composite of three separate prints: (1) the head in this photo is taken from a portrait of Grant; (2) the horse and body are those of Major General Alexander M. McCook; and (3) the backgoround is of Confederate prisoners captured at the battle of Fisher's Hill, VA.

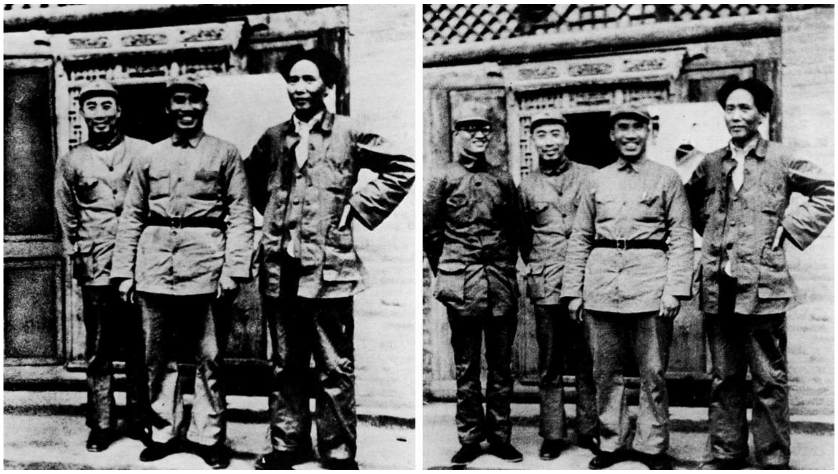


circa 1865: In this photo by famed photographer Mathew Brady, General Sherman is seen posing with his Generals. General Francis

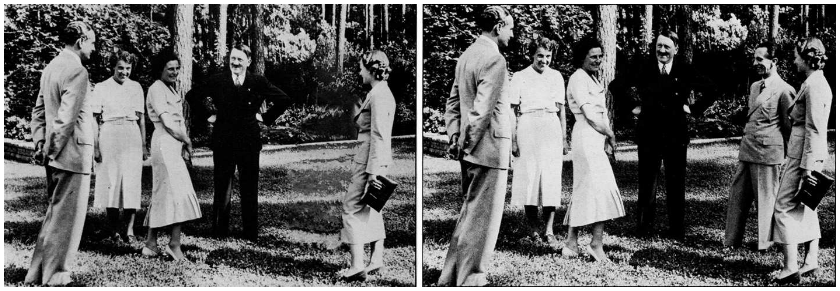
P. Blair, shown in the far right, was inserted into this photograph.



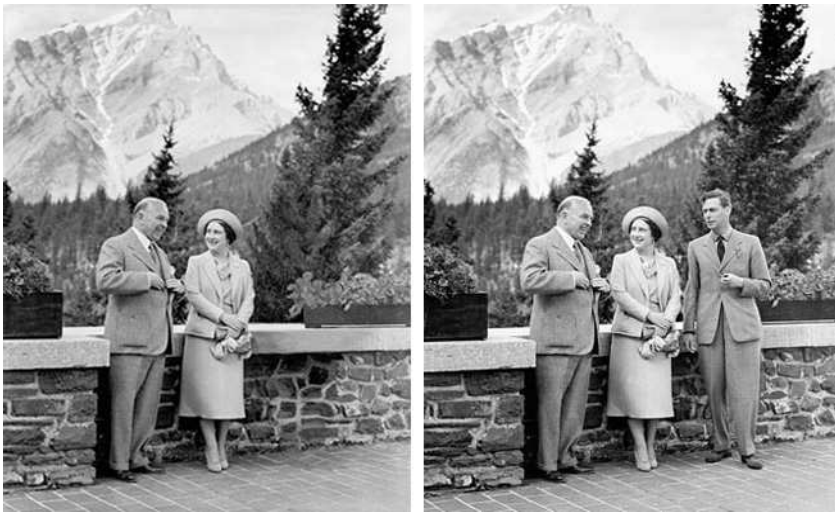
circa 1930: Stalin routinely air-brushed his enemies out of photographs. In this photograph a commissar was removed from the original photograph after falling out of favor with Stalin.



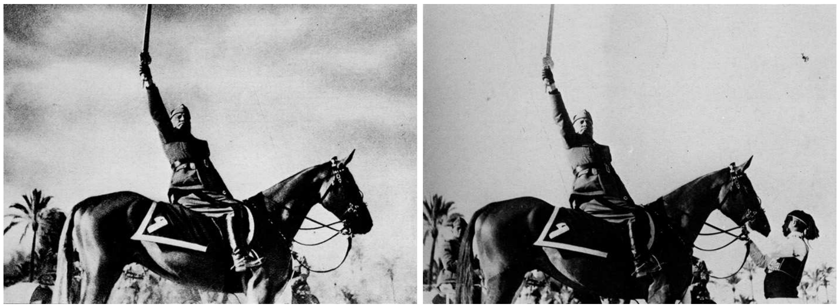
1936: In this doctored photograph, Mao Tse-tung, shown on the far right, had Po Ku removed from the original photograph, after Po Ku fell out of favor with Mao.



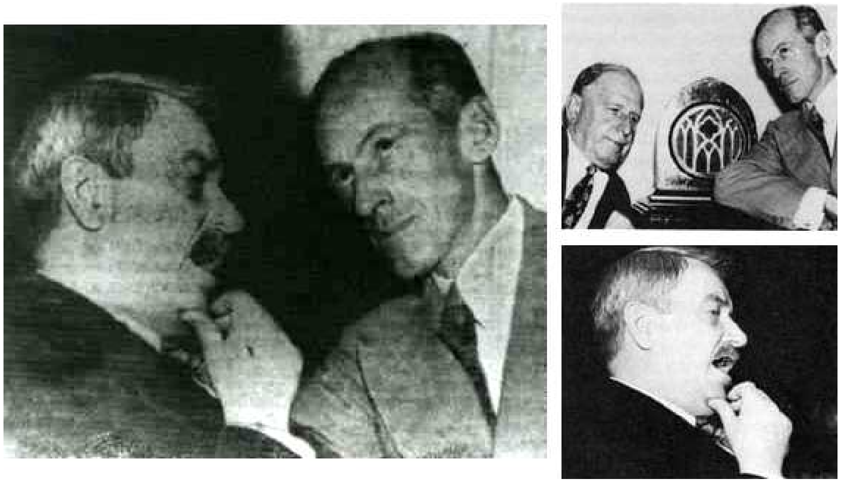
1937: In this doctored photograph, Adolf Hitler had Joseph Goebbels removed from the original photograph. It remains unclear why exactly Goebbels fell out of favor with Hitler.



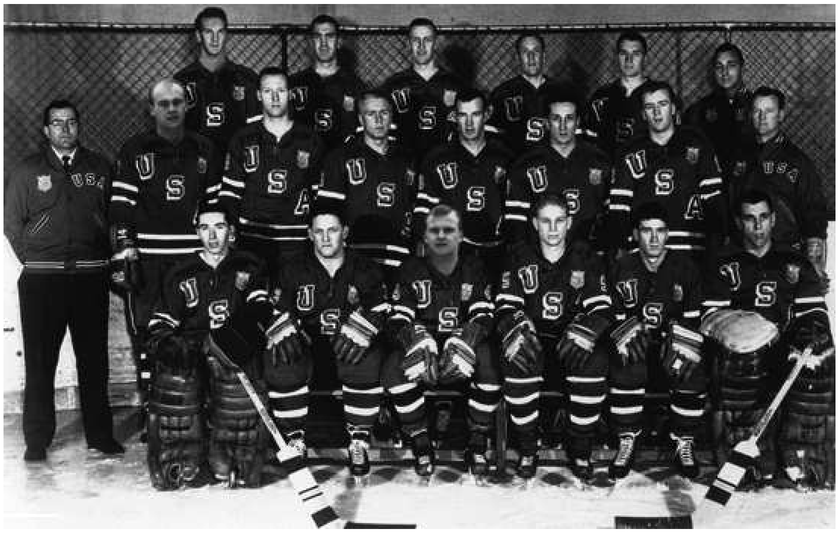
1939: In this doctored photo of Queen Elizabeth and Canadian Prime Minister William Lyon Mackenzie King in Banff, Alberta, King George VI was removed from the original photograph. This photo was used on an election poster for the Prime Minister. It is hypothesized that the Prime Minister had the photo altered because a photo of just him and the Queen painted him in a more powerful light.



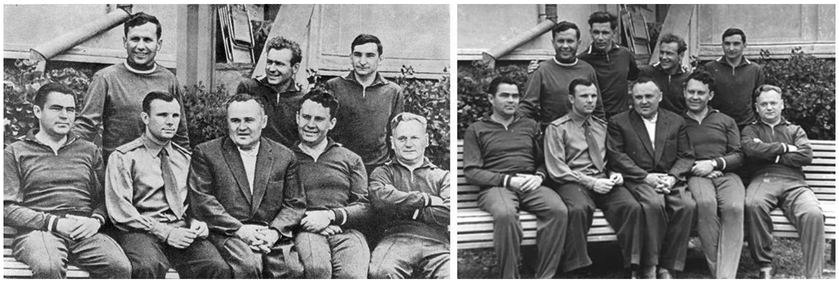
1942: In order to create a more heroic portrait of himself, Benito Mussolini had the horse handler removed from the original photograph.



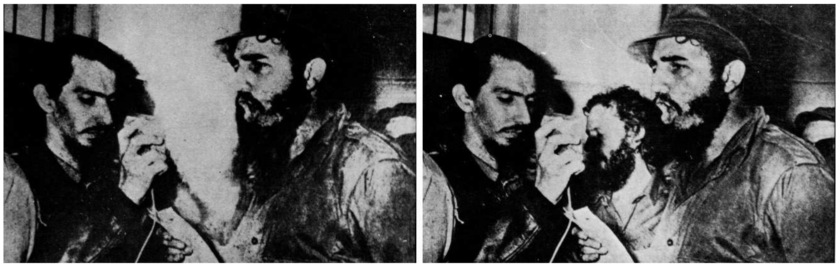
1950: It is believed that this doctored photograph contributed to Senator Millard Tydings' electoral defeat in 1950. The photo of Tydings, shown on the right, conversing with Earl Browder, a leader of the American Communist party, was meant to suggest that Tydings had communist sympathies.



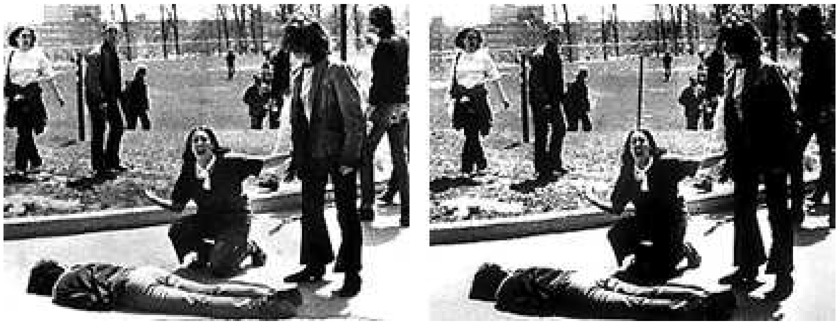
1960: In 1960 the U.S. Olympic hockey team defeated the Soviet Union and Czechoslovakia to win its first Olympic gold medal in hockey. The official team photo was doctored to include the faces of Bill Cleary (front row, third from the left), Bob Cleary (middle row, far left) and John Mayasich (top row, far left), who were not present for the team photo. These players were superimposed onto the bodies of players Bob Dupuis, Larry Alm and Herb Brooks, respectively.



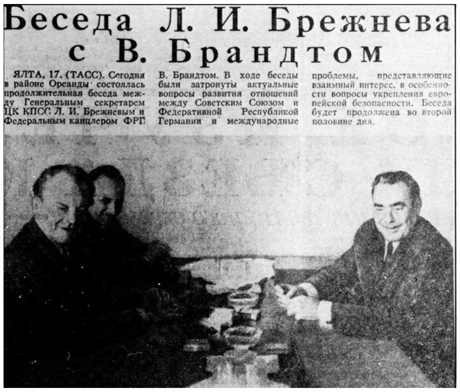
1961: On April 12, 1961 a Russian team of cosmonauts led by Yuri Gagarin were the first humans to complete an orbit of earth. One of the cosmonauts, Grigoriy Nelyubov, was removed from this photo of the team taken after their journey. Nelyubov had been expelled from the program for misbehavior.



1968: When in the summer of 1968 Fidel Castro (right) approves of the Soviet intervention in Czechoslovakia, Carlos Franqui (middle) cuts off relations with the regime and goes into exile in Italy. His image was removed from photographs. Franqui wrote about his feeling of being erased: "I discover my photographic death. Do I exist? I am a little black, I am a little white, I am a little shit, On Fidel's vest."

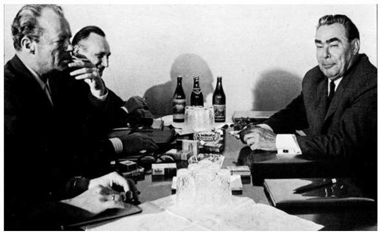


May 1970: This Pulitzer Prize winning photo by John Filo shows Mary Ann Vecchio screaming as she kneels over the body of student Jeffrey Miller at Kent State University, where National Guardsmen had fired into a crowd of demonstrators, killing four and wounding nine. When this photo was published in LIFE Magazine, the fence post directly behind Vecchio was removed.

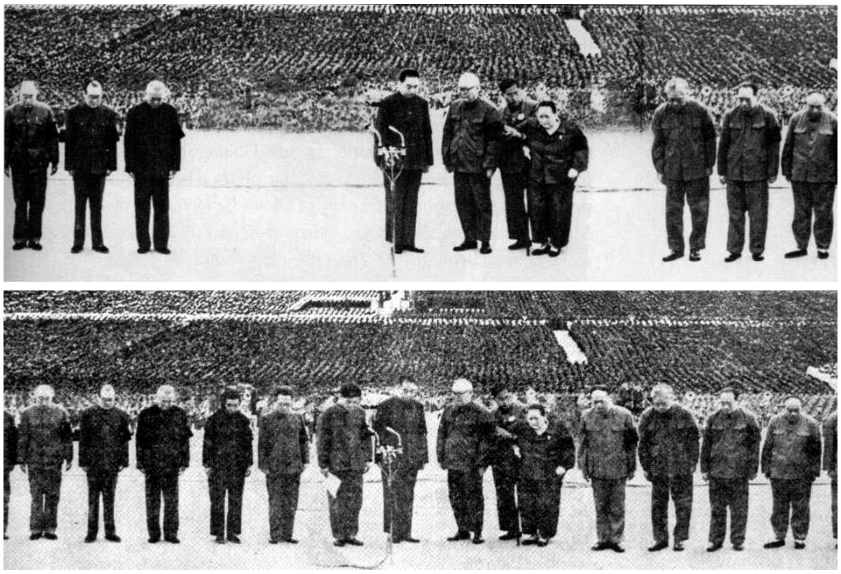


Беседа Л. И. Брежнева с В. Брандтом

ЯЛТА, 17. (ТАСС). Сегодня в районе Ореанды состоялась продолжительная беседа между Генеральным секретарем ЦК КПСС Л. И. Брежневым и Федеральным канцлером ФРГ В. Брандтом. В ходе беседы были затронуты актуальные вопросы развития отношений между Советским Союзом и Федеративной Республикой Германии и международные проблемы, представляющие взаимный интерес, в особенности вопросы укрепления европейской безопасности. Беседа будет продолжена во второй половине дня.



September 1971: The German Chancellor of West Germany, Willy Brandt (far left), meets with Leonid Brezhnev (far right), First Secretary of the Communist Party. The two smoke and drink, and it is reported that the atmosphere is cordial and that they are drunk. The German press publishes a photograph that shows the beer bottles on the table. The Soviet press, however, removed the bottles from the original photograph.



September 1976: The so called "Gang of Four" were removed from this original photograph of a memorial ceremony for Mao Tse-Tung held at Tiananmen Square.

1. D.A. Brugion. *Photo Fakery: The History and Techniques of Photographic Deception and Manipulation.* Brassey's Inc., 1999.

1. Format-Based Forensics

1.1 Fourier †

Consider a 1-D discretely sampled signal of length N:

$$f(x) \;=\; (\,1 \quad 2 \quad 4 \quad 5 \quad 3 \quad 0 \quad \dots \quad 7\,). \tag{1.1}$$

Although not made explicit, such a signal is represented with respect to a basis consisting of the canonical vectors in $\mathcal{R}^N$. That is, the signal is represented as a weighted sum of the basis vectors:

$$\begin{aligned} f(x) \;=&\; 1(\,1 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad \dots \quad 0\,) \\ +&\; 2(\,0 \quad 1 \quad 0 \quad 0 \quad 0 \quad 0 \quad \dots \quad 0\,) \\ +&\; 4(\,0 \quad 0 \quad 1 \quad 0 \quad 0 \quad 0 \quad \dots \quad 0\,) \\ +&\; \dots \\ +&\; 7(\,0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad \dots \quad 1\,). \end{aligned} \tag{1.2}$$

More compactly:

$$f(x) \;=\; \sum_{k=0}^{N-1} a_k b_k(x), \tag{1.3}$$

where $b_k(x)$ are the canonical basis vectors, and:

$$a_k \;=\; \sum_{l=0}^{N-1} f(l) b_k(l). \tag{1.4}$$

In the language of linear algebra, the weights a_k are simply an inner product between the signal $f(x)$ and the corresponding basis vector $b_k(x)$.

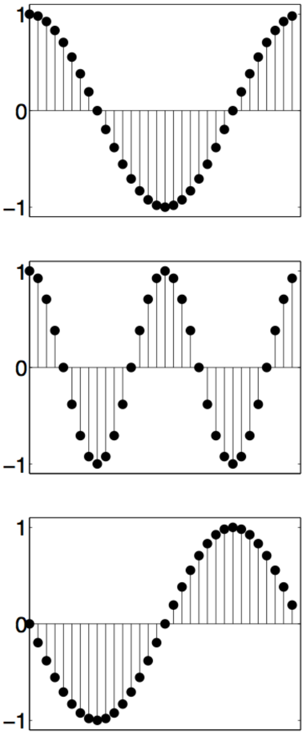


Figure 1.1 Sample 1-D Fourier basis signals.

A signal (or image) can, of course, be represented with respect to any of a number of different basis vectors. A particularly convenient and powerful choice is the Fourier basis. The Fourier basis consists of sinusoids of varying frequency and phase, Figure 1.1. Specifically, we seek to express a periodic signal as a weighted sum of the sinusoids:

$$f(x) \;=\; \frac{1}{N}\sum_{k=0}^{N-1} c_k \cos\left(\frac{2\pi k}{N}x + \phi_k\right), \tag{1.5}$$

where the frequency of the sinusoid is $\omega_k = 2\pi k/N$, the phase is ϕ_k, and the weighting (or amplitude) of the sinusoid is c_k. The sinusoids form a basis for the set of periodic signals. That is, any periodic signal can be written as a linear combination of the sinusoids. This expression is referred to as the Fourier series.

Note, however, that this basis is not fixed because the phase term, ϕ_k, is not fixed, but depends on the underlying signal $f(x)$. It is, however, possible to rewrite the Fourier series with respect to a fixed basis of zero-phase sinusoids. With the trigonometric identity:

$$\cos(A+B) \quad = \quad \cos(A)\cos(B) - \sin(A)\sin(B), \qquad (1.6)$$

the Fourier series of Equation (1.5) may be rewritten as:

$$\begin{aligned} f(x) &= \frac{1}{N}\sum_{k=0}^{N-1} c_k \cos(\omega_k x + \phi_k) \\ &= \frac{1}{N}\sum_{k=0}^{N-1} c_k \cos(\phi_k)\cos(\omega_k x) + c_k \sin(\phi_k)\sin(\omega_k x) \\ &= \frac{1}{N}\sum_{k=0}^{N-1} a_k \cos(\omega_k x) + b_k \sin(\omega_k x). \qquad (1.7) \end{aligned}$$

The basis of cosine and sine of varying frequency is now fixed. Notice the similarity to the basis representation in Equation(1.3): the signal is being represented as a weighted sum of a basis.

The Fourier series tells us that a signal can be represented in terms of the sinusoids. The Fourier transform tells us how to determine the relative weights a_k and b_k:

$$a_k = \sum_{l=0}^{N-1} f(l)\cos(\omega_k l) \quad \text{and} \quad b_k = \sum_{l=0}^{N-1} f(l)\sin(\omega_k l). \qquad (1.8)$$

As in Equation (1.4), these Fourier coefficients are determined from an inner product between the signal and corresponding basis.

A more compact notation is often used to represent the Fourier series and Fourier transform which exploits the complex exponential and its relationship to the sinusoids:

$$e^{i\omega x} \quad = \quad \cos(\omega x) + i\sin(\omega x), \qquad (1.9)$$

where i is the complex value $\sqrt{-1}$. With this complex exponential notation, the Fourier series and transform take the form:

$$f(x) = \frac{1}{N}\sum_{k=0}^{N-1} c_k e^{i\omega_k x} \quad \text{and} \quad c_k = \sum_{l=0}^{N-1} f(l) e^{-i\omega_k l}, \qquad (1.10)$$

where $c_k = a_k - ib_k$. This notation simply bundles the sine and cosine terms into a single expression.

Example 1.1 Show that if a signal $f(x)$ is zero-mean, then the Fourier coefficient $c_0 = 0$.

The Fourier coefficients c_k are complex valued. These complex valued coefficients can be analyzed in terms of their real and imaginary components, corresponding to the cosine and sine terms. This can be helpful when exploring the symmetry of the underlying signal $f(x)$, as the cosine terms are symmetric about the origin and the sine terms are asymmetric about the origin. These complex valued coefficients can also be analyzed in terms of their magnitude and phase. Considering the complex value as a vector in the real-complex space, the magnitude and phase are defined as:

$$|c_k| = \sqrt{a_k^2 + b_k^2} \qquad \text{and} \qquad \angle c_k = \tan^{-1}(b_k/a_k). \tag{1.11}$$

The magnitude describes the overall contribution of a frequency in constructing a signal, and the phase describes the relative position of each frequency.

Example 1.2 If $c = a + ib$ is a complex value, show that the following is true:

$$a = |c|\cos(\angle c) \qquad \text{and} \qquad b = |c|\sin(\angle c),$$

where $|c|$ is the magnitude and $\angle c$ is the phase.

In two dimensions, an $N \times N$ image can be expressed with respect to two-dimensional sinusoids:

$$f(x, y) \quad = \quad \frac{1}{N^2}\sum_{k=0}^{N-1}\sum_{l=0}^{N-1} c_{kl}\cos(\omega_k x + \omega_l y + \phi_{kl}), \tag{1.12}$$

with:

$$c_{kl} \quad = \quad \sum_{m=0}^{N-1}\sum_{n=0}^{N-1} f(m, n)\cos(\omega_k m + \omega_l n + \phi_{kl}), \tag{1.13}$$

Shown in Figure 1.2 are examples of the 2-D Fourier basis. From left to right are basis with increasing frequency, and from top to bottom are basis with varying orientation (i.e., relative contributions of horizontal ω_k and vertical ω_l frequencies).

As with the 1-D Fourier basis, the 2-D Fourier basis can be expressed with respect to a fixed basis as:

$$f(x,y) = \frac{1}{N^2} \sum_{k=0}^{N-1} \sum_{l=0}^{N-1} a_{kl} \cos(\omega_k x + \omega_l y) + b_{kl} \sin(\omega_k x + \omega_l y), \quad (1.14)$$

where,

$$a_{kl} \;=\; \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} f(m,n) \cos(\omega_k m + \omega_l n) \quad (1.15)$$

$$b_{kl} \;=\; \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} f(m,n) \sin(\omega_k m + \omega_l n). \quad (1.16)$$

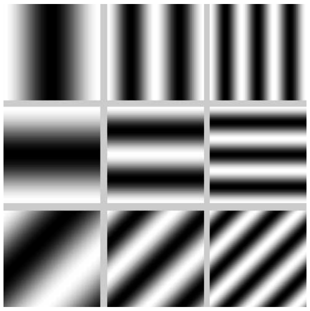


Figure 1.2 Sample 2-D Fourier basis images.

As with the 1-D Fourier basis and transform, the sine and cosine terms can be bundled using the complex exponential:

$$f(x,y) \;=\; \frac{1}{N^2} \sum_{k=0}^{N-1} \sum_{l=0}^{N-1} c_{kl} e^{i(\omega_k x + \omega_l y)} \quad (1.17)$$

$$c_{kl} \;=\; \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} f(m,n) e^{-i(\omega_k m + \omega_l n)}, \quad (1.18)$$

where $c_{kl} = a_{kl} - b_{kl}$. The Fourier transform c_{kl} is often denoted as $F(\omega_k, \omega_l)$.

Because the Fourier basis are periodic, the Fourier representation is particularly useful in discovering periodic patterns in a signal that might not otherwise be obvious when the signal is represented with respect to a canonical basis.

1.2 JPEG †

The JPEG file format has emerged as a near universal image standard employed by nearly all commercial digital cameras. Given a three channel color image (RGB), compression proceeds as follows. An image is first transformed from RGB into luminance/chrominance space (YCbCr). The two chrominance channels (CbCr) are typically subsampled by a factor of two relative to the luminance channel (Y). Each channel is then partitioned into 8 × 8 pixel blocks. These values are converted from unsigned to signed integers (e.g., from $[0, 255]$ to $[-128, 127]$). Each block, $f_c(\cdot)$, is converted to frequency space, $F_c(\cdot)$, using a two-dimensional discrete cosine transform (DCT):

$$F_c(\omega_k, \omega_l) = \sum_{m=0}^{7} \sum_{n=0}^{7} f_c(m, n) \cos(\omega_k m) \cos(\omega_l n), \qquad (1.19)$$

where $\omega_k = 2\pi k/8$, $\omega_l = 2\pi l/8$, $f_c(\cdot)$ is the underlying pixel values, and c denotes a specific image channel. Note that this representation is simply the Fourier series where only the symmetric cosine basis functions are employed.

Example 1.3 The Fourier transform assumes that the underlying signal or image is periodic. What additional assumption does the DCT make? Show how this assumption leads to a basis of only cosine terms in Equation (1.19).

Depending on the specific frequency ω_k, ω_l and channel c, each DCT coefficient, $F_c(\cdot)$, is quantized by an amount $q_c(\cdot)$:

$$\hat{F}_c(\omega_k, \omega_l) \;\; = \;\; \text{round}\left(\frac{F_c(\omega_k, \omega_l)}{q_c(\omega_k, \omega_l)} \right). \qquad (1.20)$$

This stage is the primary source of data reduction and information loss.

With some variations, the above sequence of steps is employed by all JPEG encoders. The primary source of variation in JPEG encoders is the choice of quantization values $q_c(\cdot)$, Equation (1.20). The quantization is specified as a set of three 8 × 8 tables associated with each frequency and image channel (YCbCr). For low compression rates, the values in these tables tend towards 1, and increase for higher compression rates. The quantization for the luminance channel is typically less than for the two chrominance channels, and the quantization for the lower frequency components is typically less than for the higher frequencies.

After quantization, the DCT coefficients are subjected to entropy encoding (typically Huffman coding). Huffman coding is a variable-length encoding scheme that encodes frequently occurring values with shorter codes, and less frequently occurring values with longer codes. This lossless compression scheme exploits the fact that the quantization of DCT coefficients yields many zero coefficients, which can in turn be efficiently encoded. Motivated by the fact that the statistics of the DC and AC DCT coefficients are different the JPEG standard allows for different Huffman codes for the DC and AC coefficients (the DC coefficient refers to $\omega_k = \omega_l = 0$, and the AC coefficients refer to all other frequencies). This entropy encoding is applied separately to each YCbCr channel, employing separate codes for each channel.

The JPEG standard does not enforce any specific quantization table or Huffman code. Camera and software engineers are therefore free to balance compression and quality to their own needs and tastes. The specific quantization tables and Huffman codes needed to decode a JPEG file are embedded into the JPEG header. The JPEG quantization table and Huffman codes along with other data extracted from the JPEG header have been found to form a distinct camera signature which can be used for authentication.

The first three components of the camera signature are the image dimensions, quantization table, and Huffman code. The image dimensions are used to distinguish between cameras with different sensor resolution. The set of three 8×8 quantization tables are specified as a one dimensional array of 192 values. The Huffman code is specified as six sets of 15 values corresponding to the number of codes of length $1, 2 \ldots 15$: each of three channels requires two codes, one for the DC coefficients and one for the AC coefficients. This representation eschews the actual code for a more compact representation that distinguishes codes based on the distribution of code lengths. In total, 284 values are extracted from the full resolution image: 2 image dimensions, 192 quantization values, and 90 Huffman codes.

A thumbnail version of the full resolution image is often embedded in the JPEG header. The next three components of the camera signature are extracted from this thumbnail image. A thumbnail is typically no larger in size than a few hundred square pixels, and is created by cropping, filtering and down-sampling the full-resolution image. The thumbnail is then typically compressed and stored in the header as a JPEG image. As such, the same components can be extracted from the thumbnail as from the full resolution image described in the previous section. Some camera manufacturers do not create a thumbnail image, or do not encode them as a JPEG image. In such cases, a value of zero can be assigned to all of the thumbnail parameters. Rather than being a limitation, the lack of a thumbnail is considered as a characteristic property of a camera. In total, 284 values are extracted from the thumbnail image: 2 thumbnail dimensions, 192 quantization values, and 90 Huffman codes.

The final component of the camera signature is extracted from an image's EXIF metadata. The metadata, found in the JPEG header, stores a variety of information about the camera and image. According to the EXIF standard, there are five main image file directories (IFDs) into which the metadata is organized: (1) Primary; (2) Exif; (3) Interoperability; (4) Thumbnail; and (5) GPS. Camera manufacturers are free to embed any (or no) information into each IFD. A compact representation of their choice can be extracted by counting the number of entries in each of these five IFDs. Because the EXIF standard allows for the creation of additional IFDs, the total number of any additional IFDs, and the total number of entries in each of these are also used as an additional feature. Some camera manufacturers customize their metadata in ways that do not conform to the EXIF standard, yielding errors when parsing the metadata. These errors are considered to be a feature of camera design and the total number of parser errors are used as an additional feature. In total, 8 values are extracted from the metadata: 5 entry counts from the standard IFDs, 1 for the number of additional IFDs, 1 for the number of entries in these additional IFDs, and 1 for the number of parser errors.

In summary, 284 header values are extracted from the full resolution image, a similar 284 header values from the thumbnail image, and another 8 from the EXIF metadata, for a total of 576 values. These 576 values form the signature by which images can be authenticated. To the extent that photo-editing software will employ JPEG parameters that are distinct from the camera's, any manipulation will alter the original signature, and can therefore be detected.

Specifically, photo alteration is detected by extracting the signature from an image and comparing it to a database of known authentic camera signatures. Any matching camera make and model can then be compared to the make and model specified in the image's EXIF metadata. Any mismatch is strong evidence of some form of tampering.

In [3] the camera make, model, and signature were extracted from 1.3 million images. These images span $9,163$ different distinct pairings of camera make, model, and signature and represent 33 different camera manufacturers and 773 different camera and cellphone models. A pairing of camera make, model, and signature is referred to as a camera configuration. To begin, all cameras with the same signature are placed into an equivalence class. An equivalence of size n means that n cameras of different make and model share the same signature. An equivalence class of size greater than 1 means that there is an ambiguity in identifying the camera make and model. We would, of course, like to maximize the number of camera configurations in an equivalence class of size 1 and minimize the largest equivalence class size.

It was found that 69.1% of the camera configurations are in an equivalence class of size one. 12.8% are in an equivalence class of size two, 5.7% are in an equivalence class of size three, 87.6% are in an equivalence class of size three or less, and the largest equivalence class is of size 14, with 0.2% of the camera configurations.

Because the distribution of cameras is not uniform, it is also useful to consider the likelihood of an image, as opposed to camera configuration, being in an equivalence class of size n. 62.4% of images are in an equivalence class of size one (i.e., are unique), 10.5% of images are in an equivalence class of size two, 7.5% of images are in an equivalence class of size three, and 80.4% of images are in an equivalence class of size three or less.

Shown in the table below are the percentage of camera configurations with an equivalence class size of $1 \dots 5$, and the median and maximum equivalence class size. Each row corresponds to different subsets of the complete signature. Individually, the image, thumbnail and EXIF parameters are not particularly distinct, but when combined, they provide a highly distinct signature. This suggests that the choice of parameters are not highly correlated, and hence their combination improves overall distinctiveness.

	Equivalence Class Size						
	1	2	3	4	5	median	max
image	12.9%	7.9%	6.2%	6.6%	3.4%	11	185
thumb	1.1%	1.1%	1.0%	1.1%	0.7%	694	960
EXIF	8.8%	5.4%	4.2%	3.2%	2.6%	25	188
image+thumb	24.9%	15.3%	11.3%	7.9%	3.7%	3	91
all	69.1%	12.8%	5.7%	4.0%	2.9%	1	14

The signature from Adobe Photoshop (versions 3, 4, 7, CS, CS2, CS3, CS4, CS5 at all qualities) were compared to the 9,163 camera signatures. In this case, only the image and thumbnail quantization tables and Huffman codes were used for comparison. No overlap was found between any Photoshop version/quality and camera manufacturer. As such, the Photoshop signatures, each residing in an equivalence class of size 1, are unique. This means that any photo-editing with Photoshop can be easily and unambiguously detected.

Recall that the encoding of a JPEG image involves three basic steps: DCT, quantization, and entropy encoding. The decoding of a compressed data stream involves the inverse of these steps, taken in reverse order: entropy decoding, de-quantization, and inverse DCT.

Consider the example of a generic discrete 1-D signal $f(x)$. Quantization is a point-wise operation[1] that is described by a one-parameter family of functions:

$$q_a(u) \quad = \quad \left\lfloor \frac{u}{a} \right\rfloor, \tag{1.21}$$

where a is the quantization step (a strictly positive integer), and u denotes a value in the range of $f(x)$. De-quantization brings the quantized values back to their original range: $q_a^{-1}(u) = au$. Note that the function $q_a(u)$ is not invertible, and that de-quantization is not the inverse function of quantization. Double quantization is a point-wise operation described by a two-parameter family of functions:

$$q_{ab}(u) \quad = \quad \left\lfloor \left\lfloor \frac{u}{b} \right\rfloor \frac{b}{a} \right\rfloor, \tag{1.22}$$

where a and b are the quantization steps (strictly positive integers). Notice that double quantization can be represented as a sequence of three steps: quantization with step b, followed by de-quantization with step b, followed by quantization with step a.

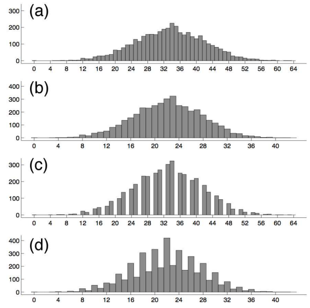


Figure 1.3 Distribution of single (a)-(b) and double (c)-(d) quantized coefficients.

Consider an example where the samples of $f(x)$ are normally distributed in the range $[0, 127]$. To illustrate the nature of the double quantization artifacts, we quantize the signal $f(x)$ in four different ways, and show the resulting histograms, Figure 1.3. Shown in panels (a) and (b) are the histograms of the same signal quantized with steps 2 and 3. Shown in panels (c) and (d) are the histograms of the same signal double quantized with steps 3 followed by 2, and 2 followed by 3. When the step size decreases some bins in the histogram are empty, Figure 1.3(c). This is not surprising since the first quantization places the samples of the original signal into 42 bins, while the second quantization re-distributes them into 64 bins. When the step size increases some bins contain more samples than their neighboring bins, Figure 1.3(d). This also is to

[1] For the purpose of illustration and in order to make the analysis easier we will use the floor function in the quantization function. Similar results can be shown if integer rounding is used instead.

be expected since the even bins receive samples from four original histogram bins, while the odd bins receive samples from only two. In both cases of double quantization, note the periodicity of the artifacts introduced into the histograms.

Example 1.4 Consider double quantization by a factor of 2 followed by 3. Using a simple bin counting argument, show that in the double quantized signal the even histogram bins receive samples from four original bins, while the odd bins receive samples from only two.

To better understand why the double quantization of a signal introduces periodic artifacts, we will analyze the dependence between the histograms of single and double quantized signals. Consider first the case of a single quantized signal denoted by $f_a(x) = q_a(f(x))$, and denote the histograms of the original and quantized signals by $H(u)$ and $H_a(v)$. Since $q_a(\cdot)$ is a many-to-one function, several values from the range of $f(x)$ will map onto the same value in the range of $f_a(x)$, i.e., several bins from H contribute to a bin in H_a. For example, let v denote a value in the range of $f_a(x)$, then the values in the range of $f(x)$ that map to it are in the range $[av, av + (a-1)]$. Therefore, the relationship between $H(u)$ and $H_a(v)$ is given by: $H_a(v) = \sum_{k=0}^{a-1} H(av+k)$. Note that there are exactly a bins in the original histogram that contribute to each bin in the histogram of the quantized signal.

Consider next the case of a double quantized signal denoted by $f_{ab}(x) = q_{ab}(f(x))$, and let its histogram be denoted by $H_{ab}(v)$. In contrast to the single quantization case, the number of bins of H that contribute to a bin of H_{ab} will depend on the double quantized bin value. Let v be a value in the range of $f_{ab}(x)$. Denote u_{min} and u_{max} as the smallest and largest values of u in the range of $f(x)$ that map to v, that is, they satisfy the following:

$$\left\lfloor \left\lfloor \frac{u}{b} \right\rfloor \frac{b}{a} \right\rfloor \quad = \quad v. \tag{1.23}$$

Using the following property of the floor function:

$$\lfloor z \rfloor = m \qquad \Rightarrow \qquad m \leq z < m+1, \tag{1.24}$$

where z is an arbitrary real number and m an integer, Equation (1.23) implies:

$$v \leq \left\lfloor \frac{u}{b} \right\rfloor \frac{b}{a} < v+1 \quad \Leftrightarrow \quad \frac{a}{b}v \leq \left\lfloor \frac{u}{b} \right\rfloor < \frac{a}{b}(v+1). \tag{1.25}$$

Since $\lfloor u/b \rfloor$ is an integer , Equation (1.25) can be rewritten using the ceiling function to include only integers:

$$\left\lceil \frac{a}{b} v \right\rceil \leq \left\lfloor \frac{u}{b} \right\rfloor \leq \left\lceil \frac{a}{b}(v+1) \right\rceil - 1. \tag{1.26}$$

From Equation (1.26) it can be seen that u_{min} must satisfy:

$$\left\lfloor \frac{u_{min}}{b} \right\rfloor = \left\lceil \frac{a}{b} v \right\rceil \quad \Rightarrow \quad u_{min} = \left\lceil \frac{a}{b} v \right\rceil b, \tag{1.27}$$

while u_{max} must satisfy:

$$\begin{aligned} \left\lfloor \frac{u_{max}}{b} \right\rfloor &= \left\lceil \frac{a}{b}(v+1) \right\rceil - 1 \quad \Rightarrow \\ u_{max} &= \left(\left\lceil \frac{a}{b}(v+1) \right\rceil - 1 \right) b + (b-1) \\ &= \left\lceil \frac{a}{b}(v+1) \right\rceil b - 1. \end{aligned} \tag{1.28}$$

Since double quantization is a monotonically increasing function, it follows that all the values between u_{min} and u_{max} will map to v through double quantization. The relationship between the original and double quantized histogram then takes the form:

$$H_{ab}(v) \quad = \sum_{u=u_{min}}^{u_{max}} H(u). \tag{1.29}$$

Note that the number of original histogram bins, $n(v)$, contributing to bin v in the double quantized histogram depends on v, and from Equations (1.27) and (1.28), can be expressed as:

$$n(v) = u_{max} - u_{min} + 1 = b \left(\left\lceil \frac{a}{b}(v+1) \right\rceil - \left\lceil \frac{a}{b} v \right\rceil \right). \tag{1.30}$$

Note that $n(v)$ is a periodic function with period b, i.e., $n(v) = n(v+b)$. This periodicity is the reason periodic artifacts appear in histograms of double quantized signals.

From Equation (1.30), the double quantization artifacts shown in Figure 1.3 can now be explained. Consider first the case of double quantization using steps $b = 3$ followed by $a = 2$, Figure 1.3(c). The number of original histogram bins contributing to double quantized histogram bins of the form $(3k+2)$ (k integer) is given by:

$$\begin{aligned} n(3k+2) &= 3 \left(\left\lceil \frac{2}{3}(3k+3) \right\rceil - \left\lceil \frac{2}{3}(3k+2) \right\rceil \right) \\ &= 3 \left(2k + 2 - 2k - \left\lceil \frac{4}{3} \right\rceil \right) \\ &= 0. \end{aligned} \tag{1.31}$$

This is consistent with the observation that every $(3k+2)^{nd}$ (k integer) bin of the double quantized histogram is empty. In the second example of double quantization in Figure 1.3, $b = 2$ and $a = 3$, it can be shown that $n(2k) = 4$ and $n(2k+1) = 2$ (k integer). Again, this is consistent with the periodic artifacts shown in Figure 1.3(d).

The periodicity in these histograms is usually clearly visible. The periodicity can also be revealed by computing the magnitude of the Fourier transform of the histogram, in which any periodicity will manifest itself as spikes in the Fourier domain.

1.5 JPEG Ghost

The previous section described a technique to determine if a JPEG image underwent double compression. This section describes a similar technique, but one that is able to localize which parts of an image underwent double compression.

Consider a set of DCT JPEG coefficients c_1 quantized by an amount q_1 which are subsequently quantized a second time by an amount q_2 to yield coefficients c_2. With the exception of $q_2 = 1$ (i.e., no second quantization), the difference between c_1 and c_2 will be minimal when $q_2 = q_1$ and will increase as the difference between q_2 and q_1 increases. Specifically, if $q_2 > q_1$ then the coefficients c_2 become increasingly more sparse relative to c_1, and if $q_2 < q_1$ then, even though the second quantization is less than the first, the coefficients c_2 shift relative to c_1 (Section 1.4).

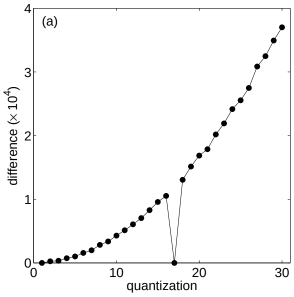


Figure 1.4 Coefficient difference at $q_1 = 17$ and $q_2 \in [1, 30]$.

Shown in Figure 1.4, for example, is the sum of squared differences between c_1 and c_2 as a function of the second quantization q_2, where $q_1 = 17$. Note that this difference increases as a function of increasing q_2, with the exception of $q_2 = q_1$, where the difference is minimal. If q_1 is not prime, as in this example, then multiple minima may appear at quality values q_2 that are integer multiples of q_1.

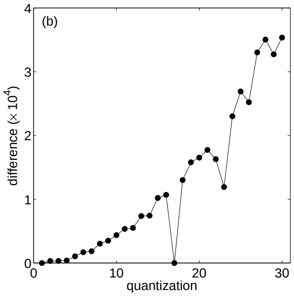


Figure 1.5 A JPEG ghost at $q_2 = 23$.

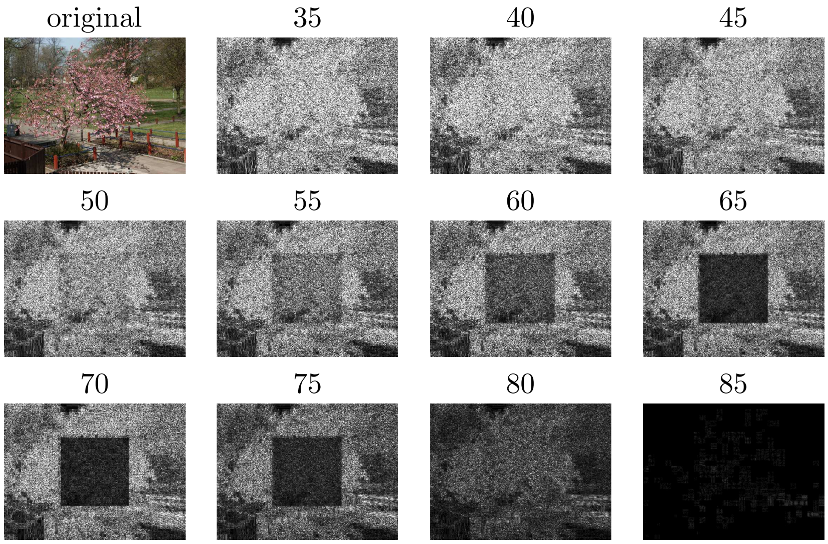
Consider now a set of coefficients c_0 quantized by an amount q_0 followed by quantization by an amount $q_1 < q_0$ to yield c_1. That is, the second quantization is at a higher quality than the first. Further quantizing c_1 by q_2 yields the coefficients c_2. As before, the difference between c_1 and c_2 will be minimal when $q_2 = q_1$. But, since the coefficients were initially quantized by q_0, we expect to find a second minimum when $q_2 = q_0$. Shown in Figure 1.5 is the sum of squared differences between c_1 and c_2, as a function of q_2, where $q_0 = 23$ and $q_1 = 17$. As before, this difference increases as a function of increasing q_2, reaches a minimum at $q_2 = q_1 = 17$, and most interestingly has a second local minimum at $q_2 = q_0 = 23$. This second minimum is a JPEG ghost, as it reveals that the coefficients were previously quantized (compressed) with a larger quantization (lower quality).

Recall that the JPEG compression scheme separately quantizes each spatial frequency within a 8 × 8 pixel block. One approach to detecting JPEG ghosts would be to separately consider each spatial frequency in each of the three luminance/color channels. However, recall that multiple minima are possible when comparing integer multiple quantization values. If, on the other hand, we consider the cumulative effect of quantization on the underlying pixel values, then this issue is far less likely to arise (unless all 192 quantization values at different JPEG qualities are integer multiples of one another – an unlikely scenario. Therefore, instead of computing the difference between the quantized DCT coefficients, we consider the difference computed directly from the pixel values, as follows:

$$d(x, y, q) \quad = \quad \frac{1}{3} \sum_{i=1}^{3} [f(x, y, i) - f_q(x, y, i)]^2, \qquad (1.32)$$

where $f(x, y, i)$, $i = 1, 2, 3$, denotes each of three RGB color channels, and $f_q(\cdot)$ is the result of compressing $f(\cdot)$ at quality q.

Shown in the top left panel of the figure below is an image whose central 200×200 pixel region was extracted, compressed at a JPEG quality of 65/100, and re-inserted into the image whose original quality was 85. Shown in each subsequent panel is the sum of squared differences, Equation (1.32), between this manipulated image, and a re-saved version compressed at different JPEG qualities. Note that the central region is clearly visible when the image is re-saved at the quality of the tampered region (65). Also note that the overall error reaches a minimum at the saved quality of 85.



A central region was extracted from the original, saved at JPEG quality 65*, and re-inserted into the image whose original quality was* 85*. Shown is the difference between this image and a re-compressed version at different qualities: a JPEG ghost is revealed at the original quality of* 65 *(and neighboring qualities).*

There are some variations in the difference images within and outside of the tampered region which could possibly confound a forensic analysis. These fluctuations are due to the underlying image content. Specifically, because the image difference is computed across all spatial frequencies, a region with small amounts of high spatial frequency content (e.g., a mostly uniform sky) will have a lower difference as compared to a highly textured region (e.g., grass). In order to compensate for these differences, a spatially averaged and normalized difference measure can be considered. The difference image is first averaged across a $b \times b$ pixel region:

$$\delta(x, y, q) = \frac{1}{3} \sum_{i=1}^{3} \frac{1}{b^2} \sum_{b_x=0}^{b-1} \sum_{b_y=0}^{b-1} [f(x + b_x, y + b_y, i) - f_q(x + b_x, y + b_y, i)]^2 \quad (1.33)$$

and then normalized so that the averaged difference at each location (x, y) is scaled into the range $[0, 1]$:

$$d(x, y, q) = \frac{\delta(x, y, q) - \min_q[\delta(x, y, q)]}{\max_q[\delta(x, y, q)] - \min_q[\delta(x, y, q)]}. \quad (1.34)$$

There are two potential complicating factors that arise when detecting JPEG ghosts in a general forensic setting. First, it is likely that different cameras and photo-editing software packages will employ different JPEG quality scales and hence quantization tables. When iterating through different qualities it would be ideal to match these qualities and tables, but this may not always be possible. Working to our advantage, however, is that the difference images are computed by averaging across all spatial frequencies. As a result small differences in the original and subsequent quantization tables will likely not have a significant impact. The second practical issue is that in the above examples it was assumed that the tampered region remains on its original 8×8 JPEG lattice after being inserted and saved. If this is not the case, then the mis-alignment may destroy the JPEG ghost since new spatial frequencies will be introduced by saving onto a new JPEG block lattice. This problem can be alleviated by sampling all 64 possible alignments (a 0 to 7 pixel shift in the horizontal and vertical directions). Specifically, an image is shifted to each of these 64 locations prior to saving at each JPEG quality.

1. Write MatLab code that simulates and reveals the effects of double JPEG compression. Specifically, your code should:

    - Generate $1,000$ random floating-point coefficients having a normal distribution.
    - Scale these coefficients into the range $[0, 255]$.
    - Quantize these coefficients by a reasonable amount.
    - Display a histogram of the quantized coefficients and the magnitude of the Fourier transform of this histogram.
    - Quantize the coefficients again with larger and smaller quantization steps.
    - Display a histogram of the double quantized coefficients and the magnitude of the Fourier transform of this histogram.
    - Explore and report on the range of quantization values that reveal the double quantization artifacts.

Example 1.1 The Fourier transform is given by:

$$c_k = \sum_{l=0}^{N-1} f(l)e^{-i\omega_k l}.$$

Recall that $\omega_k = 2\pi k/N$. The term c_0 is specified by replacing k with 0 to yield:

$$\begin{aligned} c_0 &= \sum_{l=0}^{N-1} f(l)e^{-i\frac{2\pi 0}{N}l} \\ &= \sum_{l=0}^{N-1} f(l)e^{0} \\ &= \sum_{l=0}^{N-1} f(l). \end{aligned}$$

That is, c_0 is simply the sum of the signal. Since it was assumed that the mean of the signal is zero, c_0 must be 0. And more, generally, the term c_0 (the DC term) is always proportional to the mean of the signal.

Example 1.2 Denote the complex value c as $c = a + ib$. In the complex plane shown in Figure 1.6, c is a 2-D vector with horizontal (real) component a and vertical (complex) component b. The magnitude of the vector is $\sqrt{a^2+b^2}$ and the angle of the vector relative to the horizontal axis (the phase) is $\tan^{-1}(b/a)$.

From basic geometry, we see that: $\cos(\phi) = a/\sqrt{a^2+b^2}$ or $a = \sqrt{a^2+b^2}\cos(\phi)$, where the magnitude $|c|$ is $\sqrt{a^2+b^2}$ and where the angle ϕ is simply the phase $\angle c$, yielding $a = |c|\cos(\angle c)$. A similar construction yields $b = \sqrt{a^2+b^2}\sin(\phi) = |c|\sin(\angle c)$.

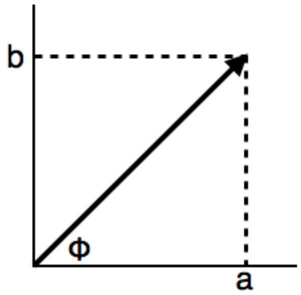


Figure 1.6 Complex plane.

Example 1.3 The DCT assumes that the underlying image is symmetric (or even). The 2-D Fourier transform is given by:

$$f(x,y) = \frac{1}{N^2}\sum_{k=0}^{N-1}\sum_{l=0}^{N-1} a_{kl}\cos(\omega_k x + \omega_l y) + b_{kl}\sin(\omega_k x + \omega_l y).$$

If we assume that the underlying image is symmetric, then the asymmetric (odd) sinusoidal component of the Fourier basis is not needed, and the 2-D Fourier transform can be written as:

$$f(x,y) = \frac{1}{N^2}\sum_{k=0}^{N-1}\sum_{l=0}^{N-1} a_{kl}\cos(\omega_k x + \omega_l y),$$

which, using the identity $\cos(A+B) = \cos(A)\cos(B) - \sin(A)\sin(B)$, can be rewritten as:

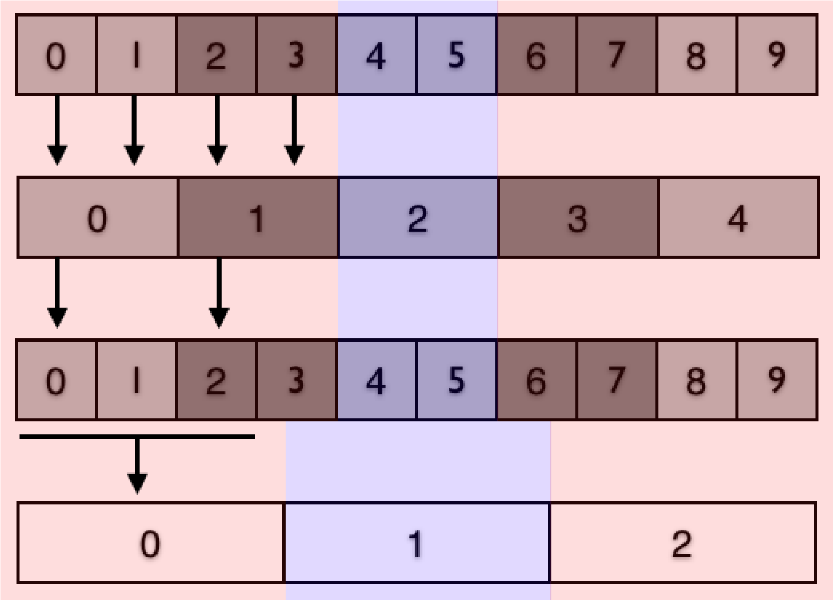
$$f(x,y) = \frac{1}{N^2} \sum_{k=0}^{N-1} \sum_{l=0}^{N-1} a_{kl} (\cos(\omega_k x)\cos(\omega_l y) - \sin(\omega_k x)\sin(\omega_l y)).$$

Once again, the sinusoidal components are not required yielding:

$$f(x,y) = \frac{1}{N^2} \sum_{k=0}^{N-1} \sum_{l=0}^{N-1} a_{kl} \cos(\omega_k x)\cos(\omega_l y),$$

which is the DCT.

Example 1.4 Consider ten bins of a histogram (top row in the figure below) that are quantized by a factor of 2 into four bins (second row). Requantization requires expanding these bins to their original nine (third row) followed by quantization by a factor of 3 into three bins (fourth row). In the final double quantized histogram bins 0 and 2 contain information from four original bins, while histogram bin 1 contains information from only (the shading and color coding denote the path of a bin's contents through quantization and double quantization).



1. H. Farid. Exposing digital forgeries from JPEG ghosts. IEEE Transactions on Information Forensics and Security, 1(4):154 160, 2009.

2. E. Kee and H. Farid. Digital image authentication from thumbnails. In *SPIE Symposium on Electronic Imaging*, San Jose, CA, 2010.

3. E. Kee, M.K. Johnson, and H. Farid. Digital image authentication from JPEG headers. *IEEE Transactions on Information Forensics and Security*, 6(3):1066-1075, 2011.

4. B. Mahdian and S. Saic. Detecting double compressed JPEG images. In *International Conference on Imaging for Crime Detection and Prevention*, 2009.

5. A.C. Popescu and H. Farid. Statistical tools for digital forensics. In *6th International Workshop on Information Hiding*, Toronto, Canada, 2004.

2. Camera-Based Forensics

2.1 Least-Squares (LS) †

You are given a collection of data points (x_i, y_i), for $i = 1 \dots n$, and asked to fit a line to these data. The model of a line:

$$y \;=\; mx + b, \tag{2.1}$$

is parametrized by two parameters: the slope m and y-intercept b. Each data point (x_i, y_i) provides one constraint on these two parameters. A total of n such constraints provides an over-constrained system that can be expressed in matrix form as:

$$\begin{pmatrix} x_1 & 1 \\ x_2 & 1 \\ \vdots & \vdots \\ x_n & 1 \end{pmatrix} \begin{pmatrix} m \\ b \end{pmatrix} \;=\; \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} \tag{2.2}$$

$$X\mathbf{u} \;=\; \mathbf{y}. \tag{2.3}$$

This linear system of equations will have a solution only when the vector $\mathbf{y}$ is contained in the column space of matrix X: that is, when $\mathbf{y}$ can be expressed as a linear combination of the columns of the matrix X. From a geometric perspective, this will occur when all points (x_i, y_i) lie precisely on a line. If, however, the points deviate even slightly from a perfect line, then $\mathbf{y}$ will not be in the column space of X, and there will be no solution to the above linear system. It is in these situations that we seek a solution that minimizes some measure of goodness of fit of a line to the points.

Shown in Figure 2.1 are six data points that lie slightly off of a line. The fit line minimizes the overall vertical displacement of the points from the line. Minimizing this vertical distance, $mx+b-y$, lends itself to a particularly straight-forward optimization, termed least-squares. In matrix form, the vertical distance between points and a line with slope m and y-intercept b is given by:

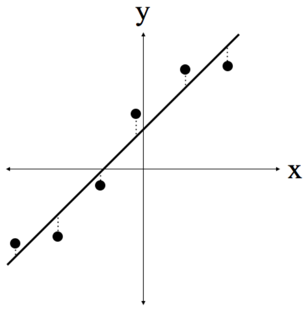


Figure 2.1 Least-squares minimizes the vertical displacement of points from the line.

$$\begin{pmatrix} x_1 & 1 \\ x_2 & 1 \\ \vdots & \vdots \\ x_n & 1 \end{pmatrix} \begin{pmatrix} m \\ b \end{pmatrix} \;-\; \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} \tag{2.4}$$

$$X\mathbf{u} - \mathbf{y}. \tag{2.5}$$

We seek to minimize the sum of these squared distances:

$$\sum_{i=1}^{n}(mx_i + b - y_i)^2 \quad = \quad (X\mathbf{u} - \mathbf{y})^T(X\mathbf{u} - \mathbf{y}). \tag{2.6}$$

The minimization of this error is expressed as a quadratic error function in the parameters of the line:

$$E(\mathbf{u}) \quad = \quad \|X\mathbf{u} - \mathbf{y}\|^2, \tag{2.7}$$

where $\|\cdot\|$ denotes vector norm. This quadratic error is minimized by differentiating:

$$\frac{dE}{d\mathbf{u}} \quad = \quad 2X^T(X\mathbf{u} - \mathbf{y}), \tag{2.8}$$

setting equal to zero and solving for $\mathbf{u}$:

$$\begin{aligned} 2X^T(X\mathbf{u} - \mathbf{y}) &= 0 && (2.9) \\ X^TX\mathbf{u} &= X^T\mathbf{y} && (2.10) \\ \mathbf{u} &= (X^TX)^{-1}X^T\mathbf{y}, && (2.11) \end{aligned}$$

yielding the least-squares estimation of the line parameters. The matrix X^TX will be singular and hence not invertible if the rows of the matrix X are linearly dependent. Geometrically, this corresponds to one of two situations: (1) the n points are identical in which case there is no unique solution to fitting a line to a single point; or (2) the n points lie on a vertical line in which case $m = \infty$.

This basic framework is, of course, applicable to estimating any model that is linear in their unknown parameters. For example, a parabola $y = ax^2+bx+c$ can be fit to n points by first constructing the following linear system:

$$\begin{aligned} \begin{pmatrix} x_1^2 & x_1 & 1 \\ x_2^2 & x_2 & 1 \\ \vdots & \vdots & \vdots \\ x_n^2 & x_n & 1 \end{pmatrix} \begin{pmatrix} a \\ b \\ c \end{pmatrix} &= \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} && (2.12) \\ X\mathbf{u} &= \mathbf{y}, && (2.13) \end{aligned}$$

and then solving using the least-squares solution in Equation (2.11). Similarly, the matrix formulation of a plane $z = ax + by + c$ is:

$$\begin{aligned} \begin{pmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ \vdots & \vdots & \vdots \\ x_n & y_n & 1 \end{pmatrix} \begin{pmatrix} a \\ b \\ c \end{pmatrix} &= \begin{pmatrix} z_1 \\ z_2 \\ \vdots \\ z_n \end{pmatrix} && (2.14) \\ X\mathbf{u} &= \mathbf{z}, && (2.15) \end{aligned}$$

and the least-squares solution is $\mathbf{u} = (X^TX)^{-1}X^T\mathbf{z}$.

Example 2.1 Where possible, write out the least-squares system for the following (where a, b, and c are the model parameters):

1. $y = a\log(x) + b$
2. $z = ax^y + b$
3. $y = x^a$

Weighted least-squares allows for a non-uniform treatment of the contribution of each individual point to the overall error. The error function of Equation (2.7) takes the form:

$$E(\mathbf{u}) \quad = \quad W\|X\mathbf{u} - \mathbf{y}\|^2, \tag{2.16}$$

where W is a diagonal $n \times n$ weighting matrix with diagonal elements w_i corresponding to the weight associated with the i^{th} point. A larger weight places more emphasis on minimizing that point's deviation from the model. This error function is minimized by differentiating:

$$\frac{dE}{d\mathbf{u}} \quad = \quad 2X^T W(X\mathbf{u} - \mathbf{y}), \tag{2.17}$$

setting equal to zero and solving for $\mathbf{u}$ to yield the weighted least-squares solution:

$$\begin{aligned} 2X^T W(X\mathbf{u} - \mathbf{y}) &= 0 && (2.18)\\ X^T W X\mathbf{u} &= X^T W\mathbf{y} && (2.19)\\ \mathbf{u} &= (X^T W X)^{-1} X^T W\mathbf{y}. && (2.20) \end{aligned}$$

Notice that if W is the identity matrix, then this solution reverts back to the least-squares solution of Equation (2.7).

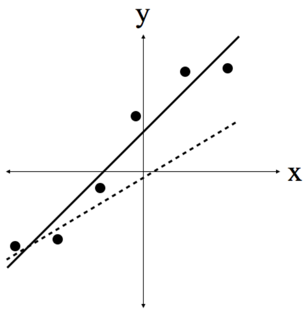


Figure 2.2
Least-squares (solid) and weighted least-squares (dashed).

Shown in Figure 2.2 are six data points fit with a line using least-squares (solid line) and weighted least squares (dashed line) where the two bottom left most points were weighted disproportionately relative to the other points. Notice how the line minimizes the error for these points at the price of significantly higher error for the remaining points.

2.2 Expectation Maximization (EM) †

The Expectation/Maximization (EM) algorithm simultaneously segments and fits data generated from multiple parametric models. For example, shown in Figure 2.3 are a collection of data points (x_i, y_i) generated from one of two linear models of the form:

$$\begin{aligned} y(i) &= a_1 x(i) + b_1 + n_1(i) && (2.21) \\ y(i) &= a_2 x(i) + b_2 + n_2(i), && (2.22) \end{aligned}$$

where the model parameters are a_1, b_1 and a_2, b_2, and the system is modeled with additive noise $n_1(i)$ and $n_2(i)$.

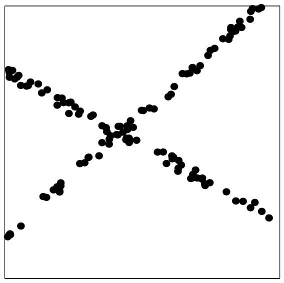


Figure 2.3 Data from two models

If we are told the model parameters, then determining which data point was generated by which model would be a simple matter of choosing, for each data point i, the model k that minimizes the error between the data and the model prediction:

$$r_k(i) = |a_k x(i) + b_k - y(i))|, \qquad (2.23)$$

for $k = 1, 2$ in our example. On the other hand, if we are told which data points were generated by which model, then estimating the model parameters reduces to solving, for each model k, an over-constrained set of linear equations:

$$\begin{pmatrix} x_k(1) & 1 \\ x_k(2) & 1 \\ \vdots & \vdots \\ x_k(n) & 1 \end{pmatrix} \begin{pmatrix} a_k \\ b_k \end{pmatrix} = \begin{pmatrix} y_k(1) \\ y_k(2) \\ \vdots \\ y_k(n) \end{pmatrix}, \qquad (2.24)$$

where the $x_k(i)$ and $y_k(i)$ all belong to model k. In either case, knowing one piece of information (the model assignment or parameters) makes determining the other relatively easy. But, lacking either piece of information makes this a considerably more difficult estimation problem. The EM algorithm is an iterative two step algorithm that estimates both the model assignment and parameters.

The "E-step" of EM assumes that the model parameters are known (initially, the model parameters can be assigned random values) and calculates the likelihood of each data point belonging to each model. In so doing the model assignment is made in a "soft" probabilistic fashion. That is, each data point is not explicitly assigned a single model, instead each data point i is assigned a probability of it belonging to each model k. For each model the residual error is first computed as:

$$r_k(i) = a_k x(i) + b_k - y(i) \qquad (2.25)$$

from which the likelihoods are calculated. We ask, what is the likelihood of point i belonging to model k given the residual error. For our two model example:

$$\begin{aligned} P(a_k, b_k | r_k(i)) &= \frac{P(r_k(i)|a_k, b_k)P(a_k, b_k)}{P(r_k(i))} \\ &= \frac{P(r_k(i)|a_k, b_k)}{P(r_1(i)|a_k, b_k) + P(r_2(i)|a_k, b_k)}, \end{aligned} \qquad (2.26)$$

for $k = 1, 2$.

The expansion of the conditional probability is from Bayes rule: $P(B|A_k) = \frac{P(A_k|B)P(B)}{\sum_l P(A_l|B)P(B)}$. If we assume a Gaussian probability distribution, then the likelihood takes the form:

$$w_k(i) = P(a_k, b_k | r_k(i)) = \frac{e^{-r_k^2(i)/\sigma}}{e^{-r_1^2(i)/\sigma} + e^{-r_2^2(i)/\sigma}}, \qquad (2.27)$$

where, σ is proportional to the amount of noise in the data.

The "M-step" of EM takes the likelihood of each data point belonging to each model, and re-estimates the model parameters using weighted least-squares. That is, the following weighted error function on the model parameters is minimized:

$$E_k(a_k, b_k) = \sum_{i=1}^{n} w_k(i)[a_k x(i) + b_k - y(i)]^2. \qquad (2.28)$$

The intuition here is that each data point contributes to the estimation of each model's parameters in proportion to the belief that it belongs to that particular model. This quadratic error function is minimized by computing the partial derivatives with respect to the model parameters, setting the result equal to zero and solving for the model parameters. Differentiating:

$$\begin{aligned} \frac{\partial E_k(a_k, b_k)}{\partial a_k} &= \sum_{i=1}^{n} 2 w_k(i) x(i)[a_k x(i) + b_k - y(i)] \\ \frac{\partial E_k(a_k, b_k)}{\partial b_k} &= \sum_{i=1}^{n} 2 w_k(i)[a_k x(i) + b_k - y(i)], \end{aligned} \qquad (2.29)$$

setting both equal to zero yields the following set of linear equations:

$$\begin{aligned} a_k \sum_{i=1}^{n} w_k(i) x(i)^2 + b_k \sum_{i=1}^{n} w_k(i) x(i) &= \sum_{i=1}^{n} w_k(i) x(i) y(i) \qquad (2.30) \\ a_k \sum_{i=1}^{n} w_k(i) x(i) + b_k \sum_{i=1}^{n} w_k(i) &= \sum_{i=1}^{n} w_k(i) y(i). \qquad (2.31) \end{aligned}$$

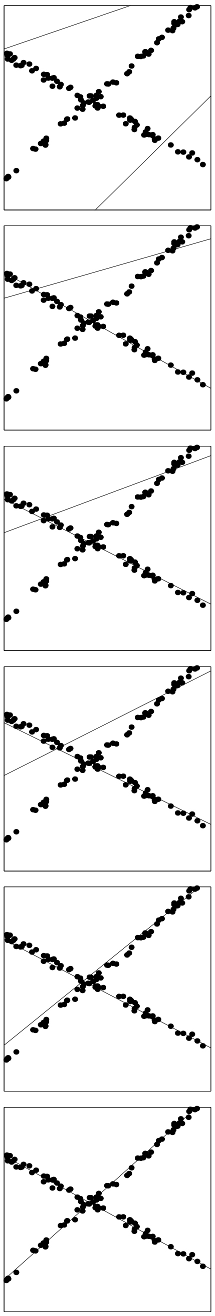


Figure 2.4 Six iterations of EM.

Rewriting in matrix form:

$$\begin{aligned}
\begin{pmatrix} \sum_{i=1}^{n} w_k(i)x(i)^2 & \sum_{i=1}^{n} w_k(i)x(i) \\ \sum_{i=1}^{n} w_k(i)x(i) & \sum_{i=1}^{n} w_k(i) \end{pmatrix} \begin{pmatrix} a_k \\ b_k \end{pmatrix} &= \begin{pmatrix} \sum_{i=1}^{n} w_k(i)x(i)y(i) \\ \sum_{i=1}^{n} w(i)y(i) \end{pmatrix} \\
A\mathbf{x_k} &= \mathbf{b} \\
\mathbf{x_k} &= A^{-1}\mathbf{b}, \qquad (2.32)
\end{aligned}$$

yields a weighted least squares solution for the model parameters.

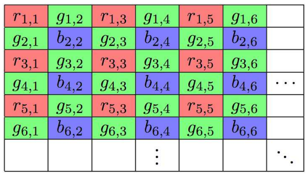
Example 2.2 Show that the solution in Equation (2.32) is identical to solving the set of linear equations in Equation (2.24) using weighted least-squares.

The EM algorithm iteratively executes the "E" and "M" step, repeatedly estimating and refining the model assignments and parameters. Shown in Figure 2.4 are several iterations of EM applied to fitting data generated from two linear models. Initially, the model parameters are randomly assigned, and after six iterations, the algorithm converges to a solution.

The EM algorithm can be sensitive to the value of σ used. It is often recommended that with a reasonable starting value, the value of σ can be updated on each EM iteration as follows:

$$\sigma = \frac{\sum_{i=1}^{n} w(i)r^2(i)}{\sum_{i=1}^{n} w(i)}. \qquad (2.33)$$

Most digital cameras capture color images using a single sensor in conjunction with an array of color filters. As a result, only one third of the samples in a color image are captured by the camera, the other two thirds are interpolated. This interpolation introduces specific correlations between the samples of a color image. When creating a digital forgery these correlations may be destroyed or altered. We describe the form of these correlations, and propose a method that quantifies and detects them in any portion of an image.



$r_{1,1}$	$g_{1,2}$	$r_{1,3}$	$g_{1,4}$	$r_{1,5}$	$g_{1,6}$	
$g_{2,1}$	$b_{2,2}$	$g_{2,3}$	$b_{2,4}$	$g_{2,5}$	$b_{2,6}$	
$r_{3,1}$	$g_{3,2}$	$r_{3,3}$	$g_{3,4}$	$r_{3,5}$	$g_{3,6}$	
$g_{4,1}$	$b_{4,2}$	$g_{4,3}$	$b_{4,4}$	$g_{4,5}$	$b_{4,6}$	$\cdots$
$r_{5,1}$	$g_{5,2}$	$r_{5,3}$	$g_{5,4}$	$r_{5,5}$	$g_{5,6}$	
$g_{6,1}$	$b_{6,2}$	$g_{6,3}$	$b_{6,4}$	$g_{6,5}$	$b_{6,6}$	
			$\vdots$			$\ddots$

Figure 2.5 Bayer array

A digital color image consists of three channels containing samples from different bands of the color spectrum, e.g., red, green, and blue. Most digital cameras, however, are equipped with a single CCD or CMOS sensor, and capture color images using a color filter array (CFA). The most frequently used CFA, the Bayer array, employs three color filters: red, green, and blue. The red and blue pixels are sampled on rectilinear lattices, while the green pixels are sampled on a quincunx lattice, Figure 2.5. Since only a single color sample is recorded at each pixel location, the other two color samples must be estimated from the neighboring samples in order to obtain a three-channel color image. Let $S(x, y)$ denote the CFA image in Figure 2.5, and $\tilde{R}(x, y)$, $\tilde{G}(x, y)$, $\tilde{B}(x, y)$ denote the red, green, and blue channels constructed from $S(x, y)$ as follows:

$$\begin{aligned} \tilde{R}(x, y) &= S(x, y) && \text{if } S(x, y) = r_{x,y} && (2.34) \\ &= 0 && \text{otherwise} \\ \tilde{G}(x, y) &= S(x, y) && \text{if } S(x, y) = g_{x,y} && (2.35) \\ &= 0 && \text{otherwise} \\ \tilde{B}(x, y) &= S(x, y) && \text{if } B(x, y) = b_{x,y} && (2.36) \\ &= 0 && \text{otherwise} \end{aligned}$$

where (x, y) span an integer lattice. A complete color image, with channels $R(x, y)$, $G(x, y)$, and $B(x, y)$ needs to be estimated. These channels take on the non-zero values of $\tilde{R}(x, y)$, $\tilde{G}(x, y)$, and $\tilde{B}(x, y)$, and replace the zeros with estimates from neighboring samples.

The estimation of the missing color samples is referred to as CFA interpolation or demosaicking. CFA interpolation has been extensively studied and many methods have been proposed. The simplest methods for demosaicking are kernel-based interpolation methods that act on each channel independently. These methods can be efficiently implemented as linear filtering operations on each color channel:

$$\begin{aligned} R(x,y) &= \sum_{u,v=-N}^{N} h_r(u,v)\tilde{R}(x-u,y-v) && (2.37) \\ G(x,y) &= \sum_{u,v=-N}^{N} h_g(u,v)\tilde{G}(x-u,y-v) && (2.38) \\ B(x,y) &= \sum_{u,v=-N}^{N} h_b(u,v)\tilde{B}(x-u,y-v), && (2.39) \end{aligned}$$

where $\tilde{R}(\cdot)$, $\tilde{G}(\cdot)$, $\tilde{B}(\cdot)$ are defined in Equations (2.34)-(2.36), and $h_r(\cdot)$, $h_g(\cdot)$, $h_b(\cdot)$ are linear filters of size $(2N+1) \times (2N+1)$. Different forms of interpolation (nearest neighbor, bilinear, bicubic, etc.) differ in the form of the interpolation filter used. For the Bayer array, the bilinear filter for the red and blue channels are separable. The 1-D filter is:

$$h_l = \begin{pmatrix} 1/2 & 1 & 1/2 \end{pmatrix}. \qquad (2.40)$$

The 2-D filter is the outer product between h_l and itself.

There are many other CFA interpolation algorithms including smooth hue transition, median filter, gradient-based, adaptive color plane, and threshold-based variable number of gradients. Regardless of their specific implementations, each CFA interpolation algorithm introduces specific statistical correlations between a subset of pixels in each color channel.

Since the color filters in a CFA are typically arranged in a periodic pattern, these correlations are periodic. Consider, for example, the red channel, $R(x,y)$, that has been sampled on a Bayer array, Figure 2.5, then CFA interpolated using bilinear interpolation. In this case, the red samples in the odd rows and even columns are the average of their closest horizontal neighbors, the red samples in the even rows and odd columns are the average of their closest vertical neighbors, and the red samples in the even rows and columns are the average of their closest diagonal neighbors:

$$\begin{aligned} R(2x+1,2y) &= \frac{R(2x+1,2y-1)}{2} + \frac{R(2x+1,2y+1)}{2} \\ R(2x,2y+1) &= \frac{R(2x-1,2y+1)}{2} + \frac{R(2x+1,2y+1)}{2} \\ R(2x,2y) &= \frac{R(2x-1,2y-1)}{4} + \frac{R(2x-1,2y+1)}{4} \\ &+ \frac{R(2x+1,2y-1)}{4} + \frac{R(2x+1,2y+1)}{4}. \end{aligned}$$

Note that in this simple case, the estimated samples are perfectly correlated to their neighbors. As such, a CFA interpolated image can be detected (in the absence of noise) by noticing, for example, that every other sample in every other row or column is perfectly correlated to its neighbors. At the same time, the non-interpolated samples are less likely to be correlated in precisely the same manner. Furthermore, it is likely that tampering will destroy these correlations, or that the splicing together of two images from different cameras will create inconsistent correlations across the composite image. As such, the lack of correlations produced by CFA interpolation can be used to expose it as a forgery.

We begin by assuming a simple linear model for the periodic correlations introduced by CFA interpolation. That is, each interpolated pixel is correlated to a weighted sum of pixels in a small neighborhood centered about itself. While perhaps overly simplistic when compared to the highly non-linear nature of most CFA interpolation algorithms, this simple model is both easy to parametrize and can reasonably approximate each of the CFA interpolation algorithms described above. Note that most CFA algorithms estimate a missing color sample from neighboring samples in all three color channels. For simplicity, however, we ignore these inter-channel correlations and treat each color channel independently.

If the specific form of the correlations is known (i.e., the parameters of the linear model), then it would be straightforward to determine which samples are correlated to their neighbors. On the other hand, if it was known which samples are correlated to their neighbors, the specific form of the correlations could be easily determined. In practice, of course, neither are known. To simultaneously estimate both we employ the expectation/maximization (EM) algorithm.

Recall that the expectation-maximization algorithm (EM) is a two-step iterative algorithm: (1) in the E-step the probability of each sample belonging to each model is estimated; and (2) in the M-step the specific form of the correlations between samples is estimated.

We first formulate the estimation for a 1-D signal. Let $f(x)$ denote a color channel (red, green, or blue) of a CFA interpolated signal. We begin by assuming that each sample in $f(x)$ belongs to one of two models: (1) M_1 if the sample is linearly correlated to its neighbors, satisfying:

$$f(x) \quad = \quad \sum_{u=-N}^{N} \alpha_u f(x+u), \tag{2.41}$$

where the model parameters are given by the linear coefficients $\alpha = \{\alpha_u | -N \leq u \leq N\}$ (N is an integer and $\alpha_0 = 0$) ; or (2) M_2 if the sample is not correlated to its neighbors, i.e., is generated by an "outlier process".

In the E-step, the residual error for the first model is:

$$r_1(x) \quad = \quad f(x) - \sum_{u=-N}^{N} \alpha_u f(x+u) \tag{2.42}$$

Assuming that the probability of observing a sampled generated by model M_1 follows a Gaussian, the likelihood then takes the form:

$$w_1(x) = Pr(\alpha \mid r_1(x)) = \frac{e^{-r_1^2(x)/\sigma}}{e^{-r_1^2(x)/\sigma} + 1/\delta}, \tag{2.43}$$

where a uniform distribution is assumed for the probability of observing a sample generated by the outlier model, M_2.

In the M-step, a new estimate of α is computed using weighted least squares to minimize the following quadratic error function:

$$E(\alpha) = \sum_x w_1(x) \left(f(x) - \sum_{u=-N}^{N} \alpha_u f(x+u) \right)^2 . \tag{2.44}$$

This error function is minimized by computing the gradient with respect to α, setting this gradient equal to zero, and solving for α yielding:

$$\alpha \quad = \quad (F^T W F)^{-1} F^T W \mathbf{f}. \tag{2.45}$$

where, if $N = 2$, for example, the vector $\mathbf{f}$ is:

$$\mathbf{f} = (\, f(3) \quad f(4) \quad f(5) \quad \ldots \,)^T , \tag{2.46}$$

and where the matrix F is:

$$F = \begin{pmatrix} f(1) & f(2) & f(4) & f(5) \\ f(2) & f(3) & f(5) & f(6) \\ f(3) & f(4) & f(6) & f(7) \\ \vdots & \vdots & \vdots & \vdots \end{pmatrix}, \tag{2.47}$$

and W is a diagonal weighting matrix with $w_1(x)$ along the diagonal. The E-step and the M-step are iteratively executed until a stable estimate of α is achieved.

In 2-D, let $f(x,y)$ denote a color channel (red, green, or blue) of a CFA interpolated image. We again assume that each sample in $f(x,y)$ belongs to one of two models: (1) M_1 if the sample is linearly correlated to its neighbors, satisfying:

$$f(x,y) \;=\; \sum_{u,v=-N}^{N} \alpha_{u,v} f(x+u,y+v), \tag{2.48}$$

where the model parameters are given by the linear coefficients $\alpha = \{\alpha_{u,v} | -N \leq u,v \leq N\}$ (N is an integer and $\alpha_{0,0} = 0$) ; or (2) M_2 if the sample is not correlated to its neighbors, i.e., is generated by an "outlier process".

In the E-step, the residual error for the first model is:

$$r_1(x,y) \;=\; f(x,y) - \sum_{u,v=-N}^{N} \alpha_{u,v} f(x+u,y+v) \tag{2.49}$$

Assuming that the probability of observing a sampled generated by model M_1 follows a Gaussian, the likelihood then takes the form:

$$w_1(x,y) = Pr(\alpha \mid r_1(x,y)) = \frac{e^{-r_1^2(x,y)/\sigma}}{e^{-r_1^2(x,y)/\sigma} + 1/\delta}, \tag{2.50}$$

where a uniform distribution is assumed for the probability of observing a sample generated by the outlier model, M_2.

In the M-step, a new estimate of α is computed using weighted least squares to minimize the following quadratic error function:

$$E(\alpha) = \sum_{x,y} w_1(x,y) \left(f(x,y) - \sum_{u,v=-N}^{N} \alpha_{u,v} f(x+u,y+v) \right)^2 \tag{2.51}$$

This error function is minimized by computing the gradient with respect to α, setting this gradient equal to zero, and solving the resulting linear system of equations. Setting equal to zero the partial derivative with respect to one of the coefficients, $\partial E / \partial \alpha_{s,t} = 0$, yields:

$$\sum_{x,y} w_1(x,y) f(x+s,y+t) \sum_{u,v=-N}^{N} \alpha_{u,v} f(x+u,y+v) \;=\; \sum_{x,y} w_1(x,y) f(x+s,y+t) f(x,y). \tag{2.52}$$

Re-ordering the terms on the left-hand side yields:

$$\begin{aligned}\sum_{u,v=-N}^{N} \alpha_{u,v} \left(\sum_{x,y} w_1(x,y) f(x+s,y+t) f(x+u,y+v) \right) \quad &= \\ \sum_{x,y} w_1(x,y) f(x+s,y+t) f(x,y). \qquad & (2.53)\end{aligned}$$

This process is repeated for each component, $\alpha_{s,t}$ to yield a system of linear equations that can be solved using standard techniques. The E-step and the M-step are iteratively executed until a stable estimate of α is achieved.

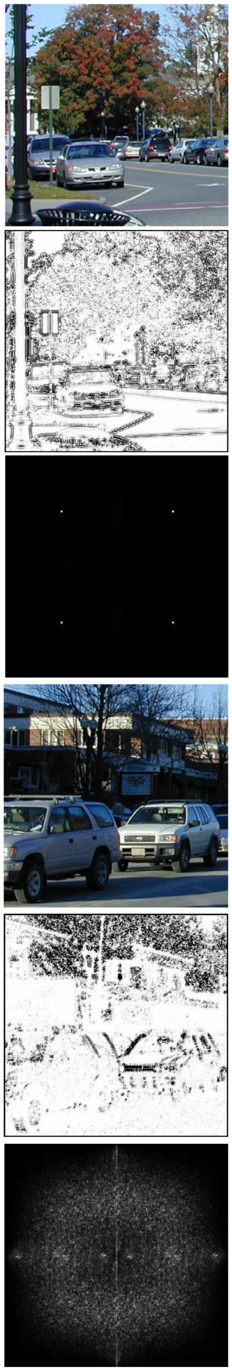


Figure 2.6 A CFA and non-CFA image and the results of CFA analysis.

Shown in the upper three panels of Figure 2.6, from top to bottom, is an original CFA interpolated image, the resulting CFA analysis displayed as a probability map, and the magnitude of the Fourier transform of the probability map. The results of the CFA analysis is displayed as the probability of each pixel belonging to model M_1 (i.e., correlated to their neighbors) in which a white pixel denotes a high probability and a black pixel denotes a low probability. The peaks in the Fourier transform correspond to the periodicity in the probability map which reveal the expected CFA correlations for an authentic image.

Shown in the lower three panels of Figure 2.6, from top to bottom, is an original CFA interpolated image, the resulting CFA analysis displayed as a probability map, and the magnitude of the Fourier transform of the probability map. In this case, the image does not contain the expected CFA correlations as can be seen by the lack of periodicity in the probability map and peaks in the Fourier transform.

2.4 Chromatic Aberration (CA)

Most images contain a variety of aberrations that result from imperfections and artifacts of the optical imaging system. In an ideal imaging system, light passes through the lens and is focused to a single point on the sensor. Optical systems, however, deviate from such ideal models in that they fail to perfectly focus light of all wavelengths. The resulting effect is known as chromatic aberration which occurs in two forms: longitudinal and lateral. Longitudinal aberration manifests itself as differences in the focal planes for different wavelengths of light. Lateral aberration manifests itself as a spatial shift in the locations where light of different wavelengths reach the sensor – this shift is proportional to the distance from the optical center. In both cases, chromatic aberration leads to various forms of color imperfections in the image. When tampering with an image, these aberrations are often disturbed and fail to be consistent across the image. We describe how to exploit these aberrations for forensic analysis.

In classical optics, the refraction of light at the boundary between two media is described by Snell's Law:

$$n \sin(\theta) \quad = \quad n_f \sin(\theta_f), \tag{2.54}$$

where θ is the angle of incidence, θ_f is the angle of refraction, and n and n_f are the refractive indices of the media through which the light passes. The refractive index of glass, n_f, depends on the wavelength of the light that traverses it. This dependency results in polychromatic light being split according to wavelength as it exits the lens and strikes the sensor. Shown in Figure 2.7, for example, is a schematic showing the splitting of short wavelength (solid blue ray) and long wavelength (dashed red ray) light. The result of this splitting of light is termed lateral chromatic aberration. In this 1-D formulation where the incident light reaches the lens at an angle θ, and is split into short wavelength (solid blue ray) and long wavelength (dashed red ray) light with an angle of refraction of θ_r and θ_b, Figure 2.7. These rays strike the sensor at positions x_r and x_b. The relationship between the angle of incidence and angles of refraction are given by Snell's law, Equation (2.54), yielding:

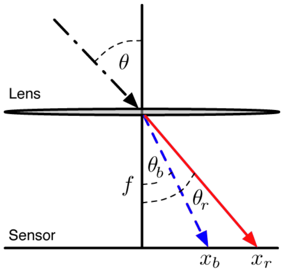


Figure 2.7 Lateral chromatic aberrations (1-D).

$$\begin{aligned} n \sin(\theta) \quad &= \quad n_r \sin(\theta_r) & (2.55) \\ n \sin(\theta) \quad &= \quad n_b \sin(\theta_b), & (2.56) \end{aligned}$$

which are combined to yield:

$$n_r \sin(\theta_r) = n_b \sin(\theta_b). \tag{2.57}$$

Dividing both sides by $\cos(\theta_b)$ gives:

$$\begin{aligned} n_r \sin(\theta_r)/\cos(\theta_b) &= n_b \tan(\theta_b) \\ &= n_b x_b/f, \end{aligned} \tag{2.58}$$

where f is the lens-to-sensor distance. If we assume that the differences in angles of refraction are relatively small, then $\cos(\theta_b) \approx \cos(\theta_r)$. Equation (2.58) then takes the form:

$$\begin{aligned} n_r \sin(\theta_r)/\cos(\theta_r) &\approx n_b x_b/f \\ n_r \tan(\theta_r) &\approx n_b x_b/f \\ n_r x_r/f &\approx n_b x_b/f \\ n_r x_r &\approx n_b x_b \\ x_r &\approx \alpha x_b, \end{aligned} \tag{2.59}$$

where $\alpha = n_b/n_r$.

For a two-dimensional lens and sensor, the distortion caused by lateral chromatic aberration takes a form similar to Equation (2.59). An incident ray reaches the lens at angles θ and ϕ, relative to the $x = 0$ and $y = 0$ planes, respectively. The application of Snell's law yields:

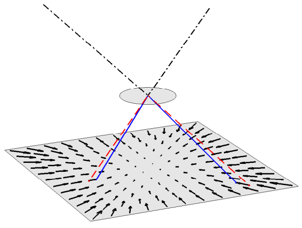


Figure 2.8 Lateral chromatic aberrations (2-D).

$$n_r \sin(\theta_r) = n_b \sin(\theta_b) \tag{2.60}$$

$$n_r \sin(\phi_r) = n_b \sin(\phi_b). \tag{2.61}$$

Following the above 1-D derivation yields the following 2-D model:

$$(x_r, y_r) \approx \alpha(x_b, y_b). \tag{2.62}$$

Shown in Figure 2.8 is vector-based depiction of this aberration, where each vector $\mathbf{v} = \begin{pmatrix} x_r - x_b & y_r - y_b \end{pmatrix}$. Note that this model is simply an expansion/contraction about the center of the image.

In real lenses, the center of optical aberrations is often different from the image center due to the complexities of multi-lens systems. The previous model can therefore be augmented with an additional two parameters, (x_0, y_0), to describe the position of the expansion/contraction center. The model now takes the form:

$$x_r = \alpha(x_b - x_0) + x_0 \tag{2.63}$$

$$y_r = \alpha(y_b - y_0) + y_0. \tag{2.64}$$

Using the green channel as reference, we would like to estimate the aberration between the red and green channels, and between the blue and green channels. Deviations or inconsistencies in these models will then be used as evidence of tampering.

Recall that the model for lateral chromatic aberration consists of three parameters, two parameters for the center of the distortion and one parameter for the magnitude of the distortion. These model parameters will be denoted (x_1, y_1, α_1) and (x_2, y_2, α_2) for the red to green and blue to green distortions, respectively.

The estimation of these model parameters can be framed as an image registration problem. Specifically, lateral chromatic aberration results in an expansion or contraction between the color channels, and hence a misalignment between the color channels. We, therefore, seek the model parameters that bring the color channels back into alignment. There are several metrics that may be used to quantify the alignment of the color channels. To help contend with the inherent intensity differences across the color channels we employ a metric based on mutual information that has proven successful in such situations.

We will describe the estimation of the red to green distortion parameters (the blue to green estimation follows a similar form). Denote the red channel of a RGB image as $R(x, y)$ and the green channel as $G(x, y)$. A corrected version of the red channel is denoted as $R(x_r, y_r)$ where:

$$\begin{aligned} x_r &= \alpha_1(x - x_1) + x_1 \qquad (2.65) \\ y_r &= \alpha_1(y - y_1) + y_1. \qquad (2.66) \end{aligned}$$

The model parameters are determined by maximizing the mutual information between $R(x_r, y_r)$ and $G(x, y)$ as follows:

$$\text{argmax}_{x_1, y_1, \alpha_1} I(\mathcal{R}; \mathcal{G}), \qquad (2.67)$$

where $\mathcal{R}$ and $\mathcal{G}$ are the random variables from which the pixel intensities of $R(x_r, y_r)$ and $G(x, y)$ are drawn. The mutual information between these random variables is defined to be:

$$I(\mathcal{R}; \mathcal{G}) = \sum_{r \in \mathcal{R}} \sum_{g \in \mathcal{G}} \Pr(r, g) \log \left(\frac{\Pr(r, g)}{\Pr(r) \Pr(g)} \right), \qquad (2.68)$$

where $\Pr(\cdot, \cdot)$ is the joint probability distribution, and $\Pr(\cdot)$ is the marginal probability distribution.

Example 2.3 Show that if a pair of random variables are independent, then the mutual information, Equation (2.68), is zero.

This metric of mutual information is maximized using a brute-force iterative search. On the first iteration, a relatively course sampling of the parameter space for x_1, y_1, α_1 is searched. On the second iteration, a refined sampling of the parameter space is performed about the maximum from the first stage. This process is repeated for a specified number of iterations. While this brute-force search can be computationally demanding, it does ensure that the global minimum is reached.

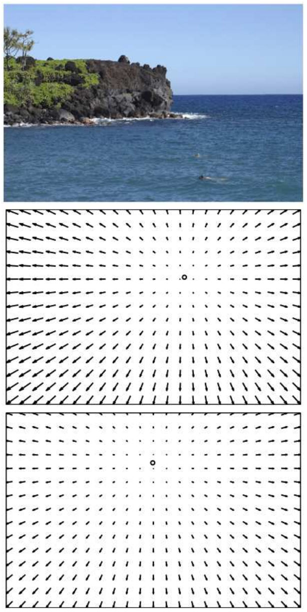


Figure 2.9 An image with CA, calibrated CA, and estimated CA.

Shown in Figure 2.9, from top to bottom, is an image, its actual red to green chromatic aberration, and the estimated chromatic aberration. The small circular mark denotes the center of the chromatic aberrations. Note both that the model of chromatic aberration captures the actual aberrations reasonably well, and that the estimated aberrations are fairly similar to the actual aberrations in terms of the magnitude and direction of the distortion.

Each pixel in a digital camera's sensor records the amount of incident light that strikes it. Slight imperfections in manufacturing introduce small amounts of noise in the recorded image. This noise is spatially varying and consistent over time and can therefore be used for forensic and ballistic purposes [1].

The image imperfections can be modeled as:

$$I(x,y) \quad = \quad I_0(x,y) + \gamma I_0(x,y)K(x,y) + N(x,y), \quad (2.69)$$

where $I_0(\cdot)$ is the noise-free image, γ is a multiplicative constant, $K(\cdot)$ is the multiplicative noise (termed photo-response non-uniformity noise (PRNU)), and $N(\cdot)$ is an additive noise term. The multiplicative PRNU factor is used for forensic and ballistic purposes.

The PRNU is estimated from a series of authentic images $I_k(x,y)$ with $k = 1 \ldots N$ taken from the camera in question. Each image is denoised with any standard denoising filter and subtracted from the original image:

$$W_k(x,y) \quad = \quad I_k(x,y) - \hat{I}_k(x,y), \quad (2.70)$$

where $\hat{I}_k(x,y)$ are the denoised images. The terms $W_k(x,y)$ suppress the underlying image content and make the estimation of the PRNU more reliable. The PRNU is estimated as:

$$K(x,y) \quad = \quad \frac{\sum_{k=1}^{n} W_k(x,y) I_k(x,y)}{\sum_{k=1}^{n} I_k^2(x,y)}. \quad (2.71)$$

The PRNU can then be used to determine if an image originated from a specific camera, or if a portion of an image has been altered. For the latter application, an image in question $I(x,y)$ is denoised and subtracted from itself to yield $W(x,y)$ as described above. The PRNU $K(x,y)$ is estimated from a set of images known to have originated from the same camera as $I(x,y)$. The correlation between the PRNU and the image being analyzed is given by:

$$\rho \quad = \quad I(x,y)K(x,y) \otimes W(x,y), \quad (2.72)$$

where $\otimes$ denotes normalized correlation. The correlation ρ is used as a measure of authenticity and can be computed locally in order to detect localized tampering.

1. Write MatLab code that implements EM for the two-line fitting problem shown in Figure 2.4. Specifically, your code should:

   - Randomly generate a slope and intercept for two lines.
   - Randomly generate points on these lines.
   - Corrupt these points with small amounts of noise.
   - With random starting conditions, use EM to iteratively estimate the model parameters and model assignments.
   - On each EM iteration, your code should render the current estimates of the lines and the data points.
   - Run one thousand simulations and report on the overall accuracy of the model estimation.

Example 2.1

1. $\begin{pmatrix} \log(x_1) & 1 \\ \vdots & \vdots \\ \log(x_n) & 1 \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix}$

2. $\begin{pmatrix} x_1^{y_1} & 1 \\ \vdots & \vdots \\ x_n^{y_n} & 1 \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} z_1 \\ \vdots \\ z_n \end{pmatrix}$

3. $y = x^a$ is non-linear in the model parameter a and can therefore not be solved using least-squares. The log of this model can, however, be solved using least-squares: $\log(y) = a\log(x)$ is linear in the parameter a. Solving this system minimizes the log of the least-squares error, which is different than minimizing the least-squares error in the original model.

Example 2.2 The weighted least-squares version of Equation (2.24) is:

$$W \begin{pmatrix} x_k(1) & 1 \\ x_k(2) & 1 \\ \vdots & \vdots \\ x_k(n) & 1 \end{pmatrix} \begin{pmatrix} a_k \\ b_k \end{pmatrix} = W \begin{pmatrix} y_k(1) \\ y_k(2) \\ \vdots \\ y_k(n) \end{pmatrix}, \qquad (2.73)$$

where W is a diagonal matrix with elements $w_k(i)$. Left-multiplying left- and right-hand sides by the transpose of the data matrix yields:

$$\begin{pmatrix} x_k(1) & \dots & x_k(n) \\ 1 & \dots & 1 \end{pmatrix} \begin{pmatrix} w_k(1) & \dots & 0 \\ \vdots & & \vdots \\ 0 & \dots & w_k(n) \end{pmatrix} \begin{pmatrix} x_k(1) & 1 \\ \vdots & \vdots \\ x_k(n) & 1 \end{pmatrix} \begin{pmatrix} a_k \\ b_k \end{pmatrix} = \begin{pmatrix} x_k(1) & \dots & x_k(n) \\ 1 & \dots & 1 \end{pmatrix} \begin{pmatrix} w_k(1) & \dots & 0 \\ \vdots & & \vdots \\ 0 & \dots & w_k(n) \end{pmatrix} \begin{pmatrix} y_k(1) \\ \vdots \\ y_k(n) \end{pmatrix}.$$

Multiplying by the weighting matrix yields:

$$\begin{pmatrix} x_k(1) & \dots & x_k(n) \\ 1 & \dots & 1 \end{pmatrix} \begin{pmatrix} w_k(1)x_k(1) & w_k(1) \\ \vdots & \vdots \\ w_k(n)x_k(n) & w_k(n) \end{pmatrix} \begin{pmatrix} a_k \\ b_k \end{pmatrix} = \begin{pmatrix} x_k(1) & \dots & x_k(n) \\ 1 & \dots & 1 \end{pmatrix} \begin{pmatrix} w_k(1)y_k(1) \\ \vdots \\ w_k(n)y_k(n) \end{pmatrix},$$

$$\begin{pmatrix} \sum_i w_k(i)x(i)^2 & \sum_i w_k(i)x(i) \\ \sum_i w_k(i)x(i) & \sum_i w_k(i) \end{pmatrix} \begin{pmatrix} a_k \\ b_k \end{pmatrix} = \begin{pmatrix} \sum_i w_k(i)x(i)y(i) \\ \sum_i w_k(i)y(i) \end{pmatrix},$$

which is identical to Equation (2.32).

Example 2.3 If two random variables X and Y are independent, then $\Pr(X,Y) = \Pr(X)\Pr(Y)$. In this case, the ratio in Equation (2.68) is 1, the log of which is 0 yielding a mutual information $I(X,Y) = 0$.

1. J. Fridrich. Digital image forensic using sensor noise. *IEEE Signal Processing Magazine*, 26(2):26-37, 2009.

2. M.K. Johnson and H. Farid. Exposing digital forgeries through chromatic aberration. In *ACM Multimedia and Security Workshop*, Geneva, Switzerland, 2006.

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

3. Pixel-Based Forensics

3.1 Resampling

Consider the creation of a digital forgery that shows a pair of famous movie stars, rumored to have a romantic relationship, walking hand-in-hand. Such a photograph could be created by splicing together individual images of each movie star and overlaying the digitally created composite onto a sunset beach. In order to create a convincing match, it is often necessary to re-size, rotate, or stretch portions of the images. This process requires re-sampling the original image onto a new sampling lattice. Although this re-sampling is often imperceptible, it introduces specific correlations into the image, which when detected can be used as evidence of digital tampering. We describe the form of these correlations, and how they can be automatically detected in any portion of an image.

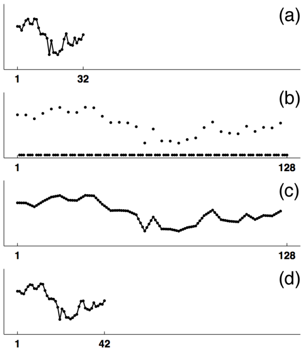


Figure 3.1 Upsampling a 1-D signal.

Consider a 1-D discretely-sampled signal $f(x)$ with m samples, Figure 3.1(a). The number of samples in this signal can be increased or decreased by a factor p/q to n samples in three steps:

1. up-sample: create a new signal $f_u(x)$ with pm samples, where $f_u(px) = f(x)$, $x = 1, 2, ..., m$, and $f_u(x) = 0$ otherwise, Figure 3.1(b).

2. interpolate: convolve $f_u(x)$ with a low-pass filter: $f_i(x) = f_u(x) \star h(x)$, Figure 3.1(c).

3. down-sample: create a new signal $f_d(x)$ with n samples, where $f_d(x) = f_i(qx)$, $t = 1, 2, ..., n$. Denote the re-sampled signal as $g(x) \equiv f_d(x)$, Figure 3.1(d).

Different types of re-sampling algorithms (e.g., linear, cubic) differ in the form of the interpolation filter $h(x)$ in step 2. Since all three steps in the re-sampling of a signal are linear, this process can be described with a single linear equation. Denoting the original and re-sampled signals in vector form, $\mathbf{f}$ and $\mathbf{g}$, respectively, re-sampling takes the form:

$$\mathbf{g} \quad = \quad A_{p/q}\mathbf{f}, \qquad (3.1)$$

where the $n \times m$ matrix $A_{p/q}$ embodies the entire re-sampling process. For example, the matrix for up-sampling by a factor of 4/3 using linear interpolation (Figure 3.1) has the form:

$$A_{4/3} = \begin{pmatrix} 1 & 0 & 0 & 0 & \\ 0.25 & 0.75 & 0 & 0 & \\ 0 & 0.50 & 0.50 & 0 & \\ 0 & 0 & 0.75 & 0.25 & \\ 0 & 0 & 0 & 1 & \\ & & & & \ddots \end{pmatrix}. \tag{3.2}$$

Depending on the re-sampling rate, the re-sampling process will introduce correlations of varying degrees between neighboring samples. For example, consider the up-sampling of a signal by a factor of two using linear interpolation. In this case, the re-sampling matrix takes the form:

$$A_{2/1} = \begin{pmatrix} 1 & 0 & 0 & \\ 0.5 & 0.5 & 0 & \\ 0 & 1 & 0 & \\ 0 & 0.5 & 0.5 & \\ 0 & 0 & 1 & \\ & & & \ddots \end{pmatrix}. \tag{3.3}$$

Here, the odd samples of the re-sampled signal $\mathbf{g}$ take on the values of the original signal $\mathbf{f}$, i.e., $g_{2i-1} = f_i, i = 1, \ldots, m$. The even samples, on the other hand, are the average of adjacent neighbors of the original signal:

$$g_{2i} = 0.5f_i + 0.5f_{i+1}, \tag{3.4}$$

where $i = 1, \ldots, m-1$. Note that since each sample of the original signal can be found in the re-sampled signal, i.e., $f_i = g_{2i-1}$ and $f_{i+1} = g_{2i+1}$, the above relationship can be expressed in terms of the re-sampled samples only:

$$g_{2i} = 0.5g_{2i-1} + 0.5g_{2i+1}. \tag{3.5}$$

That is, across the entire re-sampled signal, each even sample is precisely the same linear combination of its adjacent two neighbors. In this simple case, at least, a re-sampled signal could be detected (in the absence of noise) by noticing that every other sample is perfectly correlated to its neighbors. To be useful in a general forensic setting we need, at a minimum, for these types of correlations to be present regardless of the re-sampling rate.

Consider now re-sampling a signal by an arbitrary amount p/q. In this case we first ask, when is the i^{th} sample of a re-sampled signal equal to a linear combination of its $2N$ neighbors, that is:

$$g_i \overset{?}{=} \sum_{k=-N}^{N} \alpha_k g_{i+k}, \tag{3.6}$$

where α_k are scalar weights (and $\alpha_0 = 0$). Re-ordering terms, and re-writing the above constraint in terms of the re-sampling matrix yields:

$$\begin{aligned} g_i - \sum_{k=-N}^{N} \alpha_k g_{i+k} &= 0 & (3.7) \\ (\mathbf{a}_i \cdot \mathbf{f}) - \sum_{k=-N}^{N} \alpha_k (\mathbf{a}_{i+k} \cdot \mathbf{f}) &= 0 & (3.8) \\ \left(\mathbf{a}_i - \sum_{k=-N}^{N} \alpha_k \mathbf{a}_{i+k} \right) \cdot \mathbf{f} &= 0, & (3.9) \end{aligned}$$

where $\mathbf{a}_i$ is the i^{th} row of the re-sampling matrix $A_{p/q}$, and $\mathbf{f}$ is the original signal. We see now that the i^{th} sample of a re-sampled signal is equal to a linear combination of its neighbors when the i^{th} row of the re-sampling matrix, $\mathbf{a}_i$, is equal to a linear combination of the neighboring rows, $\sum_{k=-N}^{N} \alpha_k \mathbf{a}_{i+k}$.

For example, in the case of up-sampling by a factor of two, Equation (3.3), the even rows are a linear combination of the two adjacent odd rows. Note also that if the i^{th} sample is a linear combination of its neighbors then the $(i - kp)^{th}$ sample (k an integer) will be the same combination of its neighbors, that is, the correlations are periodic. It is, of course, possible for the constraint of Equation (3.9) to be satisfied when the difference on the left-hand side of the equation is orthogonal to the original signal $\mathbf{f}$. While this may occur on occasion, these correlations are unlikely to be periodic.

Given a signal that has been re-sampled by a known amount and interpolation method, it is possible to find a set of periodic samples that are correlated in the same way to their neighbors. Consider again the re-sampling matrix of Equation (3.2). Here, based on the periodicity of the re-sampling matrix, we see that, for example, the 3^{rd}, 7^{th}, 11^{th}, etc. samples of the re-sampled signal will have the same correlations to their neighbors.

Example 3.1 Describe how the third row of Equation (3.2) is correlated to the first, second, fourth, and fifth rows (i.e., what is α_1, α_2, α_4, α_5?). Give an intuition for why the fourth and fifth rows are not similarly correlated to their neighboring rows.

The specific form of the correlations can be determined by finding the neighborhood size, N, and the set of weights, α, that satisfy: $\mathbf{a}_i = \sum_{k=-N}^{N} \alpha_k \mathbf{a}_{i+k}$, Equation (3.9), where $\mathbf{a}_i$ is the i^{th} row of the re-sampling matrix and $i = 3, 7, 11$, etc. If, on the other-hand, we know the specific form of the correlations, α, then it is straightforward to determine which samples satisfy $g_i = \sum_{k=-N}^{N} \alpha_k g_{i+k}$, Equation (3.7).

In practice, of course, neither the re-sampling amount nor the specific form of the correlations are typically known. In order to determine if a signal has been re-sampled, we employ the expectation/maximization algorithm (EM) to simultaneously estimate a set of periodic samples that are correlated to their neighbors, and the specific form of these correlations.

Recall that the expectation-maximization algorithm (EM) is a two-step iterative algorithm: (1) in the E-step the probability of each sample belonging to each model is estimated; and (2) in the M-step the specific form of the correlations between samples is estimated.

In the E-step, the residual error for the first model is:

$$r_1(i) \quad = \quad g_i - \sum_{k=-N}^{N} \alpha_k g_{i+k}. \tag{3.10}$$

Assuming that the probability of observing a sampled generated by model M_1 follows a Gaussian, the likelihood then takes the form:

$$w_1(i) = Pr(\alpha \mid r_1(x,y)) = \frac{e^{-r_1^2(i)/\sigma}}{e^{-r_1^2(i)/\sigma} + 1/\delta}, \tag{3.11}$$

where a uniform distribution is assumed for the probability of observing a sample generated by the outlier model, M_2.

In the M-step, a new estimate of α is computed using weighted least squares, by minimizing the following quadratic error function:

$$E(\alpha) \quad = \quad \sum_i w(i) \left(g_i - \sum_{k=-N}^{N} \alpha_k g_{i+k} \right)^2 . \tag{3.12}$$

This error function is minimized by computing the gradient with respect to α, setting the result equal to zero, and solving for α, yielding:

$$\alpha \quad = \quad (G^T W G)^{-1} G^T W \mathbf{g}. \tag{3.13}$$

where, if $N = 2$, for example, the vector $\mathbf{g}$ is:

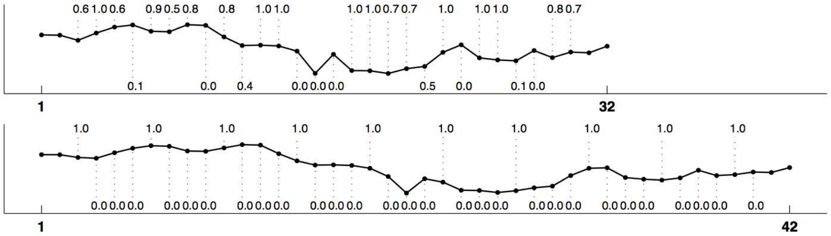
$$\mathbf{g} = \left(g_3 \quad g_4 \quad g_5 \quad \dots \right)^T , \tag{3.14}$$

and where the matrix G is:

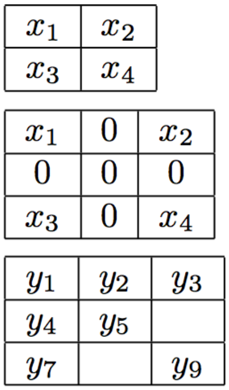
$$G = \begin{pmatrix} g_1 & g_2 & g_4 & g_5 \\ g_2 & g_3 & g_5 & g_6 \\ g_3 & g_4 & g_6 & g_7 \\ \vdots & \vdots & \vdots & \vdots \end{pmatrix} , \tag{3.15}$$

and W is a diagonal weighting matrix with $w(i)$ along the diagonal. The E-step and the M-step are iteratively executed until a stable estimate of α is achieved.

Shown below are the results of running EM on the original and re-sampled signals of Figure 3.1.



Shown on the top of this figure is the original signal where each sample is annotated with its probability of being correlated to its neighbors (the first and last two samples are not annotated due to border effects - a neighborhood size of five ($N = 2$) was used in this example). Similarly, shown on the bottom is the re-sampled signal and the corresponding probabilities. In the later case, the periodic pattern is obvious, where only every 4^{th} sample has probability 1, as would be expected from an up-sampling by a factor of 4/3. As expected, no periodic pattern is present in the original signal.



x_1	x_2
x_3	x_4

x_1	0	x_2
0	0	0
x_3	0	x_4

y_1	y_2	y_3
y_4	y_5	
y_7		y_9

Figure 3.2 Upsampling a 2-D image.

The extension to 2-D images is relatively straight-forward. As with 1-D signals, the up-sampling or down-sampling of an image is still linear and involves the same three steps: up-sampling, interpolation, and down-sampling - these steps are simply carried out on a 2-D lattice. And as with 1-D signals, the re-sampling of an image introduces periodic correlations.

Consider, for example, the simple case of up-sampling by a factor of two. Shown in Figure 3.2 is, from top to bottom, a portion of an original 2-D sampling lattice, the same lattice up-sampled by a factor of two, and a subset of the pixels of the re-sampled image. Assuming linear interpolation, these pixels are given by:

$$\begin{aligned} y_2 &= 0.5y_1 + 0.5y_3 && (3.16) \\ y_4 &= 0.5y_1 + 0.5y_7 && (3.17) \\ y_5 &= 0.25y_1 + 0.25y_3 + 0.25y_7 + 0.25y_9, && (3.18) \end{aligned}$$

where $y_1 = x_1$, $y_3 = x_2$, $y_7 = x_3$, $y_9 = x_4$. Note that of all the pixels of the re-sampled image in the odd rows and even columns (e.g., y_2) will all be the same linear combination of their two horizontal neighbors. Similarly, the pixels of the re-sampled image in the even rows and odd columns (e.g., y_4) will all be the same linear combination of their two vertical neighbors. That is, the correlations are, as with the 1-D signals, periodic. And in the same way that EM was used to uncover these periodic correlations with 1-D signals, the same approach can be used with 2-D images.

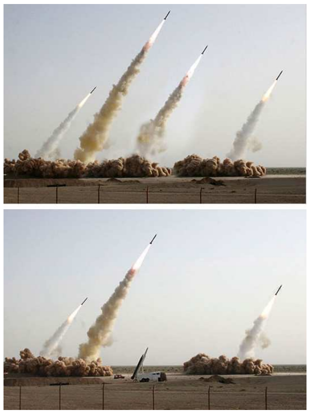


Figure 3.3 An altered photo with cloning (top) and the original photo (bottom).

A common form of photo manipulation is to copy and paste portions of an image to replicate an object or conceal a person in a scene, Figure 3.3. The presence of identical (or virtually identical) regions in an image can, therefore, be used as evidence of tampering.

Given any two regions in an image, it is a simple matter to determine how similar they are using any standard measure of image similarity (e.g., root mean square distance). Searching all possible pairs of regions and region sizes in even a modest-sized image, however, is computationally intractable. In addition, changes in geometry or color of the cloned region further increases the complexity of the search. The complexity of the search for cloned regions can be reduced by operating on salient image features, as opposed to pixels. One such approach is described here [3].

The popular scale invariant feature transform (SIFT) extracts salient keypoints and associates with each keypoint a 128 dimensional feature vector extracted from the histograms of image gradients in a local neighborhood. In order to achieve scale and rotation invariance the size of the neighborhood is determined by the dominant scale of the keypoint, and all gradients are aligned with the keypoint's dominant orientation. In order to achieve some invariance to color changes, the feature vector is normalized to unit length.

An initial matching of keypoints is then computed by finding, for each keypoint, a matching keypoint with minimal Euclidean distance (L_2 norm) between corresponding feature vectors.

The random sample consensus algorithm (RANSAC) is then used to find the largest subset of matching keypoints. To begin, we consider the case when the cloned regions are only translated copies of each other. The RANSAC algorithm is a three-step iterative algorithm: (1) a random group of three or more matched keypoints are selected, $\mathbf{p}_i$ and $\mathbf{q}_i$, $i \geq 3$; (2) the optimal translation between these matched points is estimated by minimizing $\|\mathbf{p}_i - (\mathbf{q}_i + \mathbf{t})\|^2$, where $\mathbf{t} = (\, t_x \quad t_y \,)$ is the 2-D translation vector; and (3) all remaining matched keypoints are classified as inliers if $\|\mathbf{p}_i - (\mathbf{q}_i + \mathbf{t})\|^2$ is less than a specified threshold, and outliers otherwise. These three steps are repeated a specified number of times. The RANSAC algorithm returns the largest set of matched keypoints (inliers) as a possible cloned region.

Example 3.2 Show that the translation $\mathbf{t}$ that minimizes the least-squares error $\|\mathbf{p}_i - (\mathbf{q}_i + \mathbf{t})\|^2$ is:

$$\mathbf{t} = \frac{1}{N}\sum_{i=1}^{N} \mathbf{p}_i - \mathbf{q}_i \tag{3.19}$$

This basic framework can easily be extended to consider the case when the cloned region has undergone a geometric distortion (e.g., rotation or scaling). This is done by estimating in step 2 of the RANSAC algorithm an affine M and translation $\mathbf{t}$ transformation between matched keypoints: $\|\mathbf{p}_i - (M\mathbf{q}_i + \mathbf{t})\|^2$, where M is a 2×2 affine matrix.

Shown in Figure 3.4 are the results of clone detection for the image in the top panel of Figure 3.3. The red/green and yellow/blue color coding corresponds to the detected regions.

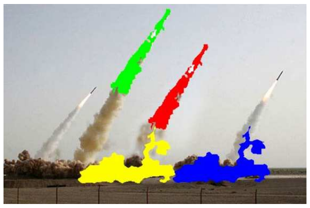


Figure 3.4 Clone detection.

A thumbnail version of the full resolution image is often embedded in an image's JPEG header. This thumbnail is created at the time of recording through a series of filtering operations, contrast adjustment, and compression. The specific choices for these operations differ between camera manufacturers and photo-editing software and can therefore be used as a signature for image authentication.

Given a full resolution image $f(x,y)$, the thumbnail is created by a series of six steps: crop, pre-filter, down-sample, post-filter, contrast and brightness adjustment, and JPEG compression. If the original resolution image is of a different aspect ratio than the thumbnail, then the image needs to either be padded or cropped accordingly. The amount of padding or cropping is specified by four parameters c_l, c_r, c_t, c_b, where c_l and c_r correspond to the padding/cropping on the left and right, and c_t and c_b on the top and bottom, Figure 3.5. A positive value corresponds to a padding (with a pixel value of 0), and a negative value corresponds to a cropping.

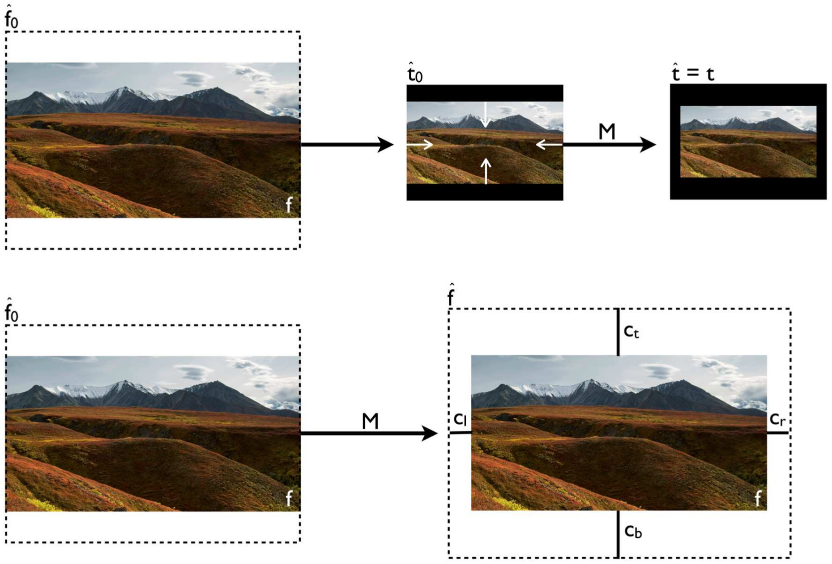


Figure 3.5 Creating the crop boundary for the full resolution image $f(x,y)$. Top row: create an initial crop boundary $\hat{f}_0(x,y)$ so that its aspect ratio matches the thumbnail $t(x,y)$; down-sample to create $\hat{t}_0(x,y)$; align with the actual thumbnail $t(x,y)$ by scaling and translating by $M = (s_x, s_y, \Delta_x, \Delta_y)$ to yield $\hat{t}(x,y)$. Bottom row: adjust the initial crop boundary $\hat{f}_0(x,y)$ by M to yield the desired crop boundary $\hat{f}(x,y)$. The crop boundary is specified by the margins c_l, c_r, c_t, c_b.

Denoting the cropped image as $\hat{f}(x, y)$, the next four processing steps are specified as follows:

$$t(x, y) \quad = \quad \alpha \left(D\{\hat{f}(x, y) \star h_1(x, y)\} \star h_2(x, y) \right) + \beta, \quad (3.20)$$

where $t(x, y)$ is the thumbnail, $h_1(\cdot)$ is the pre-filter, $D\{\cdot\}$ is the down-sampling operator, $h_2(\cdot)$ is the post-filter, $\star$ is the convolution operator, and α and β are the multiplicative contrast and additive brightness adjustment terms, respectively. The pre-filter is typically a low-pass filter applied to avoid spatial aliasing prior to down-sampling, and the optional post-filter is typically a sharpening filter. In the final step, the thumbnail is JPEG compressed with a specified quantization table. In order to simplify this model, a series of assumptions are made:

- the pre-filter is assumed to be a circularly symmetric Gaussian, $\exp(-(x^2 + y^2)/\sigma^2)$, with width σ
- the pre-filter is unit sum
- the post-filter is 3×3 pixels in size
- the post-filter is symmetric ($h_2(x, y) = h_2(-x, y)$ and $h_2(x, y) = h_2(x, -y)$), yielding a filter of the form $(a\ b\ a\ ;\ b\ c\ b\ ;\ a\ b\ a)$
- the post-filter is unit-sum, constraining $c = 1 - (4a + 4b)$.

With these constraints, the full model for creating a thumbnail is specified by 11 processing parameters: 2 for the size of the thumbnail, 4 for the cropping/padding, 1 for the pre-filter, 2 for the post-filter, 2 for the contrast and brightness. In addition, there are 128 compression parameters: the JPEG quantization table is specified by two 8×8 tables corresponding to the quantization for the luminance and chrominance channels (see also Section 1.3).[2] This yields a total of 139 model parameters. The estimation of these model parameters is described next.

In the first step of thumbnail construction, a rectangular cropping boundary relative to the full resolution image is specified. This cropping boundary is determined by anisotropically scaling and translating a bounding box of the same size as the full resolution image such that the cropped and downsampled image matches the extracted thumbnail, Figure 3.5.

The cropping parameters are estimated by first specifying an initial boundary that has the same aspect ratio as the thumbnail,

[2] It is assumed that the two chrominance channels employ the same quantization table.

are scaled to encompass the maximum image dimension, and are translated such that the image is centered within the boundary, Figure 3.5. The full resolution image $f(x, y)$ is then cropped according to this boundary, padded with zeros (where necessary), and downsampled to yield an initial thumbnail:

$$\hat{t}_0(x, y) \quad = \quad D\{\hat{f}_0(x, y)\}, \tag{3.21}$$

where $\hat{f}_0(x, y)$ is the initial cropped image, and the downsampling rate is $\max(N_x/n_x, N_y/n_y)$, where (N_x, N_y) and (n_x, n_y) are the image and thumbnail dimensions, respectively. This initial thumbnail is then anisotropically scaled and translated to match the extracted thumbnail $t(x, y)$:

$$t(x, y) \quad = \quad \hat{t}_0(s_x x + \Delta_x, s_y y + \Delta_y), \tag{3.22}$$

where (s_x, s_y) and (Δ_x, Δ_y) are the scaling and translation parameters. These parameters are estimated using a coarse-to-fine differential registration technique (see Section 7.1). This registration is performed on a grayscale version of the color thumbnail. The scaling and translation parameters are then used to specify the cropping boundary for the full resolution image. Specifically, the coordinates of the four corners of the initial cropped image $\hat{f}_0(x, y)$ are scaled and translated[3] by (s_x, s_y) and (Δ_x, Δ_y), yielding the desired rectangular cropping boundary $\hat{f}(x, y)$, Figure 3.5. This boundary is parametrized by the horizontal and vertical margins c_l, c_r, c_t, and c_b, specified as a percentage of the image dimensions, Figure 3.5. When down-sampling $\hat{f}(x, y)$, the sampling rate in the horizontal and vertical directions must be independently adjusted so that the final dimensions match that of the thumbnail $t(x, y)$.

Next, the pre-filter $h_1(\cdot)$, post-filter $h_2(\cdot)$, and contrast and brightness terms α and β are estimated by minimizing the following error function:

$$E_1(h_1, h_2) = \sum_{x,y} \left[t(x, y) - \alpha \left(D\{\hat{f}(x, y) \star h_1(x, y)\} \star h_2(x, y)\right) - \beta \ \right]^2 . \tag{3.23}$$

This error function is only specified in terms of the pre- and post-filter, and not the contrast and brightness. Given a pair of filters, the contrast and brightness are estimated by minimizing the following error function:

$$E_2(\alpha, \beta) \quad = \quad \sum_{x,y} \Big[t(x, y) - (\alpha \hat{t}_h(x, y) + \beta)\Big]^2 , \tag{3.24}$$

[3] Because the translation parameters (Δ_x, Δ_y) are specified in thumbnail coordinates, they must be scaled by the downsampling rate between $\hat{f}_0(x, y)$ and $\hat{t}_0(x, y)$.

where $\hat{t}_h(x,y) = D\{\hat{f}(x,y) \star h_1(x,y)\} \star h_2(x,y)$. This error function is minimized using standard least-squares estimation. The summation in $E_1(\cdot)$ and $E_2(\cdot)$ are performed over all three color channels of the thumbnail. The error function $E_1(\cdot)$ is minimized using a brute-force search over the filter parameters, where on each iteration of this search, the error function $E_2(\cdot)$ is minimized analytically to yield the best contrast and brightness terms for the specified filters. These parameters are refined using an iterative Nelder-Mead minimization, which is bootstrapped with the results of the brute-force minimization.

In practice, we have found that the minimization of $E_1(\cdot)$ is slightly more effective when performed in the Fourier domain. Specifically, we minimize:

$$E_1(h_1, h_2) = \sum_{\omega_x, \omega_y} W(\omega_x, \omega_y) \left(\|T(\omega_x, \omega_y)\| - \|\hat{T}(\omega_x, \omega_y)\| \right)^2 \quad (3.25)$$

where $T(\cdot)$ is the Fourier transform of the actual thumbnail $t(\cdot)$, $\hat{T}(\cdot)$ is the Fourier transform of our constructed thumbnail $\alpha(D\{\hat{f} \star h_1\} \star h_2) + \beta$, and $\|\cdot\|$ denotes the magnitude of the Fourier transform. This error function is frequency weighted with a highpass filter, $W(\cdot)$, because after the initial alignment of Equation (3.22) the low-frequencies are already well aligned.

Lastly, the thumbnail dimensions and 128 thumbnail compression parameters are extracted directly from the JPEG header. Since the original thumbnail $t(\cdot)$ was compressed with these parameters, we must also compress our estimates of the thumbnail during the parameter estimation stages described above. Specifically, $\hat{t}_0(\cdot)$ is compressed prior to estimating the scaling and translation parameters, and $\hat{t}_h(\cdot)$ is compressed prior to estimating the contrast and brightness terms.

Because the absolute value of the contrast and brightness terms may depend on the underlying image content, these parameters are combined into a single binary-valued parameter corresponding to the presence or absence of contrast/brightness adjustment. Specifically, a camera is said to apply a contrast/brightness adjustment if the camera's contrast deviates by more than 0.075 from unity, or if the brightness deviates by more than 0.05 from zero (assuming a luminance scale of $[0, 1]$).

In [1], $1,514$ images spanning 142 cameras of different make and model were analyzed. Cameras of the same make and model sometimes vary their image and thumbnail size and quantization table (variations in the thumbnail size and quantization table can, for example, be due to differences in firmware). Because these variations affect the overall thumbnail parametrization, we partitioned the $1,514$ images into 245 "camera classes" containing images from the same make, model, thumbnail size and thumbnail quantization table.

Two thumbnail models were considered to be equivalent if their integer-valued parameters (contrast/brightness, size, quantization table) were the same and if their real-valued parameters (crop boundary, pre-filter, post-filter) were within the specified thresholds described above. Cameras were then grouped into equivalence classes of identical models. From these images, 40.8% of the cameras are in an equivalence class of size one (i.e., are unique), 9.8% are in an equivalence class of size two, 8.2% are in an equivalence class of size three, and the largest equivalence class is of size 48, with 19.6% of the cameras. There is only one class of size 48 and it contains 42 Canon Powershots and 6 Canon Digital Ixus of varying models. And lastly, an image was saved with Photoshop CS3 at each of 13 possible JPEG quality settings. The thumbnail parameters were estimated for each image and compared to the 245 camera classes. None of the Photoshop thumbnail parameters were shared by any of the camera classes. This implies that any photo-editing with Photoshop can be exposed by simply examining an image's thumbnail.

3.4 Problem Set

1. You will find five images on the course webpage. Determine which of these images were up-sampled by a factor of 2.

   www.cs.dartmouth.edu/farid/Hany_Farid/Teaching/Teaching.html

Example 3.1 $\alpha_1 = -1/6$, $\alpha_2 = 2/3$, $\alpha_3 = 2/3$, and $\alpha_4 = -1/6$.

Example 3.2 The least-squares system in $\mathbf{t} = (\, t_x \quad t_y \,)^T$ is:

$$
\begin{aligned}
\begin{pmatrix} 1 & 0 \\ \vdots & \vdots \\ 1 & 0 \\ 0 & 1 \\ \vdots & \vdots \\ 0 & 1 \end{pmatrix} \begin{pmatrix} t_x \\ t_y \end{pmatrix} &= \begin{pmatrix} p_x^1 - q_x^1 \\ \vdots \\ p_x^n - q_x^n \\ p_y^1 - q_y^1 \\ \vdots \\ p_y^n - q_y^n \end{pmatrix} \\
M\mathbf{t} &= \mathbf{b}.
\end{aligned}
$$

The least-squares solution is $\mathbf{t} = (M^T M)^{-1} M^T \mathbf{b}$, where:

$$
\begin{aligned}
(M^T M)^{-1} M^T &= \left(\begin{pmatrix} 1 & \dots & 1 & 0 & \dots & 0 \\ 0 & \dots & 0 & 1 & \dots & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ \vdots & \vdots \\ 1 & 0 \\ 0 & 1 \\ \vdots & \vdots \\ 0 & 1 \end{pmatrix} \right)^{-1} \begin{pmatrix} 1 & \dots & 1 & 0 & \dots & 0 \\ 0 & \dots & 0 & 1 & \dots & 1 \end{pmatrix} \\
&= \begin{pmatrix} N & 0 \\ 0 & N \end{pmatrix}^{-1} \begin{pmatrix} 1 & \dots & 1 & 0 & \dots & 0 \\ 0 & \dots & 0 & 1 & \dots & 1 \end{pmatrix} \\
&= \begin{pmatrix} 1/N & 0 \\ 0 & 1/N \end{pmatrix} \begin{pmatrix} 1 & \dots & 1 & 0 & \dots & 0 \\ 0 & \dots & 0 & 1 & \dots & 1 \end{pmatrix} \\
&= \begin{pmatrix} 1/N & \dots & 1/N & 0 & \dots & 0 \\ 0 & \dots & 0 & 1/N & \dots & 1/N \end{pmatrix},
\end{aligned}
$$

from which:

$$
\mathbf{t} = (M^T M)^{-1} M^T \mathbf{b} = \begin{pmatrix} 1/N & \dots & 1/N & 0 & \dots & 0 \\ 0 & \dots & 0 & 1/N & \dots & 1/N \end{pmatrix} \begin{pmatrix} p_x^1 - q_x^1 \\ \vdots \\ p_x^n - q_x^n \\ p_y^1 - q_y^1 \\ \vdots \\ p_y^n - q_y^n \end{pmatrix}.
$$

To yield the solution:

$$
\begin{aligned}
t_x &= \frac{1}{N} \sum_{i=1}^{N} p_x^i - q_x^i \\
t_y &= \frac{1}{N} \sum_{i=1}^{N} p_y^i - q_y^i
\end{aligned}
$$

1. E. Kee and H. Farid. Digital image authentication from thumbnails. In *SPIE Symposium on Electronic Imaging*, San Jose, CA, 2010.

2. M. Kirchner and T. Gloe. On resampling detection in recompressed images. In *IEEE Workshop on Information Forensics and Security*, pages 21-25, 2009.

3. X. Pan and S. Lyu. Region duplication detection using image feature matching. *IEEE Transactions on Information Forensics and Security*, 5(4):857-867, 2010.

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

4. Statistical-Based Forensics

4.1 Principal Component Analysis (PCA) †

Principal component analysis (PCA) is the classic approach to reducing the complexity of analyzing high-dimensional data by projection into a lower-dimensional linear subspace. PCA projects data onto axes of maximal data variance. In so doing, the dimensionality of data is reduced while minimizing the loss of information or distortion. Shown in the top panel of Figure 4.1 are data in a 2-D space. Shown in the bottom panel is this data projected onto a 1-D linear subspace (black line) as determined by PCA.

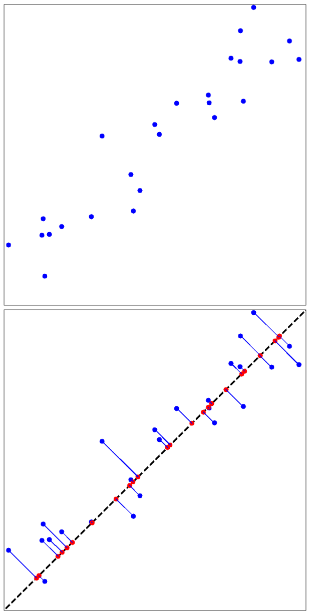


Figure 4.1 Dimensionality reduction from 2-D (blue) to 1-D (red).

Denote column vectors $\mathbf{x}_i \in \mathcal{R}^n$, $i = 1, ..., N$ as the input data. The overall mean is:

$$\mu = \frac{1}{N} \sum_{i=1}^{N} \mathbf{x}_i \qquad (4.1)$$

The zero-meaned data is packed into a $n \times N$ matrix:

$$M = \left(\mathbf{x_1} - \mu \quad \mathbf{x_2} - \mu \quad \ldots \quad \mathbf{x_N} - \mu \right) \qquad (4.2)$$

The $n \times n$ covariance matrix is computed as:

$$C = MM^T. \qquad (4.3)$$

The principle components are the eigenvectors $\mathbf{e}_j$ of the covariance matrix (i.e., $C\mathbf{e}_j = \lambda_j \mathbf{e}_j$), where the eigenvalue, λ_j is proportional to the variance of the original data along the j^{th} eigenvector. The dimensionality of each $\mathbf{x}_i$ is reduced from n to p by projecting (via an inner product) each $\mathbf{x}_i$ onto the top p eigenvalue-eigenvectors. The resulting p-dimensional vector is the reduced-dimension representation.

Example 4.1 If the dimensionality of n is larger than the number of data points N, then the eigenvectors of the smaller covariance M^TM can be computed. Show that if $\mathbf{e}_j$ is the eigenvector of M^TM, then the desired eigenvectors of the covariance matrix MM^T is $M\mathbf{e}_j$.

Note that by formulating the problem of dimensionality reduction in terms of maximizing projected variance, it is being implicitly assumed that the original data is Gaussian distributed. Significant deviations of data from this assumption can yield highly undesirable results in which significant distortions are introduced into the projected data.

4.2 Linear Discriminant Analysis (LDA) †

Linear Discriminant Analysis (LDA) is the standard approach to multi-class classification. LDA projects data onto a linear subspace so that the within-class scatter (specifically, the within-class variance) is minimized and the across-class scatter is maximized. Shown in Figure 4.2 are two classes of data in a 2-D space (red and blue solid points), and their projection onto an axis that is optimal for classification (red and blue open circles). Note that in this 1-D space, the data are perfectly separated and can be classified with a simple threshold. Novel data are projected onto the same axis and classified by comparing against a threshold.

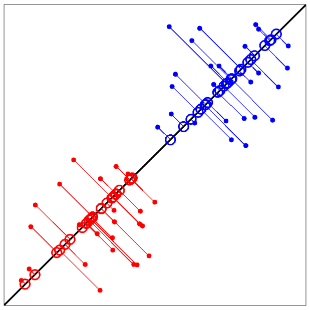


Figure 4.2 Two-class LDA.

For simplicity a two-class LDA is described – the extension to multiple classes is straight-forward. Denote column vectors $\mathbf{x}_i$, $i = 1, ..., N_x$ and $\mathbf{y}_j$, $j = 1, ..., N_y$ as training data from each of two classes. The within-class means are defined as:

$$\mu_x = \frac{1}{N_x}\sum_{i=1}^{N_x}\mathbf{x}_i, \quad \text{and} \quad \mu_y = \frac{1}{N_y}\sum_{j=1}^{N_y}\mathbf{y}_j. \tag{4.4}$$

The between-class mean is defined as:

$$\mu = \frac{1}{N_x + N_y}\left(\sum_{i=1}^{N_x}\mathbf{x}_i + \sum_{j=1}^{N_y}\mathbf{y}_j\right). \tag{4.5}$$

The within-class scatter matrix is defined as:

$$S_w = M_x M_x^T + M_y M_y^T, \tag{4.6}$$

where, the i^{th} column of matrix M_x contains the zero-meaned i^{th} exemplar given by $\mathbf{x}_i - \mu_x$. Similarly, the j^{th} column of matrix M_y contains $\mathbf{y}_j - \mu_y$. The between-class scatter matrix is defined as:

$$S_b = N_x(\mu_x - \mu)(\mu_x - \mu)^T + N_y(\mu_y - \mu)(\mu_y - \mu)^T. \tag{4.7}$$

Let $\mathbf{e}$ be the maximal generalized eigenvalue-eigenvector of S_b and S_w (i.e., $S_b\mathbf{e} = \lambda S_w\mathbf{e}$). The training data $\mathbf{x}_i$ and $\mathbf{y}_j$ are projected onto the one-dimensional linear subspace defined by $\mathbf{e}$ (i.e., $\mathbf{x}_i^T\mathbf{e}$ and $\mathbf{y}_j^T\mathbf{e}$). This projection simultaneously minimizes the within-class scatter while maximizing the between-class scatter.

Once the LDA projection axis is determined from the training set, a novel exemplar, $\mathbf{z}$, from the testing set is classified by first projecting onto the same subspace, $\mathbf{z}^T\mathbf{e}$. In the simplest case, the class to which this exemplar belongs is determined via a simple threshold.

In the case of a two-class LDA, we are guaranteed to be able to project onto a one-dimensional subspace (i.e., there will be at most one non-zero eigenvalue). In the case of a N-class LDA, the projection may be onto as high as a $N-1$-dimensional subspace.

By formulating the optimization in terms of maximizing projected variance, it is being implicitly assumed that the original data is Gaussian distributed. Significant deviations of data from this assumption can result in poor classification results.

Example 4.2 Sketch two data classes in 2-D that are completely separated, but for which LDA completely fails to separate the data.

Computer graphics rendering software is capable of generating highly realistic images that can be difficult to differentiate from photographic images. However, because computer generated (CG) images are rendered under idealized models of geometry, lighting, and camera optics and sensors, it is likely that their underlying statistics will differ from those of photographic images. To this end, we describe a method for differentiating between CG and photographic images based on based on first- and higher-order wavelet statistics.

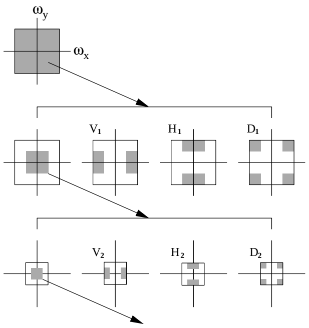


Figure 4.3 Spatial frequency decomposition of QMF.

The decomposition of images using basis functions that are localized in spatial position, orientation, and scale (e.g., wavelet) have proven extremely useful in image compression, image coding, noise removal, and texture synthesis. One reason is that such decompositions exhibit statistical regularities that can be exploited. The image decomposition described here is based on separable quadrature mirror filters (QMFs) [4] (a close relative to wavelets). As illustrated in Figure 4.3, this decomposition splits the frequency space into multiple scales, and orientations (a vertical, a horizontal, and a diagonal subband). For a color (RGB) image, the decomposition is applied independently to each color channel. The resulting vertical, horizontal, and diagonal subbands for scale i are denoted as $V_i^c(x,y)$, $H_i^c(x,y)$, and $D_i^c(x,y)$ respectively, where $c \in \{r,g,b\}$.

The distribution of coefficients in each subband are characterized by the first four order statistics (mean, variance, skewness, and kurtosis) of the subband coefficient histograms at each orientation, scale, and color channel. These statistics form the first half of the statistical model. While these statistics describe the basic coefficient distributions, they are unlikely to capture the strong correlations that exist across space, orientation, and scale. For example, salient image features such as edges tend to orient spatially in certain direction and extend across multiple scales. These image features result in substantial local energy across many scales, orientations, and spatial locations. As such, a strong coefficient in a horizontal subband may indicate that its left and right spatial neighbors in the same subband will also have a large value. Similarly, if there is a coefficient with large magnitude at scale i, it is also very likely that its "parent" at scale $i+1$ will also have a large magnitude.

In order to capture some of these higher-order statistical correlations, a second set of statistics are collected that are based on the errors in a linear predictor of coefficient magnitude. For the purpose of illustration, consider first a vertical band of the green channel at scale i, $V_i^g(x,y)$. A linear predictor for the magnitude of these coefficients in a subset of all possible spatial, orientation, scale, and color neighbors is given by:

$$\begin{aligned} |V_i^g(x,y)| &= w_1|V_i^g(x-1,y)| + w_2|V_i^g(x+1,y)| \\ &+ w_3|V_i^g(x,y-1)| + w_4|V_i^g(x,y+1)| \\ &+ w_5|V_{i+1}^g(x/2,y/2)| + w_6|D_i^g(x,y)| \\ &+ w_7|D_{i+1}^g(x/2,y/2)| + w_8|V_i^r(x,y)| \\ &+ w_9|V_i^b(x,y)|, \end{aligned} \tag{4.8}$$

where $|\cdot|$ denotes absolute value and w_k are the scalar weights. This linear relationship can be expressed more compactly in matrix form as:

$$\mathbf{v} = Q\mathbf{w}, \tag{4.9}$$

where $\mathbf{v}$ contains the coefficient magnitudes of $V_i^g(x,y)$ strung out into a column vector (to reduce sensitivity to noise, only magnitudes greater than 1 are considered), the columns of the matrix Q contain the neighboring coefficient magnitudes as specified in Equation (4.8), and $\mathbf{w} = (w_1 \ \ ... \ \ w_9)^T$. The weights $\mathbf{w}$ are determined by minimizing the following quadratic error function:

$$E(\mathbf{w}) = \|\mathbf{v} - Q\mathbf{w}\|^2 \tag{4.10}$$

using standard least-squares to yield:

$$\mathbf{w} = (Q^TQ)^{-1}Q^T\mathbf{v}. \tag{4.11}$$

Given the large number of constraints (one per pixel) in only nine unknowns, it is generally safe to assume that the 9×9 matrix Q^TQ will be invertible.

Given the linear predictor, the log error between the actual coefficient and the predicted coefficient magnitudes is:

$$\mathbf{p} = \log(\mathbf{v}) - \log(|Q\mathbf{w}|), \tag{4.12}$$

where the $\log(\cdot)$ is computed point-wise on each vector component. As with the coefficient statistics, the mean, variance, skewness, and kurtosis of this error distribution are collected. This process is repeated for scales $i = 1, ..., n-1$, and for the subbands V_i^r and

V_i^b, where the linear predictors for these subbands are of the form:

$$\begin{aligned}
|V_i^r(x,y)| &= w_1|V_i^r(x-1,y)| + w_2|V_i^r(x+1,y)| \\
&+ w_3|V_i^r(x,y-1)| + w_4|V_i^r(x,y+1)| \\
&+ w_5|V_{i+1}^r(x/2,y/2)| + w_6|D_i^r(x,y)| \\
&+ w_7|D_{i+1}^r(x/2,y/2)| + w_8|V_i^g(x,y)| \\
&+ w_9|V_i^b(x,y)|, \qquad (4.13)
\end{aligned}$$

and

$$\begin{aligned}
|V_i^b(x,y)| &= w_1|V_i^b(x-1,y)| + w_2|V_i^b(x+1,y)| \\
&+ w_3|V_i^b(x,y-1)| + w_4|V_i^b(x,y+1)| \\
&+ w_5|V_{i+1}^b(x/2,y/2)| + w_6|D_i^b(x,y)| \\
&+ w_7|D_{i+1}^b(x/2,y/2)| + w_8|V_i^r(x,y)| \\
&+ w_9|V_i^g(x,y)|. \qquad (4.14)
\end{aligned}$$

A similar process is repeated for the horizontal and diagonal subbands. As an example, the predictor for the green channel takes the form:

$$\begin{aligned}
|H_i^g(x,y)| &= w_1|H_i^g(x-1,y)| + w_2|H_i^g(x+1,y)| \\
&+ w_3|H_i^g(x,y-1)| + w_4|H_i^g(x,y+1)| \\
&+ w_5|H_{i+1}^g(x/2,y/2)| + w_6|D_i^g(x,y)| \\
&+ w_7|D_{i+1}^g(x/2,y/2)| + w_8|H_i^r(x,y)| \\
&+ w_9|H_i^b(x,y)|, \qquad (4.15)
\end{aligned}$$

and

$$\begin{aligned}
|D_i^g(x,y)| &= w_1|D_i^g(x-1,y)| + w_2|D_i^g(x+1,y)| \\
&+ w_3|D_i^g(x,y-1)| + w_4|D_i^g(x,y+1)| \\
&+ w_5|D_{i+1}^g(x/2,y/2)| + w_6|H_i^g(x,y)| \\
&+ w_7|V_i^g(x,y)| + w_8|D_i^r(x,y)| \\
&+ w_9|D_i^b(x,y)|. \qquad (4.16)
\end{aligned}$$

For the horizontal and diagonal subbands, the predictor for the red and blue channels are determined in a similar way as was done for the vertical subbands, Equations (4.13)-(4.14). For each oriented, scale and color subband, a similar error metric, Equation (4.12), and error statistics are computed.

For a multi-scale decomposition with scales $i = 1, ..., n$, the total number of basic coefficient statistics is $36(n-1)$ ($12(n-1)$ per color channel), and the total number of error statistics is also $36(n-1)$, yielding a grand total of $72(n-1)$ statistics. These statistics form the feature vector to be used to discriminate between CG and photographic images.

From the measured statistics of a training set of images labeled as CG or photographic, our goal is to build a classifier that can determine to which category a novel test image belongs. The most basic approach is to employ a two-class linear discriminant analysis (LDA), Section **??**.

In [2] a database of $40,000$ photographic and $6,000$ CG images were collected from a variety of on-line sources, from which the coefficient and error statistics were extracted. To accommodate different image sizes, only the central 256×256 region of each image was considered. For each image region, a four-level three-orientation QMF pyramid was constructed for each color channel, from which a 216-dimensional feature vector (72 per color channel) of coefficient and error statistics was collected.

From the $46,000$ feature vectors, $32,000$ photographic and $4,800$ CG feature vectors were used to train the LDA. The remaining feature vectors were used to test the classifier. In the training stage, the classifier correctly classified 58.7% and 99.4% of the photographic and CG images In the testing stage, the classifier correctly classified 54.6% and 99.2% of the photographic and CG images. A non-linear support vector machine (SVM) had better performance, correctly classifying 66.8% and 98.8% of the photographic images and CG images.

The asymmetry in the classification is by design because in a legal setting, classifying a CG as photographic has more severe implications than mis-classifying a photographic image as CG [1]. The relative classification accuracies can be controlled by adjusting the LDA threshold or SVM parameters.

The past few decades have seen tremendous advances in computer graphics rendering software and hardware. These advances have led to remarkable theatrical releases that blur the line between reality and fantasy. At the same time, this technology has resulted in challenging legal situations. Most notably, the 1996 Child Pornography Prevention Act expanded the prohibition on child pornography to include not only pornographic images of actual children but also any computer generated (CG) images that simulate a minor engaging in sexually explicit conduct. This ruling was subsequently challenged, and in 2002 the U.S. Supreme Court found that portions of the CPPA were overly broad and infringed on the first amendment *(Ashcroft v. Free Speech Coalition).* This new ruling classified computer generated child pornography, the creation of which does not involve an actual child, as protected speech. The ability to distinguish between protected (CG) and illegal (photographic) material has, therefore, become essential.

When an image is not of sufficient quality or resolution to undergo a statistical analysis, Section 4.3, we must rely on a visual examination. We describe a perceptual study that tests the ability of the average user to distinguish CG from photographic images of people [1].

Thirty CG images were downloaded from two popular computer graphics websites. The people depicted in these images vary in age, gender, race, pose, and lighting. As a control set, thirty high-resolution photographic images were downloaded that closely matched the CG images in terms of age, gender, race, and pose. The background was manually deleted from each CG and photographic image so that observers could only use the rendered or photographed person to make their judgments.

Because the sizes of the CG and photographic images varied significantly, each image was cropped to a square aspect-ratio and down-sampled so that the area of the person depicted was $122,600$ pixels (approximately 350×350 pixels). This down-sampling had the added benefit of largely removing any JPEG artifacts in the original JPEG image. The image was then saved as the highest quality JPEG.

Each CG and photographic image was then color adjusted to match the brightness (mean) and contrast (variance) of each luminance and chrominance channel. This brightness and contrast

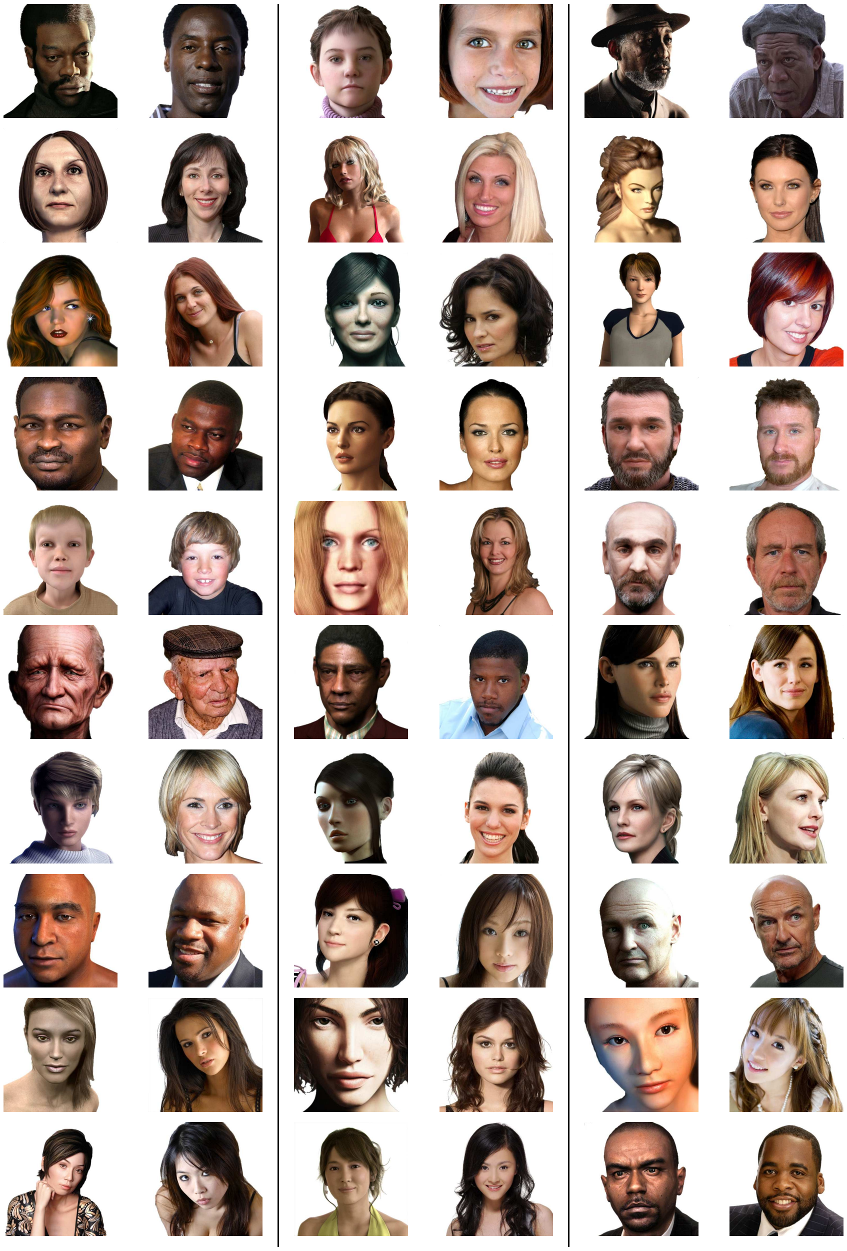


Figure 4.4 CG (left) and photographic (right) images.

matching ensured that observers could not classify images based on systematic differences in simple low-level image statistic (as they could if, for example, the CG images generally had a higher contrast than their photographic counterparts).

The cropped, masked, down-sampled, and brightness and contrast adjusted images are shown in Figure 4.4, where the paired CG and photographic images are shown side-by-side.

In addition to testing the ability of human observers to classify images as CG or photographic, the impact of resolution was also tested. Shown in Figure 4.5 is a CG image at its original resolution, and down-sized by a factor of 0.025, 0.050, 0.100, 0.250, and 0.500 along each dimension. Because the images were initially scaled to match the area of the person depicted, the absolute size of these images depends on content. Across all 60 CG and photographic images, the average size in pixels are: 13 × 13, 27 × 27, 54 × 54, 109 × 109, 218 × 218, and 436 × 436.

Four hundred and thirty six observers were recruited through Amazon's Mechanical Turk crowd sourcing utility and asked to rank each of 30 images of varying resolution, quality, and color as either CG or photographic.

Observer reliability is characterized as the probability that an image is in fact photographic when an observer believes it to be photographic. This measure is adopted because in a legal setting it is the most critical measure in assessing a defendant's guilt or innocence. This measure can be expressed as the following conditional probability, where R denotes the user response and I the image category (both variables can take on the value of "CG" or "photo"):

$$P(I = \text{photo}|R = \text{photo}). \tag{4.17}$$

This conditional probability can be estimated directly from the observer responses.

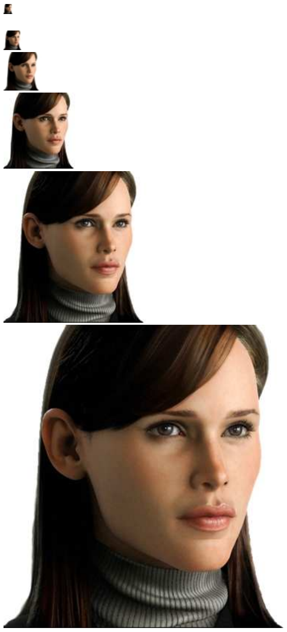


Figure 4.5 A CG image at six resolutions.

The reliability of photographic classification is, from smallest to largest scale, 62.1%, 64.8%, 73.1%, 79.6%, 84.7%, and 79.5%. An interesting and somewhat surprising result is that observers consistently perform better at one-half resolution than the full resolution. It is speculated that performance at the highest resolution is lower because the fine details in computer generated images are so accurate that observers take their presence as evidence of a photographic image. At one-half resolution, however, these details are not visible, and so observers rely on other cues which, interestingly, are more distinguishing.

In addition to the reliability of judging that an image is a photograph, it is also useful to know the reliability of judging that an image is CG. By replacing "photo" with "CG" in Equation (4.17), the conditional probability that an image is CG if an observers says it is CG is given by $P(I = \text{CG}|R = \text{CG})$. This reliability is, from smallest to largest scale, 60.25%, 72.51%, 88.11%, 90.48%, 94.42%, and 92.28%. These results are valuable in interpreting the reliability of photograph classification. If, for example, observers were highly conservative in classifying an image as photo, then they would have a high reliability for photographs but low reliability for CG. This is clearly not the case since observers have similar reliability for judging photographs and CG.

It seems very likely that the accuracies reported here are a lower bound on human performance. The observers were given no training and no incentive to perform well. Most decisions were made within five seconds. Compared with the types of images encountered in forensic settings, the images were relatively impoverished, containing only a single person depicted from the neck up against a blank background. A full body figure interacting with the environment or other people is far more difficult to render photorealistically. But while observer performance can likely be improved, there is little doubt that with time rendering technologies will also improve.

4.5 Problem Set

1. Build a two-class LDA classifier that distinguishes between the following two classes of images.

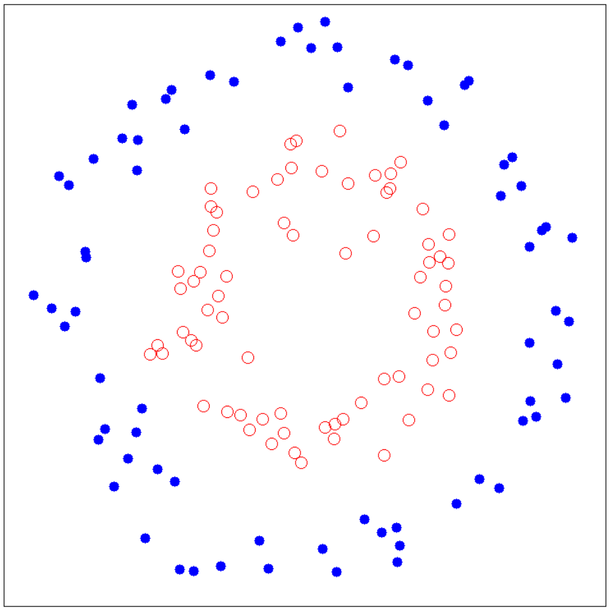
   - Class 1: Generate 1000 grayscale random noise images of size 128 × 128 using MatLab's `rand` function.
   - Class 2: Generate 1000 grayscale random noise images of size 128×128 using MatLab's `rand` function, and blur each image with the following 1-D separable blur filter $\begin{pmatrix} 1 & 4 & 6 & 4 & 1 \end{pmatrix}/16$ (see MatLab's `conv` function).
   - As a statistical feature vector compute the mean and variance of the result of convolving each image with the following 1-D high-pass filter $\begin{pmatrix} 1 & -2 & 1 \end{pmatrix}$ (after convolving and before computing the mean and variance, eliminate a few pixel boundary along the image border to avoid edge artifacts).
   - Train a LDA on 750 images from class 1 and class 2, and test on the remaining 250 images from each class.
   - Report the training and testing accuracy.
   - Display the 2-D feature vectors for the images in class 1 and 2 before and after projection onto the 1-D LDA projection axis.

Example 4.1 The eigenvectors $\mathbf{e}_j$ of the covariance matrix M^TM satisfy $M^TM\mathbf{e}_j = \lambda_j\mathbf{e}_j$. Left-multiplying this equality by M yields:

$$\begin{aligned} M(M^TM\mathbf{e}_j) &= \lambda_j M\mathbf{e}_j \\ MM^T(M\mathbf{e}_j) &= \lambda_j(M\mathbf{e}_j). \end{aligned}$$

That is, the vector $M\mathbf{e}_j$ is an eigenvector of the desired covariance matrix MM^T, where $\mathbf{e}_j$ is an eigenvector of M^TM.

Example 4.2 There are many possible configurations of data for which LDA will fail to separate the classes. Here is one example where the red open circles and blue filled circles denote different classes. Although these classes are perfectly separated in 2-D, any 1-D linear projection through this 2-D space will completely inter-mix data from the two classes.



1. H. Farid and M.J. Bravo. Perceptual Discrimination of Computer Generated and Photographic Faces. *Digital Investigation*, 2011 (in press).

2. S. Lyu and H. Farid. How realistic is photorealistic? *IEEE Transactions on Signal Processing*, 53(2):845850, 2005.

3. T-T Ng, S-F Chang, and M-P Tsui. Lessons learned from online classification of photorealistic computer graphics and photographs. In *IEEE Workshop on Signal Processing Applications for Public Security and Forensics*, 2007.

4. E.P. Simoncelli and E.H. Adelson. *Subband image coding*, chapter Subband transforms, pages 143192. Kluwer Academic, 1990.

5. Geometric-Based Forensics

5.1 Camera Model †

Under an ideal pinhole camera model, the perspective projection of arbitrary points **P** in 3-D world coordinates is given, in homogeneous coordinates, by:

$$\begin{aligned}
\begin{pmatrix} x \\ y \\ s \end{pmatrix} &= \lambda \begin{pmatrix} \alpha f & \beta & c_1 \\ 0 & f & c_2 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} R \mid \mathbf{t} \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} && (5.1) \\
\mathbf{p} &= \lambda K M \mathbf{P}, && (5.2)
\end{aligned}$$

where **p** is the 2-D projected point in homogeneous coordinates, which in non-homogeneous coordinates is $\begin{pmatrix} x/s & y/s \end{pmatrix}$, λ is a scale factor, K is the intrinsic matrix, and M is the extrinsic matrix. Within the extrinsic matrix, R is a 3×3 rotation matrix, and $\mathbf{t}$ is a 3×1 translation vector. Within the intrinsic matrix, f is the focal length, α is the aspect ratio, (c_1, c_2) is the principle point (the projection of the camera center onto the image plane), and β is the skew.

For simplicity, it is typically assumed that the pixels are square ($\alpha = 1, \beta = 0$). This is a reasonable assumptions for most modern-day cameras. With this assumption, the intrinsic matrix simplifies to:

$$K = \begin{pmatrix} f & 0 & c_1 \\ 0 & f & c_2 \\ 0 & 0 & 1 \end{pmatrix}. \qquad (5.3)$$

The camera model in Equation (5.2) specifies the perspective projection of arbitrary 3-D points from world to image coordinates. In the special case when the world points are constrained to a planar surface, the projection takes the form:

$$\begin{aligned}
\begin{pmatrix} x \\ y \\ s \end{pmatrix} &= \lambda \begin{pmatrix} f & 0 & c_1 \\ 0 & f & c_2 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} r_1 & r_2 & \mathbf{t} \end{pmatrix} \begin{pmatrix} X \\ Y \\ 1 \end{pmatrix} && (5.4) \\
\mathbf{p} &= \lambda K M \mathbf{P} && (5.5) \\
\mathbf{p} &= H \mathbf{P}, && (5.6)
\end{aligned}$$

where $\mathbf{p}$ is the 2-D projected point in homogeneous coordinates, and $\mathbf{P}$, in the appropriate coordinate system, is specified by 2-D coordinates in homogeneous coordinates. As before, λ is a scale factor, K is the intrinsic matrix, M is a now 3×3 extrinsic matrix in which $\mathbf{r}_1$, $\mathbf{r}_2$ and $\mathbf{r}_1 \times \mathbf{r}_2$ are the columns of the 3×3 rotation matrix that describes the transformation from world to camera coordinates, and as before, $\mathbf{t}$ is a 3×1 translation vector. The 3×3 matrix H, referred to as a homography, embodies the projection of a planar surface.

Example 5.1 Show that if Z is a constant in Equation (5.2), then this imaging model is the same as the model in Equation (5.6).

5.2 Calibration[†]

Recall from the previous section that a planar homography H is a scaled product of an intrinsic, K, and extrinsic, M, matrix: $H = \lambda KM$. It can be desirable to factor a homography into these components in order to determine the intrinsic camera parameters.

It is straightforward to show that $\mathbf{r}_1 = \frac{1}{\lambda} K^{-1}\mathbf{h}_1$ and $\mathbf{r}_2 = \frac{1}{\lambda} K^{-1}\mathbf{h}_2$ where $\mathbf{h}_1$ and $\mathbf{h}_2$ are the first two columns of the matrix H. The constraint that $\mathbf{r}_1$ and $\mathbf{r}_2$ are orthogonal (they are columns of a rotation matrix) and have the same norm (unknown due to the scale factor λ) yields two constraints on the unknown intrinsic matrix K:

$$\begin{aligned} \mathbf{r}_1^T \mathbf{r}_2 &= 0 \\ \mathbf{h}_1^T (K^{-T} K^{-1}) \mathbf{h}_2 &= 0, \qquad (5.7) \end{aligned}$$

and

$$\begin{aligned} \mathbf{r}_1^T \mathbf{r}_1 - \mathbf{r}_2^T \mathbf{r}_2 &= 0 \\ \mathbf{h}_1^T (K^{-T} K^{-1}) \mathbf{h}_1 - \mathbf{h}_2^T (K^{-T} K^{-1}) \mathbf{h}_2 &= 0. \qquad (5.8) \end{aligned}$$

With only two constraints, it is possible to estimate the principal point (c_1, c_2) or the focal length f, but not both. If, however, the focal length is known, then it is possible to estimate the principal point.

For notational simplicity we solve for the components of $Q = K^{-T} K^{-1}$, which contain the desired coordinates of the principal point and the assumed known focal length:

$$Q = \frac{1}{f^2} \begin{pmatrix} 1 & 0 & -c_1 \\ 0 & 1 & -c_2 \\ -c_1 & -c_2 & c_1^2 + c_2^2 + f^2 \end{pmatrix}. \qquad (5.9)$$

In terms of Q, the first constraint, Equation (5.7), takes the form:

$$\begin{aligned} h_1 h_2 + h_4 h_5 - (h_2 h_7 + h_1 h_8) c_1 - (h_5 h_7 + h_4 h_8) c_2 & \\ + h_7 h_8 (c_1^2 + c_2^2 + f^2) &= 0, \qquad (5.10) \end{aligned}$$

Note that this constraint is a second-order polynomial in the coordinates of the principal point, which can be factored as follows:

$$(c_1 - \alpha_1)^2 + (c_2 - \beta_1)^2 = \gamma_1^2, \qquad (5.11)$$

where:

$$\begin{aligned} \alpha_1 &= (h_2 h_7 + h_1 h_8)/(2 h_7 h_8), & (5.12) \\ \beta_1 &= (h_5 h_7 + h_4 h_8)/(2 h_7 h_8), & (5.13) \\ \gamma_1^2 &= \alpha_1^2 + \beta_1^2 - f^2 - (h_1 h_2 + h_4 h_5)/(h_7 h_8). & (5.14) \end{aligned}$$

Similarly, the second constraint, Equation (5.8), takes the form:

$$\begin{aligned} h_1^2 + h_4^2 + 2(h_2h_8 - h_1h_7)c_1 + 2(h_5h_8 - h_4h_7)c_2 & \\ -h_2^2 - h_5^2 + (h_7^2 - h_8^2)(c_1^2 + c_2^2 + f^2) &= 0, \end{aligned} \qquad (5.15)$$

or,

$$(c_1 - \alpha_2)^2 + (c_2 - \beta_2)^2 = \gamma_2^2, \qquad (5.16)$$

where:

$$\begin{aligned} \alpha_2 &= (h_1h_7 - h_2h_8)/(h_7^2 - h_8^2), & (5.17) \\ \beta_2 &= (h_4h_7 - h_5h_8)/(h_7^2 - h_8^2), & (5.18) \\ \gamma_2^2 &= \alpha_2^2 + \beta_2^2 - (h_1^2 + h_4^2 - h_2^2 - h_5^2)/(h_7^2 - h_8^2) - f^2. & (5.19) \end{aligned}$$

Both constraints, Equations (5.11) and (5.16) are circles in the desired coordinates of the principal point c_1 and c_2, and the solution is the intersection of the two circles.

For certain homographies this solution can be numerically unstable. For example, if $h_7 \approx 0$ or $h_8 \approx 0$, the first constraint becomes numerically unstable. Similarly, if $h_7 \approx h_8$, the second constraint becomes unstable. In order to avoid these instabilities, an error function with a regularization term can be introduced. We start with the following error function to be minimized:

$$E(c_1, c_2) = g_1(c_1, c_2)^2 + g_2(c_1, c_2)^2, \qquad (5.20)$$

where $g_1(c_1, c_2)$ and $g_2(c_1, c_2)$ are the constraints on the principal point given in Equations (5.10) and (5.15), respectively. To avoid numerical instabilities, a regularization term is added to penalize deviations of the principal point from the image center $(0, 0)$ (in normalized coordinates). This augmented error function takes the form:

$$E(c_1, c_2) = g_1(c_1, c_2)^2 + g_2(c_1, c_2)^2 + \Delta(c_1^2 + c_2^2), \qquad (5.21)$$

where Δ is a scalar weighting factor. This error function is a nonlinear least-squares problem, which can be minimized using a Levenberg-Marquardt iteration. The image center $(0, 0)$ is used as the initial condition for the iteration.

If, on the other hand, we assume that the principal point is the image center $(0, 0)$, then the focal length f can be estimated. In this case, the intrinsic matrix simplifies to:

$$K = \begin{pmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{pmatrix}, \qquad (5.22)$$

to yield a homography of the form:

$$H = \lambda \begin{pmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{pmatrix} \left(\mathbf{r}_1 \quad \mathbf{r}_2 \quad \mathbf{t} \right). \tag{5.23}$$

Left-multiplying by K^{-1} yields:

$$\begin{pmatrix} \frac{1}{f} & 0 & 0 \\ 0 & \frac{1}{f} & 0 \\ 0 & 0 & 1 \end{pmatrix} H = \lambda \left(\mathbf{r}_1 \quad \mathbf{r}_2 \quad \mathbf{t} \right) \tag{5.24}$$

$$\begin{pmatrix} \frac{1}{f} & 0 & 0 \\ 0 & \frac{1}{f} & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & h_9 \end{pmatrix} = \lambda \left(\mathbf{r}_1 \quad \mathbf{r}_2 \quad \mathbf{t} \right) \tag{5.25}$$

As before, because $\mathbf{r}_1$ and $\mathbf{r}_2$ are the first two columns of a rotation matrix, their inner product, $\mathbf{r}_1^T \cdot \mathbf{r}_2$, is zero, leading to the following constraint:

$$\left[\begin{pmatrix} \frac{1}{f} & 0 & 0 \\ 0 & \frac{1}{f} & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} h_1 \\ h_4 \\ h_7 \end{pmatrix} \right]^T \cdot \left[\begin{pmatrix} \frac{1}{f} & 0 & 0 \\ 0 & \frac{1}{f} & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} h_2 \\ h_5 \\ h_8 \end{pmatrix} \right] = 0 \tag{5.26}$$

$$\left(h_1 \quad h_4 \quad h_7 \right) \begin{pmatrix} \frac{1}{f^2} & 0 & 0 \\ 0 & \frac{1}{f^2} & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} h_2 \\ h_5 \\ h_8 \end{pmatrix} = 0. \tag{5.27}$$

The focal length is estimated by solving the above linear system for f:

$$f = \sqrt{-\frac{h_1 h_2 + h_4 h_5}{h_7 h_8}}. \tag{5.28}$$

The additional constraint that $\mathbf{r}_1$ and $\mathbf{r}_2$ are each unit length, $\mathbf{r}_1^T \cdot \mathbf{r}_1 = \mathbf{r}_2^T \cdot \mathbf{r}_2$, can also be used to estimate the focal length.

Example 5.2 The scale factor λ can be determined by exploiting the unit norm constraint on the columns of the rotation matrix. Describe how to estimate this scale factor.

If there is no relative rotation between the world and camera coordinate systems, then there is an inherent ambiguity between the world to camera translation in X and Y and the position of the principal point, and between the translation in Z (depth) and the focal length. As such, the factorization of the homography is not unique in the case of a fronto-parallel view.

5.3 Lens Distortion†

The imaging model described in the previous two sections assumes an idealized pinhole camera. In practice, however, cameras have multiple lenses that can deviate substantially from this model. Most significantly, lenses introduce geometric distortion whereby straight lines in the world appear curved in the image.

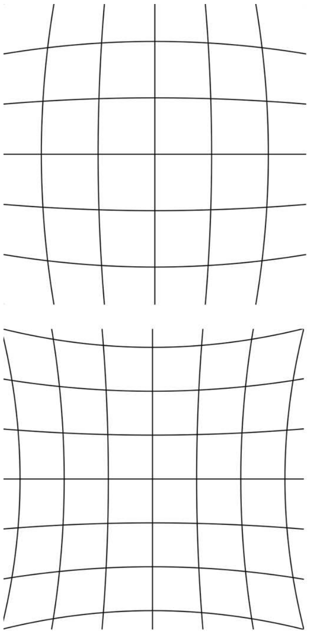


Figure 5.1 Barrel and pin cushion lens distortion.

Geometric lens distortions can be modeled with a one-parameter radially symmetric model. Given an ideal undistorted image $f_u(x,y)$, the distorted image is denoted as $f_d(\tilde{x},\tilde{y})$, where the distorted spatial parameters are given by:

$$\tilde{x} = x + \kappa x r^2 \quad \text{and} \quad \tilde{y} = y + \kappa y r^2, \tag{5.29}$$

where $r^2 = x^2 + y^2$, and κ controls the amount of distortion. Shown in Figure 5.1 are the results of distorting a rectilinear grid with a negative (barrel distortion) and positive (pincushion distortion) value of κ.

This model assumes that the center of the image coincides with the principal axis of the lens. If, however, this is not the case, then it is necessary to add additional parameters to the model to account for a spatial offset of the distortion center. This new model takes the form:

$$\tilde{x} = x + \kappa(x - c_x)r^2 \quad \text{and} \quad \tilde{y} = y + \kappa(y - c_y)r^2, \tag{5.30}$$

where $r^2 = (x - c_x)^2 + (y - c_y)^2$, and (c_x, c_y) corresponds to the center of the distortion (i.e., the principal point).

Lens distortion should be removed when considering the geometric techniques described in this chapter so that the image formation more closely matches Equation (5.2). Lens distortion can be manually estimated and removed by distorting an image according to Equation (5.29) or (5.30) until lines that are known to be straight in the world appear straight in the image. For the one-parameter model, this is relatively easy to do. For the three-parameter model, however, this manual calibration can be difficult and should be automated. This can be done by specifying curved lines in the image that are known to be straight in the world, and searching the three model parameters until the curved lines are mapped to straight lines.

Note that, unlike the 3×4 projection matrix in Equation (5.2), the 3×3 planar homography in Equation (5.6) is invertible. This implies that if the homography H can be estimated, then the original world coordinates $\mathbf{P}$ can be determined from the projected image coordinates $\mathbf{p}$.

It is straight-forward to see that $\mathbf{p} \times H\mathbf{P} = \mathbf{0}$, where $\times$ denotes cross product. Specifically, the cross product is defined as:

$$\mathbf{a} \times \mathbf{b} \quad = \quad \mathbf{n}\|a\|\|b\| \sin(\theta), \tag{5.31}$$

where $\mathbf{n}$ is mutually orthogonal to $\mathbf{a}$ and $\mathbf{b}$, and θ is the angle between $\mathbf{a}$ and $\mathbf{b}$. If $\mathbf{a} = \mathbf{b}$, then $\theta = 0$ and $\mathbf{a} \times \mathbf{b} = 0$. This identity yields the following:

$$\begin{aligned}
\mathbf{p} \times H\mathbf{P} &= \mathbf{0} &(5.32)\\
\begin{pmatrix} x \\ y \\ s \end{pmatrix} \times \begin{pmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & h_9 \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} &= \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} &(5.33)\\
\begin{pmatrix} x \\ y \\ s \end{pmatrix} \times \begin{pmatrix} h_1X + h_2Y + h_3Z \\ h_4X + h_5Y + h_6Z \\ h_7X + h_8Y + h_9Z \end{pmatrix} &= \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}, &(5.34)
\end{aligned}$$

where it can be assumed that $Z = 1$ and $s = 1$ because the homography will only be estimated to within an unknown scale factor. Evaluating the cross product on the left-hand side yields:

$$\begin{pmatrix} y(h_7X + h_8Y + h_9Z) - s(h_4X + h_5Y + h_6Z) \\ s(h_1X + h_2Y + h_3Z) - x(h_7X + h_8Y + h_9Z) \\ x(h_4X + h_5Y + h_6Z) - y(h_1X + h_2Y + h_3Z) \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \tag{5.35}$$

Re-ordering the terms yields a linear system $A\mathbf{h} = \mathbf{0}$, where the matrix A is:

$$\begin{pmatrix} 0 & 0 & 0 & -sX & -sY & -sZ & yX & yY & yZ \\ sX & sY & sZ & 0 & 0 & 0 & -xX & -xY & -xZ \\ -yX & -yY & -yZ & xX & xY & xZ & 0 & 0 & 0 \end{pmatrix}, \tag{5.36}$$

and:

$$\mathbf{h} = \begin{pmatrix} h_1 & h_2 & h_3 & h_4 & h_5 & h_6 & h_7 & h_8 & h_9 \end{pmatrix}^T \tag{5.37}$$

Given the known coordinates of a point, $\mathbf{P}$, on a plane in the world and its corresponding projected coordinates, $\mathbf{x}$, the above system seemingly provides three constraints in the nine unknowns of $\mathbf{h}$. Note, however, that the rows of the matrix A are not linearly independent (the third row is a linear combination of the first two rows). As such, this system provides only two constraints in the nine unknowns. Because the homography can only be estimated to within an unknown scale factor, the number of unknowns reduces from nine to eight.

As such, in order to solve for the projective transformation matrix H, four or more points with known coordinates $\mathbf{P}$ and $\mathbf{p}$ are required. The coordinates of these points are placed into the rows of matrix A to yield the following quadratic error function to be minimized:

$$E(\mathbf{h}) \quad = \quad \|A\mathbf{h}\|^2. \tag{5.38}$$

Note that minimizing this function using least-squares will lead to the degenerate solution $\mathbf{h} = 0$. In order to avoid this degenerate solution we constrain $\mathbf{h}$ to have unit sum $\|\mathbf{h}\|^2 = 1$ (hence the scale ambiguity in estimating the homography). This added constraint yields a total least-squares optimization. The optimal unit vector $\mathbf{h}$ is the minimal eigenvalue eigenvector of $A^T A$.

With a known projective transformation matrix H, an image can be warped according to H^{-1} to yield a rectified image, Figure 5.2.

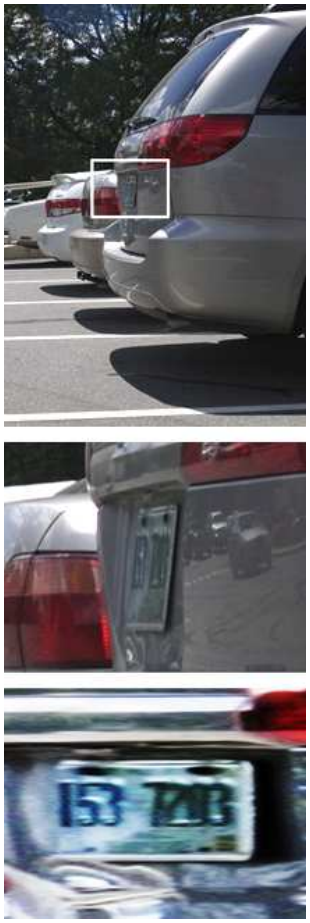


Figure 5.2 A an original photo (top), a magnified view of the license plate (middle), and the planar rectified license plate (bottom).

Although the estimation of H is straight-forward, there are a few implementation details that should be considered. For the sake of numerical stability, it is recommended that the image coordinates $\mathbf{p}$ and world coordinates $\mathbf{P}$ are transformed so that their respective centroids are at the origin and that their respective mean distance from the origin is $\sqrt{2}$. In homogeneous coordinates, this transformation matrix takes the form:

$$T \quad = \quad \begin{pmatrix} \alpha & 0 & -\alpha c_1 \\ 0 & \alpha & -\alpha c_2 \\ 0 & 0 & 1 \end{pmatrix}, \tag{5.39}$$

where α is the multiplicative scaling and c_1 and c_2 are the additive offsets. The homography H estimated using these normalized coordinates is then transformed back to the original coordinates as $T_1^{-1} H T_2$, where T_1 and T_2 are the transformation matrices for the image and world coordinates, respectively.

Example 5.3 Show that after the image and world coordinates are transformed by T_1 and T_2, Equation (5.39), the estimated homography H should be transformed by $T_1^{-1} H T_2$.

When creating a photo composite it is often necessary to translate, scale, rotate, etc. portions of an image. Such image-based manipulations can change the effective intrinsic camera parameters. Therefore, differences in these estimated parameters can be used as evidence of tampering.

For example, a translation in the image is equivalent to translating the camera's principal point. In homogeneous coordinates, translation is represented by multiplication with a matrix T. With a horizontal and vertical translation in the image of (d_1, d_2), the mapping of planar points in the world $\mathbf{P}$ to (translated) image coordinates $\mathbf{p}$ is:

$$\begin{aligned}
\mathbf{p} &= TH\mathbf{P} \\
&= \lambda TKM\mathbf{P} \\
&= \lambda \begin{pmatrix} 1 & 0 & d_1 \\ 0 & 1 & d_2 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} f & 0 & c_1 \\ 0 & f & c_2 \\ 0 & 0 & 1 \end{pmatrix} M\mathbf{P} \\
&= \lambda \begin{pmatrix} f & 0 & c_1 + d_1 \\ 0 & f & c_2 + d_2 \\ 0 & 0 & 1 \end{pmatrix} M\mathbf{P}. \qquad (5.40)
\end{aligned}$$

That is, translation in the image is equivalent to translating the principal point. As such this transformation can be detected by estimating the principal point for different parts of an image, from which any inconsistencies can be used as evidence of tampering.

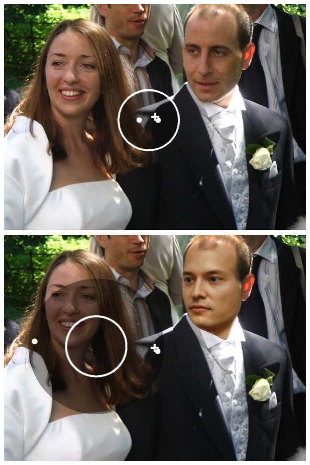


Figure 5.3 An authentic and fake photo – the white dots denote the estimated principal points for each person.

Shown in Figure 5.3 is an authentic (top) and fake (bottom) photo. The plus sign denotes the center of the image, and the two small white dots denote the estimated principal point for each person. In the authentic photo, the principal points are similar and near the image center, while in the fake photo, the principal point for the man is significantly different than for the woman.

Although planar reflecting surfaces do not appear with great frequency in natural environments (aside from lakes and other still bodies of water), many man-made environments feature mirrors, windows, computer screens, tabletops, and other flat specular surfaces that create reflections. We next consider the nature of the geometric inconsistencies that arise when fake reflections are inserted into a photo or when a photo containing reflections is manipulated.

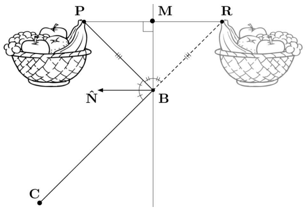


Figure 5.4 Geometry of reflections.

A ray from an observer, **C**, toward a point, **R**, in the observed reflection will strike the reflecting surface at some point, **B**, and a ray from that point to the point, **P**, on the reflected object will be symmetric about the surface normal with the ray from the observer, Figure 5.4. A ray between the reflected object and the apparent location of its reflection must appear to be perpendicular to the reflecting surface. To see why, consider that to the observer it appears as if the ray toward the reflecting surface continues through the surface and strikes a virtual object behind the mirror. The isosceles triangle formed by the points **R**, **P**, and **B** is bisected by the plane of the reflecting surface at **B** and therefore the line containing **R** and **P** must be perpendicular to that plane.

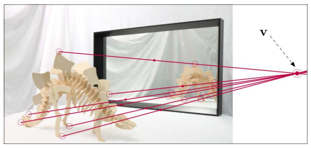


Figure 5.5 Reflection vanishing point.

The above reasoning applies for all points on reflected objects and it implies that for a given planar reflecting surface, the set of lines connecting points on objects to the corresponding points on their reflections will all appear as if they were rays in the three-dimensional scene that are perpendicular to the surface and therefore mutually parallel. Assuming that the image of the scene was created by linear perspective projection, the image of these parallel lines will form a bundle that converges to a common vanishing point, **v**, Figure 5.5. In the special case where the reflecting surface is parallel to the view-plane normal then the lines will remain parallel in the image and the vanishing point for these parallel lines will be at infinity in the direction defined by the lines.

Lines in the image between a point on an object and its reflection in the surface must all converge to the same point. Further, this vanishing point is the vanishing point for lines perpendicular to the surface and any scene elements that are known to be perpendicular to the reflecting surface must also be consistent with this vanishing point.

Shown in Figure 5.6 are an authentic and fake photo. All three dinosaurs share a common vanishing point in the authentic photo. The fake photo was created by combining individual photographs where the mirror's position was varied. As a result, each object has a well-defined reflection vanishing point, but they are mutually inconsistent. The reflections of the herbivorous dinosaurs correspond to the pictured mirror location. The carnivorous dinosaurs reflection corresponds to a mirror location that is more rotated toward the viewer.

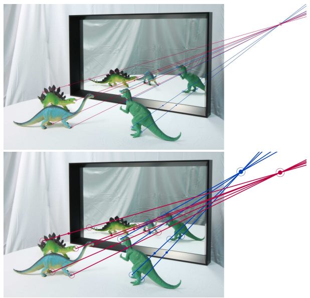


Figure 5.6 An authentic and fake photo with consistent and inconsistent vanishing points.

Since light travels in a straight line, a point on a shadow, its corresponding point on the object, and the light source all lie on a single line. Therefore, the light source will always lie on a line that connects every point on a shadow with its corresponding point on an object, regardless of scene geometry. In an image, the projection of these lines will always intersect at the 2-D projection of the light position.

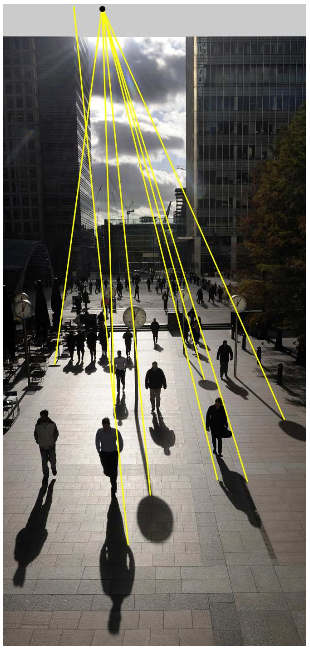


Figure 5.7
A consistent and an inconsistent shadow.

Shown in Figure 5.7 is a photo in which seven lines connect points on a shadow with their corresponding points on the object. All of these lines intersect at a single point, which in this case can clearly be seen to be the location of the sun. Also shown in this figure is a clearly fake shadow where the line deviates significantly from the intersection point computed from neighboring shadows. Although all of the shadows in this example are cast onto a single planar surface, this is not required to perform this simple shadow analysis: cast shadows on any surface must all intersect at the 2-D projection of the light position. In addition, depending on the orientation of the camera relative to the scene, the projected light position can be below the ground plane.

In practice, there are some limitations to this geometric analysis of light position. Care must be taken to select appropriately matched points on the shadow and the object; this is best achieved when the object has a distinct shape (e.g., the tip of a cone). If the dominant light is the sun, then the lines may be nearly parallel, making the computation of their intersection vulnerable to numerical instability.

It is natural to question if geometric and mathematical analyses are required to detect inconsistencies in reflections. After all, the visual system is remarkable, capable of hyperacuity, rapid scene understanding, and robust face recognition. We will show, however, that the visual system can be quite inept at judging the consistency and inconsistency of reflections.

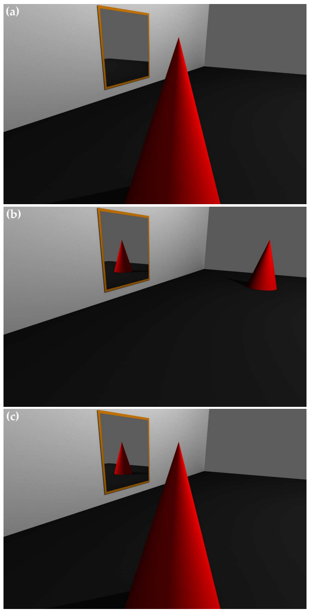


Figure 5.8 Consistent (a)-(b) and inconsistent (c) reflections.

Shown in the Figure 5.8(a) is a rendered 3-D scene containing a red cone and a mirror. Panel (b) of this figure is the same scene with the cone displaced relative to the mirror. Panel (c) is a composite created by replacing the correct reflection in panel (a) with that from panel (b) to create a physically impossible scene.

Three-dimensional rendered scenes were generated such that the reflection was either consistent or inconsistent with the scene geometry. The scenes were rendered with the viewer in one of three locations relative to the reflective mirror, either 10° (nearly frontoparallel) or ±60° relative to the mirror. For each viewing direction, the object (red cone) was moved to one of three locations along the ground plane. The inconsistent scenes were generated by combining the reflection from one scene with the object from another, always taken from scenes with the same viewing direction. Twenty observers were each presented with these scenes (14 consistent and 28 inconsistent) and given unlimited time to determine if the reflection in each was correct. The average accuracy over all viewing conditions was only 55.7%, slightly better than chance. The average response time was 7.6 seconds, indicating that observers spent a reasonable amount of time inspecting each scene. These experiments reveal that the visual system is quite inept at judging the consistency of reflections.

It is also natural to question if geometric and mathematical analyses are required to detect inconsistencies in shadows. We will show that as with reflections, the visual system can be quite inept at judging the consistency and inconsistency of shadows.

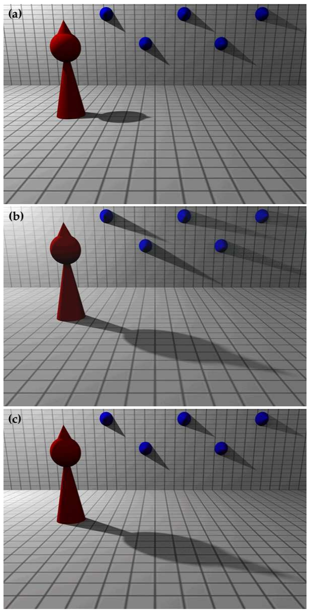


Figure 5.9 Consistent (a)-(b) and inconsistent (c) shadows.

Shown in Figure 5.9(a) is a rendered 3-D scene illuminated by a single light that produces cast shadows on the ground plane and back wall. Panel (b) of this figure is the same scene with the light moved to a different location. Panel (c) is a composite created by combining the back wall from panel (a) and the ground plane from panel (b) to create a scene with shadows that are inconsistent with a single light.

One hundred and forty rendered scenes were created such that the cast shadows were either consistent or inconsistent with a single light. For the consistent scenes, the light was positioned either on the left or right side of the room and in one of nine different locations that varied in distance from the ground plane and from the back wall. For the inconsistent scenes, the back walls from scenes with different lighting were interchanged. Twenty observers were each given unlimited time to judge whether the original and composite scenes were consistent with a single light. Their performance was nearly perfect (95.5%) for inconsistent scenes that were the combination of lights from opposites sides of the room (i.e., the cast shadows ran in opposite directions). For all other cases, however, observer accuracy was near chance (52.8%). The average response time was 4.9 seconds, indicating that observers spent a reasonable amount of time inspecting each scene. These experiments reveal that the visual system is quite inept at judging the consistency of cast shadows.

1. Write MatLab code that computes vanishing points from three or more lines. Specifically, your code should:

    - Display an image and allow the user to select any number of lines in the image.
    - Each line can be specified by two user selected points.
    - With three or more lines it is highly unlikely that they will all intersect at a single vanishing point.
    - Your code should, therefore, determine the vanishing point by optimizing an objective function that minimizes the deviation of all pairs of line intersections from their center of mass, while minimizing the deviation of the user selected points from their initial values.
    - In other words, allow the user selected lines to wiggle a little bit (but not too much) in order to obtain a well defined vanishing point.
    - You may use MatLab's built-in optimization `fminsearch`.
    - Compute the vanishing points for each of the images found on the course webpage:

        www.cs.dartmouth.edu/farid/Hany_Farid/Teaching/Teaching.html

Example 5.1 Without loss of generality, assume that the intrinisc matrix K is the identity matrix, and that the overall scale factor $\lambda = 1$. The perspective projection model of Equation (5.2), in non-homogeneous coordinates, takes the form:

$$\begin{aligned} \mathbf{p}_i &= R\mathbf{P}_i + \mathbf{T} \\ \mathbf{p}_i &= R \begin{pmatrix} X_i \\ Y_i \\ Z_i \end{pmatrix} + \mathbf{T}. \end{aligned}$$

If Z is a constant c then:

$$\begin{aligned} \mathbf{p}_i &= R \begin{pmatrix} X_i \\ Y_i \\ 0 \end{pmatrix} + R \begin{pmatrix} 0 \\ 0 \\ c \end{pmatrix} + \mathbf{T} \\ &= \begin{pmatrix} \mathbf{r}_1 & \mathbf{r}_2 & \mathbf{r}_3 \end{pmatrix} \begin{pmatrix} X_i \\ Y_i \\ 0 \end{pmatrix} + \begin{pmatrix} \mathbf{r}_1 & \mathbf{r}_2 & \mathbf{r}_3 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ c \end{pmatrix} + \mathbf{T} \\ &= \begin{pmatrix} \mathbf{r}_1 & \mathbf{r}_2 & \mathbf{r}_3 \end{pmatrix} \begin{pmatrix} X_i \\ Y_i \\ 0 \end{pmatrix} + \begin{pmatrix} T_x \\ T_y \\ Tz + c' \end{pmatrix} \end{aligned}$$

where $c' = \mathbf{r}_3 \cdot (\, 0 \quad 0 \quad c \,)$. The third term of the world points $\mathbf{P}_i$ are all zero, therefore the third column of the rotation matrix R is irrelevant, yielding:

$$\mathbf{p}_i = \begin{pmatrix} \mathbf{r}_1 & \mathbf{r}_2 \end{pmatrix} \begin{pmatrix} X_i \\ Y_i \end{pmatrix} + \mathbf{T}',$$

where $\mathbf{T}' = (\, T_x \quad T_y \quad Tz + c' \,)^T$. Rewriting in homogeneous equations yields:

$$\mathbf{p}_i = \begin{pmatrix} \mathbf{r}_1 & \mathbf{r}_2 & \mathbf{T}' \end{pmatrix} \begin{pmatrix} X_i \\ Y_i \\ 1 \end{pmatrix},$$

which is the same imaging model as in Equation (5.6).

Example 5.2 Recall the relationship between the first column of the rotation matrix $\mathbf{r}_1$ and the scale factor λ, the intrinsic matrix K, and the first column of the homography $\mathbf{h}_1$:

$$\frac{1}{\lambda} K^{-1} \mathbf{h}_1 = \mathbf{r}_1.$$

Imposing the unit norm constraint on $\mathbf{r}_1$ yields:

$$\begin{aligned} 1/\lambda \| K^{-1}\mathbf{h}_1 \| &= \| \mathbf{r}_1 \| \\ 1/\lambda \| K^{-1}\mathbf{h}_1 \| &= 1 \\ \lambda &= \| K^{-1}\mathbf{h}_1 \|. \end{aligned}$$

Example 5.3 We seek to estimate the homography H that describes the transformation between image $\mathbf{p}$ and world $\mathbf{P}$ coordinates:

$$\mathbf{p} = H\mathbf{P}.$$

The normalization of the image and world points, however, means that we are estimating the following homography:

$$T_1\mathbf{p} = HT_2\mathbf{P}.$$

The desired homography is therefore simply:

$$\mathbf{p} = (T_1^{-1}HT_2)\mathbf{P}.$$

5.12 Readings

1. H. Farid and M.J. Bravo. Image forensic analyses that elude the human visual system. *In SPIE Symposium on Electronic Imaging*, San Jose, CA, 2010.

2. R. Hartley. In defense of the eight-point algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(6), 1997.

3. R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision.* Cambridge University Press, 2004.

4. M.K. Johnson and H. Farid. Detecting photographic composites of people. In *6th International Workshop on Digital Watermarking*, Guangzhou, China, 2007.

6. Physics-Based Forensics

6.1 2-D Lighting

Consider the creation of a forgery showing two movie stars, rumored to be romantically involved, walking down a sunset beach. Such an image might be created by splicing together individual images of each movie star. In so doing, it is often difficult to exactly match the lighting effects due to directional lighting (e.g., the sun on a clear day). Differences in lighting can, therefore, be a telltale sign of tampering. To the extent that the direction of the light source can be estimated for different objects/people in an image, inconsistencies in the lighting direction can be used as evidence of tampering.

The standard approaches for estimating light source direction begin by making some simplifying assumptions: (1) the surface of interest is Lambertian (the surface reflects light isotropically); (2) the surface has a constant reflectance value; (3) the surface is illuminated by a point light source infinitely far away; and (4) the angle between the surface normal and the light direction is between -90° and 90°. Under these assumptions, the image intensity can be expressed as:

$$I(x,y) \quad = \quad R(\mathbf{N}^T(x,y) \cdot \mathbf{L}) + A, \tag{6.1}$$

where R is the constant reflectance value, $\mathbf{L}$ is a 3-vector pointing in the direction of the light source, $\mathbf{N}(x,y)$ is a 3-vector representing the surface normal at the point (x,y), and A is a constant ambient light term. The intuition behind the contribution of the dot product between $\mathbf{N}$ and $\mathbf{L}$ is that the amount of light that strikes a surface is proportional to the angle between the surface normal and light direction.

If we are only interested in the direction of the light source, then the reflectance term, R, can be considered to have unit-value, understanding that the estimation of $\mathbf{L}$ will only be within an unknown scale factor. The resulting linear equation provides a single constraint in four unknowns, the three components of $\mathbf{L}$ and the ambient term A.

With at least four points with the same reflectance, R, and distinct surface normals, $\mathbf{N}$, the light source direction and ambient term can be solved for using standard least-squares estimation. To begin, a quadratic error function, embodying the imaging model of Equation (6.1), is given by:

$$
\begin{aligned}
E(\mathbf{L}, A) &= \left\| M \begin{pmatrix} L_x \\ L_y \\ L_z \\ A \end{pmatrix} - \begin{pmatrix} I(x_1, y_1) \\ I(x_2, y_2) \\ \vdots \\ I(x_p, y_p) \end{pmatrix} \right\|^2 && (6.2) \\
&= \left|\left| M\mathbf{v} - \mathbf{b} \right|\right|^2, && (6.3)
\end{aligned}
$$

where $||\cdot||$ denotes vector norm, L_x, L_y, and L_z denote the components of the light source direction $\mathbf{L}$, and

$$
M = \begin{pmatrix} N_x(x_1, y_1) & N_y(x_1, y_1) & N_z(x_1, y_1) & 1 \\ N_x(x_2, y_2) & N_y(x_2, y_2) & N_z(x_2, y_2) & 1 \\ \vdots & \vdots & \vdots & \vdots \\ N_x(x_p, y_p) & N_y(x_p, y_p) & N_z(x_p, y_p) & 1 \end{pmatrix}, \qquad (6.4)
$$

where $N_x(x_i, y_i)$, $N_y(x_i, y_i)$, and $N_z(x_i, y_i)$ denote the components of the surface normal $\mathbf{N}$ at image coordinate (x_i, y_i). The quadratic error function above is minimized by differentiating with respect to the unknown, $\mathbf{v}$, setting the result equal to zero, and solving for $\mathbf{v}$ to yield the least-squares estimate:

$$
\mathbf{v} \;=\; (M^T M)^{-1} M^T \mathbf{b}. \qquad (6.5)
$$

Note that this solution requires knowledge of 3-D surface normals from at least four distinct points ($p \geq 4$) on a surface with the same reflectance. With only a single image and no objects of known geometry in the scene, it is not immediately clear from where these normals will be determined.

Under an assumption of orthographic projection, the z-component of the surface normal is zero along the occluding contour of an object, $N_z = 0$. In addition, the x- and y-components of the surface normal, N_x and N_y, can be estimated directly from the image. In this case, the error function of Equation (6.3) takes the form:

$$
\begin{aligned}
E(\mathbf{L}, A) &= \left\| M \begin{pmatrix} L_x \\ L_y \\ A \end{pmatrix} - \begin{pmatrix} I(x_1, y_1) \\ I(x_2, y_2) \\ \vdots \\ I(x_p, y_p) \end{pmatrix} \right\|^2 && (6.6) \\
&= \left|\left| M\mathbf{v} - \mathbf{b} \right|\right|^2, && (6.7)
\end{aligned}
$$

where,

$$M = \begin{pmatrix} N_x(x_1,y_1) & N_y(x_1,y_1) & 1 \\ N_x(x_2,y_2) & N_y(x_2,y_2) & 1 \\ \vdots & \vdots & \vdots \\ N_x(x_p,y_p) & N_y(x_p,y_p) & 1 \end{pmatrix} \tag{6.8}$$

This error function is minimized, as before, using standard least-squares to yield the same solution as in Equation (6.5), but with the matrix M taking the form given in Equation (6.8). In this case, the solution requires knowledge of 2-D surface normals from at least three distinct points ($p \geq 3$) on a surface with the same reflectance.

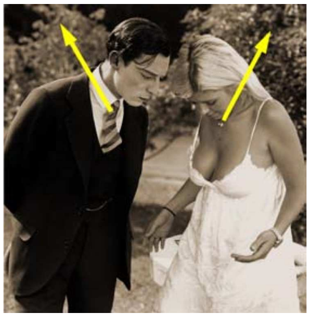


Figure 6.1 Light direction (yellow arrow) is inconsistent.

Example 6.1 Consider a circle parametrized as $c(\theta) = (\, r\cos(\theta) \quad r\sin(\theta)\,)$. Give an expression for the 2-D surface normal at any point on the circle.

Shown in Figure 6.1 is a composite photo from which the 2-D lighting direction was estimated for each person (yellow arrows). The light direction for the man (Buster Keaton) is 120° and for the woman (Tara Reid) is 62°. Shown in Figure 6.2 is a photo of Supermen in which the lighting is consistent, although the photo is fake: inconsistent lighting is good evidence of fakery, but consistent lighting does not necessarily imply authenticity.

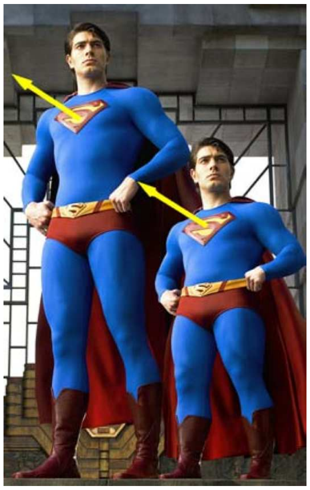


Figure 6.2 Light direction (yellow arrow) is consistent across Supermen.

The lighting of a scene can be complex – any number of lights can be placed in any number of positions, creating different lighting environments.

Under the assumption of distant lighting, an arbitrary lighting environment can be expressed as a non-negative function on the sphere, $L(\mathbf{V})$, where $\mathbf{V}$ is a unit vector in Cartesian coordinates and the value of $L(\mathbf{V})$ is the intensity of the incident light along direction $\mathbf{V}$. If the object being illuminated is convex, the irradiance (light received) at any point on the surface is due to only the lighting environment; i.e., there are no cast shadows or interreflections. As a result, the irradiance, $E(\mathbf{N})$, can be parametrized by the unit length surface normal $\mathbf{N}$ and written as a convolution of the reflectance function of the surface, $R(\mathbf{V}, \mathbf{N})$, with the lighting environment $L(\mathbf{V})$:

$$E(\mathbf{N}) \quad = \quad \int_{\Omega} L(\mathbf{V})\, R(\mathbf{V}, \mathbf{N})\; d\Omega, \tag{6.9}$$

where Ω represents the surface of the sphere and $d\Omega$ is an area differential on the sphere. For a Lambertian surface, the reflectance function is a clamped cosine:

$$R(\mathbf{V}, \mathbf{N}) \quad = \quad \max\left(\mathbf{V} \cdot \mathbf{N}, 0\right), \tag{6.10}$$

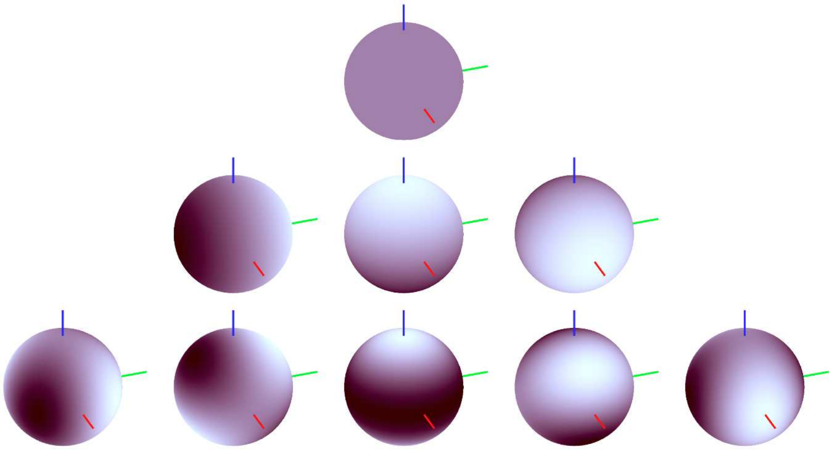
which is either the cosine of the angle between vectors $\mathbf{V}$ and $\mathbf{N}$, or zero when the angle is greater than 90 degrees. This reflectance function effectively limits the integration in Equation (6.9) to the hemisphere about the surface normal $\mathbf{N}$. In addition, while we have assumed no cast shadows, Equation (6.10) explicitly models attached shadows, i.e., shadows due to surface normals facing away from the direction $\mathbf{V}$.

The convolution in Equation (6.9) can be simplified by expressing both the lighting environment and the reflectance function in terms of spherical harmonics. Spherical harmonics form an orthonormal basis for piecewise continuous functions on the sphere and are analogous to the Fourier basis on the line or plane. The first three orders of spherical harmonics are defined in terms of the surface normal $\mathbf{N} = (\, x \quad y \quad z\,)$ of a sphere:

$$\begin{aligned} Y_{0,0}(\mathbf{N}) \quad &= \quad \frac{1}{\sqrt{4\pi}} \\ Y_{1,-1}(\mathbf{N}) \quad &= \quad \sqrt{\frac{3}{4\pi}} y \end{aligned}$$

$$
\begin{aligned}
Y_{1,0}(\mathbf{N}) &= \sqrt{\frac{3}{4\pi}}z \\
Y_{1,1}(\mathbf{N}) &= \sqrt{\frac{3}{4\pi}}x \\
Y_{2,-2}(\mathbf{N}) &= 3\sqrt{\frac{5}{12\pi}}xy \\
Y_{2,-1}(\mathbf{N}) &= 3\sqrt{\frac{5}{12\pi}}yz \\
Y_{2,0}(\mathbf{N}) &= \frac{1}{2}\sqrt{\frac{5}{4\pi}}(3z^2-1) \\
Y_{2,1}(\mathbf{N}) &= 3\sqrt{\frac{5}{12\pi}}xz \\
Y_{2,2}(\mathbf{N}) &= \frac{3}{2}\sqrt{\frac{5}{12\pi}}(x^2-y^2).
\end{aligned}
$$

Shown below are the first three orders of spherical harmonics: $Y_{0,0}$ (row 1), $Y_{1,*}$ (row 2), and $Y_{2,*}$ (row 3).



The lighting environment expanded in terms of spherical harmonics is:

$$L(\mathbf{V}) = \sum_{n=0}^{\infty}\sum_{m=-n}^{n} l_{n,m}Y_{n,m}(\mathbf{V}), \qquad (6.11)$$

where $Y_{n,m}(\cdot)$ is the m^{th} spherical harmonic of order n, and $l_{n,m}$ is the corresponding coefficient of the lighting environment. Similarly, the reflectance function for Lambertian surfaces, $R(\mathbf{V},\mathbf{N})$, can be expanded in terms of spherical harmonics, and due to its symmetry about the surface normal, only harmonics with $m = 0$ appear in the expansion:

$$R(\mathbf{V},\mathbf{N}) = \sum_{n=0}^{\infty} r_n Y_{n,0}\left(\begin{pmatrix} 0 & 0 & \mathbf{V}\cdot\mathbf{N} \end{pmatrix}^T\right). \qquad (6.12)$$

Note that for $m = 0$, the spherical harmonic $Y_{n,0}(\cdot)$ depends only on the z-component of its argument.

Convolutions of functions on the sphere become products when represented in terms of spherical harmonics. As a result, the irradiance, Equation (6.9), takes the form:

$$E(\mathbf{N}) = \sum_{n=0}^{\infty} \sum_{m=-n}^{n} \hat{r}_n l_{n,m} Y_{n,m}(\mathbf{N}), \tag{6.13}$$

where

$$\hat{r}_n = \sqrt{\frac{4\pi}{2n+1}} r_n. \tag{6.14}$$

A key observation is that the coefficients $\hat{r}_n$ for a Lambertian reflectance function decay rapidly, and thus the infinite sum in Equation (6.13) can be well approximated by the first nine terms:

$$E(\mathbf{N}) \approx \sum_{n=0}^{2} \sum_{m=-n}^{n} \hat{r}_n l_{n,m} Y_{n,m}(\mathbf{N}). \tag{6.15}$$

Since the constants $\hat{r}_n$ are known for a Lambertian reflectance function, the irradiance of a convex Lambertian surface under arbitrary distant lighting can be well modeled by the nine lighting environment coefficients $l_{n,m}$ up to order two.

Irradiance describes the total amount of light reaching a point on a surface. For a Lambertian surface, the reflected light, or radiosity, is proportional to the irradiance by a reflectance term ρ. In addition, Lambertian surfaces emit light uniformly in all directions, so the amount of light received by a viewer (i.e., camera) is independent of the view direction.

A camera maps its received light to intensity through a camera response function $f(\cdot)$. Assuming the reflectance term ρ is constant across the surface, the measured intensity at a point $\mathbf{x}$ in the image is given by:

$$I(\mathbf{x}) = f(\rho t E(\mathbf{N}(\mathbf{x}))). \tag{6.16}$$

where $E(\cdot)$ is the irradiance, $\mathbf{N}(\mathbf{x})$ is the surface normal at point $\mathbf{x}$, and t is the exposure time. For simplicity, we assume a linear camera response, and thus the intensity is related to the irradiance by an unknown multiplicative factor, which is assumed to have unit value—this assumption implies that the lighting coefficients can only be estimated to within an unknown scale factor. Under these assumptions, the relationship between image intensity and irradiance is simply:

$$I(\mathbf{x}) \quad = \quad E(\mathbf{N}(\mathbf{x})). \tag{6.17}$$

Since, under our assumptions, the intensity is equal to irradiance, Equation (6.17) can be written in terms of spherical harmonics by expanding Equation (6.15):

$$\begin{aligned} I(\mathbf{x}) &= l_{0,0}\pi Y_{0,0}(\mathbf{N}) + l_{1,-1}\tfrac{2\pi}{3}Y_{1,-1}(\mathbf{N}) + l_{1,0}\tfrac{2\pi}{3}Y_{1,0}(\mathbf{N}) \\ &+ l_{1,1}\tfrac{2\pi}{3}Y_{1,1}(\mathbf{N}) + l_{2,-2}\tfrac{\pi}{4}Y_{2,-2}(\mathbf{N}) + l_{2,-1}\tfrac{\pi}{4}Y_{2,-1}(\mathbf{N}) \\ &+ l_{2,0}\tfrac{\pi}{4}Y_{2,0}(\mathbf{N}) + l_{2,1}\tfrac{\pi}{4}Y_{2,1}(\mathbf{N}) + l_{2,2}\tfrac{\pi}{4}Y_{2,2}(\mathbf{N}). \end{aligned} \tag{6.18}$$

Note that this expression is linear in the nine lighting environment coefficients, $l_{0,0}$ to $l_{2,2}$. As such, given 3-D surface normals at $p \geq 9$ points on the surface of an object, the lighting environment coefficients can be estimated as the least-squares solution to the system $M\mathbf{v} = \mathbf{b}$, where the matrix M contains the sampled spherical harmonics, $\mathbf{v}$ is the vector of unknown lighting environment coefficients, and $\mathbf{b}$ is the vector of intensities at p points:

$$\mathbf{v} = \begin{pmatrix} l_{0,0} & l_{1,-1} & \dots & l_{2,2} \end{pmatrix}^T \tag{6.19}$$

$$\mathbf{b} = \begin{pmatrix} I(\mathbf{x}_1) & I(\mathbf{x}_2) & \dots & I(\mathbf{x}_p) \end{pmatrix}^T, \tag{6.20}$$

and

$$M = \begin{pmatrix} \pi Y_{0,0}(\mathbf{N}(\mathbf{x}_1)) & \frac{2\pi}{3}Y_{1,-1}(\mathbf{N}(\mathbf{x}_1)) & \dots & \frac{\pi}{4}Y_{2,2}(\mathbf{N}(\mathbf{x}_1)) \\ \pi Y_{0,0}(\mathbf{N}(\mathbf{x}_2)) & \frac{2\pi}{3}Y_{1,-1}(\mathbf{N}(\mathbf{x}_2)) & \dots & \frac{\pi}{4}Y_{2,2}(\mathbf{N}(\mathbf{x}_2)) \\ \vdots & \vdots & \ddots & \vdots \\ \pi Y_{0,0}(\mathbf{N}(\mathbf{x}_p)) & \frac{2\pi}{3}Y_{1,-1}(\mathbf{N}(\mathbf{x}_p)) & \dots & \frac{\pi}{4}Y_{2,2}(\mathbf{N}(\mathbf{x}_p)) \end{pmatrix}. \tag{6.21}$$

The least-squares solution to this system is:

$$\mathbf{v} \quad = \quad \left(M^T M\right)^{-1} M^T\mathbf{b}. \tag{6.22}$$

This solution requires 3-D surface normals from at least nine points on the surface of an object. Without multiple images or known geometry, however, this requirement may be difficult to satisfy from an arbitrary image.

Under an assumption of orthographic projection, the z-component of the surface normal is zero along the occluding contour of an object, $N_z = 0$. Therefore, the intensity profile along an occluding contour simplifies to:

$$\begin{aligned} I(\mathbf{x}) &= A + l_{1,-1}\tfrac{2\pi}{3}Y_{1,-1}(\mathbf{N}) + l_{1,1}\tfrac{2\pi}{3}Y_{1,1}(\mathbf{N}) \\ &+ l_{2,-2}\tfrac{\pi}{4}Y_{2,-2}(\mathbf{N}) + l_{2,2}\tfrac{\pi}{4}Y_{2,2}(\mathbf{N}), \end{aligned} \tag{6.23}$$

where:

$$A \quad = \quad l_{0,0}\frac{\pi}{2\sqrt{\pi}} - l_{2,0}\frac{\pi}{16}\sqrt{\frac{5}{\pi}}. \tag{6.24}$$

Note that the functions $Y_{i,j}(\cdot)$ depend only on the x and y components of the surface normal $\mathbf{N}$. That is, the five lighting coefficients can be estimated from only 2-D surface normals, which are relatively simple to estimate from a single image. In addition, Equation (6.23) is still linear in its now five lighting environment coefficients, which can be estimated as the least-squares solution to $M\mathbf{v} = \mathbf{b}$, where:

$$\mathbf{v} \quad = \quad (\, A \quad l_{1,-1} \quad l_{1,1} \quad l_{2,-2} \quad l_{2,2}\,)^T \tag{6.25}$$

$$\mathbf{b} \quad = \quad (\, I(\mathbf{x}_1) \quad I(\mathbf{x}_2) \quad \dots \quad I(\mathbf{x}_p)\,)^T \tag{6.26}$$

and where M is:

$$\begin{pmatrix} 1 & \frac{2\pi}{3}Y_{1,-1}(\mathbf{N}(\mathbf{x}_1)) & \frac{2\pi}{3}Y_{1,1}(\mathbf{N}(\mathbf{x}_1)) & \frac{\pi}{4}Y_{2,-2}(\mathbf{N}(\mathbf{x}_1)) & \frac{\pi}{4}Y_{2,2}(\mathbf{N}(\mathbf{x}_1)) \\ 1 & \frac{2\pi}{3}Y_{1,-1}(\mathbf{N}(\mathbf{x}_2)) & \frac{2\pi}{3}Y_{1,1}(\mathbf{N}(\mathbf{x}_2)) & \frac{\pi}{4}Y_{2,-2}(\mathbf{N}(\mathbf{x}_2)) & \frac{\pi}{4}Y_{2,2}(\mathbf{N}(\mathbf{x}_2)) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & \frac{2\pi}{3}Y_{1,-1}(\mathbf{N}(\mathbf{x}_p)) & \frac{2\pi}{3}Y_{1,1}(\mathbf{N}(\mathbf{x}_p)) & \frac{\pi}{4}Y_{2,-2}(\mathbf{N}(\mathbf{x}_p)) & \frac{\pi}{4}Y_{2,2}(\mathbf{N}(\mathbf{x}_p)) \end{pmatrix}$$

This solution only provides five of the nine lighting environment coefficients.

Shown in Figure 6.3 are two composites into which the ducks (top) and football coach (bottom) were inserted into a seemingly unlikely setting. In each case, the 2-D lighting environment was estimated from various people and ducks in the photo (dashed white lines). The spheres shown in this same figure are rendered with the estimated lighting environments, and reveal fairly significant lighting differences.

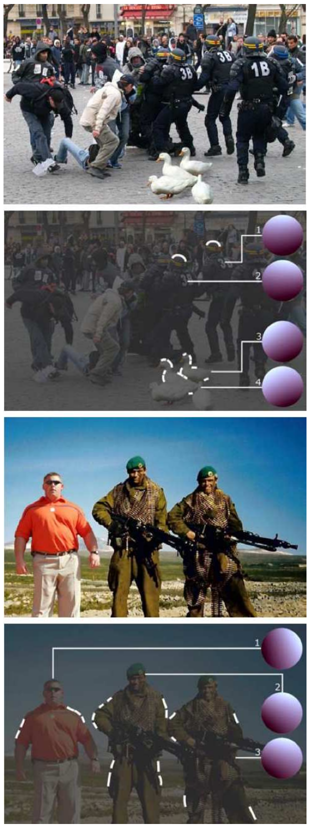


Figure 6.3 Photo composites with inconsistent lighting.

The estimates of lighting described in the previous section consist of only a subset of the complete lighting information. This is because it was assumed that 3-D models were not available and therefore only 2-D surface normals at an object's occluding boundary were used. We next describe how to estimate the full 3-D lighting environment in images of people. In order to extract the required 3-D surface normals, 3-D models are fit to an image of a person's head and automatically aligned to an arbitrary head pose. With such a model the full 3-D light position or lighting environment can be estimated.

In [1], the authors describe a 3-D morphable model for the analysis and synthesis of human faces. The model was derived by collecting a set of 3-D laser scanned faces and projecting them into a lower-dimensional linear subspace. New faces (geometry, texture/color, and expressions) are modeled as linear combinations of the resulting linear basis.

The 3-D model parameters can be estimated from a paired frontal and profile view or from only a single frontal view. This estimation requires the manual selection of several fiducial points on the face (11 on the frontal view and 9 on the profile view), from which the 3-D model is then automatically estimated. Shown in Figure 6.4 (a)-(b), for example, is a frontal and profile view with the selected fiducial points. The estimated model is shown in Figure 6.4 (d)-(f), which can be seen to be in good agreement with the original head poses shown in panels (a)-(c).

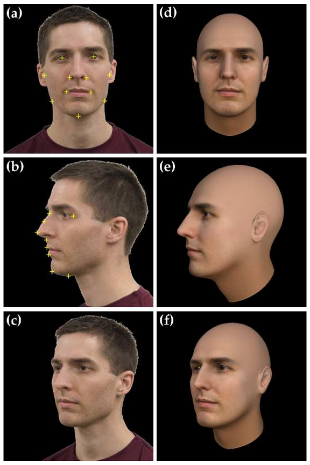


Figure 6.4
Images (a-b) used to estimate the model (d-f).

Once estimated, the 3-D model is registered to the face being analyzed. This is done by maximizing an objective function over the camera intrinsic and extrinsic parameters that aligns the 3-D model to the image of the face. Specifically, we seek the rotation matrix R, translation vector $\mathbf{t}$, focal length f and camera center (c_x, c_y) that maximizes the correlation between the image $I(\cdot)$ and the rendered model $I_m(\cdot)$:

$$E(R, \mathbf{t}, f, c_x, c_y) \quad = \quad I(x,y) * I_m(x,y), \qquad (6.27)$$

where $*$ denotes correlation, the spatial coordinates of the rendered model $I_m(\cdot)$ ($x = x_s/s$ and $y = y_s/s$) are given by:

$$\begin{pmatrix} x_s \\ y_s \\ s \end{pmatrix} \quad = \quad \begin{pmatrix} f & 0 & c_x \\ 0 & f & c_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} R & \mathbf{t} \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}, \qquad (6.28)$$

and (X, Y, Z) are the 3-D coordinates of the face model.

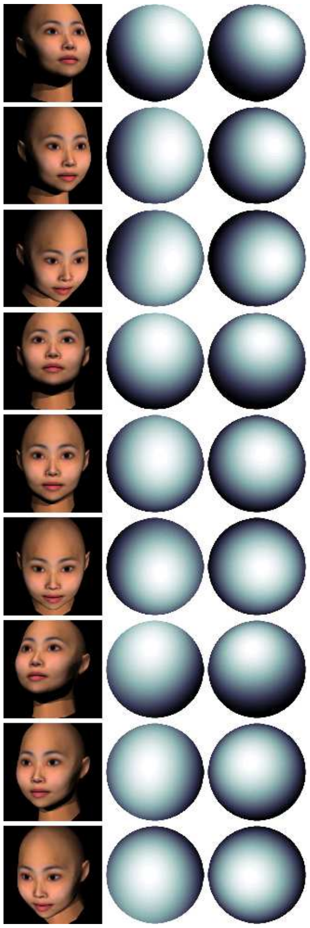


Figure 6.5 Actual (middle) and estimated (right) 3-D lighting environments.

The error function in Equation (6.27) is maximized as follows. The 3-D model is first manually rotated to approximately align it with the image $I(\cdot)$. At least three corresponding points are selected on the model and image (e.g., the center of each eye and base of the nose), from which the optimal translation is estimated using standard least-squares. With this initial alignment as a starting configuration, a brute force search that maximizes Equation (6.27) is performed over the three rotation parameters, focal length, and camera center. On each iteration of this search, the translation vector is estimated as described above. In order to reduce the effects of lighting, a high-pass filter is applied to both the image $I(\cdot)$ and rendered model $I_m(\cdot)$ prior to computing the correlation in Equation (6.27).

Once the model has been estimated and registered, 3-D surface normals and corresponding intensities are used to estimate the lighting environment as described in the previous sections.

Shown in Figure 6.5 are nine images from which the 3-D lighting environment was estimated. Shown in the middle column is a sphere rendered with the actual lighting environment and shown in the last column is a sphere rendered with the estimated lighting environment.

United States President John F. Kennedy was assassinated on November 22nd, 1963. Shortly afterwards, Lee Harvey Oswald was arrested and charged with the crime. Because he was killed before his trial many questions surrounding the assassination remained unanswered. Since this time, numerous theories have circulated suggesting that Oswald acted as part of a larger criminal conspiracy involving a variety of government, international, or criminal groups. Many of these theories point to purported inconsistencies in the events of November 22nd and in the evidence collected against Oswald. One such example is a photograph of Oswald in his backyard holstering a pistol and holding a rifle in one hand and Marxist newspapers in the other, Figure 6.6. Oswald claimed that this photo was fake. In addition, many have argued that the photo is riddled with multiple inconsistencies, including inconsistent lighting, shadows, geometry, and proportions.

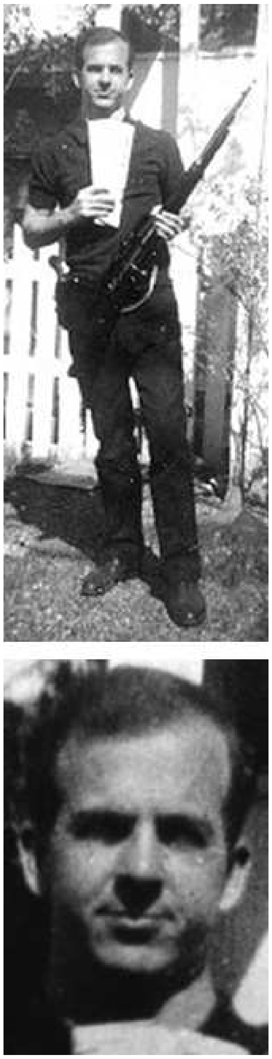


Figure 6.6
Lee Harvey Oswald in his backyard and a magnified view of his head.

It may appear that the shadow cast by Oswald's body onto the ground, and the shadow under his nose are inconsistent with a single light source, Figure 6.6, and hence evidence of photo tampering. Specifically, the nose appears to be illuminated from above, and the body seems to be illuminated from Oswald's upper left. By measuring the location of the light in this scene, and building 3-D models of Oswald's face, body, and other parts of the scene, we can determine if, in fact, these shadows are inconsistent.

In [1], the authors describe a 3-D morphable model for the analysis and synthesis of human faces. The model was derived by collecting a large set of 3-D laser scanned faces and projecting them into a lower-dimensional linear subspace. New faces (geometry, texture/color, and expressions) are modeled as linear combinations of the resulting low-parameter linear basis. The model parameters can be estimated from a paired profile and frontal image or from only a single frontal image.

We are fortunate to have access to contemporaneous profile and frontal views of Oswald in the form of a mugshot taken shortly after his arrest, Figure 6.7. These photographs provide the ideal input for constructing a 3-D model from the commercially available FaceGen, (*Singular Inversions*). Two views of the resulting 3-D model show a good agreement with the original mugshot photo in Figure 6.7.

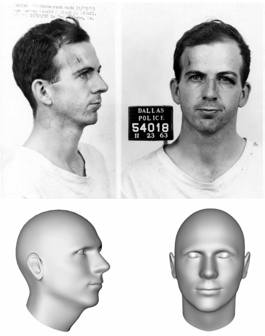
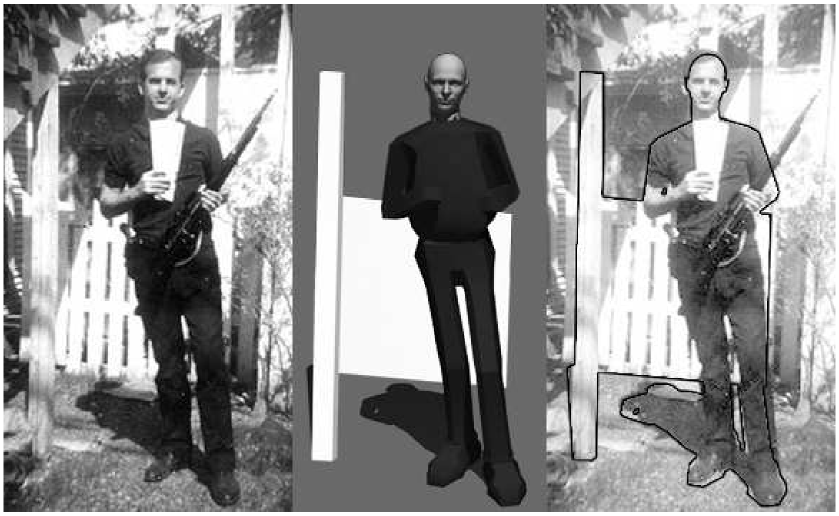


Figure 6.7 A profile and frontal view of Oswald (top) used to construct a 3-D model (bottom).

This 3-D head model was combined with a generic articulated 3-D body and rendered in the 3-D modeling software Maya (*Autodesk*). The ground plane, fence, and post under the stairs were created from simple 3-D primitives. The 3-D direction to a distant light source (i.e., the sun) was estimated as described in the previous section. The overall scene geometry and camera position were manually positioned until they matched the original photo.

Shown below is, from left to right, the original photo, our 3-D rendering, and a superposition of the original photo and the outline of our rendering. This figure shows a good agreement between the model and the original photo. Note that in this 3-D model both the scene geometry and lighting position are estimated, allowing for both a geometric and lighting analysis, as described next.



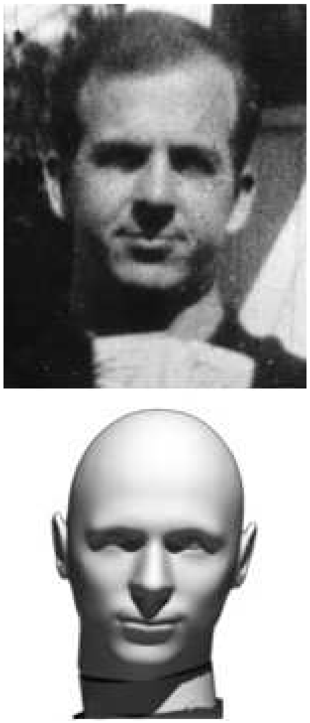


Figure 6.8 Oswald's head and the 3-D model.

With a 3-D model of the scene geometry and lighting, it is relatively easy to compare the shadows in the original photo and rendered scene. Shown in Figure 6.8 is a side-by-side comparison of the shadows. Note that the shadows on the nose, eyes, lower lip and neck are well matched, as are the shadow cast by the body onto the ground plane, and the thin sliver of a shadow from the vertical post onto the ground plane. These shadows, which at first glance appeared inconsistent, are in fact perfectly consistent with a single light source.

At first glance it may also appear that Oswald's chin in the backyard photo is too wide to be consistent with his chin in other photos (e.g., his mugshot) and hence evidence of a photo composite. Shown in the left column of Figure 6.9 is a photo of Oswald from his mugshot (top) and from the backyard photo (bottom). The yellow guidelines are drawn at the point in the top photo where the chin meets the jaw line. Note that the chin appears to be much wider in the backyard photo. Shown in the right column of Figure 6.9 are the corresponding 3-D renderings with neutral front lighting (top) and lighting to match the backyard photo (bottom). The yellow guidelines, of the same width as on the left, show the same apparent widening of the chin. From these 3-D renderings, it is clear that the apparent widening of the chin is due to the shading along the chin and jaw, and not to photo manipulation.

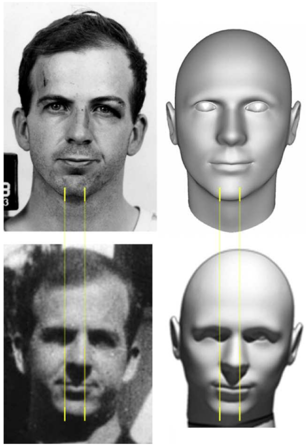


Figure 6.9 Oswald under neutral lighting and in his backyard.

6.5 Problem Set

1. Write MatLab code that estimates the $x-$ and $y-$ components of light direction, as described in Section 5.1. Specifically, your code should:

   - Assume that the object in the image is circular (in order to simplify the user interface).
   - Allow the user to select any number of points along the contour of the object.
   - Fit a circle to this contour.
   - Extract analytic estimates of surface normals from which the 2-D light estimate can be determined.
   - Compute the light direction for each of the images found on the course webpage:

     www.cs.dartmouth.edu/farid/Hany_Farid/Teaching/Teaching.html

Example 6.1 With the circle parametrized as $c(\theta) = (\, r\cos(\theta) \quad r\sin(\theta)\,)^T$, the tangent along the circle is given by:

$$\frac{dc(\theta)}{d\theta} = \begin{pmatrix} -r\sin(\theta) \\ r\cos(\theta) \end{pmatrix}.$$

The surface normal is orthogonal to the tangent, and is computed by rotating the tangent by 90°:

$$\begin{aligned} \mathbf{N} &= \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} \begin{pmatrix} -r\sin(\theta) \\ r\cos(\theta) \end{pmatrix} \\ &= \begin{pmatrix} r\cos(\theta) \\ r\sin(\theta) \end{pmatrix}. \end{aligned}$$

Shown below is Matlab code for computing and visualizing surface normals along a unit circle's boundary.

```
t   = [0 : 10 : 359];
x   = cosd(t); % circle
y   = sind(t); % circle
nx  = cosd(t); % normal
ny  = sind(t); % normal
plot( x, y, '.' );
for k = 1 : length(t)
   line( [x(k) x(k)+nx(k)], [y(k) y(k)+ny(k)] );
end
axis square;
```

1. V. Blanz and T. Vetter. A morphable model for the synthesis of 3D faces. In *SIGGRAPH, Computer Graphics Proceedings*, pages 187-194, Los Angeles, CA, 1999.

2. H. Farid. The Lee Harvey Oswald backyard photos: Real or fake? *Perception*, 11(38):1731-1734, 2009.

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

4. M.K. Johnson and H. Farid. Exposing digital forgeries in complex lighting environments. *IEEE Transactions on Information Forensics and Security*, 3(2):450-461, 2007.

5. E. Kee and H. Farid. Exposing digital forgeries from 3-D lighting environments. In *IEEE International Workshop on Information Forensics and Security*, Seattle, WA, 2010.

7. Video Forensics

7.1 Motion †

Differential motion estimation has proven highly effective at computing inter-frame motion. We model the motion between two sequential frames, or portion of frames, $f(x, y, t)$ and $f(x, y, t - 1)$, with a 6-parameter affine transform:

$$f(x, y, t) = f(m_1 x + m_2 y + m_5, m_3 x + m_4 y + m_6, t - 1), \quad (7.1)$$

where m_1, m_2, m_3, m_4 form the 2×2 affine matrix A and m_5 and m_6 the translation vector $\mathbf{t}$ given by:

$$A = \begin{pmatrix} m_1 & m_2 \\ m_3 & m_4 \end{pmatrix} \qquad \text{and} \qquad \mathbf{t} = \begin{pmatrix} m_5 \\ m_6 \end{pmatrix}. \quad (7.2)$$

With relatively few parameters, the affine model captures a rich range of motions such as translation, scale, and rotation. The basic framework described here can easily be extended to accommodate higher-order polynomial models that capture a richer set of motions (or even a lower-order model that models the inter-frame motion with only a translation). In order to estimate the affine parameters, we define the following quadratic error function to be minimized:

$$E(\mathbf{m}) = \sum_{x,y \in \Omega} [f(x, y, t) - f(m_1 x + m_2 y + m_5, m_3 x + m_4 y + m_6, t - 1)]^2 \quad (7.3)$$

where $\mathbf{m}^T = (\, m_1 \quad \ldots \quad m_6 \,)$ and where Ω denotes a user specified region of interest (ROI). Since this error function is non-linear in its unknowns, it cannot be minimized analytically. To simplify the minimization, we approximate this error function using a first-order truncated Taylor series expansion:

$$\begin{aligned} E(\mathbf{m}) &= \sum_{x,y \in \Omega} [f_t - (m_1 x + m_2 y + m_5 - x) f_x - (m_3 x + m_4 y + m_6 - y) f_y]^2 \\ &= \sum_{x,y \in \Omega} \left[k - \mathbf{c}^T \mathbf{m}\right]^2 \qquad (7.4) \end{aligned}$$

where, for notational convenience, the spatial/temporal parameters are dropped, and where the scalar k and vector $\mathbf{c}$ are given as:

$$\begin{aligned} k &= f_t + x f_x + y f_y \qquad (7.5) \\ \mathbf{c}^T &= (\, x f_x \quad y f_x \quad x f_y \quad y f_y \quad f_x \quad f_y \,). \qquad (7.6) \end{aligned}$$

The quadratic error function is now linear in its unknowns, **m**, and can therefore be minimized analytically by differentiating with respect to **m**:

$$\frac{dE(\mathbf{m})}{d\mathbf{m}} = \sum_{\Omega} 2\mathbf{c}\left[k - \mathbf{c}^T\mathbf{m}\right], \qquad (7.7)$$

setting the result equal to zero, and solving for **m** to yield:

$$\mathbf{m} = \left[\sum_{\Omega} \mathbf{c}\mathbf{c}^T\right]^{-1} \left[\sum_{\Omega} \mathbf{c}k\right]. \qquad (7.8)$$

This solution assumes that the first term, a 6×6 matrix, is invertible. This can usually be guaranteed by integrating over a large enough ROI (Ω) with sufficient image content.

Example 7.1 The Taylor series expansion of a 1-D function $f(x)$ about a point $x = x_0$ is given by:

$$f(x) = f(x_0) + f^{(1)}(x_0)(x - x_0) + \frac{f^{(2)}(x - x_0)^2}{2!} + \frac{f^{(3)}(x - x_0)^3}{3!} + \ldots, \qquad (7.9)$$

where $f^{(n)}$ denotes the n^{th}-order derivative. Consider modeling the motion between a pair of 1-D signals with a simple translation Δ: $f_0(x) = f_1(x + \Delta)$. Using a first-order truncated Taylor series expansion, provide a quadratic error function that is linear in the unknown parameter Δ. [HINT: expand $f_1(x+\Delta)$ by first performing a variable substitution $y = x+\Delta$ and then expanding about the point $y = x$ (i.e., $\Delta = 0$).]

Central to the above motion estimation are the calculation of the spatial/temporal derivatives. Given a pair of frames $f(x, y, t)$ and $f(x, y, t - 1)$, these derivatives are computed as follows:

$$\begin{aligned} f_x(x,y,t) &= (0.5f(x,y,t) + 0.5f(x,y,t-1)) \star d(x) \star p(y) && (7.10)\\ f_y(x,y,t) &= (0.5f(x,y,t) + 0.5f(x,y,t-1)) \star p(x) \star d(y) && (7.11)\\ f_t(x,y,t) &= (0.5f(x,y,t) - 0.5f(x,y,t-1)) \star p(x) \star p(y), && (7.12) \end{aligned}$$

where $\star$ denotes the convolution operator, and $d(\cdot)$ and $p(\cdot)$ are 1-D separable filters:

$$d(x) = \begin{pmatrix} 0.5 & -0.5 \end{pmatrix} \quad \text{and} \quad p(x) = \begin{pmatrix} 0.5 & 0.5 \end{pmatrix}, \qquad (7.13)$$

and where $p(y)$ and $d(y)$ are the same filters oriented vertically instead of horizontally.

The required spatial/temporal derivatives have finite support thus fundamentally limiting the amount of motion that can be estimated. A coarse-to-fine scheme should be adopted in order to contend with larger motions. A L-level Gaussian pyramid is built for each frame, $f(x,y,t)$ and $f(x,y,t-1)$. The motion estimated at pyramid level l is used to warp the frame at the next higher level $l-1$, until the finest level of the pyramid is reached (the full resolution frame at $l=1$). In so doing, large motions are estimated at the coarse levels, and are iteratively refined through the pyramid levels.

The repeated warping through the pyramid levels can introduce significant blurring and therefore adversely affect the motion estimation at the higher pyramid levels. This can be avoided by always operating on the original frame. Specifically, if the estimated motion at pyramid level l is m_1, m_2, m_3, m_4, m_5, and m_6, then the original frame should be warped with the affine matrix A and the translation vector $\mathbf{t}$ given by:

$$A = \begin{pmatrix} m_1 & m_2 \\ m_3 & m_4 \end{pmatrix} \qquad \text{and} \qquad \mathbf{t} = \begin{pmatrix} 2^{l-1}m_5 \\ 2^{l-1}m_6 \end{pmatrix}. \tag{7.14}$$

Working through each level of the pyramid requires that the original frame will have to be repeatedly warped according to the motion estimates at each pyramid level. These individual warps can be accumulated so that the original frame need only undergo one warp. Specifically, two affine matrices A_1 and A_2 and corresponding translation vectors $\mathbf{t}_1$ and $\mathbf{t}_2$ are combined as follows:

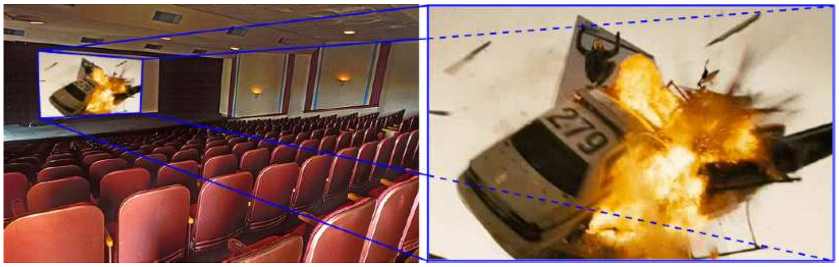
$$A = A_2 A_1 \qquad \text{and} \qquad \mathbf{t} = A_2\mathbf{t}_1 + \mathbf{t}_2, \tag{7.15}$$

which is equivalent to applying A_1 and $\mathbf{t}_1$ followed by A_2 and $\mathbf{t}_2$. The warped original frame is then decomposed into the next required pyramid level, and the process repeated through all pyramid levels.

Example 7.2 Show that the application of an affine matrix A_1 and translation $\mathbf{t}_1$ followed by another affine matrix A_2 and translation $\mathbf{t}_2$, is equivalent to applying the affine matrix and translation vector specified in Equation (7.15).

Often only hours after their release, major motion pictures can find their way onto the Internet. A simple and popular way to create such bootleg video is to record a movie from the theater screen. Although these video are certainly not of the same quality as their subsequent DVD releases, increasingly compact and high resolution video recorders are affording better quality video recordings.

We describe how to automatically detect a video that was recorded from a screen. Shown below (right), for example, is a movie scene.



Also shown in this figure (left) is the same scene as viewed on a theater screen. Note that due to the angle of the video camera relative to the screen, a perspective distortion has been introduced into this second recording. We show that this re-projection can introduce a distortion into the intrinsic camera parameters

Consider the effect of first projecting arbitrary points in 3-D world coordinates into 2-D image coordinates, and then projecting these points a second time. The first projection is given by:

$$\mathbf{p}_1 = \lambda_1 K_1 M_1 \mathbf{P}. \qquad (7.16)$$

The second planar projection is given by:

$$\begin{aligned} \mathbf{p}_2 &= \lambda_2 K_2 M_2 \mathbf{p}_1 & (7.17)\\ &= \lambda_2 K_2 M_2 \left(\lambda_1 K_1 M_1 \mathbf{P}\right) & (7.18)\\ &= \lambda_2 \lambda_1 K_2 M_2 \left(K_1 M_1 \mathbf{P}\right). & (7.19) \end{aligned}$$

We assume that the camera skew (the $(1,2)$ entry in the 3×3 intrinsic matrix) in both projection matrices are zero. It will be shown that a re-projection can yield a non-zero skew in the final effective intrinsic matrix. As such, significant deviations of the skew from zero in the estimated intrinsic matrix can be used as evidence that a video has been re-projected.

The final re-projection matrix $K_2M_2K_1M_1$ can be factored into a product of a scale factor and intrinsic and extrinsic matrices:

$$K_2M_2K_1M_1 \quad = \quad \lambda KM. \tag{7.20}$$

Expressing each 3×4 extrinsic matrix M_1 and M in terms of their rotation and translation components yields:

$$\begin{aligned} K_2M_2K_1(R_1 \mid \mathbf{t}_1) &= \lambda K(R \mid \mathbf{t}) \\ K_2M_2K_1R_1 &= \lambda KR. \end{aligned} \tag{7.21}$$

Reshuffling a few terms yields:

$$\begin{aligned} K^{-1}K_2M_2K_1R_1 &= \lambda R \\ K^{-1}K_2M_2K_1 &= \lambda RR_1^T. \end{aligned} \tag{7.22}$$

Note that the right-hand side of this relationship is an orthogonal matrix – this will be exploited later. On the left-hand side, the left-most matrix is the inverse of the effective intrinsic matrix:

$$K^{-1} \quad = \quad \begin{pmatrix} \frac{1}{\alpha f} & -\frac{s}{\alpha f^2} & \frac{sc_y - c_x f}{\alpha f^2} \\ 0 & \frac{1}{f} & -\frac{c_y}{f} \\ 0 & 0 & 1 \end{pmatrix}. \tag{7.23}$$

And the product of the next three matrices is:

$$\begin{aligned} K_2M_2K_1 &= \begin{pmatrix} f_2 & 0 & 0 \\ 0 & f_2 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} m_{11} & m_{21} & t_1 \\ m_{12} & m_{22} & t_2 \\ m_{13} & m_{23} & t_3 \end{pmatrix} \begin{pmatrix} f_1 & 0 & 0 \\ 0 & f_1 & 0 \\ 0 & 0 & 1 \end{pmatrix}, \\ &= \begin{pmatrix} f_1f_2m_{11} & f_1f_2m_{21} & f_2t_1 \\ f_1f_2m_{12} & f_1f_2m_{22} & f_2t_2 \\ f_1m_{13} & f_1m_{23} & t_3 \end{pmatrix} \\ &= \begin{pmatrix} \mathbf{q}_1^T \\ \mathbf{q}_2^T \\ \mathbf{q}_3^T \end{pmatrix}, \end{aligned} \tag{7.24}$$

where f_2 and f_1 are the focal lengths of the original projections. The product of the four matrices on the left-hand side of Equation (7.22) is then:

$$K^{-1}K_2M_2K_1 = \begin{pmatrix} \left(\frac{1}{\alpha f}\mathbf{q}_1 - \frac{s}{\alpha f^2}\mathbf{q}_2 + \frac{sc_y - c_x f}{\alpha f^2}\mathbf{q}_3 \right)^T \\ \left(\frac{1}{f}\mathbf{q}_2 - \frac{c_y}{f}\mathbf{q}_3 \right)^T \\ \mathbf{q}_3^T \end{pmatrix}. \tag{7.25}$$

Recall that $K^{-1}K_2M_2K_1 = \lambda RR_1^T$, Equation (7.22), and that R and R_1^T are each orthonormal. Since the product of two orthonormal matrices is orthonormal, $K^{-1}K_2M_2K_1$ is orthogonal (the rows/columns will not be unit length when $\lambda \neq 1$). This orthogonality constrains the above matrix rows as follows:

$$\begin{aligned} \mathbf{q}_3^T \left(\frac{1}{f}\mathbf{q}_2 - \frac{c_y}{f}\mathbf{q}_3 \right) &= 0 \qquad (7.26) \\ \left(\frac{1}{f}\mathbf{q}_2 - \frac{c_y}{f}\mathbf{q}_3 \right)^T \left(\frac{1}{\alpha f}\mathbf{q}_1 - \frac{s}{\alpha f^2}\mathbf{q}_2 + \frac{sc_y - c_x f}{\alpha f^2}\mathbf{q}_3 \right) &= 0. \qquad (7.27) \end{aligned}$$

Solving Equation (7.26) for c_y yields:

$$c_y = \frac{\mathbf{q}_3^T \mathbf{q}_2}{\|\mathbf{q}_3\|^2}. \tag{7.28}$$

Substituting for c_y into Equation (7.27), followed by some simplifications, yields:

$$s = f \frac{\mathbf{q}_2^T \mathbf{q}_1 \|\mathbf{q}_3\|^2 - (\mathbf{q}_3^T \mathbf{q}_2)(\mathbf{q}_3^T \mathbf{q}_1)}{\|\mathbf{q}_2\|^2 \|\mathbf{q}_3\|^2 - (\mathbf{q}_3^T \mathbf{q}_2)^2}. \tag{7.29}$$

Note that the skew, s, is expressed only in terms of the effective focal length f, the pair of intrinsic matrices K_1 and K_2, and the second transformation matrix M_2. We can now see under what conditions $s = 0$.

First, note that the denominator of Equation (7.29) cannot be zero. If $\|\mathbf{q}_2\|^2 \|\mathbf{q}_3\|^2 - (\mathbf{q}_3^T \mathbf{q}_2)^2 = 0$ then, $\mathbf{q}_2 \propto \mathbf{q}_3$, in which case $K_2 M_2 K_1$ is singular, which it cannot be, since each matrix in this product is full rank. And, since $f \neq 0$, the skew is zero only when the numerator of Equation (7.29) is zero:

$$\begin{aligned} \mathbf{q}_2^T \mathbf{q}_1 \|\mathbf{q}_3\|^2 - (\mathbf{q}_3^T \mathbf{q}_2)(\mathbf{q}_3^T \mathbf{q}_1) &= 0 \\ f_1^2 p_{31} p_{32} - t_1 t_2 + p_{33}^2 t_1 t_2 + p_{31} p_{32} t_3^2 - p_{32} p_{33} t_1 t_3 - p_{31} p_{33} t_2 t_3 &= 0 \qquad (7.30) \end{aligned}$$

where p_{3i} is the i^{th} element of $\mathbf{p}_3 = \mathbf{p}_1 \times \mathbf{p}_2$. There are a few intuitive cases that can be seen from the above constraint. For example, if the world to camera rotation is strictly about the z-axis, then $p_{31} = p_{32} = 0$ and $p_{33} = 1$, and the skew $s = 0$. This situation arises when the image plane of the second projection is perfectly parallel to the screen being imaged. As another example, if $\mathbf{t} = \pm f_1 \mathbf{p}_3$, then the skew $s = 0$. This situations arises when the translation of the second projection is equal to the third column of the rotation matrix scaled by focal length of the first projection – a perhaps somewhat unlikely configuration.

Although there are clearly many situations under which $s = 0$, simulations suggest that under realistic camera motions, this condition is rarely satisfied. Specifically, we computed the skew, Equation (7.29), from one million randomly generated camera configurations. The relative position of the second camera to the planar projection screen was randomly selected with the rotation in the range $[-45, 45]$ degrees, X and Y translation in the range [-1000,1000], Z translation in the range $[4000, 6000]$, and focal length in the range $[25, 75]$. The average skew was 0.295, and only 48 of the $1,000,000$ configurations had a skew less than 10^{-5} (in a similar simulation, the estimated skew for a single projection is on the order of 10^{-12}).

We next review how to estimate camera skew from a video sequence containing a known planar surface.

Let H describe the projection of a planar surface in the video. Recall that a 3×3 planar homography H can be expressed as:

$$H \quad = \quad \lambda K M = K \left(\mathbf{r}_1 \; \mathbf{r}_2 \mid \mathbf{t} \right). \tag{7.31}$$

The orthonormality of $\mathbf{r}_1$ and $\mathbf{r}_2$, yields the following two constraints:

$$\mathbf{r}_1^T \mathbf{r}_2 = 0 \qquad \text{and} \qquad \mathbf{r}_1^T \mathbf{r}_1 = \mathbf{r}_2^T \mathbf{r}_2, \tag{7.32}$$

which in turn imposes the following constraints on H and K:

$$\begin{pmatrix} h_{11} \\ h_{12} \\ h_{13} \end{pmatrix}^T K^{-T} K^{-1} \begin{pmatrix} h_{21} \\ h_{22} \\ h_{23} \end{pmatrix} \quad = \quad 0 \tag{7.33}$$

$$\begin{pmatrix} h_{11} \\ h_{12} \\ h_{13} \end{pmatrix}^T K^{-T} K^{-1} \begin{pmatrix} h_{11} \\ h_{12} \\ h_{13} \end{pmatrix} \quad = \quad \begin{pmatrix} h_{21} \\ h_{22} \\ h_{23} \end{pmatrix}^T K^{-T} K^{-1} \begin{pmatrix} h_{21} \\ h_{22} \\ h_{23} \end{pmatrix}. \tag{7.34}$$

For notational ease, denote $B = K^{-T} K^{-1}$, where B is a symmetric matrix parametrized with three degrees of freedom:

$$B \quad = \quad \begin{pmatrix} b_{11} & b_{12} & 0 \\ b_{12} & b_{22} & 0 \\ 0 & 0 & 1 \end{pmatrix}. \tag{7.35}$$

Notice that by parametrizing the intrinsic matrix in this way, we have bundled all of the anomalies of a double projection into the estimate of the camera skew. Substituting the matrix B into the constraints of Equations (7.33)-(7.34) yields the following constraints:

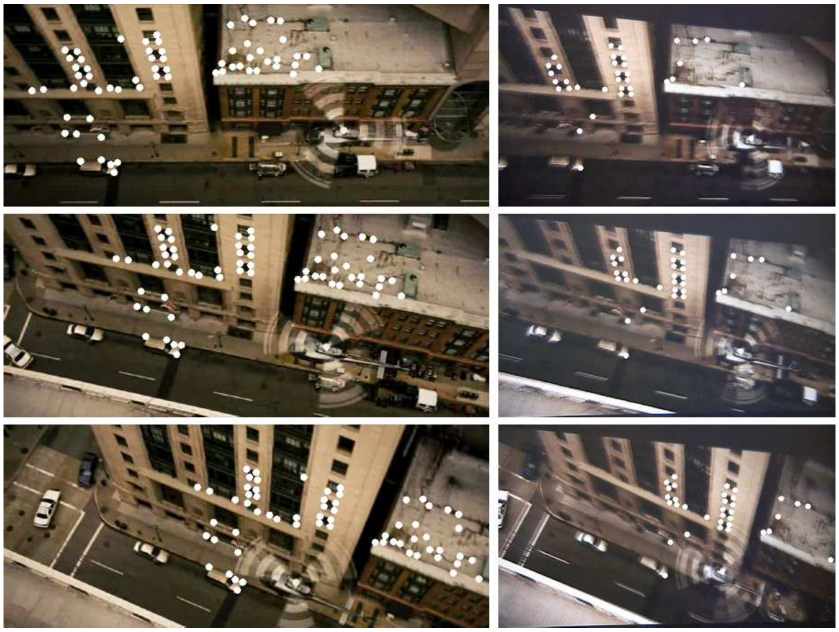
$$\begin{pmatrix} h_{11}h_{21} & h_{12}h_{21} + h_{11}h_{22} & h_{12}h_{22} \\ h_{11}^2 - h_{21}^2 & 2\left(h_{11}h_{12} - h_{21}h_{22}\right) & h_{12}^2 - h_{22}^2 \end{pmatrix} \begin{pmatrix} b_{11} \\ b_{12} \\ b_{22} \end{pmatrix} = - \begin{pmatrix} h_{13}h_{23} \\ h_{13}^2 - h_{23}^2 \end{pmatrix} \tag{7.36}$$

Each image of a planar surface enforces two constraints on the three unknowns b_{ij}. The matrix $B = K^{-T} K^{-1}$ can, therefore, be estimated from two or more views of the same planar surface using standard least-squares estimation. The desired skew can then be determined from the estimated matrix B as:

$$s = -f\frac{b_{12}}{b_{11}}. \tag{7.37}$$

A cameras skew can be estimated from two or more views of a planar surface. This approach has the advantage that it affords a closed-form linear solution, but has the disadvantage that it only applies to frames that contain a known planar surface. As described in [3], it is possible to estimate the camera skew without having to assume a known planar geometry.

Shown below (left) are three frames of 42-frame movie segment. Superimposed on each frame are 64 features tracked across all frames. These features were then used to estimate the skew. The estimated skew for the original movie was 0.029. This 42-frame segment was then displayed on a 20-inch LCD computer monitor with 1600 1200 pixel resolution, and recorded with a digital video camera, three frames of which are shown below (right). A similar set of features were tracked in this video segment, from which the skew was estimated. The estimated skew was 0.25, an order of magnitude larger than the skew from the authentic video.



Increasingly sophisticated video editing and special effects software has made it possible to create forged video sequences that appear to contain realistic dynamic motion. For example, video sharing websites are littered with titles like "Seriously Amazing Best Beer Pong Shots," "Dude Perfect Amazing Basketball Shots," and "Epic Pool Jump." These videos appear to show spectacular basketball shots, gravity-defying acrobatics, and the bone-crushing results of daredevil leaps and jumps. Some of these videos are real, but many are fake. We describe a forensic technique that is tailored to determine if video of a purportedly ballistic motion, such as a ball being thrown or a person jumping through the air, or a motorcycle soaring off of a ramp, is consistent with the geometry and physics of a free-falling projectile.

The center of mass of a projectile, in the absence of air resistance or any other external forces, follows a ballistic trajectory that can be described in three dimensions with a time-parametrized parabolic curve:

$$\mathbf{p}_t \quad = \quad \mathbf{p}_0 + \mathbf{v}t + \tfrac{1}{2}\mathbf{a}t^2, \tag{7.38}$$

where t denotes time, $\mathbf{p}_0$ is the initial position, $\mathbf{v}$ is the initial velocity, and $\mathbf{a}$ is the acceleration (due to gravity). In the absence of any other external forces, the path of the projectile is planar. Assuming linear perspective projection under a pinhole camera model, the image of a projectile's trajectory, in homogeneous coordinates, is:

$$\tilde{\mathbf{q}}_t \quad = \quad \mathbf{H}\mathbf{p}_t, \tag{7.39}$$

where $\mathbf{H}$ is a 3×3 matrix embodying the planar perspective projection (a homography).

Consider the special case where the optical axis of the camera is orthogonal to the plane of motion, the focal length is unit length, and the principal point is the image center. In this case, the z-component of the velocity is zero, and the x- and z-components of the acceleration are zero. The trajectory, Equation (7.38), then simplifies to:

$$\begin{pmatrix} p_t^x \\ p_t^y \\ p_t^z \end{pmatrix} \quad = \quad \begin{pmatrix} p_0^x \\ p_0^y \\ p_0^z \end{pmatrix} + \begin{pmatrix} v^x \\ v^y \\ 0 \end{pmatrix} t + \frac{1}{2} \begin{pmatrix} 0 \\ -g \\ 0 \end{pmatrix} t^2, \tag{7.40}$$

where g is gravity and the world coordinate system is defined such that the z-axis is the optical axis and the positive y-axis points upward. In addition, the world to image transformation is simply $\tilde{\mathbf{q}}_t = \mathbf{p}_t$ ($\mathbf{H} = \mathbf{I}$). In non-homogeneous coordinates, this yields:

$$\tilde{q}_t^x = \frac{p_0^x + v^x t}{p_0^z} \tag{7.41}$$

$$\tilde{q}_t^y = \frac{p_0^y + v^y t - \frac{1}{2}gt^2}{p_0^z}. \tag{7.42}$$

Note that in this special case the projectile's path maps to a parabola in image coordinates, which can be seen more clearly by rewriting the above equations as:

$$\begin{pmatrix} \tilde{q}_t^x \\ \tilde{q}_t^y \end{pmatrix} = \begin{pmatrix} \frac{p_0^x}{p_0^z} \\ \frac{p_0^y}{p_0^z} \end{pmatrix} + \begin{pmatrix} \frac{v^x}{p_0^z} \\ \frac{v^y}{p_0^z} \end{pmatrix} t + \frac{1}{2} \begin{pmatrix} 0 \\ \frac{-g}{p_0^z} \end{pmatrix} t^2. \tag{7.43}$$

Under an arbitrary homography, however, the image of a projectile will not necessarily be a parabola. Specifically:

$$\tilde{\mathbf{q}}_t = \mathbf{H}\mathbf{p}_t = \begin{pmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & h_9 \end{pmatrix} \mathbf{p}_t. \tag{7.44}$$

In non-homogeneous image coordinates, the projectile takes the form:

$$\tilde{q}_t^x = \frac{\mathbf{h}^1 \cdot \mathbf{p}_t}{\mathbf{h}^3 \cdot \mathbf{p}_t} = \frac{\mathbf{h}^1 \cdot (\mathbf{p}_0 + \mathbf{v}t + \frac{1}{2}\mathbf{a}t^2)}{\mathbf{h}^3 \cdot (\mathbf{p}_0 + \mathbf{v}t + \frac{1}{2}\mathbf{a}t^2)} \tag{7.45}$$

$$\tilde{q}_t^y = \frac{\mathbf{h}^2 \cdot \mathbf{p}_t}{\mathbf{h}^3 \cdot \mathbf{p}_t} = \frac{\mathbf{h}^2 \cdot (\mathbf{p}_0 + \mathbf{v}t + \frac{1}{2}\mathbf{a}t^2)}{\mathbf{h}^3 \cdot (\mathbf{p}_0 + \mathbf{v}t + \frac{1}{2}\mathbf{a}t^2)}, \tag{7.46}$$

where $\cdot$ is inner product and $\mathbf{h}^i$ is the i^{th} row of the homography $\mathbf{H}$. Note that these image coordinates follow a rational parabola described by a ratio of second-order polynomials, and cannot be expressed as a single second-order polynomial as in Equation (7.43).

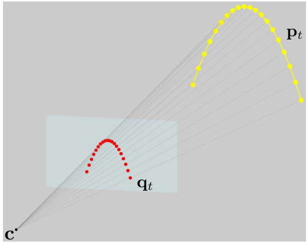


Figure 7.1 A ballistic trajectory $\mathbf{p}_t$ projects to a rational parabola $\mathbf{q}_t$.

The motion of a projectile described above was specified in a world coordinate system, Equation (7.38), and its projection was specified in an image coordinate system, Equation (7.39). We will now specify all coordinates with respect to a common three-dimensional coordinate system in which the origin is the camera center, the image plane is f units from the origin (the focal length), and the optical axis is orthogonal to the image plane. With this notation, a projectile is specified as $\mathbf{p}_t$ in Equation (7.38), its projection is specified as a three-vector $\mathbf{q}_t$ in which the z-component $q_t^z = f$, and the camera center, $\mathbf{c}$, is the origin of the three-dimensional coordinate system, Figure 7.1.

For each moment in time $t = 1 \ldots n$, define a line from the camera center, $\mathbf{c}$, through the image of the projectile's center of mass, $\mathbf{q}_t$, as:

$$\mathbf{l}_t \;=\; \mathbf{c} + s_t(\mathbf{q}_t - \mathbf{c}), \tag{7.47}$$

where s_t is a parametric variable for the line. If the projectile $\mathbf{p}_t$ follows a parabolic trajectory then there exists a value of the parametric variable at each moment in time that satisfies:

$$\begin{aligned} \mathbf{p}_t &= \mathbf{l}_t && (7.48) \\ \mathbf{p}_0 + \mathbf{v}t + \tfrac{1}{2}\mathbf{a}t^2 &= \mathbf{c} + s_t(\mathbf{q}_t - \mathbf{c}), && (7.49) \end{aligned}$$

for some values of s_t. Expanding in terms of the individual components yields:

$$\begin{aligned} p_0^x + v^x t + \tfrac{1}{2}a^x t^2 &= c^x + s_t(q_t^x - c^x) && (7.50) \\ p_0^y + v^y t + \tfrac{1}{2}a^y t^2 &= c^y + s_t(q_t^y - c^y) && (7.51) \\ p_0^z + v^z t + \tfrac{1}{2}a^z t^2 &= c^z + s_t(q_t^z - c^z). && (7.52) \end{aligned}$$

This system of equations is linear in terms of both the nine unknowns that specify the object's trajectory through space ($\mathbf{p}_0$, $\mathbf{v}$, and $\mathbf{a}$) and the n unknowns that specify the object's parametric location along the line from the camera (s_t). With a sufficient number of frames ($n \geq 5$) where the projectile's position can be observed, it is possible to solve for these unknowns by performing a least-squares fit to the data. However, in the absence of noise, the linear system will always be rank deficient so that the least-squares solution is not unique. This deficiency corresponds to a scale ambiguity in the solution: any solution could be scaled about the origin and still satisfy the constraints, Figure 7.1

Although the perspective projection of a parabola makes an ideal model for representing the image of ballistic trajectory, the previously mentioned scale ambiguity can lead to bad behavior when data are noisy or if the underlying motion is not actually ballistic. In these cases, the solution may be skewed dramatically in the otherwise unconstrained part of the solution space. This skew typically manifests as trajectories in planes that are nearly parallel to the view lines with unreasonably high velocities and accelerations.

To prevent these degenerate solutions, an optional constraint can be imposed based on the variation in size of a projectile over time. With the assumption that the actual projectile is of constant size, then its projected size in the image is inversely proportional to the distance between the object and camera center as measured orthogonal to the image plane. Accordingly, the additional constraints require that the trajectory's distance to the image plane vary based on measurements of the object's size in the input images.

Consider a spherical projectile with diameter d at position (x_1, y_1, z_1) relative to the camera center with the z-axis being perpendicular to the image plane. The projected size of this projectile will be $\tilde{d}_1 = fd/z_1$, where f is the camera focal length. As the projectile moves to another position (x_2, y_2, z_2), the projected size is $\tilde{d}_2 = fd/z_2$. The ratio of these projections is $\tilde{d}_2/\tilde{d}_1 = z_1/z_2$. Note that this ratio does not depend on the focal length f or diameter d.

This basic constraint takes the form:

$$\frac{l_1^z - c^z}{l_k^z - c^z} \quad = \quad \frac{\tilde{d}_k}{\tilde{d}_1}, \tag{7.53}$$

where c^z is the z-component of the camera center $\mathbf{c}$, $\tilde{d}_1$ and $\tilde{d}_k$ are the measured sizes of the projectile in the image at times $t = 1$ and $t = k$, and l_1^z and l_k^z are the z-components of $\mathbf{l}_1$ and $\mathbf{l}_k$ as defined in Equation (7.47). This constraint expands to:

$$s_1 \left[\frac{q_1^z - c^z}{q_k^z - c^z}\right] - s_k \left[\frac{\tilde{d}_k}{\tilde{d}_1}\right] \quad = \quad 0. \tag{7.54}$$

Note that this constraint is linear in the unknown parametric variables s_1 and s_k. These linear constraints for all $k = 2 \ldots n$ can be included when solving for the trajectory parameters.

All of the constraints can now be assembled into a linear system of equations. This system can be over-constrained, either because the data cannot be fit by a parabola or because of noise caused by small measurement errors. This system can also be under-constrained due to a scale ambiguity. The following least-squares solution contends with all of these cases.

The unknown parameters, $\mathbf{p}_0$, $\mathbf{v}$, $\mathbf{a}$, and s_t $(t = 1 \ldots n)$, are gathered into a length $n + 9$ vector:

$$\mathbf{u} = [p_0^x \; v^x \; a^x \; p_0^y \; v^y \; a^y \; p_0^z \; v^z \; a^z \; s_1 \cdots s_n]^{\mathsf{T}}. \tag{7.55}$$

The constraints, Equations (7.50)-(7.52), are assembled into a linear system $\mathbf{Mu} = \mathbf{b}$ where:

$$\mathbf{M}=\begin{pmatrix} 1 & t_1 & \frac{t_1^2}{2} & 0 & 0 & 0 & 0 & 0 & 0 & c^x - q_1^x & 0 & \dots & 0 \\ 0 & 0 & 0 & 1 & t_1 & \frac{t_1^2}{2} & 0 & 0 & 0 & c^y - q_1^y & 0 & \dots & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & t_1 & \frac{t_1^2}{2} & c^z - q_1^z & 0 & \dots & 0 \\ 1 & t_2 & \frac{t_2^2}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & c^x - q_2^x & \dots & 0 \\ 0 & 0 & 0 & 1 & t_2 & \frac{t_2^2}{2} & 0 & 0 & 0 & 0 & c^y - q_2^y & \dots & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & t_2 & \frac{t_2^2}{2} & 0 & c^z - q_2^z & \dots & 0 \\ \vdots & & & & & & & \ddots & & & & & \vdots \\ 1 & t_n & \frac{t_n^2}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \dots & c^x - q_n^x \\ 0 & 0 & 0 & 1 & t_n & \frac{t_n^2}{2} & 0 & 0 & 0 & 0 & 0 & \dots & c^y - q_n^y \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & t_n & \frac{t_n^2}{2} & 0 & 0 & \dots & c^z - q_n^z \end{pmatrix} \tag{7.56}$$

and

$$\mathbf{b} = [c^x \; c^y \; c^z \; c^x \; c^y \; c^z \cdots c^x \; c^y \; c^z]^{\mathsf{T}} \,. \tag{7.57}$$

If the optional size constraints are used, then $\mathbf{M}$ is extended by appending an additional $n-1$ rows with a corresponding number of zeros appended to $\mathbf{b}$:

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{q_1^z - c^z}{q_2^z - c^z} & -\frac{\tilde{d}_2}{\tilde{d}_1} & 0 & \dots & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{q_1^z - c^z}{q_3^z - c^z} & 0 & -\frac{\tilde{d}_3}{\tilde{d}_1} & \dots & 0 \\ \vdots & & & & & & & \ddots & & & & & & \vdots \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{q_1^z - c^z}{q_n^z - c^z} & 0 & 0 & \dots & -\frac{\tilde{d}_n}{\tilde{d}_1} \end{pmatrix} . \tag{7.58}$$

The least-squares solution to the system of constraints is given by $\mathbf{u} = \mathbf{M}^{+}\mathbf{b}$, where $\mathbf{M}^{+}$ denotes pseudo-inverse computed as $\mathbf{M}^{+} = \mathbf{V}\mathbf{S}^{-1}\mathbf{U}^{\mathsf{T}}$, where $\mathbf{M} = \mathbf{USV}$ is the singular-value decomposition of $\mathbf{M}$.

Due to the possible scale ambiguity, this solution may not be unique. This occurs when the smallest-magnitude singular value in $\mathbf{S}$ is zero. If there is no zero singular value then $\mathbf{M}^{+}\mathbf{b}$ is our final solution. Otherwise the ambiguity is resolved by finding the solution where $\|\mathbf{a}\| = 9.8\text{m/s}^2$. Let $\mathbf{u}^*$ be the column of $\mathbf{V}$ corresponding to the zero singular value. The final solution is $\mathbf{u} = \mathbf{M}^{+}\mathbf{b} + \alpha\mathbf{u}^*$ where α is selected by solving the quadratic equation:

$$\left\| [0 \; 0 \; a^x \; 0 \; 0 \; a^y \; 0 \; 0 \; a^z \; 0 \cdots 0]^{\mathsf{T}} \cdot (\mathbf{M}^{+}\mathbf{b} + \alpha\mathbf{u}^*) \right\|^2 = 9.8^2. \tag{7.59}$$

This quadratic constraint on the acceleration will give us the scaling that corresponds to its size in the physical world. More importantly it also avoids the trivial solution of $\mathbf{u} = \mathbf{0}$ when $\mathbf{b} = \mathbf{0}$.

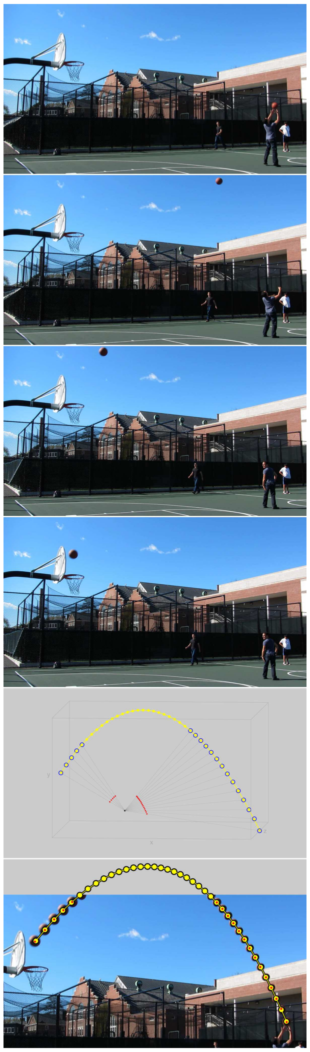


Figure 7.2 An authentic video.

The estimation of the projectile motion described above yields two parametrizations of the projectile path. The first is the initial position, velocity and acceleration ($\mathbf{p}_0$, $\mathbf{v}$, $\mathbf{a}$) as specified in Equation (7.38). The second is a non-parametric representation specified by the variables s_t.

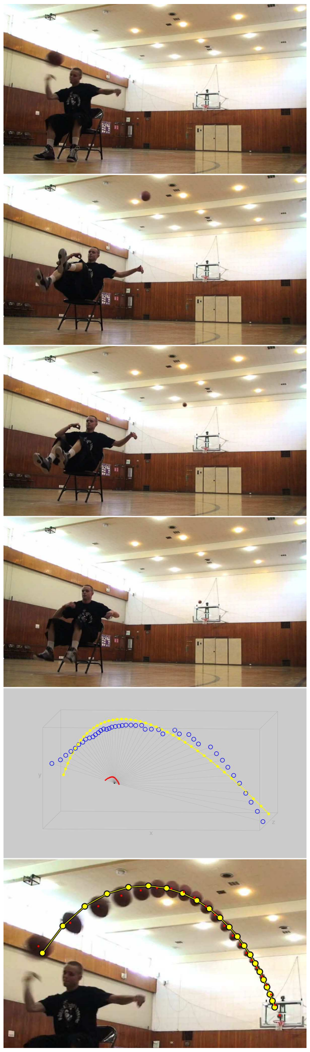


Figure 7.3 A fake video.

For authentic video of a ballistic projectile, these representations should be in agreement, while an imperfectly faked projectile motion will yield inconsistencies in these representations. We quantify the error in these representations as the average Euclidean distance between the pair of representations of the projectile motion:

$$
\begin{aligned}
E &= \frac{1}{n}\sum_{t=1}^{n} \|\mathbf{p}_t - \mathbf{l}_t\| && (7.60)\\
&= \frac{1}{n}\sum_{t=1}^{n} \|(\mathbf{p}_0 + \mathbf{v}t + \tfrac{1}{2}\mathbf{a}t^2) - (\mathbf{c} + s_t(\mathbf{q}_t - \mathbf{c}))\|, && (7.61)
\end{aligned}
$$

where $\|\cdot\|$ denotes a vector 2-norm. Specifically, an error E above a specified threshold is taken to be evidence of tampering.

This average error will be large when an overall path does not correspond to a ballistic trajectory. For situations where only small parts of an otherwise correct motion have been altered, the maximum error may also be informative.

Although not described here, this basic formulation is applicable to the case when the camera is moving, however it is necessary to estimate the inter-frame camera motion so that the projectile can be expressed over time in the same coordinate system.

Shown in Figure 7.2 are (from top to bottom) four sample video frames. Shown in the next panel are two parametrizations of the estimated trajectory: the parametric trajectory specified by $\mathbf{p}_0$, $\mathbf{v}$, $\mathbf{a}$ (filled yellow dots and solid line), and the non-parametric trajectory specified by the variables s_t (open blue circles). The small black dot corresponds to the camera center and the small red dots correspond to the projection of the projectile in each video frame. Note that the two parametrizations are in agreement, as expected for an authentic video. Shown in bottom panel is the estimated parametric trajectory projected into the image plane (filled yellow dots) and the tracked position of the projectile (small red dots). These locations are in agreement, as expected for an authentic video. Shown in Figure 7.3 are the results for a fake video. The two parametrizations are not in agreement, and the estimated and tracked positions in the bottom panel are not in agreement, revealing this video to be a fake.

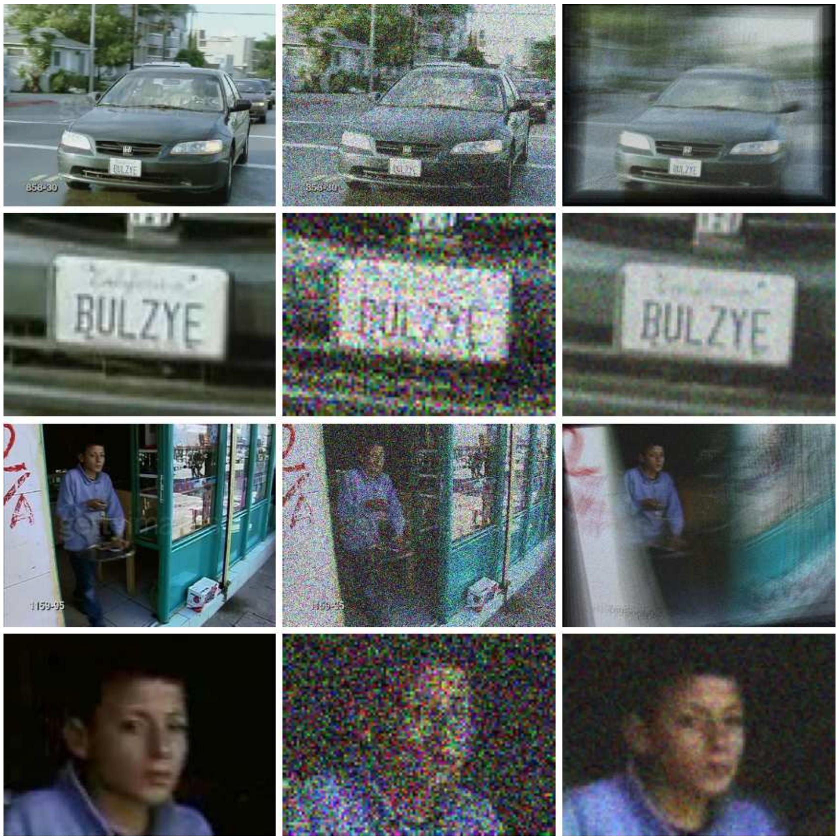
The poor image quality of many video surveillance cameras effectively renders them useless for the purposes of identifying a person, a license plate, etc. Under certain conditions, however, it may be possible to combine multiple video frames for such identification tasks.

The goal of video stabilization is to create a new video sequence where the motion between frames (or parts of a frame) has effectively been removed.

Inter-frame motion between two frames can be estimated as described in Section 6.1. For a full video sequence $f(x, y, t)$, $t \in [1, N]$, the inter-frame motion is computed between all pairs of neighboring frames to yield motion estimates $\mathbf{m}_t$. With the ROI (Ω in Equation (7.8)) specified on the first frame, the ROI must be warped on each successive frame to account for the underlying motion between frames. The ROI at time t ($t > 2$) is therefore simply warped according to the previous frame's estimated motion. With the motion estimated between all pairs of neighboring frames, each frame is warped to align with the last frame of the sequence. Successive warps are combined according to Equation (7.15).

Once stabilized, the video sequence is combined to yield a single high quality image. We assume that the corrupting noise on each video frame is identically and independently drawn (iid) from a zero-mean distribution. Under this assumption a temporal mean or median filter is the optimal approach to removing the noise – the temporal mean minimizes the $L2$ norm and the temporal median minimizes the $L1$ norm between the true and estimated value at each pixel. Specifically, denote $f(x, y, t)$, $t \in [1, N]$ as the original video sequence, and $\hat{f}(x, y, t)$ as the stabilized video. The enhanced frame is estimated by computing a pixel-wise mean or median across $\hat{f}(x, y, t)$. Although typically comparable, one temporal filter may give better results than the other depending on the specific noise characteristics.

Shown in the figure below is one frame of an original video, corrupted video, and enhanced video. In these examples, the motion estimation was applied to a grayscale version of each color frame, and the stabilization and enhancement applied separately to each color channel. The ROI was a rectangular region encompassing the license plate and face.


858-30
BULZYE
1153-95

7.5 Problem Set

1. Write MatLab code for computing motion between two video frames as described in Equation (7.8). You need not implement the coarse-to-fine portion typically required by differential motion estimation techniques. Compute the motion for the image sequences found on the course webpage:

   www.cs.dartmouth.edu/farid/Hany_Farid/Teaching/Teaching.html

Example 7.1 Consider the estimation of the translation Δ between a pair of 1-D signals:

$$f_0(x) \;=\; f_1(x+\Delta).$$

We seek to linearize $f_1(x+\Delta)$ in order to be able to estimate Δ using least-squares estimation. We compute the Taylor series expansion of $f_1(x+\Delta)$. Let $y = x+\Delta$, and then expand $f_1(y)$ about the point $y_0 = x$ (i.e., $\Delta = 0$):

$$f_1(y) \;=\; f_1(y_0) + f_1'(y_0)(y - y_0).$$

A variable substitution then yields:

$$\begin{aligned} f_1(x+\Delta) &= f_1(x) + f_1'(x)(x+\Delta-x) \\ &= f_1(x) + f_1'(x)\Delta, \end{aligned}$$

which is now linear in the unknown Δ. The quadratic error function then takes the form:

$$E(\Delta) \;=\; [f_0(x) - (f_1(x) + \Delta f_1'(x))]^2.$$

Example 7.2 Consider the application of affine matrix A_1 and translation $\mathbf{t}_1$:

$$A_1\mathbf{x} + \mathbf{t}_1,$$

followed by the application of a second affine matrix A_2 and translation $\mathbf{t}_2$:

$$A_2(A_1\mathbf{x} + \mathbf{t}_1) + \mathbf{t}_2 \;=\; (A_2A_1)\mathbf{x} + (A_2\mathbf{t}_1 + \mathbf{t}_2),$$

which is the same as applying one affine matrix A_2A_1 and one translation vector $A_2\mathbf{t}_1 + \mathbf{t}_2$.

1. V. Conotter, J.F. O'Brien, and H. Farid. Exposing Digital Forgeries in Ballistic Motion. *IEEE Transactions on Information Forensics and Security*, 2011 (in review).

2. H. Farid and J.B. Woodward. Video stabilization and enhancement. Technical Report TR2007-605, Department of Computer Science, Dartmouth College, 2007.

3. W. Wang and H. Farid. Detecting re-projected video. In *10th International Workshop on Information Hiding*, Santa Barbara, CA, 2008.

8. Printer Forensics

8.1 Clustering

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:

$$\text{cut}(A, B) \quad = \quad \sum_{u \in A} \sum_{v \in B} w(u, v). \tag{8.1}$$

The optimal bipartioning of a graph is that which minimizes the cut. Shown in Figure 8.1, 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.

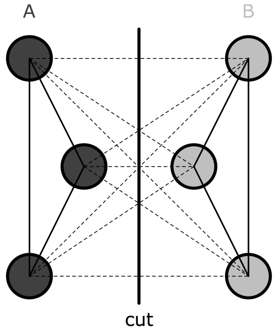


Figure 8.1 Clustering by graph cut.

Computing the optimal cut is NP-complete, as there are an exponential number of partitions to consider. There are, however, efficient approximation algorithms. We describe one such technique, normalized cuts. When minimizing the graph cut, Equation (8.1), there is a natural tendency to simply cut a small number of low-cost edges. The normalized cut was introduced to remove this bias and is defined as:

$$\text{Ncut}(A, B) \quad = \quad \frac{\text{cut}(A, B)}{\text{assoc}(A, V)} + \frac{\text{cut}(A, B)}{\text{assoc}(B, V)}, \tag{8.2}$$

where,

$$\text{assoc}(A, V) \quad = \quad \sum_{u \in A} \sum_{v \in V} w(u, v) \tag{8.3}$$

and

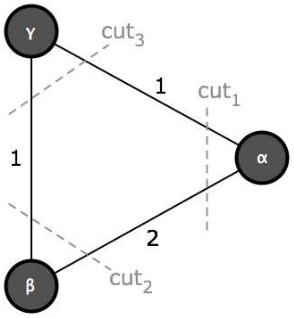
$$\text{assoc}(B, V) \quad = \quad \sum_{u \in B} \sum_{v \in V} w(u, v). \tag{8.4}$$

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.

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 $(D - W)\mathbf{e} = \lambda D\mathbf{e}$, for the eigenvector $\mathbf{e}$ with the second smallest eigenvalue λ. Let the sign of each component of $\mathbf{e}$ (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.

Example 8.1 Consider the following simple 3-node graph with edge weights 1, 1, and 2.



What is the normalized cut, Equation (8.2), for each of the three cuts (denoted with a dashed line)?

In a forensic setting it is likely that one will have to contend with partitioning a graph with 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 the solution to 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 Lanczos method can be employed for estimating large, sparse and symmetric eigenproblems. This technique is particularly efficient when only a few of the extremal (maximum or minimum) eigenvalue eigenvectors are needed, as in our case.

8.2 Banding

The printer forensic technique described here works by modeling the intensity variations in a printed grayscale or color image [1]. These banding artifacts are a result of intrinsic printer imperfections. Because these imperfections are distinct, they can be used to link a printed image to a specific printer.

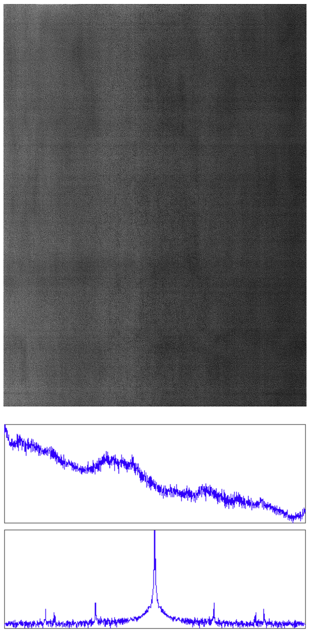


Figure 8.2 Banding artifacts.

Shown in Figure 8.2, for example, is a uniform gray image printed on a standard laser printer. Also shown in this figure is the result of vertically averaging the intensity to yield a 1-D plot of the intensity variation as a function of horizontal position on the page. The significant deviations from a constant value are the result of banding artifacts. Because these artifacts tend to be periodic, their form can be quantified in the Fourier domain. Shown in the bottom panel of Figure 8.2 is the power spectrum of the 1-D vertical average of intensity variation. The peaks correspond to the periodicity in the banding pattern.

8.3 Profiling

The printer forensic technique described here works by modeling the geometric degradation on a per-character basis [3]. Instead of explicitly modeling this usually complex degradation, a data driven approach is taken in which a linear basis is generated from a set of degraded characters (e.g., the letter *e*). This basis representation embodies the printer degradation and can be used to determine if parts of a printed document are consistent with a single printer.

In this first stage, multiple copies of the same character are located in a scanned document. A single character is then selected as the reference character. Each character is brought into spatial alignment with this reference character using a coarse-to-fine differential registration technique as described in Section 7.1.

With all of the characters properly aligned, a profile is constructed that embodies the degradation introduced by the printer. Specifically, a principal components analysis (PCA) is applied, Section 4.1.,to the aligned characters to create a new linear basis that embodies the printer degradation. Although this approach is limited to a linear basis, this degradation model is easy to compute and is able to capture fairly complex degradations that are not easily embodied by a low-dimensional parametric model.

The final profile consists of both the mean character, μ, (subtracted off prior to the PCA) and the top p eigenvalue eigenvectors $\mathbf{e}_i$, $i \in [1,p]$. Note that this printer profile is constructed on a per character basis, e.g., for the letter e of the same font and size. Shown in Figure 8.3, for example, are the printer profiles (μ and $\mathbf{e}_1$) for three printers (HP LasertJet 4300, Xerox Phaser 5500DN, and Xerox Phaser 8550DP).

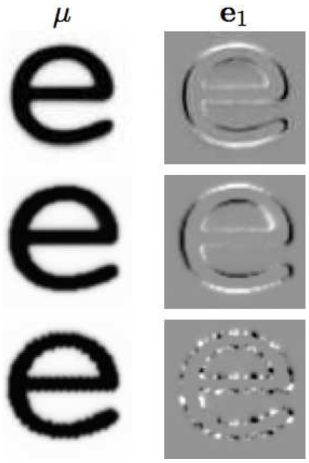


Figure 8.3 Printer profile for three printers.

In a forensics setting, we are interested in determining if part of a document has been manipulated by, for example, splicing in portions from a different document, or digitally editing a previously printed and scanned document and then printing the result. To this end, a printed and scanned document is processed to construct a printer profile $\mathcal{P} = \{\mu, \mathbf{e}_1, ..., \mathbf{e}_p\}$. Each character $\mathbf{c}_j$ is then projected onto each basis vector:

$$\alpha_{ji} \quad = \quad (\mathbf{c}_j - \mu)^T \mathbf{e}_i, \tag{8.5}$$

where a character $\mathbf{c}_j$ is a 2-D grayscale image reshaped into vector form. The basis weights for each character $\mathbf{c}_j$ are denoted as $\alpha_j = (\, \alpha_{j1} \quad \alpha_{j2} \quad \ldots \quad \alpha_{jp}\,)$. With the assumption that tampering will disturb the basis representation α_j, the weights α_j are subjected to a normalized graph cut partitioning, Section 8.1, to determine if they form distinct clusters.

Specifically, a weighted undirected graph $G = (V, E)$ is constructed with vertices V and edges E. Each vertex corresponds to a character $\mathbf{c}_j$ with $j \in [1, m]$, and the weight on each edge connecting vertices k and l is given by:

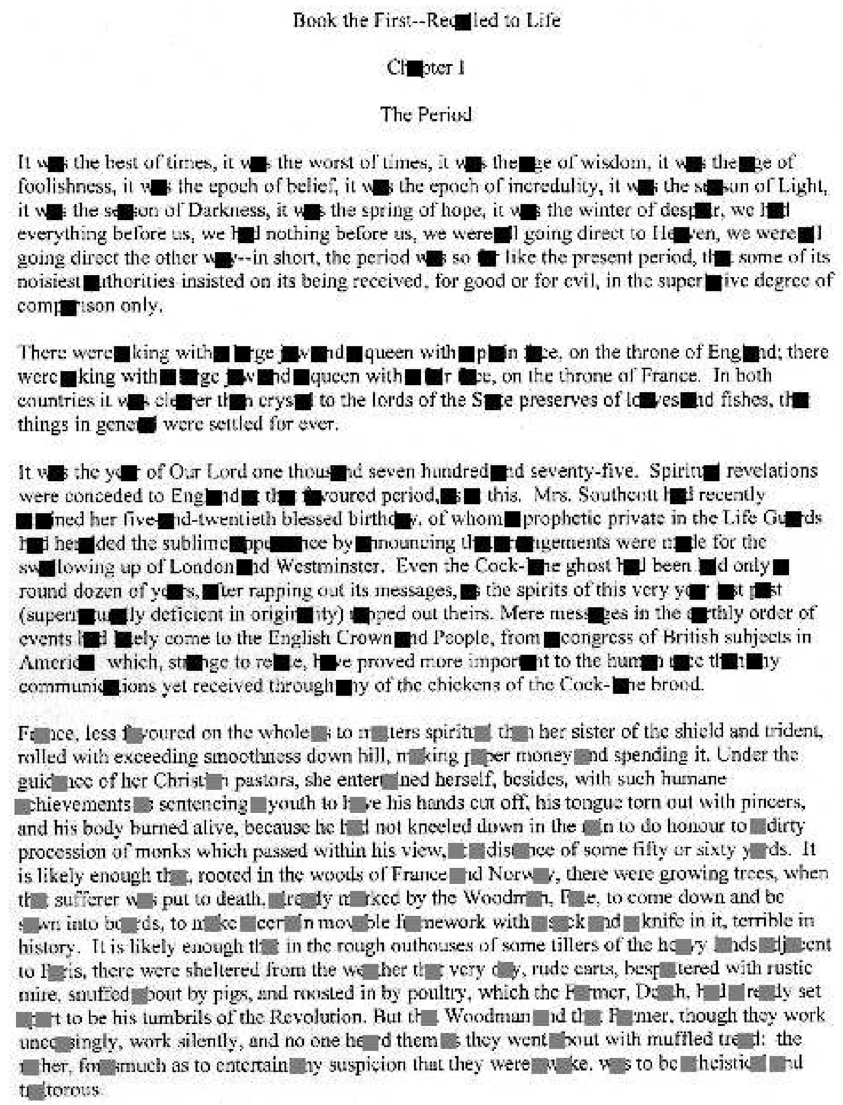
$$w(k,l) \quad = \quad \exp\left(-\frac{d_\alpha^2(\alpha_k, \alpha_l)}{\sigma_\alpha^2}\right) \cdot \exp\left(-\frac{d_c^2(k,l)}{\sigma_c^2}\right), \tag{8.6}$$

where $d_\alpha(\cdot)$ is the Mahalanobis distance defined as:

$$d_\alpha(\mathbf{x}, \mathbf{y}) \quad = \quad \sqrt{(\mathbf{x} - \mathbf{y})\Sigma^{-1}(\mathbf{x} - \mathbf{y})}, \tag{8.7}$$

and where Σ is the covariance matrix. The second term in the weighting function, $d_c(\cdot)$, is the distance between two characters, defined as the linear distance in scan line order (i.e., in the order in which the text is read, left to right, top to bottom). This additional term makes it more likely for characters in close proximity to be grouped together.

Shown in Figure 8.4 is a page from *The Tale of Two Cities*, where the top half was printed on a HP LaserJet 4350 and the bottom half was printed on a Xerox Phaser 5500DN. These documents were scanned and combined and printed on an HP LaserJet 4300 printer. A printer profile was created from 200 copies of the letter *a*. Shown in Figure 8.4 are the classification results, which correctly classify the top and bottom halves of the document as originating from different printers. The cost of this clustering was 0.05 which is an order of magnitude smaller than that which occurs in the authentic document.



Book the First--Rec■lled to Life

Ch■pter I

The Period

It w■s the best of times, it w■s the worst of times, it w■s the■ge of wisdom, it w■s the■ge of foolishness, it w■s the epoch of belief, it w■s the epoch of incredulity, it w■s the s■son of Light, it w■s the s■son of Darkness, it w■s the spring of hope, it w■s the winter of desp■r, we h■d everything before us, we h■d nothing before us, we were■l going direct to He■ven, we were■l going direct the other w■y--in short, the period w■s so f■r like the present period, th■ some of its noisiest■thorities insisted on its being received, for good or for evil, in the super■ive degree of comp■rison only.

There were■king with■l■rge j■w■nd■queen with■p■in f■ce, on the throne of Eng■nd; there were■king with■l■rge j■w■nd■queen with■f■r f■ce, on the throne of France. In both countries it w■s cle■rer th■n crys■l to the lords of the S■te preserves of lo■ves■nd fishes, th■ things in gene■l were settled for ever.

It w■s the ye■r of Our Lord one thous■nd seven hundred■nd seventy-five. Spiritu■l revelations were conceded to Eng■nd■t th■t f■voured period, ■s■ this. Mrs. Southcott h■d recently ■■ned her five-■nd-twentieth blessed birthd■y, of whom■prophetic private in the Life Gu■rds h■d her■lded the sublime■ppe■nce by■nnouncing th■■rr■ngements were m■de for the sw■llowing up of London■nd Westminster. Even the Cock-■ne ghost h■d been ■d only■ round dozen of ye■rs, ■fter rapping out its messages, ■s the spirits of this very ye■r ■st p■st (supern■tu■lly deficient in origi■ity) ■pped out theirs. Mere mess■ges in the ■rthly order of events h■d ■tely come to the English Crown■nd People, from■congress of British subjects in Americ■: which, st■nge to re■e, h■ve proved more import■nt to the hum■n r■ce th■n■ny communic■tions yet received through■ny of the chickens of the Cock-■ne brood.

Fr▨nce, less f▨voured on the whole ▨s to m▨tters spiritu▨l th▨n her sister of the shield and trident, rolled with exceeding smoothness down hill, m▨king p▨per money▨nd spending it. Under the guid▨nce of her Christi▨n pastors, she entert▨ned herself, besides, with such humane ▨chievements ▨s sentencing▨youth to h▨ve his hands cut off, his tongue torn out with pincers, and his body burned alive, because he h▨d not kneeled down in the r▨n to do honour to▨dirty procession of monks which passed within his view, ▨t▨dist▨nce of some fifty or sixty y▨rds. It is likely enough th▨t, rooted in the woods of France▨nd Norw▨y, there were growing trees, when th▨t sufferer w▨s put to death, ▨lre▨dy m▨rked by the Woodm▨n, F▨te, to come down and be s▨wn into bo▨rds, to m▨ke▨cert▨n mov▨ble fr▨mework with▨s▨ck▨nd▨knife in it, terrible in history. It is likely enough th▨t in the rough outhouses of some tillers of the he▨vy l▨nds▨dj▨cent to P▨ris, there were sheltered from the we▨ther th▨t very d▨y, rude carts, besp▨ttered with rustic mire, snuffed▨bout by pigs, and roosted in by poultry, which the F▨rmer, De▨th, h▨d▨lre▨dy set ▨p▨rt to be his tumbrils of the Revolution. But th▨t Woodman▨nd th▨t F▨rmer, though they work unce▨singly, work silently, and no one he▨rd them▨s they went▨bout with muffled tre▨d: the r▨ther, for▨smuch as to entertain▨ny suspicion that they were▨w▨ke, w▨s to be▨theistic▨l▨nd tr▨torous.

Figure 8.4 Each small square region denotes a single letter 'a', and the color coding (gray/black) denotes the cluster assignment.

8.4 Problem Set

1. Generate a 50×50 sized grayscale image of the letter 'e'. Generate 25 versions of the 'e' by adding small amounts of noise. Generate another 25 versions of the 'e' by slightly blurring and adding small amounts of noise. Build a printer profile (μ and $\mathbf{e}_1$) from these 50 characters. Subject the printer profile representation to normalized cut clustering.

Example 8.1 The normalized cut (Ncut) for cut_1 is:

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

where $V = \{\alpha, \beta, \gamma\}$, $A = \{\alpha\}$, $B = \{\beta, \gamma\}$, and where:

$$\begin{aligned} \text{cut}(A,B) &= 1+2=3 \\ \text{assoc}(A,V) &= 1+2=3 \\ \text{assoc}(B,V) &= 3+2=5, \end{aligned}$$

to yield an Ncut of $3/3+3/5 = 1.6$. The Ncut for cut_2 is the same as cut_1 because the same valued edges are cut. The Ncut for cut_3 is:

$$\begin{aligned} \text{cut}(A,B) &= 1+1=2 \\ \text{assoc}(A,V) &= 1+1=2 \\ \text{assoc}(B,V) &= 3+3=6, \end{aligned}$$

to yield the minimum Ncut of $2/2 + 2/6 = 1.3$.

1. G.N. Ali, P.-J. Chiang, A.K. Mikkilineni, J.P. Allebach, G.T.-C. Chiu, and E.J. Delp. Intrinsic and extrinsic signatures for information hiding and secure printing with electrophotographic devices. In *International Conference on Digital Printing Technologies*, pages 511515, 2003.

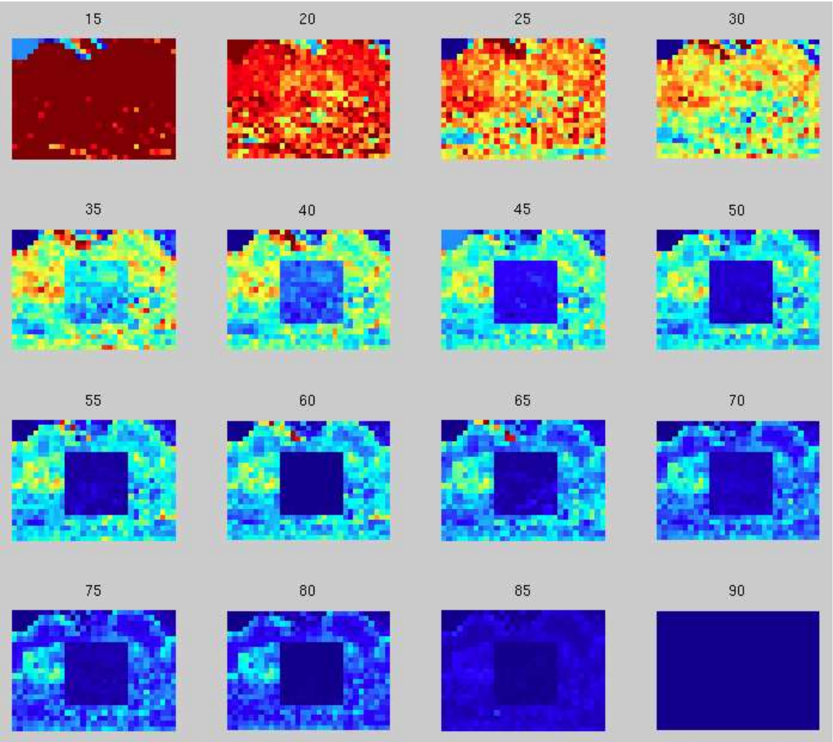
2. O. Bulan, J. Mao, and G. Sharma. Geometric distortion signatures for printer identification. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 14011404, 2009.

3. E. Kee and H. Farid. Printer profiling for forensics and ballistics. In *ACM Multimedia and Security Workshop*, pages 310, 2008.

9. MatLab Code

9.1 JPEG Ghost

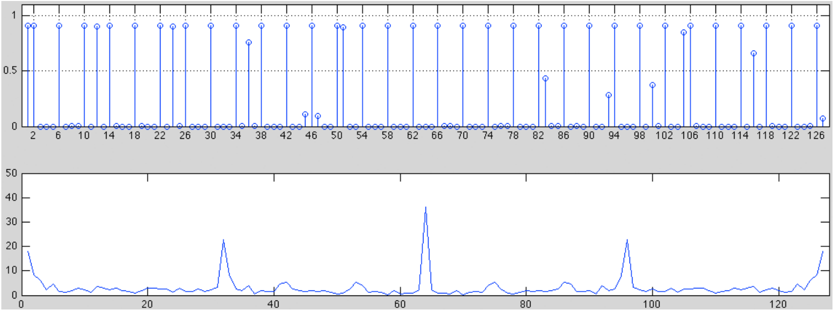
This code generates a seamless forgery in which a central portion (size `sz x sz`) of an image is compressed at a quality of `q2` and inserted into an image of quality `q1` (where `q2` < `q1`). This forgery is then compressed at qualities specified by `qrng`. Shown below is a sample output where the ghost can clearly be seen at quality 60 (and neighboring qualities). The differences are shown on a normalized scale of 0 (blue) to 1 (red). This code assumes that the altered region remains on the original JPEG sampling lattice.



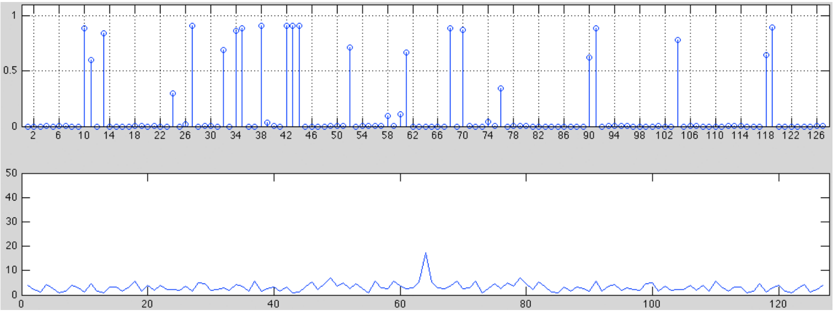
```
clear;

b         = 16; % block size
q1        = 90; % JPEG quality for first image
q2        = 60; % JPEG quality for second image (q2 < q1)
qrng      = [15 : 5 : 90]; % range of qualities to test against
sz        = 200; % size of altered region (in pixels)

%%% LOAD ORIGINAL
im1 = imread( '*** YOUR IMAGE ***' );
[ydim,xdim,zdim] = size(im1);

%%% SAVE WITH DIFFERENT JPEG QUALITIES
imwrite( im1, 'tmp1.jpg', 'quality', q1 );
imwrite( im1, 'tmp2.jpg', 'quality', q2 );

%%% MAKE NEW IMAGE AS  COMBINATION OF im1 AND im2
im1        = imread( 'tmp1.jpg' );
im2        = imread( 'tmp2.jpg' );
xrng       = round( [xdim/2-sz/2 : xdim/2+sz/2-1] ); % altered region
yrng       = round( [ydim/2-sz/2 : ydim/2+sz/2-1] ); % altered region
im1(yrng,xrng,:) = im2(yrng,xrng,:); % splice
imwrite( im1, 'tmp3.jpg', 'quality', q1 ); % re-save

%%% LOAD ALTERED IMAGE
im3 = imread( 'tmp3.jpg' );
[ydim,xdim,zdim] = size(im3);

%%% COMPUTE DIFFERENCE BETWEEN im3 AND JPEG COMPRESSED VERSIONS OF im3
nQ = length(qrng);
map = zeros( ydim, xdim, length(qrng) );
c = 1;
for q = qrng
   imwrite( uint8(im3), 'tmp4.jpg', 'quality', q ) ;
   im4 = double( imread( 'tmp4.jpg' ) );
   for z = 1 : zdim % compute difference: average over RGB
      map(:,:,c) = map(:,:,c) + (double(im3(:,:,z)) - im4(:,:,z)).^2;
   end
   map(:,:,c) = map(:,:,c) / zdim;
   c = c + 1;
end

%%% COMPUTE DIFFERENCE (PER bxb BLOCK)
blkE = zeros( floor((ydim-b)/b), floor((xdim-b)/b), nQ );
for c = 1 : nQ
   cy = 1;
   for y = 1 : b : ydim-b
      cx = 1;
      for x = 1 : b : xdim-b
         bE = map(y:y+b-1,x:x+b-1,c);
         blkE(cy,cx,c) = mean(bE(:));
         cx = cx + 1;
      end
      cy = cy + 1;
   end
end

%%% NORMALIZE DIFFERENCE INTO [0,1]
minval = min( blkE, [], 3 );
maxval = max( blkE, [], 3 );
for c = 1 : nQ
   blkE(:,:,c) = blkE(:,:,c) - minval;
   blkE(:,:,c) = blkE(:,:,c) ./ (maxval-minval);
end

%%% DISPLAY RESULTS
sp = ceil( sqrt(nQ) );
for c = 1 : nQ
   subplot(sp,sp,c);
   imagesc( blkE(:,:,c), [0 1] );
   axis image off;
   title( sprintf( ’%d’, qrng(c) ) );
   drawnow;
end
```

This code generates a synthetic signal where every fourth sample is correlated to their neighbors ($g(x) = 0.5g(x-1) + 0.5g(x+1)$). The EM algorithm is initialized randomly with the correlations (`alpha`) and then iteratively computes the probability that each sample is correlated to its neighbors (`w`), and re-estimates the correlations (`alpha`). On each iteration, the probability is plotted in the upper panel, and the magnitude of the Fourier transform of the probability is plotted in the lower panel. Spikes in the Fourier domain denote periodic correlations.

Shown below is a sample output where EM correctly finds the correlations.

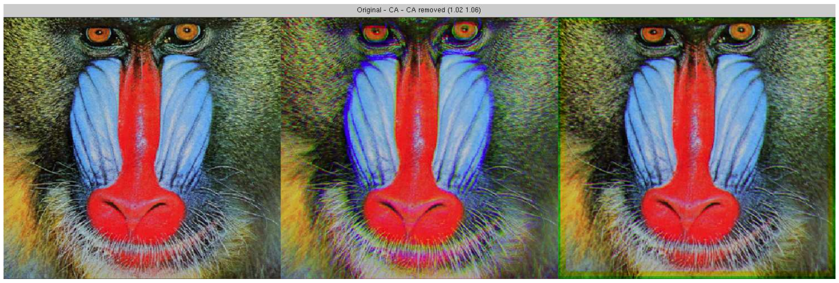


And shown below is a sample output where EM fails to correctly find the correlations (this is typically the result of starting conditions that deviate significantly from the actual correlation).



```
clear;

%%% MAKE SIGNAL
N = 128;
f = rand(1,N);
f = f - min(f);
f = f / max(f);
g = f;

%%% ADD CORRELATION
for k = 2 : 4 : N
    g(k) = 0.5*f(k-1) + 0.5*f(k+1);
end

%%% EM
alpha = rand(2,1); % INITIAL COEFFICIENTS
sigma = 0.005; % VARIANCE ON GAUSSIAN
delta = 10; % UNIFORM
while(1)
    %%% E-STEP
    for k = 2 : N-1
        r(k) = g(k) - (alpha(1)*g(k-1) + alpha(2)*g(k+1)); % RESIDUAL
        w(k) = exp(-r(k)^2/sigma) / (exp(-r(k)^2/sigma) + 1/delta); % PROBABILITY
    end

    %%% PLOT
    subplot(211);
    stem( w );
    set( gca, ’Xtick’, [2:4:N], ’Ytick’, [0 0.5 1] );
    title( sprintf( ’[%.2f %.2f]’, alpha(1), alpha(2) ) );
    axis( [0 N 0 1.1] );
    grid on;
    subplot(212);
    plot(fftshift(abs(fft(w))));
    axis( [0 N 0 50] );
    drawnow; pause(0.25);

    %%% M-STEP
    M = [g(1:N-2)’ g(3:N)’];
    b = g(2:N-1)’;
    r = r(2:end); % remove edge point;
    w = w(2:end); % remove edge point
    W = diag(w);
    alpha_new = inv(M’*W*M)*M’*W*b; % WLS
    if( norm( alpha - alpha_new ) < 0.01 )
        break; % STOPPING CONDITION
    end
    alpha = alpha_new;

    %%% UPDATE SIGMA
    sigma = sum(w.*(r.^2)) /sum(w);
end
```

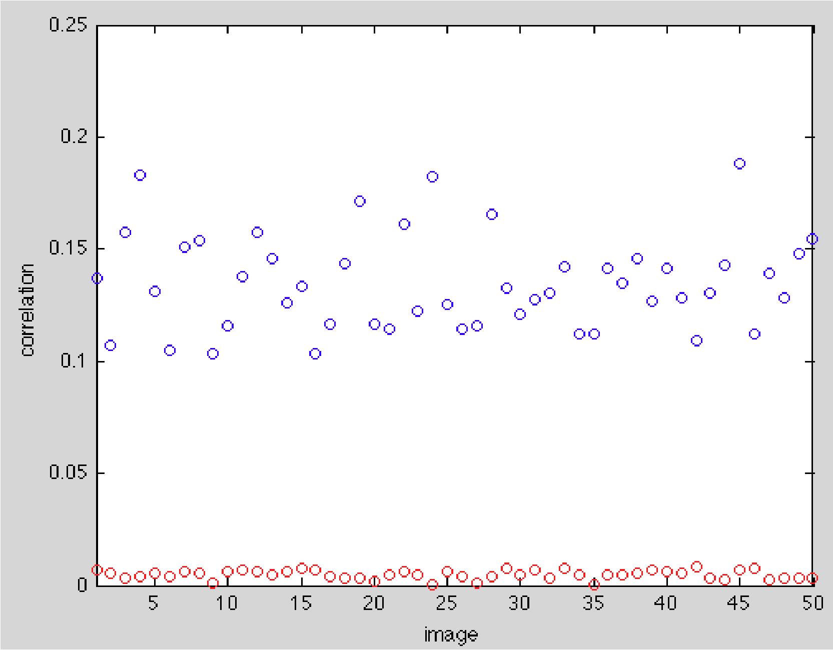
This code estimates chromatic aberration for a one-parameter distortion model (the center of the chromatic distortion is assumed to be the image center). The main function (`ca`) takes as input a color image (`filename`) and the amount of red (`aR`) and blue (`aB`) chromatic aberration to be synthetically introduced into the image. This value of the chromatic aberration should be in the range 0.8 to 1.2. A brute-force search is then performed over this parameter space to maximize the correlation between the red/green and blue/green channels. The output figure shown below consists of the original image, the image with chromatic aberration, and this image after the estimated chromatic aberration is removed. The estimated red and blue chromatic aberration can be found in the image title. If the estimation is correct, then the first and last panel should be the same (except for possible edge artifacts). This code assumes that the center of the chromatic aberration is the image center.



```
function[] = ca( filename, aR, aB )

    %%% LOAD IMAGE AND ADD CHROMATIC ABERRATION TO RED/BLUE CHANNELS
    im          = double( imread( filename ) );
    [ydim,xdim,zdim] = size( im );
    imCA        = im;
    imCA(:,:,1) = scaleim( imCA(:,:,1), aR );
    imCA(:,:,3) = scaleim( imCA(:,:,3), aB );

    %%% ESTIMATE CHROMATIC ABERRATION FOR RED/BLUE CHANNELS BY ALIGNING
    %%% TO GREEN CHANNEL
    k = 1;
    green = imCA(:,:,2);
    for alpha = [0.80 : 0.02 : 1.20] % alpha search range
       red    = scaleim( imCA(:,:,1), 1/alpha );
       ind    = find( red > 0 );
       C(k,1) = corr2( red(ind), green(ind) );

       blue   = scaleim( imCA(:,:,3), 1/alpha );
       ind    = find( blue > 0 );
       C(k,2) = corr2( blue, green );

       C(k,3) = alpha;
       k      = k + 1;
    end

    %%% REMOVE ESTIMATED CHROMATIC ABERRATIONS
    [maxval,maxindR] = max(C(:,1));
    [maxval,maxindG] = max(C(:,2));
    alphaR           = C(maxindR,3);
    alphaG           = C(maxindG,3);
    imCORRECT        = imCA;
    imCORRECT(:,:,1) = scaleim( imCA(:,:,1), 1/alphaR );
    imCORRECT(:,:,3) = scaleim( imCA(:,:,3), 1/alphaG );

    imagesc( uint8([im imCA imCORRECT]) ); axis image off;
    title( sprintf( 'Original - CA - CA removed (%0.2f %0.2f)\n', ...
               alphaR, alphaG ) );

%%% SCALE IMAGE BY A FACTOR OF ALPHA
function[im2] = scaleim( im, alpha )

   [ydim,xdim,zdim] = size(im);

   if( alpha > 1 ) % central crop to match size(im)
      im2 = imresize( im, alpha );
      [ydim2,xdim2,zdim2] = size(im2);
      cy  = floor( (ydim2-ydim)/2 );
      cx  = floor( (xdim2-xdim)/2 );
      im2 = im2( cy+1:ydim2-cy, cx+1:xdim2-cx, : ); % crop
      im2 = imresize( im2, [ydim xdim] ); % deal with off-by-one sizes
   else % pad with zeros to match size(im)
      im2 = imresize( im, alpha );
      [ydim2,xdim2,zdim2] = size( im2 );
      im3 = zeros( ydim, xdim, zdim );
      cy  = floor( (ydim-ydim2)/2 );
      cx  = floor( (xdim-xdim2)/2 );
      im3( [1:ydim2]+cy, [1:xdim2]+cx, : ) = im2;
      im2 = im3;
   end
```

This code generates (`n`) synthetic fractal images images, adds a sensor noise pattern (`K`), and then estimates this noise pattern (`Kest`). This estimated noise pattern is then compared against 50 images with the same and with different noise patterns.

Shown below is a sample output of the code where the blue dots correspond to the correlation for 50 images with the noise pattern (`K`) and the red dots correspond to the correlation for 50 images with a different noise pattern.



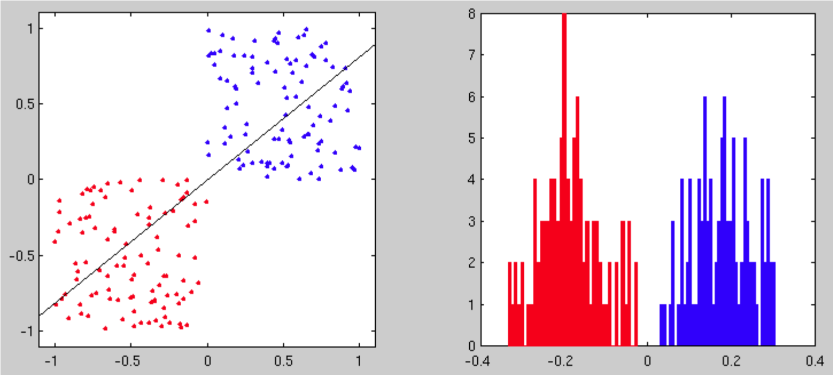
```
function[] = prnu()

%%% NOISE AND IMAGE PARAMETERS
dim = 512;              % size of image (dim x dim)
n   = 10;               % number of images for estimating PRNU
N   = rand(dim,dim,n); % noise (additive)
K   = rand(dim,dim);    % PRNU (multiplicative)
L   = 0.01*rand+0.01;   % multiplicative factor on PRNU

%%% NOISE-CORRUPTED IMAGES
I = zeros(dim,dim,n);
for j = 1 : n
    I0 = mkFract([dim dim]); % noise-free images
    I0 = I0 - min(I0(:));
    I0 = 255 * I0/max(I0(:));
    I(:,:,j) = I0 + L*(I0.*K) + N(:,:,j);
end

%%% DENOISE
for j = 1 : n
    W(:,:,j) = I(:,:,j) - wiener2(I(:,:,j),[5 5]);
end

%%% ESTIMATE PRNU
Kest = sum( W.*I, 3 ) ./ sum( I.*I, 3 );

%%% CORRELATE Kest AGAINST IMAGES WITH SAME PRNU
cla;
axis( [1 50 0 0.25] ); xlabel(’image’); ylabel(’correlation’); box on;
hold on;
for j = 1 : 50
    N     = rand(dim,dim);
    I0    = mkFract([dim dim]);
    I0    = I0 - min(I0(:));
    I0    = 255 * I0/max(I0(:));
    Itest = I0 + L*(I0.*K) + N; % test image (use L and K from above)
    c1(j)  = corr2( Itest.*Kest, Itest-wiener2(Itest) ); % correlation
    plot(j,c1(j),’bo’);
    drawnow;
end

%%% CORRELATE Kest AGAINST IMAGES WITH DIFFERENT PRNU
for j = 1 : 50
    N     = rand(dim,dim);
    K     = rand(dim,dim);
    L     = 0.01*rand+0.01;
    I0    = mkFract([dim dim]);
    I0    = I0 - min(I0(:));
    I0    = 255 * I0/max(I0(:));
    Itest = I0 + L*(I0.*K) + N; % test image
    c2(j)  = corr2( Itest.*Kest, Itest-wiener2(Itest) ); % correlation
    plot(j,c2(j),’ro’);
    drawnow;
end
hold off;

% Make a matrix of dimensions SIZE (a [Y X] 2-vector, or a scalar)
% containing fractal (pink) noise with power spectral density of the
% form: 1/f^(5-2*FRACT_DIM).  Image variance is normalized to 1.0.
% FRACT_DIM defaults to 1.0 (from E. Simoncelli’s matlabPyrTools)
function [res] = mkFract(dims, fract_dim)

if (exist(’fract_dim’) ~= 1)
    fract_dim = 1.0;
end

res = randn(dims);
```

```
fres = fft2(res);

sz = size(res);
ctr = ceil((sz+1)./2);

shape = ifftshift(mkR(sz, -(2.5-fract_dim), ctr));
shape(1,1) = 1;  %%DC term

fres = shape .* fres;
fres = ifft2(fres);

if (max(max(abs(imag(fres)))) > 1e-10)
    error('Symmetry error in creating fractal');
else
    res = real(fres);
    res = res / sqrt(var2(res));
end

% ----------------------------------------------------------------------

% Compute a matrix of dimension SIZE (a [Y X] 2-vector, or a scalar)
% containing samples of a radial ramp function, raised to power EXPT
% (default = 1), with given ORIGIN (default = (size+1)/2, [1 1] =
% upper left).  All but the first argument are optional. (from E.
% Simoncelli's matlabPyrTools)
function [res] = mkR(sz, expt, origin)

sz = sz(:);
if (size(sz,1) == 1)
    sz = [sz,sz];
end
if (exist('expt') ~= 1)
    expt = 1;
end
if (exist('origin') ~= 1)
    origin = (sz+1)/2;
end

[xramp,yramp] = meshgrid( [1:sz(2)]-origin(2), [1:sz(1)]-origin(1) );
res = (xramp.^2 + yramp.^2).^(expt/2);

% ----------------------------------------------------------------------

% Sample variance of a matrix. Passing MEAN (optional) makes the
% calculation faster. (from E. Simoncelli's matlabPyrTools)
function [res] = var2(mtx, mn)

if (exist('mn') ~= 1)
    mn =  mean2(mtx);
end
if (isreal(mtx))
    res = sum(sum(abs(mtx-mn).^2)) / max((prod(size(mtx)) - 1),1);
```

```
else
    res = sum(sum(real(mtx-mn).^2)) + i*sum(sum(imag(mtx-mn).^2));
    res = res  / max((prod(size(mtx)) - 1),1);
end
```

This code generates two synthetic classes of data in a 2-D space and computes the 1-D linear subspace onto which the data are projected in order to perform classification.

Shown in the left panel below is a sample output of the code where the red and blue dots on the left denote the original 2-D data, and the 1-D linear subspace (black line) as computed by LDA. Shown in the right panel are distributions of the data after projection onto the 1-D linear subspace. Note that the data are perfectly separated on either side of the origin.



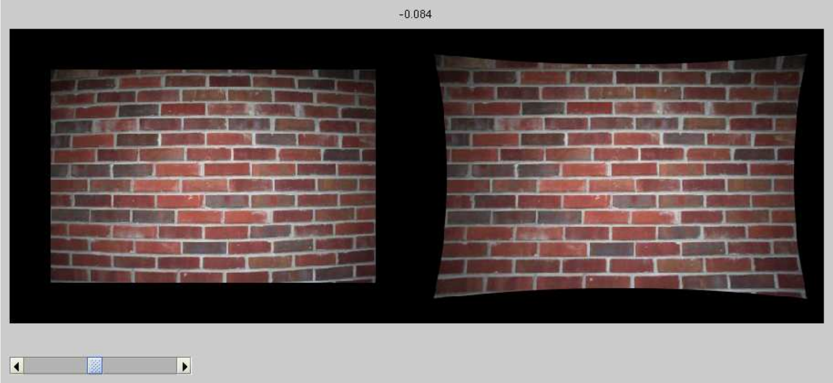
This code (except for the display part) generalizes to multi-class LDA. For example, for a third class `C`, add `C` to the overall mean, compute the mean for class `C`, and compute the between `Sb` and within `Sw` class matrices as follows:

```
mu   = mean( [A ; B ; C] ); % OVERALL MEAN
muC  = mean( C ); % CLASS MEAN
Sb   = size(A,1)*((muA-mu)’*(muA-mu)) + size(B,1)*((muB-mu)’*(muB-mu)) + ...
       size(C,1)*((muC-mu)’*(muB-mu));
Sw   = M1 * M1’ + M2 * M2’ + M3 * M3’; % where  M3(k,:) = C(:,k)’ - muC(k);
```

For three classes, the LDA returns two linear subspaces specified by the largest and second largest generalized eigenvector.

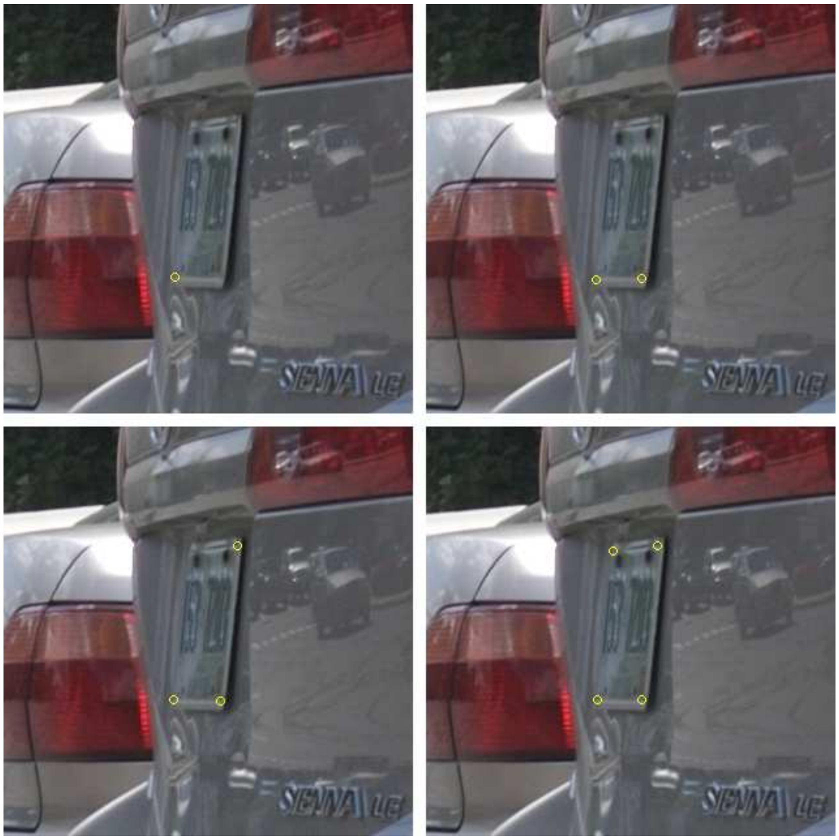
```
clear;

%%% DATA
n   = 100; % NUMBER OF DATA POINTS IN EACH CLASS
N   = 2; % DIMENSIONALITY OF DATA (MUST BE 2 FOR DISPLAY TO WORK)
A   = rand(n,N); % CLASS 1
B   = -rand(n,N); % CLASS 2

%%% BETWEEN (Sb) and WITHIN (Sw) CLASS SCATTER MATRICES
mu  = mean( [A ; B] ); % OVERALL MEAN
muA = mean( A ); % CLASS MEAN
muB = mean( B ); % CLASS MEAN

Sb  = size(A,1)*((muA-mu)'*(muA-mu)) + size(B,1)*((muB-mu)'*(muB-mu)); % BETWEEN CLASS
for k = 1 : N
   M1(k,:) = A(:,k)' - muA(k);
   M2(k,:) = B(:,k)' - muB(k);
end
Sw  = M1 * M1' + M2 * M2'; % WITHIN CLASS

%%% LDA
[vec,val] = eig( Sb, Sw );
[maxval,ind] = max(diag(real(val)));
e = real( vec(:,ind) ); % LINEAR DISCRIMINANT AXIS

%%% PROJECT TRAINING ONTO MAXIMAL EIGENVALUE-EIGENVECTOR
projA = (A * e)';
projB = (B * e)';

%%% DISPLAY
subplot(121); cla; hold on;
h = plot( A(:,1), A(:,2), 'b.' ); set(h,'MarkerSize',14);
h = plot( B(:,1), B(:,2), 'r.' ); set(h,'MarkerSize',14);
h = line( [-10*e(1) 10*e(1)], [-10*e(2) 10*e(2)] ); set( h, 'Color', 'k' );
axis( [-1.1 1.1 -1.1 1.1] ); axis square; box on;
hold off;

subplot(122); cla; hold on;
[N,X] = hist( projA, 40 ); h = bar(X,N); set(h,'FaceColor','b','EdgeColor','b');
[N,X] = hist( projB, 40 ); h = bar(X,N); set(h,'FaceColor','r','EdgeColor','r');
axis square; box on;
hold off;
```

This code removes lens distortion from an image assuming that the principal point is the image center. The slider in the lower left is used to specify the amount of distortion (`K`), and the numeric value is displayed in the image title. The original image is shown on the left and the corrected image is shown on the right.

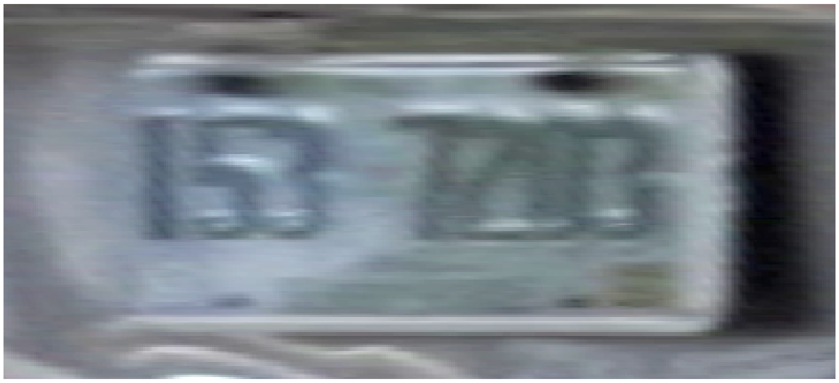


```
function[imK] = lensdistort( filename )

   if( isempty( get( gcf, 'UserData' ) ) )
      h = uicontrol; % GUI to control amount of lens distortion
      set( h, 'Style', 'slider' );
      set( h, 'Position', [20 20 200 20] );
      set( h, 'Value', 0, 'Min', -1, 'Max', 1 );
      set( h, 'SliderStep', [0.001 0.01] );
      set( h, 'Callback', sprintf( 'lensdistort(''%s'');', filename ) );
      set( gcf, 'UserData', h );
   end

   h                = get( gcf, 'UserData' );
   K                = get( h, 'Value' );                          % lens distortion
   im               = double( imread( filename ) );
   [ydim,xdim,zdim] = size( im );
   impad            = zeros( ydim+100, xdim+100, zdim );          % pad with zeros
   impad(51:ydim+50,51:xdim+50,:) = im;                           % pad with zeros
   im               = impad;                                      % pad with zeros
   [ydim,xdim,zdim] = size( im );
   [xramp,yramp]    = meshgrid( [1:xdim]-xdim/2, [1:ydim]-ydim/2 ); % image lattice
   xramp            = xramp / (xdim/2);                           % normalize into [0,1]
   yramp            = yramp / (ydim/2);                           % normalize into [0,1]
   rad              = sqrt( xramp.^2 + yramp.^2 );
   xrampK           = xramp .* (1 + K*rad.^2);                    % new image lattice
   yrampK           = yramp .* (1 + K*rad.^2);                    % new image lattice

   imK = zeros( ydim, xdim, zdim );                               % initialize new image
   for z = 1 : zdim                                               % remove lens distortion
      imK(:,:,z) = interp2( xramp, yramp, im(:,:,z), xrampK, yrampK, 'cubic' );
   end
   imshow( uint8([im imK]) );
   title( K );
```

This code estimates a planar homography of a license plate and warps the image according to the inverse homography. Shown below is the point selection of four points on the image of a license plate (left to right, top to bottom).



And shown below is a magnified and contrast enhanced version of the planar rectified license plate.



```
function[] = rectify()

%%% WORLD
x1(:,1) = [0 0 1]’;
x1(:,2) = [200 0 1]’;
x1(:,3) = [200 100 1]’;
x1(:,4) = [0 100 1]’;

%%% IMAGE
%%% license.jpg available at: www.cs.dartmouth.edu/farid/teaching/trento11/license.jpg
im = imread( ’license.jpg’ );
imshow(im);
x2 = zeros( size(x1) );
for k = 1 : 4
    p = ginput(1);
    plot( p(1), p(2), ’yo’ );
    x2(1:2,k) = p’;
    x2(3,k) = 1;
end

%%% HOMOGRAPHY
H = homography( x1, x2 );
H = H / H(9);

%%% RECTIFY
T1  = maketform( ’projective’, inv(H)’);
im2 = imtransform( im, T1 );
imshow( im2 );
set( gca, ’Ydir’, ’normal’ );

%%% --------------------------------------
function[H] = homography(x1,x2)

[x1, T1] = normalise(x1); % WORLD COORDINATES
[x2, T2] = normalise(x2); % IMAGE COORDINATES

N = length(x1);
A = zeros(3*N,9);
O = [0 0 0];
for n = 1 : N
    X = x1(:,n)’;
    x = x2(1,n);
    y = x2(2,n);
    s = x2(3,n);
    A(3*n-2,:) = [  O  -s*X  y*X];
    A(3*n-1,:) = [ s*X   O  -x*X];
    A(3*n  ,:) = [-y*X  x*X   O ];
end
[U,D,V] = svd(A,0); % TOTAL LEAST-SQUARES
H = reshape(V(:,9),3,3)’;
H = inv(T2)*H*T1;
```

```
%%% -----------------------------------------
function[P2,T] = normalise(P)

P(1,:) = P(1,:)./P(3,:);
P(2,:) = P(2,:)./P(3,:);
P(3,:) = 1;

c = mean(P(1:2,:)')';    % TRANSLATE
P1(1,:) = P(1,:) - c(1);
P1(2,:) = P(2,:) - c(2);

dist = sqrt(P1(1,:).^2 + P1(2,:).^2); % SCALE
scale = sqrt(2)/mean(dist);

T = [scale   0   -scale*c(1)
    0     scale -scale*c(2)
    0       0      1      ];

P2 = T*P; % NORMALIZE POINTS
```

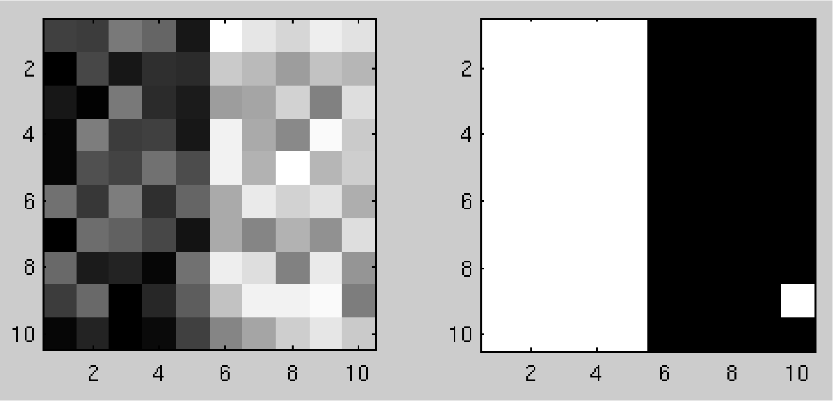
This code is for video stabilization and enhancement. The function `videostabilize` takes as input: (1) a data structure `frames` with field `im` that contains the original video sequence; (2) the number of Gaussian pyramid levels `L`; and (3) a binary image `roi` that specifies the region of interest, with value of 1 for those pixels to be considered in the motion estimation, and 0 otherwise. It is assumed that each frame is a grayscale image. The output of this function is a data structure `stable` with field `im` that contains the stabilized video sequence, and a data structure `motion` with fields `A` and `T` that contain the estimated affine and translation parameters.

The function `videoenhance` takes as input a data structure `stable` with field `im` that contains a stabilized video sequence (the output of `videostabilize`). The outputs of this function are two images `tempmean` and `tempmedian` that contain the results of temporally filtering the stabilized video.

```
%%% STABILIZE VIDEO
function[ motion, stable ] = videostabilize( frames, roi, L )

   N = length( frames );
   roiorig = roi;

   %%% ESTIMATE PAIRWISE MOTION
   Acum = [1 0 ; 0 1];
   Tcum = [0 ; 0];
   stable(1).roi = roiorig;
   for k = 1 : N-1
      [A,T] = opticalflow( frames(k+1).im, frames(k).im, roi, L );
      motion(k).A = A;
      motion(k).T = T;
      [Acum,Tcum] = accumulatewarp( Acum, Tcum, A, T );
      roi = warp( roiorig, Acum, Tcum );
   end

   %%% STABILIZE TO LAST FRAME
   stable(N).im = frames(N).im;
   Acum = [1 0 ; 0 1];
   Tcum = [0 ; 0];
   for k = N-1 : -1 : 1
      [Acum,Tcum] = accumulatewarp( Acum, Tcum, motion(k).A, motion(k).T );
      stable(k).im = warp( frames(k).im, Acum, Tcum );
   end

% -------------------------------------------------------------------------
```

```
%%% ALIGN TWO FRAMES (f2 to f1)
function[ Acum, Tcum ] = opticalflow( f1, f2, roi, L )

   f2orig = f2;
   Acum = [1 0 ; 0 1];
   Tcum = [0 ; 0];

   for k = L : -1 : 0
      %%% DOWN-SAMPLE
      f1d = down( f1, k );
      f2d = down( f2, k );
      ROI = down( roi, k );

      %%% COMPUTE MOTION
      [Fx,Fy,Ft] = spacetimederiv( f1d, f2d );
      [A,T] = computemotion( Fx, Fy, Ft, ROI );
      T = (2^k) * T;
      [Acum,Tcum] = accumulatewarp( Acum, Tcum, A, T );

      %%% WARP ACCORDING TO ESTIMATED MOTION
      f2 = warp( f2orig, Acum, Tcum );
   end
% --------------------------------------------------------------------------

%%% COMPUTE MOTION
function[ A, T ] = computemotion( fx, fy, ft, roi )

   [ydim,xdim] = size(fx);
   [x,y] = meshgrid( [1:xdim]-xdim/2, [1:ydim]-ydim/2 );

   %%% TRIM EDGES
   fx   = fx( 3:end-2, 3:end-2 );
   fy   = fy( 3:end-2, 3:end-2 );
   ft   = ft( 3:end-2, 3:end-2 );
   roi  = roi( 3:end-2, 3:end-2 );
   x    = x( 3:end-2, 3:end-2 );
   y    = y( 3:end-2, 3:end-2 );

   ind  = find( roi > 0 );
   x    = x(ind); y = y(ind);
   fx   = fx(ind); fy = fy(ind); ft = ft(ind);
   xfx  = x.*fx; xfy = x.*fy; yfx  = y.*fx; yfy = y.*fy;

   M(1,1) = sum( xfx .* xfx ); M(1,2) = sum( xfx .* yfx ); M(1,3) = sum( xfx .* xfy );
   M(1,4) = sum( xfx .* yfy ); M(1,5) = sum( xfx .* fx );  M(1,6) = sum( xfx .* fy );
   M(2,1) = M(1,2);            M(2,2) = sum( yfx .* yfx ); M(2,3) = sum( yfx .* xfy );
   M(2,4) = sum( yfx .* yfy ); M(2,5) = sum( yfx .* fx );  M(2,6) = sum( yfx .* fy );
   M(3,1) = M(1,3);            M(3,2) = M(2,3);            M(3,3) = sum( xfy .* xfy );
   M(3,4) = sum( xfy .* yfy ); M(3,5) = sum( xfy .* fx );  M(3,6) = sum( xfy .* fy );
   M(4,1) = M(1,4);            M(4,2) = M(2,4);            M(4,3) = M(3,4);
   M(4,4) = sum( yfy .* yfy ); M(4,5) = sum( yfy .* fx );  M(4,6) = sum( yfy .* fy );
   M(5,1) = M(1,5);            M(5,2) = M(2,5);            M(5,3) = M(3,5);
   M(5,4) = M(4,5);            M(5,5) = sum( fx .* fx );   M(5,6) = sum( fx .* fy );
   M(6,1) = M(1,6);            M(6,2) = M(2,6);            M(6,3) = M(3,6);
   M(6,4) = M(4,6);            M(6,5) = M(5,6);            M(6,6) = sum( fy .* fy );

   k = ft + xfx + yfy;
   b(1) = sum( k .* xfx );     b(2) = sum( k .* yfx );
   b(3) = sum( k .* xfy );     b(4) = sum( k .* yfy );
   b(5) = sum( k .* fx );      b(6) = sum( k .* fy );

   v = inv(M) * b';
   A = [v(1) v(2) ; v(3) v(4)];
   T = [v(5) ; v(6)];

% -------------------------------------------------------------------------

%%% WARP IMAGE
function[ f2 ] = warp( f, A, T )

   [ydim,xdim] = size( f );
   [xramp,yramp] = meshgrid( [1:xdim]-xdim/2, [1:ydim]-ydim/2 );

   P       = [xramp(:)' ; yramp(:)'];
   P       = A * P;
   xramp2  = reshape( P(1,:), ydim, xdim ) + T(1);
   yramp2  = reshape( P(2,:), ydim, xdim ) + T(2);
   f2      = interp2( xramp, yramp, f, xramp2, yramp2, 'bicubic' ); % warp
   ind     = find( isnan(f2) );
   f2(ind) = 0;

% -------------------------------------------------------------------------

%%% BLUR AND DOWNSAMPLE (L times)
function[ f ]  = down( f, L );

   blur = [1 2 1]/4;
   for k = 1 : L
      f = conv2( conv2( f, blur, 'same' ), blur', 'same' );
      f = f(1:2:end,1:2:end);
   end

% -------------------------------------------------------------------------

%%% SPACE/TIME DERIVATIVES
function[ fx, fy, ft ] = spacetimederiv( f1, f2 )

   %%% DERIVATIVE FILTERS
   pre   = [0.5 0.5];
   deriv = [0.5 -0.5];

   %%% SPACE/TIME DERIVATIVES
   fpt = pre(1)*f1 + pre(2)*f2; % pre-filter in time
   fdt = deriv(1)*f1 + deriv(2)*f2; % differentiate in time

   fx = conv2( conv2( fpt, pre', 'same' ), deriv, 'same' );
   fy = conv2( conv2( fpt, pre, 'same' ), deriv', 'same' );
   ft = conv2( conv2( fdt, pre', 'same' ), pre, 'same' );

% -------------------------------------------------------------------------

%%% ACCUMULATE WARPS
function[ A2, T2 ] = accumulatewarp( Acum, Tcum, A, T )

   A2 = A * Acum;
   T2 = A*Tcum + T;

% -------------------------------------------------------------------------

%%% ENHANCE STABILIZED VIDEO
function[ tempmean, tempmedian ] = videoenhance( stable )

   N = length( stable );
   [ydim,xdim] = size( stable(1).im );

   %%% BUILD A 3-D IMAGE STACK FROM THE INPUT SEQUENCE
   stack = zeros( ydim, xdim, N )
   for k = 1 : N
      stack(:,:,k) = stable(k).im;
   end

   %%% FILTER
   tempmean   = mean( stack, 3 );
   tempmedian = median( stack, 3 );
```

This code implements clustering based on normalized cuts. The code generates a `N x N` grayscale image whose left side is, on average, darker than the right side. The weighting matrix `W` is based on the difference in intensity between each pixel. Shown below is a sample input (left) and output (right). The output is a binary image where white corresponds to one cluster and black to another (one pixel on the far right is mis-classified because its intensity is more similar to the left side of the image).



```
clear;

N  = 10;                     % image dimensions (NxN)
I1 = 0.5*rand(N,N/2);        % left half of image
I2 = 0.5*rand(N,N/2) + 0.5;  % right half of image
I  = [I1 I2];                % image
W  = zeros(N^2,N^2);         % weighting matrix
D  = zeros(N^2,N^2);
for j = 1 : N^2
   for k = 1 : N^2
      W(j,k) = exp(-(I(j)-I(k)).^2/1);
   end
end
for j = 1 : N^2
   D(j,j) = sum(W(j,:));
end
[eigvec,eigval] = eig( D-W, D );        % generalized eigenvector
[val,ind]   = sort( diag(eigval) );     % sort eigenvalues
cluster     = sign( eigvec(:,ind(2)) ); % 2nd smallest eigenvector
cluster     = reshape( cluster, N, N );

subplot(121); imagesc(I); axis image; colormap gray;
subplot(122); imagesc(cluster); axis image; colormap gray;
```

Bibliography

[1] Donald Adjeroh, Deng Cao, Marco Piccirilli, and Arun Ross. Predictability and correlation in human metrology. In *Workshop on Information Forensics and Security*, 2010.

[2] Yoshinori Akao, Kazuhiko Kobayashi, Shigeru Sugawara, and Yoko Seki. Discrimination of inkjet-printed counterfeits by spur marks and feature extraction by spatial frequency analysis. In *SPIE Conference on Optical Security and Counterfeit Deterrence Techniques IV*, pages 129–137, 2002.

[3] Gazi N. Ali, Pei-Ju Chiang, Aravind K. Mikkilineni, Jan P. Allebach, George T.-C. Chiu, and Edward J. Delp. Intrinsic and extrinsic signatures for information hiding and secure printing with electrophotographic devices. In *International Conference on Digital Printing Technologies*, pages 511–515, 2003.

[4] Gazi N. Ali, Pei-Ju Chiang, Aravind K. Mikkilineni, George T.-C. Chiu, Edward J. Delp, and Jan P. Allebach. Application of principal components analysis and gaussian mixture models to printer identification. In *International Conference on Digital Printing Technologies*, pages 301–305, 2004.

[5] Erwin J. Alles, Zeno J. Geradts, and Cor. J. Veenman. Source camera identification for low resolution heavily compressed images. In *International Conference on Computational Sciences and Its Applications*, pages 557–567, 2008.

[6] Erwin J. Alles, Zeno J. Geradts, and Cor. J. Veenman. Source camera identification for heavily JPEG compressed low resolution still images. *Journal of Forensic Sciences*, 54(3):628–638, 2009.

[7] Paul Alvarez. Using extended file information (exif) file headers in digital evidence analysis. *International Journal of Digital Evidence*, 2(3):1–5, 2004.

[8] Irene Amerini, Lamberto Ballan, Roberto Caldelli, Alberto Del Bimbo, and Giuseppe Serra. Geometric tampering estimation by means of a sift-based forensic analysis. In *International Conference on Acoustics, Speech, and Signal Processing*, pages 1702–1705, 2010.

[9] Irene Amerini, Lamberto Ballan, Roberto Caldelli, Alberto Del Bimbo, and Giuseppe Serra. A sift-based forensic method for copymove attack detection and transformation recovery. *IEEE Transactions on Information Forensics and Security*, 6(3):1099–1110, 2011.

[10] Irene Amerini, Roberto Caldelli, Vito Cappellini, Francesco Picchioni, and Alessandro Piva. Analysis of denoising filters for photo response non uniformity noise extraction in source camera identification. In *International Conference on Digital Signal Processing*, pages 511–517, 2009.

[11] Irene Amerini, Roberto Caldelli, Vito Cappellini, Francesco Picchioni, and Alessandro Piva. Estimate of prnu noise based on different noise models for source camera identification. *International Journal of Digital Crime and Forensics*, 2(2):21–33, 2010.

[12] Fernanda A. Andalo, Gabriel Taubin, and Siome Goldenstein. Detecting vanishing points by segment clustering on the projective plane for single-view photogrammetry. In *Workshop on Information Forensics and Security*, 2010.

[13] Edoardo Ardizzone, Alessandro Bruno, and Giuseppe Mazzola. Detecting multiple copies in tampered images. In *International Conference on Image Processing*, 2010.

[14] Edoardo Ardizzone and Giuseppe Mazzola. Detection of duplicated regions in tampered digital images by bit-plane analysis. In *Image Analysis and Processing*, pages 893–901, 2009.

[15] Osman Arslan, Roy M. Kumontoy, Pei-Ju Chiang, Aravind K. Mikkilineni, Jan P. Allebach, George T.-C. Chiu, and Edward J. Delp. Identification of inkjet printers for forensic applications. In *International Conference on Digital Printing Technologies*, pages 235–238, 2005.

[16] Ismail Avcibas, Sevinc Bayram, Nasir Memon, Mahalingam Ramkumar, and Bulent Sankur. A classifier design for detecting image manipulations. In *IEEE International Conference on Image Processing*, pages 2645–2648, 2004.

[17] Mauro Barni, Andrea Costanzo, and Lara Sabatini. Identification of cut & paste tampering by means of double-JPEG detection and image segmentation. In *International Symposium on Circuits and Systems*, 2010.

[18] Md. Khayrul Bashar, Keiji Noda, Noboru Ohnishi, Hiroaki Kudo, Tetsuya Matsumoto, and Yoshinori Takeuchi. Wavelet-based multiresolution features for detecting duplications in images. In *Machine Vision Application*, pages 264–267, 2007.

[19] Philip Bateman, Anthony T. S. Ho, and Alan Woodward. Image forensics of digital cameras by analysing image variations using statistical process control. In *International Conference on Information, Communications and Signal Processing*, 2009.

[20] Philip Bateman, Anthony T. S. Ho, and Alan Woodward. Accurate detection of out-of-control variations from digital camera devices. In *IEEE International Symposium on Circuits and Systems*, 2010.

[21] Sebastiano Battiato and Giuseppe Messina. Digital forgery estimation into DCT domain: A critical analysis. In *ACM Multimedia and Security Workshop*, pages 37–42, 2009.

[22] Federica Battisti, Marco Carli, and Alessandro Neri. Image forgery detection by means of no-reference quality metrics. In *SPIE Conference on Media Waternarking, Security, and Forensics*, pages 8303–17, 2012.

[23] Sevinc Bayram, Ismail Avcibas, Bulent Sankur, and Nasir Memon. Image manipulation detection with binary similarity measures. In *European Signal Processing Conference*, pages 752–755, 2005.

[24] Sevinc Bayram, Ismail Avcibas, Bulent Sankur, and Nasir Memon. Image manipulation detection. *Journal of Electronic Imaging*, 15(4):041102, 2006.

[25] Sevinc Bayram, Husrev T. Sencar, and Nasir Memon. Improvements on source camera model identification based on cfa interpolation. In *International Conference on Digital Forensics*, 2006.

[26] Sevinc Bayram, Husrev T. Sencar, and Nasir Memon. Classification of digital camera-models based on demosaicing artifacts. *Digital Investigation*, 5:46–59, 2008.

[27] Sevinc Bayram, Husrev T. Sencar, and Nasir Memon. A survey of copy-move forgery detection techniques. In *IEEE Western New York Image Processing Workshop*, 2008.

[28] Sevinc Bayram, Husrev T. Sencar, and Nasir Memon. Video copy detection based on source device characteristics: A complementary approach to content-based methods. In *ACM International Conference on Multimedia Information Retrieval*, pages 435–442, 2008.

[29] Sevinc Bayram, Husrev T. Sencar, and Nasir Memon. An efficient and robust method for detecting copy-move forgery. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 1053–1056, 2009.

[30] Sevinc Bayram, Husrev T. Sencar, and Nasir Memon. Efficient techniques for sensor fingerprint matching in large image and video databases. In *SPIE Conference on Media Forensics and Security*, 2010.

[31] Sevinc Bayram, Husrev T. Sencar, Nasir Memon, and Ismail Avcibas. Source camera identification based on cfa interpolation. In *IEEE International Conference on Image Processing*, 2005.

[32] Tiziano Bianchi and Alessandro Piva. Analysis of non-aligned double JPEG artifacts for the localization of image forgeries. In *Workshop on Information Forensics and Security*, 2011.

[33] Tiziano Bianchi and Alessandro Piva. Detection of non-aligned double JPEG compression with estimation of primary compression parameters. In *International Conference on Image Processing*, 2011.

[34] Tiziano Bianchi, Alessia De Rosa, and Alessandro Piva. Improved DCT coefficient analysis for forgery localization in JPEG images. In *International Conference on Acoustics, Speech and Signal Processing*, 2011.

[35] Jurrien Bijhold, Arnout Ruifrok, Michael Jessen, Zeno J. Geradts, Sabine Ehrhardt, and Ivo Alberink. Forensic audio and visual evidence. In *International Forensic Science Symposium*, pages 372–413, 2007.

[36] Greg J. Bloy. Blind camera fingerprinting and image clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(3):532–535, 2008.

[37] Rainer Böhme, Felix Freiling, Thomas Gloe, and Matthias Kirchner. Multimedia forensics is not computer forensics. In Zeno J. Geradts, Katrin Y. Franke, and Cor. J. Veenman, editors, *Third International Workshop on Computational Forensics*, pages 90–103, 2009.

[38] Sergio Bravo-Solorio and Asoke K. Nandi. Passive method for detecting duplicated regions affected by reflection, rotation and scaling. In *European Signal Processing Conference*, pages 824–828, 2009.

[39] Sergio Bravo-Solorio and Asoke K. Nandi. Automated detection and localisation of duplicated regions affected by reflection, rotation and scaling in image forensics. *Signal Processing*, 91(8):1759–1770, 2011.

[40] Eddy B. Brixen. Further investigation into the enf criterion for forensic authentication. In *123rd AES Convention*, 2007.

[41] Eddy B. Brixen. Techniques for the authentication of digital audio recordings. In *123rd AES Convention*, 2007.

[42] Eddy B. Brixen. Enf; quantification of the magnetic field. In *32nd International Conference on Audio Engineering*, 2008.

[43] Eddy B. Brixen. How to extract the enf from digital audio recordings. In *ACFEI National Conference*, 2008.

[44] Dino A. Brugioni. *Photo Fakery: The History and Techniques of Photographic Deception and Manipulation*. Brassey's, 1999.

[45] Robert Buchholz, Christian Krätzer, and Jana Dittmann. Microphone classification using fourier coefficients. In Stefan Katzenbeisser and Ahmad-Reza Sadeghi, editors, *International Workshop on Information Hiding*, pages 235–246, 2009.

[46] Orhan Bulan, Junwen Mao, and Gaurav Sharma. Geometric distortion signatures for printer identification. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 1401–1404, 2009.

[47] Roberto Caldelli, Irene Amerini, and Francesco Picchioni. Distinguishing between camera and scanned images by means of frequency analysis. In Matthew Sorell, editor, *International Conference on Forensics in Telecommunications, Information and Multimedia*, 2009.

[48] Roberto Caldelli, Irene Amerini, and Francesco Picchioni. A dft-based analysis to discern between camera and scanned images. *International Journal of Digital Crime and Forensics*, 2(1):21–29, 2010.

[49] Roberto Caldelli, Irene Amerini, Francesco Picchioni, and Matteo Innocenti. Fast image clustering of unknown source images. In *Workshop on Information Forensics and Security*, 2010.

[50] Roberto Caldelli, Irene Amerini, Francesco Picchioni, Alessia De Rosa, and Vito Cappellini. Multimedia forensics: Technologies for imagery investigation. In *Electronic Information, the Visual Arts and Beyond*, pages 87–92, 2009.

[51] Roberto Caldelli, Irene Amerini, Francesco Picchioni, Alessia De Rosa, and Francesca Uccheddu. *Handbook of Research on Computational Forensics, Digital Crime and Investigation: Methods and Solutions*, chapter Multimedia Forensic Techniques for Acquisition Device Identification and Digital Image Authentication, pages 130–154. Information Science Reference, 2009.

[52] Gang Cao, Yao Zhao, and Rongrong Ni. Image composition detection using object-based color consistency. In *IEEE International Conference on Signal Processing*, pages 1186–1189, 2008.

[53] Gang Cao, Yao Zhao, and Rongrong Ni. Detection of image sharpening based on histogram aberration and ringing artifacts. In *IEEE International Conference on Multimedia and Expo*, pages 1026–1029, 2009.

[54] Gang Cao, Yao Zhao, and Rongrong Ni. Edge-based blur metric for tamper detection. *Journal of Information Hiding and Multimedia Signal Processing*, 1(1):20–27, 2010.

[55] Gang Cao, Yao Zhao, and Rongrong Ni. Forensic estimation of gamma correction in digital images. In *International Conference on Image Processing*, 2010.

[56] Gang Cao, Yao Zhao, and Rongrong Ni. Forensic identification of resampling operators: A semi non-intrusive approach. *Forensic Science International*, 2011.

[57] Gang Cao, Yao Zhao, Rongrong Ni, and Alex C. Kot. Unsharp masking sharpening detection via overshoot artifacts analysis. *IEEE Signal Processing Letters*, 18(10):603–606, 2011.

[58] Gang Cao, Yao Zhao, Rongrong Ni, and Huawei Tian. Anti-forensics of contrast enhancement in digital images. In *ACM Multimedia and Security Workshop*, pages 25–34, 2010.

[59] Gang Cao, Yao Zhao, Rongrong Ni, Lifang Yu, and Huawei Tian. Forensic detection of median filtering in digital images. In *IEEE International Conference on Multimedia and Expo*, 2010.

[60] Hong Cao and Alex C. Kot. A generalized model for detection of demosaicing characteristics. In *IEEE International Conference on Multimedia and Expo*, pages 1513–1516, 2008.

[61] Hong Cao and Alex C. Kot. Accurate detection of demosaicing regularity for digital image forensics. *IEEE Transactions on Information Forensics and Security*, 4(4):899–910, 2009.

[62] Hong Cao and Alex C. Kot. Raw tool identification through detected demosaicing regularity. In *IEEE International Conference on Image Processing*, pages 2885–2888, 2009.

[63] Hong Cao and Alex C. Kot. Detection of tampering inconsistencies on mobile photos. In *International Workshop on Digital Watermarking*, pages 105–119, 2010.

[64] Hong Cao and Alex C. Kot. Identification of recaptured photographs on lcd screens. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 1790–1793, 2010.

[65] Hong Cao and Alex C. Kot. Mobile camera identification using demosaicing features. In *International Symposium on Circuits and Systems*, 2010.

[66] Hong Cao and Alex C. Kot. Similar dslr processor identification using compact model templates. In *Asia Pacific Signal and Information Processing Association*, 2011.

[67] Oya Celiktutan, Ismail Avcibas, and Bulent Sankur. Blind identification of cellular phone cameras. In Edward J. Delp and Ping Wah Wong, editors, *SPIE Conference on Security, Steganography, and Watermarking of Multimedia Contents*, volume 6505, 2007.

[68] Oya Celiktutan, Ismail Avcibas, Bulent Sankur, and Nasir Memon. Source cell-phone identification. *IEEE Signal Processing and Communications Applications*, pages 1–3, April 2006.

[69] Oya Celiktutan, Bulent Sankur, and Ismail Avcibas. Blind identification of source cell-phone model. *IEEE Transactions on Information Forensics and Security*, 3(3):553–566, 2008.

[70] Shih-Fu Chang. Blind passive media forensics: Motivation and opportunity. In *Multimedia Content Analysis and Mining*, pages 57–59, 2007.

[71] Dongmei Chen, Jianhua Li, Shilin Wang, and Shenghong Li. Identifying computer generated and digital camera images using fractional lower order moments. In *IEEE Conference on Industrial Electronics and Applications*, pages 230–235, 2009.

[72] Mo Chen, Jessica Fridrich, and Miroslav Goljan. Digital imaging sensor identification (further study). In Edward J. Delp and Ping Wah Wong, editors, *SPIE Conference on Security, Steganography, and Watermarking of Multimedia Contents*, volume 6505, 2007.

[73] Mo Chen, Jessica Fridrich, Miroslav Goljan, and Jan Lukas. Source digital camcorder identification using sensor photo-response non-uniformity. In Edward J. Delp and Ping Wah Wong, editors, *SPIE Conference on Security, Steganography, and Watermarking of Multimedia Contents*, volume 6505, 2007.

[74] Mo Chen, Jessica Fridrich, Miroslav Goljan, and Jan Lukas. Determining image origin and integrity using sensor noise. *IEEE Transactions on Information Forensics and Security*, 3(1):74–90, 2008.

[75] Sz-Han Chen and Chiou-Ting Hsu. Source camera identification based on camera gain histogram. In *IEEE International Conference on Image Processing*, volume 4, pages IV–429–IV–432, 2007.

[76] Wen Chen and Yun Qing Shi. Detection of double mpeg video compression using first digits statistics. In *International Workshop on Digital Watermarking*, 2008.

[77] Wen Chen, Yun Qing Shi, and Wei Su. Image splicing detection using 2-d phase congruency and statistical moments of characteristic function. In Edward J. Delp and Ping Wah Wong, editors, *SPIE Conference on Security, Steganography, and Watermarking of Multimedia Contents*, volume 6505, 2007.

[78] Wen Chen, Yun Qing Shi, and Guo Rong Xuan. Identifying computer graphics using hsv color model and statistical moments of characteristic functions. In *IEEE International Conference on Multimedia and Expo*, 2007.

[79] Wen Chen, Yun Qing Shi, Guo Rong Xuan, and Wei Su. Computer graphics identification using genetic algorithm. In *IEEE International Conference on Pattern Recognition*, 2008.

[80] Yi-Lei Chen and Chiou-Ting Hsu. Image tampering detection by blocking periodicity analysis in JPEG compressed images. In *IEEE Workshop on Multimedia Signal Processing*, pages 803–808, 2008.

[81] Yi-Lei Chen and Chiou-Ting Hsu. Detecting doubly compressed images based on quantization noise model and image restoration. In *IEEE Workshop on Multimedia Signal Processing*, 2009.

[82] Yi-Lei Chen and Chiou-Ting Hsu. Detecting recompression of JPEG images via periodicity analysis of compression artifacts for tampering detection. *IEEE Transactions on Information Forensics and Security*, 6(2):396–406, 2011.

[83] H. R. Chennamma and Lalitha Rangarajan. Image splicing detection using inherent lens radial distortion. *International Journal of Computer Science Issues*, 7(6):149–158, 2010.

[84] H. R. Chennamma and Lalitha Rangarajan. Source camera identification based on sensor readout noise. *International Journal of Digital Crime and Forensics*, 2(3):28–42, 2010.

[85] H. R. Chennamma, Lalitha Rangarajan, and M. S. Rao. Robust near duplicate image matching for digital image forensics. *International Journal of Digital Crime and Forensics*, 1(3):62–79, 2009.

[86] H. R. Chennamma, Lalitha Rangarajan, and M. S. Rao. *New Technologies for Digital Crime and Forensics: Devices, Applications and Software*, chapter Efficient Image Matching using Local Invariant Features for Copy Detection. IGI Global Publisher, 2010.

[87] Girija Chetty and Monica Singh. Nonintrusive image tamper detection based on fuzzy fusion. *International Journal of Computer Science and Network Security*, 10(9):86–90, 2010.

[88] Pei-Ju Chiang, Nitin Khanna, Aravind K. Mikkilineni, Maria V. Ortiz Segovia, Sungjoo Suh, Jan P. Allebach, George T.-C. Chiu, and Edward J. Delp. Printer and scanner forensics. *IEEE Signal Processing Magazine*, 26(2):72–83, 2009.

[89] Chang-Hee Choi, Min-Jeong Lee, and Heung-Kyu Lee. Scanner identification using spectral noise in the frequency domain. In *International Conference on Image Processing*, 2010.

[90] Jung-Ho Choi, Dong-Hyuck Im, Hae-Yeoun Lee, Jun-Taek Oh, Jin-Ho Ryu, and Heung-Kyu Lee. Color laser printer identification by analyzing statistical features on discrete wavelet transform. In *IEEE International Conference on Image Processing*, pages 1505–1508, 2009.

[91] Jung-Ho Choi, Hae-Yeoun Lee, and Heung-Kyu Lee. Color laser printer forensics with noise texture analysis. In *ACM Workshop on Multimedia and Security*, pages 19–24, 2010.

[92] Kai San Choi, Edmund Y. Lam, and Kenneth K. Y. Wong. Automatic source camera identification using intrinsic lens radial distortion. *Optics Express*, 14(24):11551–11565, 2006.

[93] Kai San Choi, Edmund Y. Lam, and Kenneth K. Y. Wong. Feature selection in source camera identification. In *IEEE International Conference on Systems, Man and Cybernetics*, pages 3176–3180, 2006.

[94] Kai San Choi, Edmund Y. Lam, and Kenneth K. Y. Wong. Source camera identification by JPEG compression statistics for image forensics. In *IEEE Region 10 Conference (TENCON 2006)*, 2006.

[95] Kai San Choi, Edmund Y. Lam, and Kenneth K. Y. Wong. Source camera identification using footprints from lens aberration. In Nitin Sampat, Jeffrey M. DiCarlo, and Russel A. Martin, editors, *SPIE Conference on Digital Photography*, volume 6069, 2006.

[96] Vincent Christlein, Christian Riess, and Elli Angelopoulou. On rotation invariance in copy-move forgery detection. In *Workshop on Information Forensics and Security*, 2010.

[97] Vincent Christlein, Christian Riess, and Elli Angelopoulou. A study on features for the detection of copy-move forgeries. In *Information Security Solutions Europe*, 2010.

[98] Wei-Hong Chuang, Hui Su, and Min Wu. Exploring compression effects for improved source camera identification using strongly compressed video. In *International Conference on Image Processing*, 2011.

[99] Wei-Hong Chuang, Ashwin Swaminathan, and Min Wu. Tampering identification using empirical frequency response. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 1517–1520, apr 2009.

[100] Wei-Hong Chuang and Min Wu. Semi non-intrusive training for cell-phone camera model linkage. In *Workshop on Information Forensics and Security*, 2010.

[101] Chen Chunhua, Yun Qing Shi, and Su Wei. A machine learning based scheme for double JPEG compression detection. In *International Conference on Pattern Recognition*, pages 1–4, 2008.

[102] Valentina Conotter, Giulia Boato, and Hany Farid. Detecting photo manipulation on signs and billboards. In *International Conference on Image Processing*, 2010.

[103] Valentina Conotter and Lorenzo Cordin. Detecting photographic and computer generated composites. In *SPIE Conference on Media Forensics and Security*, 2011.

[104] Andrea Cortiana, Valentina Conotter, Giulia Boato, and Francesco G. B. De Natale. Performance comparison of denoising filters for source camera identification. In *SPIE Conference on Media Watermarking, Security, and Forensics*, 2011.

[105] Brian D'Alessandro and Yun Qing Shi. MP3 bit rate quality detection through frequency spectrum analysis. In *ACM Multimedia and Security Workshop*, pages 57–62, 2009.

[106] Nahuel Dalgaard, Carlos Mosquera, and Fernando Perez-Gonzalez. On the role of differentiation for resampling detection. In *International Conference on Image Processing*, 2010.

[107] Sintayehu Dehnie, Husrev T. Sencar, and Nasir Memon. Digital image forensics for identifying computer generated and digital camera images. In *IEEE International Conference on Image Processing*, pages 2313–2316, 2006.

[108] Wei Deng, Qinghu Chen, Feng Yuan, and Yuchen Yan. Printer identification based on distance transform. In *International Conference on Intelligent Networks and Intelligent Systems*, pages 565–568, 2008.

[109] Zanoni Dias, Anderson Rocha, and Siome Goldenstein. First steps toward image phylogeny. In *Workshop on Information Forensics and Security*, 2010.

[110] Ahmet Emir Dirik, Sevinc Bayram, Husrev T. Sencar, and Nasir Memon. New features to identify computer generated images. In *IEEE International Conference on Image Processing*, volume 4, pages IV–433–IV–436, 2007.

[111] Ahmet Emir Dirik and Nasir Memon. Image tamper detection based on demosaicing artifacts. In *IEEE International Conference on Image Processing*, pages 1509–1512, 2009.

[112] Ahmet Emir Dirik, Husrev T. Sencar, and Nasir Memon. Source camera identification based on sensor dust characteristics. In *IEEE Workshop on Signal Processing Applications for Public Security and Forensics*, pages 1–6, 2007.

[113] Ahmet Emir Dirik, Husrev T. Sencar, and Nasir Memon. Digital single lens reflex camera identification from traces of sensor dust. *IEEE Transactions on Information Forensics and Security*, 3(3):539–552, 2008.

[114] Ahmet Emir Dirik, Husrev T. Sencar, and Nasir Memon. Flatbed scanner identification based on dust and scratches over scanner platen. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 1385–1388, 2009.

[115] Jing Dong, Wei Wang, Tieniu Tan, and Yun Qing Shi. Run-length and edge statistics based approach for image splicing detection. In *International Workshop on Digital Watermarking*, pages 76–87, 2008.

[116] Brandon Dybala, Brian Jennings, and David Letscher. Detecting filtered cloning in digital images. In *ACM Multimedia and Security Workshop*, pages 43–50, 2007.

[117] Jiayuan Fan, Alex C. Kot, Hong Cao, and Farook Sattar. Modeling the exif-image correlation for image manipulation detection. In *International Conference on Image Processing*, 2011.

[118] Na Fan, Cheng Jin, and Yizhen Huang. A pixel-based digital photo authentication framework via demosaicking inter-pixel correlation. In *ACM Multimedia and Security Workshop*, pages 125–130, 2009.

[119] Zhigang Fan and Ricardo de Queiroz. Identification of bitmap compression history: JPEG detection and quantizer estimation. *IEEE Transactions on Image Processing*, 12(2):230–235, 2003.

[120] Yanmei Fang, Ahmet Emir Dirik, Xiaoxi Sun, and Nasir Memon. Source class identification for dslr and compact cameras. In *IEEE Workshop on Multimedia Signal Processing*, 2009.

[121] Yanmei Fang and Jing Yin. Digital image forensics for photographic copying. In *SPIE Conference on Media Waternarking, Security, and Forensics*, pages 8303–12, 2012.

[122] Zhen Fang, Shuozhong Wang, and Xinpeng Zhang. Image splicing detection using camera characteristic inconsistency. In *International Conference on Multimedia Information Networking and Security*, pages 20–24, 2009.

[123] Hany Farid. Detecting digital forgeries using bispectral analysis. Technical Report AIM-1657, AI Lab, Massachusetts Institute of Technology, 1999.

[124] Hany Farid. Creating and detecting doctored and virtual images: Implications to the child pornography prevention act. Technical Report TR2004-518, Department of Computer Science, Dartmouth College, 2004.

[125] Hany Farid. Digital doctoring: How to tell the real from the fake. *Significance*, 3(4):162–166, 2006.

[126] Hany Farid. Digital image ballistics from JPEG quantization. Technical Report TR2006-583, Department of Computer Science, Dartmouth College, 2006.

[127] Hany Farid. Exposing digital forgeries in scientific images. In *ACM Multimedia and Security Workshop*, pages 29–36, 2006.

[128] Hany Farid. Digital image ballistics from JPEG quantization: A followup study. Technical Report TR2008-638, Department of Computer Science, Dartmouth College, 2008.

[129] Hany Farid. Digital image forensics. *Scientific American*, 6(298):66–71, 2008.

[130] Hany Farid. *Advances in Computers*, volume 77, chapter Photo Fakery and Forensics, pages 1–55. Elsevier, 2009.

[131] Hany Farid. *Deception: Methods, Motives, Contexts and Consequences*, chapter Digital Doctoring: can we trust photographs?, pages 95–108. Stanford University Press, 2009.

[132] Hany Farid. Exposing digital forgeries from JPEG ghosts. *IEEE Transactions on Information Forensics and Security*, 1(4):154–160, 2009.

[133] Hany Farid. The lee harvey oswald backyard photos: Real or fake? *Perception*, 11(38):1731–1734, 2009.

[134] Hany Farid. Seeing is not believing. *IEEE Spectrum*, 8(46):44–48, 2009.

[135] Hany Farid. A survey of image forgery detection. *IEEE Signal Processing Magazine*, 2(26):16–25, 2009.

[136] Hany Farid. A 3-d lighting and shadow analysis of the jfk zapruder film (frame 317). Technical Report TR2010-677, Department of Computer Science, Dartmouth College, 2010.

[137] Hany Farid. A 3-d photo forensic analysis of the lee harvey oswald backyard photo. Technical Report TR2010-669, Department of Computer Science, Dartmouth College, 2010.

[138] Hany Farid and Mary J. Bravo. Photorealistic rendering: How realistic is it? In *Vision Sciences*, 2007.

[139] Hany Farid and Mary J. Bravo. Image forensic analyses that elude the human visual system. In *SPIE Conference on Media Forensics and Security*, 2010.

[140] Hany Farid and Siwei Lyu. Higher-order wavelet statistics and their application to digital forensics. In *IEEE Workshop on Statistical Analysis in Computer Vision (in conjunction with CVPR)*, 2003.

[141] Hany Farid and Jeffrey B. Woodward. Video stabilization and enhancement. Technical Report TR2007-605, Department of Computer Science, Dartmouth College, 2007.

[142] Xiaoying Feng. Fld-based detection of re-compressed speech signals. In *ACM Workshop on Multimedia and Security*, pages 43–48, 2010.

[143] Xiaoying Feng and Gwenael Doerr. JPEG recompression detection. In *SPIE Conference on Media Forensics and Security*, 2010.

[144] Tomas Filler, Jessica Fridrich, and Miroslav Goljan. Using sensor pattern noise for camera model identification. In *IEEE International Conference on Image Processing*, pages 1296–1299, 2008.

[145] Claude S. Fillion and Gaurav Sharma. Detecting content adaptive scaling of images for forensic applications. In *SPIE Conference on Media Forensics and Security*, 2010.

[146] Marco Fontani, Tiziano Bianchi, Alessia De Rosa, Alessandro Piva, and Mauro Barni. A dempster-shafer framework for decision fusion in image forensics. In *Workshop on Information Forensics and Security*, 2011.

[147] Marco Fontani, Andrea Costanzo, Mauro Barni, Tiziano Bianchi, Alessia De Rosa, and Alessandro Piva. Two decision fusion frameworks for image forensics. In *Annual GTTI Meeting*, 2011.

[148] Jessica Fridrich. Digital image forensic using sensor noise. *IEEE Signal Processing Magazine*, 26(2):26–37, 2009.

[149] Jessica Fridrich, Mo Chen, and Miroslav Goljan. Imaging sensor noise as digital x-ray for revealing forgeries. In *International Workshop on Information Hiding*, pages 342–358, 2007.

[150] Jessica Fridrich, David Soukal, and Jan Lukas. Detection of copy move forgery in digital images. In *Digital Forensic Research Workshop*, August 2003.

[151] Dongdong Fu, Yun Qing Shi, and Wei Su. Detection of image splicing based on hilbert-huang transform and moments of characteristic functions with wavelet decomposition. In Byeungwoo Jeon, editor, *International Workshop on Digital Watermarking*, pages 177–187, 2006.

[152] Dongdong Fu, Yun Qing Shi, and Wei Su. A generalized benford's law for JPEG coefficients and its applications in image forensics. In Edward J. Delp and Ping Wah Wong, editors, *SPIE Conference on Security, Steganography, and Watermarking of Multimedia Contents*, volume 6505, 2007.

[153] Andrew C. Gallagher. Detection of linear and cubic interpolation in JPEG compressed images. In *Second Canadian Conference on Computer and Robot Vision*, pages 65–72, 2005.

[154] Andrew C. Gallagher and Tsu-Han Chen. Image authentication by detecting traces of demosaicing. In *IEEE Workshop on Vision of the Unseen (in conjunction with CVPR)*, pages 1–8, 2008.

[155] Xinting Gao, Tian-Tsong Ng, Bo Qiu, and Shih-Fu Chang. Single-view recaptured image detection based on physics-based features. In *IEEE International Conference on Multimedia and Expo*, 2010.

[156] Xinting Gao, Bo Qiu, JingJing Shen, Tian-Tsong Ng, and Yun Qing Shi. A smart phone image database for single image recapture. In *International Workshop on Digital Watermarking*, pages 90–104, 2010.

[157] Daniel Garcia-Romero and Carol Espy-Wilson. Automatic acquisition device identification from speech recordings. In *International Conference on Acoustics, Speech, and Signal Processing*, pages 1806–1809, 2010.

[158] Ravi Garg, Avinash L. Varna, and Min Wu. ”seeing” enf: Natural time stamp for digital video via optical sensing and signal processing. In *ACM Multimedia*, 2011.

[159] Matthew D. Gaubatz and Steven J. Simske. Printer-scanner identification via analysis of structured security deterrents. In *IEEE Workshop on Information Forensics and Security*, pages 151–155, London, UK, 2009.

[160] Matthew D. Gaubatz, Robert Ulichney, and David Rouse. A low-complexity reduced-reference print identification algorithm. In *IEEE International Conference on Image Processing*, pages 1281–1284, 2009.

[161] Zeno J. Geradts, Jurrien Bijhold, Martijn Kieft, Kenji Kurosawa, Kenro Kuroki, and Naoki Saitoh. Methods for identification of images acquired with digital cameras. In Simon K. Bramble, Edward M. Carapezza, and Lenny I. Rudin, editors, *SPIE Conference on Enabling Technologies for Law Enforcement and Security*, volume 4232, pages 505–512, 2001.

[162] Zeno J. Geradts, Jurrien Bijhold, Martijn Kieft, Kenji Kurosawa, Kenro Kuroki, and Naoki Saitoh. Digital camera identification. *Journal of Forensic Identification*, 52(5):621–631, 2002.

[163] Sandeep Gholap and P. K. Bora. Illuminant colour based image forensics. In *IEEE Region 10 Conference (TENCON 2008)*, 2008.

[164] Thomas Gloe and Rainer Böhme. The ’dresden image database’ for benchmarking digital image forensics. In *ACM Symposium on Applied Computing*, volume 2, pages 1584–1590, 2010.

[165] Thomas Gloe, Karsten Borowka, and Antje Winkler. Feature-based camera model identification works in practice: Results of a comprehensive evaluation study. In Stefan Katzenbeisser and Ahmad-Reza Sadeghi, editors, *International Workshop on Information Hiding*, pages 262–276, 2009.

[166] Thomas Gloe, Elke Franz, and Antje Winkler. Forensics for flatbed scanners. In Edward J. Delp and Ping Wah Wong, editors, *SPIE Conference on Security, Steganography, and Watermarking of Multimedia Contents*, volume 6505, 2007.

[167] Thomas Gloe, Matthias Kirchner, Antje Winkler, and Rainer Böhme. Can we trust digital image forensics? In *International Conference on Multimedia*, pages 78–86, 2007.

[168] Thomas Gloe, Antje Winkler, and Karsten Borowka. Efficient estimation and large-scale evaluation of lateral chromatic aberration for digital image forensics. In *SPIE Conference on Media Forensics and Security*, 2010.

[169] Siome Goldenstein and Anderson Rocha. High-profile forensic analysis of images. In *International Conference on Imaging for Crime Detection and Prevention*, pages 1–6, 2009.

[170] Miroslav Goljan. Digital camera identification from images – estimating false acceptance probability. In *International Workshop on Digital Watermarking*, pages 454–468, 2008.

[171] Miroslav Goljan, Mo Chen, and Jessica Fridrich. Identifying common source digital camera from image pairs. In *IEEE International Conference on Image Processing*, 2007.

[172] Miroslav Goljan and Jessica Fridrich. Camera identification from scaled and cropped images. In Edward J. Delp and Ping Wah Wong, editors, *SPIE Conference on Security, Forensics, Steganography, and Watermarking of Multimedia Contents*, volume 6819, 2008.

[173] Miroslav Goljan and Jessica Fridrich. Determining approximate age of digital images using sensor defects. In *SPIE Conference on Media Watermarking, Security, and Forensics*, 2011.

[174] Miroslav Goljan, Jessica Fridrich, and Mo Chen. Sensor noise camera identification: Countering counter-forensics. In *SPIE Conference on Media Forensics and Security*, 2010.

[175] Miroslav Goljan, Jessica Fridrich, and Mo Chen. Defending against fingerprint-copy attack in sensor-based camera identification. *IEEE Transactions on Information Forensics and Security*, 6(1):227–236, 2011.

[176] Miroslav Goljan, Jessica Fridrich, and Tomas Filler. Large scale test of sensor fingerprint camera identification. In *SPIE Conference on Media Forensics and Security*, 2009.

[177] Miroslav Goljan, Jessica Fridrich, and Tomas Filler. Managing a large database of camera fingerprints. In *SPIE Conference on Media Forensics and Security*, 2010.

[178] Miroslav Goljan, Jessica Fridrich, and Jan Lukas. Camera identification from printed images. In Edward J. Delp, Ping Wah Wong, Jana Dittmann, and Nasir Memon, editors, *SPIE Conference on Security, Forensics, Steganography, and Watermarking of Multimedia Contents*, volume 6819, 2008.

[179] E. S. Gopi. Digital image forgery detection using artificial neural network and independent component analysis. *Applied Mathematics and Computation*, 194(2):540–543, 2007.

[180] E. S. Gopi, Lakshmanan Nataraj, T. Gokul, S. KumaraGanesh, and Prerak. R. Shah. Digital image forgery detection using artificial neural network and auto regressive coefficients. In *Canadian Conference on Electrical and Computer Engineering*, 2006.

[181] Hongmei Gou, Ashwin Swaminathan, and Min Wu. Noise features for image tampering detection and steganalysis. In *IEEE International Conference on Image Processing*, 2007.

[182] Hongmei Gou, Ashwin Swaminathan, and Min Wu. Robust scanner identification based on noise features. In *SPIE Conference on Security, Steganography, and Watermarking of Multimedia Contents*, February 2007.

[183] Hongmei Gou, Ashwin Swaminathan, and Min Wu. Intrinsic sensor noise features for forensic analysis on scanners and scanned images. *IEEE Transactions on Information Forensics and Security*, 4(3):476–491, sep 2009.

[184] Catalin Grigoras. Digital audio recording analysis: the electric network frequency (enf) criterion. *Speech, Language and the Law*, 12(1):63–76, 2005.

[185] Catalin Grigoras. Applications of enf analysis method in forensic authentication of digital audio and video recordings. In *123rd AES Convention*, 2007.

[186] Catalin Grigoras. Applications of enf criterion in forensic audio, video, computer and telecommunication analysis. *Forensic Science International*, 167(2-3):136–145, 2007.

[187] Catalin Grigoras, Alan Cooper, and Marcin Michalek. Forensic speech and audio analysis working group - best practice guidelines for enf analysis in forensic authentication of digital evidence, 2009.

[188] Jiri Grim, Petr Somol, and Pavel Pudil. Digital image forgery detection by local statistical models. In *International Conference on Intelligent Information Hiding and Multimedia Signal Processing*, pages 579–582, 2010.

[189] Gökhan Gül and Ismail Avcibas. Source cell phone camera identification based on singular value decomposition. In *IEEE Workshop on Information Forensics and Security*, pages 171–175, London, UK, 2009.

[190] Gökhan Gül, Ismail Avcibas, and Fatih Kurugollu. SVD based image manipulation detection. In *International Conference on Image Processing*, 2010.

[191] Gaurav Gupta, Sanjoy Kumar Saha, Shayok Chakraborty, and Chandan Mazumdar. Document frauds: Identification and linking fake document to scanners and printers. In *International Conference on Theory and Applications*, pages 497–501, 2007.

[192] Junfeng He, Zhouchen Lin, Lifeng Wang, and Xiaoou Tang. Detecting doctored JPEG images via DCT coefficient analysis. In *European Conference on Computer Vision*, 2006.

[193] John Ho, Oscar C. Au, and Jiantao Zhou. Inter-channel demosaicking traces for digital image forensics. In *IEEE International Conference on Multimedia and Expo*, 2010.

[194] Dun-Yu Hsiao and Soo-Chang Pei. Detecting digital tampering by blur estimation. In *International Workshop on Systematic Approaches to Digital Forensic Engineering*, pages 264–278, 2005.

[195] Pei-Lun Hsieh, Yu-Ming Liang, and Hong-Yuan Mark Liao. Recognition of blurred license plate images. In *Workshop on Information Forensics and Security*, 2010.

[196] Chih-Chung Hsu, Tzu-Yi Hung, Chia-Wen Lin, and Chiou-Ting Hsu. Video forgery detection using correlation of noise residue. In *IEEE Workshop on Multimedia Signal Processing*, pages 170–174, 2008.

[197] Yu-Feng Hsu and Shih-Fu Chang. Detecting image splicing using geometry invariants and camera characteristics consistency. In *IEEE International Conference on Multimedia and Expo*, 2006.

[198] Yu-Feng Hsu and Shih-Fu Chang. Image splicing detection using camera response function consistency and automatic segmentation. In *IEEE International Conference on Multimedia and Expo*, 2007.

[199] Yu-Feng Hsu and Shih-Fu Chang. Statistical fusion of multiple cues for image tampering detection. In *Asilomar Conference on Signals, Systems, and Computers*, 2008.

[200] Yu-Feng Hsu and Shih-Fu Chang. Camera response functions for image forensics: An automatic algorithm for splicing detection. *IEEE Transactions on Information Forensics and Security*, 5(4):816–825, 2010.

[201] Fangjun Huang, Jiwu Huang, and Yun Qing Shi. Detecting double JPEG compression with the same quantization matrix. *IEEE Transactions on Information Forensics and Security*, 5(4):848–856, 2010.

[202] Hailing Huang, Weiqiang Guo, and Yu Zhang. Detection of copy-move forgery in digital images using sift algorithm. In *Pacific-Asia Workshop on Computational Intelligence and Industrial Application*, pages 272–276, 2008.

[203] Yanping Huang, Wei Lu, Wei Sun, and Dongyang Long. Improved DCT-based detection of copy-move forgery in images. *Forensic Science International*, 206(1):178–184, 2011.

[204] Yizhen Huang. Can digital image forgery detection be unevadable? a case study: Color filter array interpolation statistical feature recovery. In Shipeng Li, Fernando Pereira, Heung-Yeung Shum, and Andrew G. Tescher, editors, *SPIE Conference on Visual Communications and Image Processing*, volume 5960, 2005.

[205] Yizhen Huang and Na Fan. Learning from interpolated images using neural networks for digital forensics. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 177–182, 2010.

[206] Yizhen Huang and Yangjing Long. Demosaicking recognition with applications in digital photo authentication based on a quadratic pixel correlation model. In *Computer Vision and Pattern Recognition*, pages 1–8, 2008.

[207] Maarten Huijbregtse and Zeno J. Geradts. Using the enf criterion for determining the time of recording of short digital audio recordings. In Zeno J. Geradts, Katrin Y. Franke, and Cor. J. Veenman, editors, *International Workshop on Computational Forensics*, pages 116–124, 2009.

[208] Dai-Kyung Hyun, Seung-Jin Ryu, Min-Jeong Lee, Jun-Hee Lee, Hae-Yeoun Lee, and Heung-Kyu Lee. Source camcorder identification with cropped and scaled videos. In *SPIE Conference on Media Waternarking, Security, and Forensics*, pages 8303–11, 2012.

[209] Alain Jaubert. *Le commissariat aux archives (French Edition)*. Barrault, 1992.

[210] Arshad Jhumka, Chris Levett, and Sarabjot Anand. Towards event ordering in digital forensics. In *ACM Workshop on Multimedia and Security*, 2010.

[211] Micah Kimo Johnson. *Lighting and Optical Tools for Image Forensics*. PhD thesis, Department of Computer Science, Dartmouth College, Hanover, NH, 2007.

[212] Micah Kimo Johnson and Hany Farid. Exposing digital forgeries by detecting inconsistencies in lighting. In *ACM Multimedia and Security Workshop*, pages 1–10, 2005.

[213] Micah Kimo Johnson and Hany Farid. Exposing digital forgeries through chromatic aberration. In *ACM Multimedia and Security Workshop*, pages 48–55, 2006.

[214] Micah Kimo Johnson and Hany Farid. Metric measurements on a plane from a single image. Technical Report TR2006-579, Department of Computer Science, Dartmouth College, 2006.

[215] Micah Kimo Johnson and Hany Farid. Detecting photographic composites of people. In *International Workshop on Digital Watermarking*, pages 19–33, 2007.

[216] Micah Kimo Johnson and Hany Farid. Exposing digital forgeries in complex lighting environments. *IEEE Transactions on Information Forensics and Security*, 3(2):450–461, 2007.

[217] Micah Kimo Johnson and Hany Farid. Exposing digital forgeries through specular highlights on the eye. In *International Workshop on Information Hiding*, pages 311–325, 2007.

[218] Mateusz Kajstura, Agata Trawinska, and Jacek Hebenstreit. Application of the electrical network frequency (enf) criterion: A case of a digital recording. *Forensic Science International*, 155(2-3):165–171, 2005.

[219] Pravin Kakar, Sudha Natarajan, and Wee Ser. Detecting digital image forgeries through inconsistent motion blur. In *IEEE International Conference on Multimedia and Expo*, 2010.

[220] Pravin Kakar, Sudha Natarajan, and Wee Ser. Image authentication by motion blur consistency verification. In *IEEE Region 10 Conference (TENCON 2010)*, pages 188–193, 2010.

[221] Pravin Kakar, Sudha Natarajan, and Wee Ser. Exposing digital image forgeries by detecting discrepancies in motion blur. *IEEE Transactions on Multimedia*, 13(3):443–452, 2011.

[222] Xiangui Kang, Yinxiang Li, Zhenhua Qu, and Jiwu Huang. Enhancing roc performance of trustworthy camera source identification. In *SPIE Conference on Media Watermarking, Security, and Forensics*, 2011.

[223] Xiao-Bing Kang and Sheng-Min Wei. Identifying tampered regions using singular value decomposition in digital image forensics. In *International Conference on Computer Science and Software Engineering*, pages 926–930, 2008.

[224] Yan Ke, Rahul Sukthankar, and Larry Huston. Efficient near-duplicate detection and sub-image retrieval. In *ACM International Conference on Multimedia*, pages 869–876, 2004.

[225] Eric Kee and Hany Farid. Printer profiling for forensics and ballistics. In *ACM Multimedia and Security Workshop*, pages 3–10, 2008.

[226] Eric Kee and Hany Farid. Detecting photographic composites of famous people. Technical Report TR2009-656, Department of Computer Science, Dartmouth College, 2009.

[227] Eric Kee and Hany Farid. Digital image authentication from thumbnails. In *SPIE Conference on Media Forensics and Security*, 2010.

[228] Eric Kee and Hany Farid. Exposing digital forgeries from 3-d lighting environments. In *Workshop on Information Forensics and Security*, 2010.

[229] Eric Kee and Hany Farid. A perceptual metric for photo retouching. *Proceedings of the National Academy of Sciences*, 108(50):19907–19912, 2011.

[230] Eric Kee, Micah K. Johnson, and Hany Farid. Digital image authentication from JPEG headers. *IEEE Transactions on Information Forensics and Security*, 6(3):1066–1075, 2011.

[231] Saiqa Khan and Arun Kulkarni. An efficient method for detection of copy-move forgery using discrete wavelet transform. *International Journal on Computer Science and Engineering*, 2(5):1801–1806, 2010.

[232] Nitin Khanna, George T.-C. Chiu, Jan P. Allebach, and Edward J. Delp. Forensic technqiues for classifying scanner, computer generated and digital camera images. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 1653–1656, 2008.

[233] Nitin Khanna, George T.-C. Chiu, Jan P. Allebach, and Edward J. Delp. Scanner identification with extension to forgery detection. In *SPIE Conference on Security, Forensics, Steganography, and Watermarking of Multimedia Contents*, 2008.

[234] Nitin Khanna and Edward J. Delp. Source scanner identification for scanned documents. In *IEEE Workshop on Information Forensics and Security*, pages 166–170, London, UK, 2009.

[235] Nitin Khanna and Edward J. Delp. Intrinsic signatures for scanned documents forensics : Effect of font shape and size. In *International Symposium on Circuits and Systems*, 2010.

[236] Nitin Khanna, Aravind K. Mikkilineni, Pei-Ju Chiang, Maria V. Ortiz, Vivek Shah, Sungjoo Suh, George T.-C. Chiu, Jan P. Allebach, and Edward J. Delp. Printer and sensor forensics. In *IEEE Workshop on Signal Processing Applications for Public Security and Forensics*, 2007.

[237] Nitin Khanna, Aravind K. Mikkilineni, Pei-Ju Chiang, Maria V. Ortiz, Sungjoo Suh, George T.-C. Chiu, Jan P. Allebach, and Edward J. Delp. Sensor forensics: Printers, cameras and scanners, they never lie. In *IEEE International Conference on Multimedia and Expo*, pages 20–23, 2007.

[238] Nitin Khanna, Aravind K. Mikkilineni, George T.-C. Chiu, Jan P. Allebach, and Edward J. Delp. Forensic classification of imaging sensor types. In Edward J. Delp and Ping Wah Wong, editors, *SPIE Conference on Security, Steganography, and Watermarking of Multimedia Contents*, volume 6505, 2007.

[239] Nitin Khanna, Aravind K. Mikkilineni, George T.-C. Chiu, Jan P. Allebach, and Edward J. Delp. Scanner identification using sensor pattern noise. In Edward J. Delp and Ping Wah Wong, editors, *SPIE Conference on Security, Steganography, and Watermarking of Multimedia Contents*, volume 6505, 2007.

[240] Nitin Khanna, Aravind K. Mikkilineni, George T.-C. Chiu, Jan P. Allebach, and Edward J. Delp. Survey of scanner and printer forensics at purdue university. In *International Workshop on Computational Forensics*, pages 22–34, 2008.

[241] Nitin Khanna, Aravind K. Mikkilineni, and Edward J. Delp. Forensic camera classification: Verification of sensor pattern noise approach. *Forensic Science Communications*, 11(1), 2009.

[242] Nitin Khanna, Aravind K. Mikkilineni, and Edward J. Delp. Scanner identification using feature-based processing and analysis. *IEEE Transactions on Information Forensics and Security*, 4(1):123–139, 2009.

[243] Nitin Khanna, Aravind K. Mikkilineni, and Edward J. Delp. Texture based attacks on intrinsic signature based printer identification. In *SPIE Conference on Media Forensics and Security*, 2010.

[244] Nitin Khanna, Aravind K. Mikkilineni, Anthony F. Martone, Gazi N. Ali, George T.-C. Chiu, Jan P. Allebach, and Edward J. Delp. A survey of forensic characterization methods for physical devices. *Digital Investigation*, 3(Supplement 1):17–28, 2006.

[245] Mehdi Kharrazi, Husrev T. Sencar, and Nasir Memon. Blind source camera identification. In *IEEE International Conference on Image Processing*, pages 709–712, 2004.

[246] Stefan Kiltz, Mario Hildebrandt, Robert Altschaffel, and Jana Dittmann. A transparent bridge for forensic sound network traffic data acquisition. In *Sicherheit - Schutz und Zuverlässigkeit*, pages 93–104, 2010.

[247] Stefan Kiltz, Tobias Hoppe, and Jana Dittmann. A new forensic model and its application to the collection, extraction and long term storage of screen content off a memory dump. In *International Conference on Digital Signal Processing*, pages 1135–1140, 2009.

[248] Stefan Kiltz, Tobias Hoppe, Jana Dittmann, and Claus Vielhauer. Video surveillance: A new forensic model for the forensically sound retrival of picture content off a memory dump. In Stefan Fischer, Erik Maehle, and Ruediger Reischuk, editors, *Informatik2009 - Digitale Multimedia-Forensik*, pages 1619–1633, 2009.

[249] Hyoung J. Kim, Soomin Lim, Jongsub Moon, Boram Kim, and Eui S. Jung. A photographic forensic case study: Myths, principles and techniques. *Mathematical and Computer Modelling*, 55(1-2):3–11, 2012.

[250] Matthias Kirchner. Fast and reliable resampling detection by spectral analysis of fixed linear predictor residue. In *ACM Multimedia and Security Workshop*, pages 11–20, 2008.

[251] Matthias Kirchner. On the detectability of local resampling in digital images. In Edward J. Delp, Ping Wah Wong, Jana Dittmann, and Nasir Memon, editors, *SPIE Conference on Security, Forensics, Steganography, and Watermarking of Multimedia Contents*, volume 6819, 2008.

[252] Matthias Kirchner. Efficient estimation of cfa pattern configuration in digital camera images. In *SPIE Conference on Media Forensics and Security*, 2010.

[253] Matthias Kirchner. Linear row and column predictors for the analysis of resized images. In *ACM Workshop on Multimedia and Security*, pages 13–18, 2010.

[254] Matthias Kirchner and Rainer Böhme. Tamper hiding: Defeating image forensics. In Teddy Furon, Francois Cayre, Gwenael Doerr, and Patrick Bas, editors, *International Workshop on Information Hiding*, pages 326–341, 2007.

[255] Matthias Kirchner and Rainer Böhme. Hiding traces of resampling in digital images. *IEEE Transactions on Information Forensics and Security*, 3(4):582–592, 2008.

[256] Matthias Kirchner and Rainer Böhme. Synthesis of color filter array pattern in digital images. In Edward J. Delp, Jana Dittmann, Nasir Memon, and Ping Wah Wong, editors, *SPIE Conference on Media Forensics and Security*, 2009.

[257] Matthias Kirchner and Jessica Fridrich. On detection of median filtering in digital images. In *SPIE Conference on Media Forensics and Security*, 2010.

[258] Matthias Kirchner and Thomas Gloe. On resampling detection in recompressed images. In *IEEE Workshop on Information Forensics and Security*, pages 21–25, 2009.

[259] Simon Knight, Simon Moschou, and Matthew Sorell. Analysis of sensor photo response non-uniformity in raw images. In Matthew Sorell, editor, *International Conference on Forensics in Telecommunications, Information and Multimedia*, pages 130–141, 2009.

[260] Michihiro Kobayashi, Takahiro Okabe, and Yoichi Sato. Detecting video forgeries based on noise characteristics. In Toshikazu Wada, Fay Huang, and Stephen Lin, editors, *Advances in Image and Video Technology, Third Pacific Rim Symposium*, pages 306–317, 2009.

[261] Michihiro Kobayashi, Takahiro Okabe, and Yoichi Sato. Detecting forgery from static-scene video based on inconsistency in noise level functions. *IEEE Transactions on Information Forensics and Security*, 5(4):883–892, 2010.

[262] Jesse D. Kornblum. Using JPEG quantization tables to identify imagery processed by software. *Digital Investigation*, 5(Supplement 1):S21–S25, 2008.

[263] Christian Krätzer and Jana Dittmann. Der einfluss gleichgewichteter fusion in der mikrofonforensik unter beispielhafter nutzung von zwei klassifikatoren. In *GI Jahrestagung*, pages 1604–1618, 2009.

[264] Christian Krätzer, Andrea Oermann, Jana Dittmann, and Andreas Lang. Digital audio forensics: A first practical evaluation on microphone and environment classification. In *ACM Multimedia and Security Workshop*, pages 63–74, 2007.

[265] Christian Krätzer, Kun Qian, Maik Schott, and Jana Dittmann. A context model for microphone forensics and its application in evaluations. In *SPIE Conference on Media Watermarking, Security, and Forensics*, 2011.

[266] Christian Krätzer, Maik Schott, and Jana Dittmann. Unweighted fusion in microphone forensics using a decision tree and linear logistic regression models. In *ACM Multimedia and Security Workshop*, pages 49–56, 2009.

[267] Kenro Kuroki, Kenji Kurosawa, and Naoki Saitoh. An approach to individual video camera identification. *Journal of Forensic Sciences*, 47(1):97–102, 2002.

[268] Kenji Kurosawa, Kenro Kuroki, and Naoki Saitoh. Ccd fingerprint method - identification of a video camera from videotaped images. In *IEEE International Conference on Image Processing*, volume 3, pages 537–540, 1999.

[269] Kenji Kurosawa and Naoki Saitoh. Fundamental study on identification of cmos cameras. In *SPIE Conference on Visual Information Processing XII*, pages 202–209, 2003.

[270] Cheng-Liang Lai and Yu-Shiang Chen. The application of intelligent system to digital image forensics. In *International Conference on Machine Learning and Cybernetics*, volume 5, pages 2991–2998, 2009.

[271] Aaron Langille and Minglun Gong. An efficient match-based duplication detection algorithm. In *Canadian Conference on Computer and Robot Vision*, 2006.

[272] Tran Van Lanh, Kai-Sen Chong, Sabu Emmanuel, and Mohan S. Kankanhalli. A survey on digital camera image forensic methods. In *IEEE International Conference on Multimedia and Expo*, pages 16–19, 2007.

[273] Ji-Won Lee, Min-Jeong Lee, Tae-Woo Oh, Seung-Jin Ryu, and Heung-Kyu Lee. Screenshot identification using combing artifact from interlaced video. In *ACM Workshop on Multimedia and Security*, pages 49–54, 2010.

[274] Sangwon Lee, David A. Shamma, and Bruce Gooch. Detecting false captioning using common sense reasoning. *Digital Investigation*, 3(Supplement 1):65–70, 2006.

[275] Frederic Lefebvre, Bertrand Chupeau, Ayoub Massoudi, and Eric Diehl. Image and video fingerprinting: Forensic applications. In *SPIE Conference on Media Forensics and Security*, 2009.

[276] Bin Li, Yun Qing Shi, and Jiwu Huang. Detecting doubly compressed JPEG images by using mode based first digit features. In *IEEE Workshop on Multimedia Signal Processing*, pages 730–735, 2008.

[277] Chang-Tsun Li. Detection of block artifacts for digital forensic analysis. In Matthew Sorell, editor, *International Conference on Forensics in Telecommunications, Information and Multimedia*, pages 173–178, 2009.

[278] Chang-Tsun Li. Source camera identification using enhanced sensor pattern noise. In *IEEE International Conference on Image Processing*, 2009.

[279] Chang-Tsun Li. Source camera linking using enhanced sensor pattern noise extracted from images. In *International Conference on Imaging for Crime Detection and Prevention*, 2009.

[280] Chang-Tsun Li. Source camera identification using enhanced sensor pattern noise. *IEEE Transactions on Information Forensics and Security*, 5(2):280–287, 2010.

[281] Chang-Tsun Li. Unsupervised classification of digital images using enhanced sensor pattern noise. In *IEEE International Symposium on Circuits and Systems*, 2010.

[282] Chang-Tsun Li, Chih-Yuan Chang, and Yue Li. On the repudiability of device identification and image integrity verification using sensor pattern noise. In *International Conference on Information Security and Digital Forensics*, pages 19–25, 2009.

[283] Chang-Tsun Li and Yue Li. Digital camera identification using colour-decoupled photo response non-uniformity noise pattern. In *International Symposium on Circuits and Systems*, 2010.

[284] Guo-Hui Li, Qiong Wu, Dan Tu, and Shao-Jie Sun. A sorted neighborhood approach for detecting duplicated regions in image forgeries based on DWT and SVD. In *IEEE International Conference on Multimedia and Expo*, pages 1750–1753, 2007.

[285] Weihai Li and Nenghai Yu. Rotation robust detection of copy-move forgery. In *International Conference on Image Processing*, 2010.

[286] Weihai Li, Nenghai Yu, and Yuan Yuan. Doctored JPEG image detection. In *IEEE International Conference on Multimedia and Expo*, pages 253–256, 2008.

[287] Weihai Li, Yuan Yuan, and Nenghai Yu. Detecting copy-paste forgery of JPEG image via block artifact grid extraction. In *International Workshop on Local and Non-Local Approximation in Image Processing*, 2008.

[288] Weihai Li, Yuan Yuan, and Nenghai Yu. Passive detection of doctored JPEG image via block artifact grid extraction. *IEEE Transactions on Signal Processing*, 89(9):1821–1829, 2009.

[289] Wenxiang Li, Tao Zhang, Xi-Jian Ping, and Ergong Zheng. Identifying photorealistic computer graphics using second-order difference statistics. In *International Conference on Fuzzy Systems and Knowledge Discovery*, pages 2316–2319, 2010.

[290] Yue Li and Chang-Tsun Li. Decomposed photo response non-uniformity for digital forensic analysis. In Matthew Sorell, editor, *International Conference on Forensics in Telecommunications, Information and Multimedia*, pages 166–172, 2009.

[291] Zhe Li and Jiangbin Zheng. Blind detection of digital forgery image based on the local entropy of the gradient. In *International Workshop on Digital Watermarking*, pages 161–169, 2008.

[292] Dandan Liao, Rui Yang, Hongmei Liu, Jian Li, and Jiwu Huang. Double h.264/avc compression detection using quantized nonzero ac coefficients. In *SPIE Conference on Media Watermarking, Security, and Forensics*, 2011.

[293] Cheng-Chang Lien, Cheng-Lun Shih, and Chih-Hsun Chou. Fast forgery detection with the intrinsic resampling properties. In *International Conference on Intelligent Information Hiding and Multimedia Signal Processing*, pages 232–235, 2010.

[294] Hwei-Jen Lin, Chun-Wei Wang, and Yang-Ta Kao. Fast copy-move forgery detection. *WSEAS Transactions on Signal Processing*, 5(5):188–197, 2009.

[295] W. Sabrina Lin, Steven Tjoa, H. Vicky Zhao, and K. J. Ray Liu. Image source coding forensics via intrinsic fingerprints. In *IEEE International Conference on Multimedia and Expo*, pages 1127–1130, 2007.

[296] W. Sabrina Lin, Steven Tjoa, H. Vicky Zhao, and K. J. Ray Liu. Digital image source coder forensics via intrinsic fingerprints. *IEEE Transactions on Information Forensics and Security*, 4(3):460–475, 2009.

[297] Zhouchen Lin, Junfeng He, Xiaoou Tang, and Chi-Keung Tang. Fast, automatic and fine-grained tampered JPEG image detection via DCT coefficient analysis. *Pattern Recognition*, 42(11):2492–2501, 2009.

[298] Zhouchen Lin, Rongrong Wang, Xiaoou Tang, and Heung-Yeung Shum. Detecting doctored images using camera response normality and consistency. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2005.

[299] Bei-Bei Liu, Yongjian Hu, and Heung-Kyu Lee. Source camera identification from significant noise residual regions. In *International Conference on Image Processing*, 2010.

[300] Bei-Bei Liu, Heung-Kyu Lee, Yongjian Hu, and Chang-Hee Choi. On classification of source cameras: A graph based approach. In *Workshop on Information Forensics and Security*, 2010.

[301] Ming Liu, Nenghai Yu, and Weihai Li. Camera model identification for JPEG images via tensor analysis. In *International Conference on Intelligent Information Hiding and Multimedia Signal Processing*, pages 462–465, 2010.

[302] Qiguang Liu, Xiaochun Cao, Chao Deng, and Xiaojie Guo. Identifying image composites through shadow matte consistency. *IEEE Transactions on Information Forensics and Security*, 6(3):1111–1122, 2011.

[303] Qingzhong Liu and Andrew H. Sung. A new approach for JPEG resize and image splicing detection. In *ACM Multimedia and Security Workshop*, pages 43–48, 2009.

[304] Yangjing Long and Yizhen Huang. Image based source camera identification using demosaicking. In *IEEE Workshop on Multimedia Signal Processing*, pages 419–424, 2006.

[305] Wei Lu, Fu-Lai Chung, and Hong-Tao Lu. Blind fake image detection scheme using SVD. *IEICE Transactions on Communications*, E89-B(5):1726–1728, 2006.

[306] Wei Lu, Wei Sun, Fu-Lai Chung, and Hong-Tao Lu. Revealing digital fakery using multiresolution decomposition and higher order statistics. *Engineering Applications of Artificial Intelligence*, 24(4):666–672, 2011.

[307] Wei Lu, Wei Sun, Jiwu Huang, and Hong-Tao Lu. Digital image forensics using statistical features and neural network classifier. In *International Conference on Machine Learning and Cybernetics*, pages 2831–2834, 2008.

[308] Jan Lukas. Digital image authentication using image filtering techniques. In *Conference on Scientific Computing*, 2000.

[309] Jan Lukas and Jessica Fridrich. Estimation of primary quantization matrix in double compressed JPEG images. In *Digital Forensic Research Workshop*, August 2003.

[310] Jan Lukas, Jessica Fridrich, and Miroslav Goljan. Digital bullet scratches for images. In *IEEE International Conference on Image Processing*, volume 3, pages 65–68, 2005.

[311] Jan Lukas, Jessica Fridrich, and Miroslav Goljan. Detecting digital image forgeries using sensor pattern noise. In Edward J. Delp and Ping Wah Wong, editors, *SPIE Conference on Security, Steganography, and Watermarking of Multimedia Contents*, volume 6072, 2006.

[312] Jan Lukas, Jessica Fridrich, and Miroslav Goljan. Digital camera identification from sensor noise. *IEEE Transactions on Information Forensics and Security*, 1(2):205–214, 2006.

[313] Weiqi Luo, Jiwu Huang, and Guoping Qiu. Robust detection of region-duplication forgery in digital images. In *International Conference on Pattern Recognition*, pages 746–749, 2006.

[314] Weiqi Luo, Jiwu Huang, and Guoping Qiu. A novel method for block size forensics based on morphological operations. In *International Workshop on Digital Watermarking*, pages 229–239, 2008.

[315] Weiqi Luo, Jiwu Huang, and Guoping Qiu. JPEG error analysis and its applications to digital image forensics. *IEEE Transactions on Information Forensics and Security*, 5(3):480–491, 2010.

[316] Weiqi Luo, Zhenhua Qu, Jiwu Huang, and Guoping Qiu. A novel method for detecting cropped and recompressed image block. In *IEEE Conference on Acoustics, Speech and Signal Processing*, pages 217–220, 2007.

[317] Weiqi Luo, Min Wu, and Jiwu Huang. Mpeg recompression detection based on block artifacts. In *SPIE Conference on Security, Forensics, Steganography, and Watermarking of Multimedia Contents*, January 2008.

[318] Siwei Lyu. *Natural Image Statistics for Digital Image Forensics*. PhD thesis, Department of Computer Science, Dartmouth College, Hanover, NH, 2005.

[319] Siwei Lyu. Estimating vignetting function from a single image for image authentication. In *ACM Workshop on Multimedia and Security*, pages 3–12, 2010.

[320] Siwei Lyu and Hany Farid. How realistic is photorealistic? *IEEE Transactions on Signal Processing*, 53(2):845–850, 2005.

[321] Babak Mahdian and Stanislav Saic. Detection of copy move forgery using a method based on blur moment invariants. *Forensic Science International*, 171:180–189, 2007.

[322] Babak Mahdian and Stanislav Saic. On periodic properties of interpolation and their application to image authentication. In *International Symposium on Information Assurance and Security*, pages 439–446, 2007.

[323] Babak Mahdian and Stanislav Saic. Blind authentication using periodic properties of interpolation. *IEEE Transactions on Information Forensics and Security*, 3(3):529–538, 2008.

[324] Babak Mahdian and Stanislav Saic. Blind methods for detecting image fakery. In *42nd Annual IEEE International Carnahan Conference on Security Technology*, pages 280–286, 2008.

[325] Babak Mahdian and Stanislav Saic. Detection of resampling supplemented with noise inconsistencies analysis for image forensics. In *International Conference on Computational Sciences and Its Applications*, pages 546–556, 2008.

[326] Babak Mahdian and Stanislav Saic. A cyclostationarity analysis applied to image forensics. In *IEEE Workshop on Applications of Computer Vision*, pages 389–399, 2009.

[327] Babak Mahdian and Stanislav Saic. Detecting double compressed JPEG images. In *International Conference on Imaging for Crime Detection and Prevention*, 2009.

[328] Babak Mahdian and Stanislav Saic. Detection and description of geometrically transformed digital images. In Edward J. Delp, Jana Dittmann, Nasir Memon, and Ping Wah Wong, editors, *SPIE Conference on Media Forensics and Security*, 2009.

[329] Babak Mahdian and Stanislav Saic. Using noise inconsistencies for blind image authentication. *Image and Vision Computing*, 27(10):1497–1503, 2009.

[330] Babak Mahdian and Stanislav Saic. A bibliography on blind methods for identifying image forgery. *Signal Processing: Image Communication*, 25(6):389–399, 2010.

[331] Babak Mahdian and Stanislav Saic. Blind methods for detecting image fakery. *IEEE Aerospace and Electronic Systems Magazine*, 25(4):18–24, 2010.

[332] Babak Mahdian and Stanislav Saic. Identifying image forgeries using change point detection. In *SPIE Conference on Media Watermarking, Security, and Forensics*, 2011.

[333] Babak Mahdian, Stanislav Saic, and Radim Nedbal. JPEG quantization tables forensics: A statistical approach. In *International Workshop on Computational Forensics*, pages 150–159, 2010.

[334] Robert C. Maher. Acoustical characterization of gunshots. In *Signal Processing Applications for Public Security and Forensics*, pages 109–113, 2007.

[335] Hafiz Malik and Hany Farid. Audio forensics from acoustic reverberation. In *International Conference on Acoustics, Speech, and Signal Processing*, pages 1710–1713, 2010.

[336] Junwen Mao, Orhan Bulan, Gaurav Sharma, and Suprakash Datta. Device temporal forensics: An information theoretic approach. In *IEEE International Conference on Image Processing*, pages 1501–1504, 2009.

[337] Anthony F. Martone, Aravind K. Mikkilineni, and Edward J. Delp. Forensics of things. In *IEEE Southwest Symposium on Image Analysis and Interpretation*, pages 149–152, 2006.

[338] Scott McCloskey. Confidence weighting for sensor fingerprinting. In *Computer Vision and Pattern Recognition*, pages 1–6, 2008.

[339] Christine McKay, Ashwin Swaminathan, Hongmei Gou, and Min Wu. Image acquisition forensics: Forensic analysis to identify imaging source. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 1657–1660, April 2008.

[340] Fanjie Meng, Xiangwei Kong, and Xingang You. A new feature-based method for source camera identification. In Indrajit Ray and Sujeet Shenoi, editors, *Advances in Digital Forensics IV*, volume 285, pages 207–218, 2008.

[341] Xian-Zhe Meng, Shao-Zhang Niu, and Jian-Chen Zou. Tamper detection for shifted double JPEG compression. In *International Conference on Intelligent Information Hiding and Multimedia Signal Processing*, pages 434–437, 2010.

[342] Aravind K. Mikkilineni, Osman Arslan, Pei-Ju Chiang, Roy M. Kumontoy, Jan P. Allebach, George T.-C. Chiu, and Edward J. Delp. Printer forensics using SVM techniques. In *International Conference on Digital Printing Technologies*, pages 223–226, 2005.

[343] Aravind K. Mikkilineni, Pei-Ju Chiang, Gazi N. Ali, George T.-C. Chiu, Jan P. Allebach, and Edward J. Delp. Printer identification based on textural features. In *International Conference on Digital Printing Technologies*, pages 306–311, 2004.

[344] Aravind K. Mikkilineni, Pei-Ju Chiang, Gazi N. Ali, George T.-C. Chiu, Jan P. Allebach, and Edward J. Delp. Printer identification based on graylevel co-occurence features for security and forensic applications. In *SPIE International Conference on Security, Steganography, and Watermarking of Multimedia Contents*, pages 430–440, 2005.

[345] Aravind K. Mikkilineni, Nitin Khanna, and Edward J. Delp. Forensic printer detection using intrinsic signatures. In *SPIE Conference on Media Watermarking, Security, and Forensics*, 2011.

[346] N. Mondaini, Roberto Caldelli, Alessandro Piva, Mauro Barni, and Vito Cappellini. Detection of malevolent changes in digital video for forensic applications. In Edward J. Delp and Ping Wah Wong, editors, *SPIE Conference on Security, Steganography, and Watermarking of Multimedia Contents*, volume 6505, 2007.

[347] A. N. Myna, M. G. Venkateshmurthy, and C. G. Patil. Detection of region duplication forgery in digital images using wavelets and log-polar mapping. In *International Conference on Computational Intelligence and Multimedia Applications*, volume 3, pages 371–377, 2007.

[348] Gopal Narayanan and Yun Qing Shi. A statistical model for quantized ac block DCT coefficients in JPEG compression and its application to detecting. In *International Workshop on Digital Watermarking*, pages 75–89, 2010.

[349] Lakshmanan Nataraj, Anindya Sarkar, and Bangalore S. Manjunath. Adding gaussian noise to ”denoise” JPEG for detecting image resizing. In *IEEE International Conference on Image Processing*, pages 1493–1496, 2009.

[350] Lakshmanan Nataraj, Anindya Sarkar, and Bangalore S. Manjunath. Improving re-sampling detection by adding noise. In *SPIE Conference on Media Forensics and Security*, 2010.

[351] Ramesh Neelamani, Ricardo de Queiroz, Zhigang Fan, Sanjeeb Dash, and Richard G. Baraniuk. JPEG compression history estimation for color images. *IEEE Transactions on Image Processing*, 15(6):1365–1378, 2006.

[352] Tian-Tsong Ng. Camera response function signature for digital forensics - part ii: Signature extraction. In *IEEE Workshop on Information Forensics and Security*, pages 161–165, December 2009.

[353] Tian-Tsong Ng and Shih-Fu Chang. Classifying photographic and photorealistic computer graphic images using natural image statistics. Technical report, ADVENT Technical Report, #220-2006-6, Columbia University, October 2004.

[354] Tian-Tsong Ng and Shih-Fu Chang. A data set of authentic and spliced image blocks. Technical report, ADVENT Technical Report, #203-2004-3, Columbia University, June 2004.

[355] Tian-Tsong Ng and Shih-Fu Chang. A model for image splicing. In *IEEE International Conference on Image Processing*, 2004.

[356] Tian-Tsong Ng and Shih-Fu Chang. An online system for classifying computer graphics images from natural photographs. In *SPIE Conference on Security, Steganography, and Watermarking of Multimedia Contents*, January 2006.

[357] Tian-Tsong Ng and Shih-Fu Chang. Identifying and prefiltering images. *IEEE Signal Processing Magazine*, pages 49–58, March 2009.

[358] Tian-Tsong Ng, Shih-Fu Chang, Yu-Feng Hsu, and Martin Pepeljugoski. Columbia photographic images and photorealistic computer graphics dataset. Technical report, ADVENT Technical Report, #203-2004-3, Columbia University, February 2005.

[359] Tian-Tsong Ng, Shih-Fu Chang, Yu-Feng Hsu, Lexing Xie, and Mao-Pei Tsui. Physics-motivated features for distinguishing photographic images and computer graphics. In *ACM Multimedia*, November 2005.

[360] Tian-Tsong Ng, Shih-Fu Chang, Ching-Yung Lin, and Qibin Sun. *Multimedia Security Technologies for Digital Rights*, chapter Passive-Blind Image Forensics, pages 383–412. Elsevier, 2006.

[361] Tian-Tsong Ng, Shih-Fu Chang, and Qibin Sun. Blind detection of photomontage using higher order statistics. In *International Symposium on Circuits and Systems*, page 688691, 2004.

[362] Tian-Tsong Ng, Shih-Fu Chang, and Mao-Pei Tsui. Lessons learned from online classification of photorealistic computer graphics and photographs. In *IEEE Workshop on Signal Processing Applications for Public Security and Forensics*, 2007.

[363] Tian-Tsong Ng, Shih-Fu Chang, and Mao-Pei Tsui. Using geometry invariants for camera response function estimation. In *IEEE Conference on Computer Vision and Pattern Recognition*, June 2007.

[364] Tian-Tsong Ng and Mao-Pei Tsui. Camera response function signature for digital forensics - part i: Theory and data selection. In *IEEE Workshop on Information Forensics and Security*, pages 156–160, December 2009.

[365] Hieu Cuong Nguyen and Stefan Katzenbeisser. Security of copy-move forgery detection techniques. In *International Conference on Acoustics, Speech and Signal Processing*, 2011.

[366] Daniel Patricio Nicolalde and Jose Antonio Apolinario, Jr. Evaluating digital audio authenticity with spectral distances and enf phase change. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 1417–1420, 2009.

[367] Andrea Oermann, Andreas Lang, and Jana Dittmann. Verifier-tuple for audio-forensic to determine speaker environment. In *ACM Multimedia and Security Workshop*, pages 57–62, 2005.

[368] Andrea Oermann, Claus Vielhauer, and Jana Dittmann. Sensometrics: Identifying pen digitizers by statistical multimedia signal processing. In *SPIE Conference on Multimedia on Mobile Devices III*, 2007.

[369] Martin S. Olivier. On the probability of a false positive match for digital camera identification based on sensor dirt location. *South African Computer Journal*, 41:21–28, 2008.

[370] Martin S. Olivier. Using sensor dirt for toolmark analysis of digital photographs. In Indrajit Ray and Sujeet Shenoi, editors, *Advances in Digital Forensics IV*, volume 285, pages 193–206, 2008.

[371] Feng Pan and Jiwu Huang. Discriminating computer graphics images and natural images using hidden markov tree model. In *International Workshop on Digital Watermarking*, pages 23–28, 2010.

[372] Xunyu Pan and Siwei Lyu. Detecting image region duplication using sift features. In *International Conference on Acoustics, Speech, and Signal Processing*, pages 1706–1709, 2010.

[373] Xunyu Pan and Siwei Lyu. Region duplication detection using image feature matching. *IEEE Transactions on Information Forensics and Security*, 5(4):857 –867, 2010.

[374] Marie-Charlotte Poilpre, Patrick Perrot, and Hugues Talbot. Image tampering detection using bayer interpolation and JPEG compression. In *International Conference on Forensic Applications and Techniques in Telecommunications, Information, and Multimedia*, 2008.

[375] Stephen B. Pollard, Steven J. Simske, and Guy B. Adams. Model based print signature profile extraction for forensic analysis of individual text glyphs. In *Workshop on Information Forensics and Security*, 2010.

[376] Alin C. Popescu. *Statistical Tools for Digital Image Forensics*. PhD thesis, Department of Computer Science, Dartmouth College, Hanover, NH, 2005.

[377] Alin C. Popescu and Hany Farid. Exposing digital forgeries by detecting duplicated image regions. Technical Report TR2004-515, Department of Computer Science, Dartmouth College, 2004.

[378] Alin C. Popescu and Hany Farid. Statistical tools for digital forensics. In *International Workshop on Information Hiding*, pages 128–147, 2004.

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

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

[381] S. Prasad and K. R. Ramakrishnan. On resampling detection and its application to detect image tampering. In *IEEE International Conference on Multimedia and Expo*, pages 1325–1328, July 2006.

[382] Ghulam Qadir, Xi Zhao, and Anthony T. S. Ho. Estimating JPEG2000 compression for image forensics using benford's law. In *SPIE Conference on Optics, Photonics, and Digital Technologies for Multimedia Applications*, 2010.

[383] Zhenhua Qu, Weiqi Luo, and Jiwu Huang. A convolutive mixing model for shifted double JPEG compression with application to passive image authentication. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 1661–1664, 2008.

[384] Zhenhua Qu, Guoping Qiu, and Jiwu Huang. Detect digital image splicing with visual cues. In Stefan Katzenbeisser and Ahmad-Reza Sadeghi, editors, *International Workshop on Information Hiding*, pages 247–261, 2009.

[385] Judith A. Redi, Wiem Taktak, and Jean-Luc Dugelay. Digital image forensics: A booklet for beginners. *Multimedia Tools and Applications*, 51(1):133–162, 2010.

[386] Christian Riess and Elli Angelopoulou. Scene illumination as an indicator of image manipulation. In *International Workshop on Information Hiding*, 2010.

[387] Anderson Rocha and Siome Goldenstein. Is it fake or real? In *Brazilian Symposium of Computer Graphics and Image Processing*, 2006.

[388] Anderson Rocha, Walter Scheirer, Terrance E. Boult, and Siome Goldenstein. Vision of the unseen: Current trends and challenges in digital image and video forensics. *ACM Computing Surveys (CSUR)*, 43(4):26:1–26:42, 2011.

[389] Daniel P. N. Rodriguez, Jose A. Apolinario, and Luiz W. P. Biscainho. Audio authenticity: Detecting enf discontinuity with high precision phase analysis. *IEEE Transactions on Information Forensics and Security*, 5(3):534–543, 2010.

[390] Alessia De Rosa, Francesca Uccheddu, Alessandro Piva, Mauro Barni, and Andrea Costanzo. Exploring image dependencies: A new challenge in image forensics. In *SPIE Conference on Media Forensics and Security*, 2010.

[391] Kurt Rosenfeld and Husrev T. Sencar. A study of the robustness of prnu-based camera identification. In Edward J. Delp, Jana Dittmann, Nasir Memon, and Ping Wah Wong, editors, *SPIE Conference on Media Forensics and Security*, 2009.

[392] Dmitry Rublev, Vladimir Fedorov, and Oleg Makarevich. Digital camera identification based on rotated images. In *3rd International Conference on Security of Information and Networks*, pages 90–93, 2010.

[393] Seung-Jin Ryu, Hae-Yeoun Lee, Dong-Hyuck Im, Jung-Ho Choi, and Heung-Kyu Lee. Electrophotographic printer identification by halftone texture analysis. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 1846–1849, 2010.

[394] Seung-Jin Ryu, Min-Jeong Lee, and Heung-Kyu Lee. Detection of copy-rotate-move forgery using zernike moments. In *International Workshop on Information Hiding*, 2010.

[395] Priscila Saboia, Tiago Carvalho, and Anderson Rocha. Eye specular highlights telltales for digital forensics: A machine learning approach. In *International Conference on Image Processing*, 2011.

[396] Dario L. M. Sacchi, Franca Agnoli, and Elizabeth F. Loftus. Changing history: Doctored photographs affect memory for past public events. *Applied Cognitive Psychology*, (21):1005–1022, 2007.

[397] Naoki Saitoh, Kenji Kurosawa, Kenro Kuroki, Norimitsu Akiba, Zeno J. Geradts, and Jurrien Bijhold. Ccd fingerprint method for digital still cameras. In *SPIE Conference on Investigative Image Processing II*, pages 37–48, 2002.

[398] Richard W. Sanders. Digital audio authenticity using the electric network frequency. In *AES International Conference on Audio Forensics, Theory and Practice*, 2008.

[399] Gopinath Sankar, H. Vicky Zhao, and Yee-Hong Yang. Feature based classification of computer graphics and real images. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 1513–1516, 2009.

[400] Anindya Sarkar, Lakshmanan Nataraj, and Bangalore S. Manjunath. Detection of seam carving and localization of seam insertions in digital images. In *ACM Multimedia and Security Workshop*, pages 107–116, 2009.

[401] Husrev T. Sencar and Nasir Memon. *Algorithms, Architecture and Information Systems Security*, chapter Overview of State-of-the-Art in Digital Image Forensics, pages 325–348. World Scientific, 2007.

[402] Husrev T. Sencar and Nasir Memon. *Algorithms, Architectures and Information Systems Security*, chapter Overview of State-of-the-art in Digital Image Forensics, pages 325–348. World Scientific Press, 2008.

[403] Yun Qing Shi. First digit law and its application to digital forensics. In *International Workshop on Digital Watermarking*, 2008.

[404] Yun Qing Shi, Chunhua Chen, and Wen Chen. A natural image model approach to splicing detection. In *ACM Multimedia and Security Workshop*, pages 51–62, 2007.

[405] B. L. Shivakumar and S. Santhosh Baboo. Detecting copy-move forgery in digital images: A survey and analysis of current methods. *Global Journal of Computer Science and Technology*, 10(7):61–65, 2011.

[406] Matthew Sorell. Conditions for effective detection and identification of primary quantization of re-quantized JPEG images. In *International Conference on Forensic Applications and Techniques in Telecommunications, Information, and Multimedia*, 2008.

[407] Matthew Stamm and K. J. Ray Liu. Blind forensics of contrast enhancement in digital images. In *IEEE International Conference on Image Processing*, pages 3112–3115, 2008.

[408] Matthew Stamm and K. J. Ray Liu. Forensic detection of image tampering using intrinsic statistical fingerprints in histograms. In *Proc. APSIPA Annual Summit and Conference*, 2009.

[409] Matthew Stamm and K. J. Ray Liu. Forensic detection of image manipulation using statistical intrinsic fingerprints. *IEEE Transactions on Information Forensics and Security*, 5(3):492–506, 2010.

[410] Matthew Stamm and K. J. Ray Liu. Forensic estimation and reconstruction of a contrast enhancement mapping. In *International Conference on Acoustics, Speech, and Signal Processing*, pages 1698–1701, 2010.

[411] Matthew Stamm and K. J. Ray Liu. Wavelet-based image compression anti-forensics. In *International Conference on Image Processing*, 2010.

[412] Matthew Stamm and K. J. Ray Liu. Anti-forensics for frame deletion/addition in mpeg video. In *International Conference on Acoustics, Speech and Signal Processing*, 2011.

[413] Matthew Stamm and K. J. Ray Liu. Anti-forensics of digital image compression. *IEEE Transactions on Information Forensics and Security*, 6(3):1050–1065, 2011.

[414] Matthew Stamm, Steven Tjoa, W. Sabrina Lin, and K. J. Ray Liu. Anti-forensics of JPEG compression. In *International Conference on Acoustics, Speech, and Signal Processing*, pages 1694–1697, 2010.

[415] Matthew Stamm, Steven Tjoa, W. Sabrina Lin, and K. J. Ray Liu. Undetectable image tampering through JPEG compression anti-forensics. In *International Conference on Image Processing*, 2010.

[416] Martin Steinebach, Huajian Liu, Peishuai Fan, and Stefan Katzenbeisser. Cell phone camera ballistics: Attacks and countermeasures. In *SPIE Conference on Multimedia on Mobile Devices*, 2010.

[417] Martin Steinebach, Mohamed El Ouariachi, Huajian Liu, and Stefan Katzenbeisser. On the reliability of cell phone camera fingerprint recognition. In *International ICST Conference on Digital Forensics and Cyber Crime*, pages 69–76, 2009.

[418] Shao-Jie Sun, Qiong Wu, and Guo-Hui Li. Detection of image compositing based on a statistical model for natural images. *Acta Automatica Sinica*, 35(12):1564–1568, 2009.

[419] Yagiz Sutcu, Sevinc Bayram, Husrev T. Sencar, and Nasir Memon. Improvements on sensor noise based source camera identification. In *IEEE International Conference on Multimedia and Expo*, pages 24–27, 2007.

[420] Yagiz Sutcu, Baris Coskun, Husrev T. Sencar, and Nasir Memon. Tamper detection based on regularity of wavelet transform coefficients. In *IEEE International Conference on Image Processing*, September 2007.

[421] Patchara Sutthiwan, Xiao Cai, Yun Qing Shi, and Hong Zhang. Computer graphics classification based on markov process model and boosting feature selection technique. In *IEEE International Conference on Image Processing*, pages 2913–2916, 2009.

[422] Patchara Sutthiwan, Yun Qing Shi, Jing Dong, Tieniu Tan, and Tian-Tsong Ng. New developments in color image tampering detection. In *International Symposium on Circuits and Systems*, 2010.

[423] Patchara Sutthiwan, Yun Qing Shi, Wei Su, and Tian-Tsong Ng. Rake transform and edge statistics for image forgery detection. In *IEEE International Conference on Multimedia and Expo*, 2010.

[424] Patchara Sutthiwan, Jingyu Ye, and Yun Qing Shi. An enhanced statistical approach to identifying photorealistic images. In Anthony T. S. Ho, Yun Qing Shi, Hyoung-Joonq Kim, and Mauro Barni, editors, *International Workshop on Digital Watermarking*, pages 323–335, 2009.

[425] Ariawan Suwendi and Jan P. Allebach. Nearest-neighbor and bilinear resampling factor estimation to detect blockiness or blurriness of an image. *Journal of Electronic Imaging*, 17(2):94–101, 2008.

[426] Ashwin Swaminathan, Min Wu, and K. J. Ray Liu. Component forensics of digital cameras: A non-intrusive approach. In *Information Sciences and Systems*, pages 1194–1199, March 2006.

[427] Ashwin Swaminathan, Min Wu, and K. J. Ray Liu. Image tampering identification using blind deconvolution. In *IEEE International Conference on Image Processing*, pages 2309–2312, 2006.

[428] Ashwin Swaminathan, Min Wu, and K. J. Ray Liu. Non-intrusive forensic analysis of visual sensors using output images. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, May 2006.

[429] Ashwin Swaminathan, Min Wu, and K. J. Ray Liu. A component estimation framework for information forensics. In *Workshop on Multimedia Signal Processing*, pages 397–400, October 2007.

[430] Ashwin Swaminathan, Min Wu, and K. J. Ray Liu. Intrinsic fingerprints for image authentication and steganalysis. In *SPIE Conference on Security, Steganography, and Watermarking of Multimedia Contents*, February 2007.

[431] Ashwin Swaminathan, Min Wu, and K. J. Ray Liu. Nonintrusive component forensics of visual sensors using output images. *IEEE Transactions on Information Forensics and Security*, 2(1):91–106, mar 2007.

[432] Ashwin Swaminathan, Min Wu, and K. J. Ray Liu. Optimization of input pattern for semi non-intrusive component forensics of digital cameras. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 225–228, April 2007.

[433] Ashwin Swaminathan, Min Wu, and K. J. Ray Liu. Digital image forensics via intrinsic fingerprints. *IEEE Transactions on Information Forensics and Security*, 3(1):101–117, 2008.

[434] Ashwin Swaminathan, Min Wu, and K. J. Ray Liu. A pattern classification framework for theoretical analysis of component forensics. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 1665–1668, April 2008.

[435] Ashwin Swaminathan, Min Wu, and K. J. Ray Liu. Component forensics. *IEEE Signal Processing Magazine*, 26(2):38–48, mar 2009.

[436] Jun Takamatsu, Yasuyuki Matsushita, Tsukasa Ogasawara, and Katsushi Ikeuchi. Estimating demosaicing algorithms using image noise variance. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2010.

[437] Steven Tjoa, W. Sabrina Lin, and K. J. Ray Liu. Transform coder classification for digital image forensics. In *IEEE International Conference on Image Processing*, September 2007.

[438] Steven Tjoa, W. Sabrina Lin, H. Vicky Zhao, and K. J. Ray Liu. Block size forensic analysis in digital images. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2007.

[439] Min-Jen Tsai, Cheng-Liang Lai, and Jung Liu. Camera/mobile phone source identification for digital forensics. In *IEEE International Confernece on Acoustics, Speech, and Signal Processing*, volume 2, pages II–221–II–224, 2007.

[440] Min-Jen Tsai and Cheng-Sheng Wang. Adaptive feature selection for digital camera source identification. In *IEEE International Symposium on Circuits and Systems*, pages 412–415, 2008.

[441] Min-Jen Tsai, Cheng-Sheng Wang, and Jung Liu. A hybrid model for digital camera source identification. In *IEEE International Conference on Image Processing*, pages 2901–2904, 2009.

[442] Min-Jen Tsai and Guan-Hui Wu. Using image features to identify camera sources. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 2, 2006.

[443] Chih-Hsuan Tzeng and Wen-Hsiang Tsai. A new technique for authentication of image/video for multimedia applications. In *International Multimedia Conference on Multimedia and Security: New Challenges*, pages 23–26, 2001.

[444] Giuseppe Valenzise, Vitaliano Nobile, Marco Tagliasacchi, and Stefano Tubaro. Countering JPEG anti-forensics. In *International Conference on Image Processing*, 2011.

[445] Giuseppe Valenzise, Marco Tagliasacchi, and Stefano Tubaro. The cost of JPEG compression anti-forensics. In *International Conference on Acoustics, Speech and Signal Processing*, 2011.

[446] Wiger Van Houten and Zeno J. Geradts. Source video camera identification for multiply compressed videos originating from youtube. *Digital Investigation*, 6(1-2):48–60, 2009.

[447] Wiger Van Houten and Zeno J. Geradts. Using sensor noise to identify low resolution compressed videos from youtube. In Zeno J. Geradts, Katrin Y. Franke, and Cor. J. Veenman, editors, *International Workshop on Computational Forensics*, pages 104–115, 2009.

[448] Tran Van Lanh, Sabu Emmanuel, and Mohan S. Kankanhalli. Identifying source cell phone using chromatic aberration. In *IEEE International Conference on Multimedia and Expo*, July 2007.

[449] David Vazquez-Padin, Carlos Mosquera, and Fernando Perez-Gonzalez. Two-dimensional statistical test for the presence of almost cyclostationarity on images. In *International Conference on Image Processing*, 2010.

[450] David Vazquez-Padin and Fernando Perez-Gonzalez. Prefilter design for forensic resampling estimation. In *Workshop on Information Forensics and Security*, 2011.

[451] Bo Wang, Xiangwei Kong, and Xingang You. Source camera identification using support vector machines. In *Advances in Digital Forensics V*, pages 107–118, 2009.

[452] Jing Wang and Hongbin Zhang. Exposing digital forgeries by detecting traces of image splicing. In *International Conference on Signal Processing*, 2006.

[453] Jingwei Wang, Byung-Ho Cha, Seong-Ho Cho, and C. C. Jay Kuo. Understanding benford's law and its vulnerability in image forensics. In *IEEE International Conference on Multimedia and Expo*, pages 1568–1571, 2009.

[454] Junwen Wang, Guangjie Liu, Hongyuan Li, Yuewei Dai, and Zhiquan Wang. Detection of image region duplication forgery using model with circle block. In *International Conference on Multimedia Information Networking and Security*, pages 25–29, 2009.

[455] Junwen Wang, Guangjie Liu, Zhan Zhang, Yuewei Dai, and Zhiquan Wang. Fast and robust forensics for image region-duplication forgery. *Acta Automatica Sinica*, 35(12):1488–1495, 2009.

[456] Wei Wang, Jing Dong, and Tieniu Tan. Effective image splicing detection based on image chroma. In *IEEE International Conference on Image Processing*, pages 1257–1260, 2009.

[457] Wei Wang, Jing Dong, and Tieniu Tan. A survey of passive image tampering detection. In Anthony T. S. Ho, Yun Qing Shi, Hyoung-Joonq Kim, and Mauro Barni, editors, *International Workshop on Digital Watermarking*, pages 308–322, 2009.

[458] Wei Wang, Jing Dong, and Tieniu Tan. Image tampering detection based on stationary distribution of markov chain. In *International Conference on Image Processing*, 2010.

[459] Wei Wang, Jing Dong, and Tieniu Tan. Tampered region localization of digital color images based on JPEG compression noise. In *International Workshop on Digital Watermarking*, pages 120–133, 2010.

[460] Wei Wang, Jing Dong, and Tieniu Tan. Exploring DCT coefficient quantization effect for image tampering localization. In *Workshop on Information Forensics and Security*, 2011.

[461] Weihong Wang. *Digital Video Forensics*. PhD thesis, Department of Computer Science, Dartmouth College, Hanover, NH, 2009.

[462] Weihong Wang and Hany Farid. Exposing digital forgeries in video by detecting double mpeg compression. In *ACM Multimedia and Security Workshop*, pages 37–47, 2006.

[463] Weihong Wang and Hany Farid. Exposing digital forgeries in interlaced and de-interlaced video. *IEEE Transactions on Information Forensics and Security*, 3(2):438–449, 2007.

[464] Weihong Wang and Hany Farid. Exposing digital forgeries in video by detecting duplication. In *ACM Multimedia and Security Workshop*, pages 35–42, 2007.

[465] Weihong Wang and Hany Farid. Detecting re-projected video. In *International Workshop on Information Hiding*, pages 72–86, 2008.

[466] Weihong Wang and Hany Farid. Exposing digital forgeries in video by detecting double quantization. In *ACM Multimedia and Security Workshop*, pages 39–48, 2009.

[467] Xin Wang, Bo Xuan, and Si-Long Peng. Digital image forgery detection based on the consistency of defocus blur. In *International Conference on Intelligent Information Hiding and Multimedia Signal Processing*, pages 192–195, 2008.

[468] Ying Wang and Pierre Moulin. On discrimination between photorealistic and photographic images. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2006.

[469] Weimin Wei, Shuozhong Wang, and Zhenjun Tang. Estimation of rescaling factor and detection of image splicing. In *International Conference on Communication Technology*, pages 676–679, 2008.

[470] Weimin Wei, Shuozhong Wang, Xinpeng Zhang, and Zhenjun Tang. Estimation of image rotation angle using interpolation-related spectral signatures with application to blind detection of image forgery. *IEEE Transactions on Information Forensics and Security*, 5(3):507–517, 2010.

[471] Dave Wolin. Document verification and traceability through image quality analysis. In *International Conference on Digital Printing Technologies*, pages 214–217, 2002.

[472] Jie Wu, Markad V. Kamath, and Skip Poehlman. Detecting differences between photographs and computer generated images. In *International Conference on Signal Processing, Pattern Recognition, and Applications*, pages 268–273, 2006.

[473] Qiong Wu, Shao-Jie Sun, Wei Zhu, Guo-Hui Li, and Dan Tu. Detection of digital doctoring in exemplar-based inpainted images. In *International Conference on Machine Learning and Cybernetics*, pages 1222–1226, July 2008.

[474] Ruoyu Wu, Xiaolong Li, and Bin Yang. Identifying computer generated graphics via histogram features. In *International Conference on Image Processing*, 2011.

[475] Yubao Wu, Xiangwei Kong, Xingang You, and Yiping Guo. Printer forensics based on page document's geometric distortion. In *IEEE International Conference on Image Processing*, pages 2909–2912, 2009.

[476] Guanshuo Xu, Shang Gao, Yun Qing Shi, RuiMin Hu, and Wei Su. Camera-model identification using Markovian transition probability matrix. In Anthony T. S. Ho, Yun Qing Shi, Hyoung-Joonq Kim, and Mauro Barni, editors, *International Workshop on Digital Watermarking*, pages 294–307, 2009.

[477] Guanshuo Xu, Yun Qing Shi, and Wei Su. Camera brand and model identification using moments of 1-d and 2-d characteristic functions. In *IEEE International Conference on Image Processing*, pages 2917–2920, 2009.

[478] Toshihiko Yamasaki, Tomoaki Matsunami, and Kiyoharu Aizawa. Detecting resized JPEG images by analyzing high frequency elements in DCT coefficients. In *International Conference on Intelligent Information Hiding and Multimedia Signal Processing*, pages 567–570, 2010.

[479] Rui Yang, Zhenhua Qu, and Jiwu Huang. Detecting digital audio forgeries by checking frame offsets. In *ACM Multimedia and Security Workshop*, pages 21–26, 2008.

[480] Rui Yang, Yun Qing Shi, and Jiwu Huang. Defeating fake-quality mp3. In *ACM Multimedia and Security Workshop*, pages 117–124, 2009.

[481] Rui Yang, Yun Qing Shi, and Jiwu Huang. Detecting double compression of audio signal. In *SPIE Conference on Media Forensics and Security*, 2010.

[482] Shuiming Ye, Qibin Sun, and Ee-Chien Chang. Detecting digital image forgeries by measuring inconsistencies of blocking artifact. In *IEEE International Conference on Multimedia and Expo*, pages 12–15, 2007.

[483] Ido Yerushalmy and Hagit Hel-Or. Digital image forgery detection based on lens and sensor aberration. *International Journal of Computer Vision*, 92(1):71–91, 2011.

[484] Hang Yu, Tian-Tsong Ng, and Qibin Sun. Recaptured photo detection using specularity distribution. In *IEEE International Conference on Image Processing*, October 2008.

[485] Jun Yu, Scott A. Craver, and Enping Li. Toward the identification of dslr lenses by chromatic aberration. In *SPIE Conference on Media Watermarking, Security, and Forensics*, 2011.

[486] Chi Zhang and Hongbin Zhang. Detecting digital image forgeries through weighted local entropy. In *IEEE International Symposium on Signal Processing and Information Technology*, pages 62–67, December 2007.

[487] Chi Zhang and Hongbin Zhang. Digital camera identification based on canonical correlation analysis. In *IEEE Workshop on Multimedia Signal Processing*, pages 769–773, 2008.

[488] Chi Zhang and Hongbin Zhang. Digital camera identification based on curvelet transform. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 1389–1392, 2009.

[489] Chi Zhang and Hongbin Zhang. Identifying color image origin using curvelet transform. In *International Conference on Image Processing*, 2010.

[490] Jing Zhang, Zhanlei Feng, and Yuting Su. A new approach for detecting copy-move forgery in digital images. In *IEEE Singapore International Conference on Communication Systems*, pages 362–366, 2008.

[491] Jing Zhang, Yuting Su, and Mingyu Zhang. Exposing digital video forgery by ghost shadow artifact. In *ACM Multimedia and Security Workshop*, pages 49–54, 2009.

[492] Jing Zhang, Haiying Wang, and Yuting Su. Detection of double-compression in JPEG2000 images. In *International Symposium on Intelligent Information Technology Application*, volume 1, pages 418–421, 2008.

[493] Pin Zhang and Xiangwei Kong. Detecting image tampering using feature fusion. In *International Conference on Availability, Reliability and Security*, pages 335–340, Fukuoka, Japan, 2009.

[494] Ting Zhang and Rang-Ding Wang. Copy-move forgery detection based on SVD in digital image. In *International Congress on Image and Signal Processing*, 2009.

[495] Wei Zhang, Xiaochun Cao, Zhiyong Feng, Jiawan Zhang, and Ping Wang. Detecting photographic composites using two-view geometrical constraints. In *IEEE International Conference on Multimedia and Expo*, pages 1078–1081, 2009.

[496] Wei Zhang, Xiaochun Cao, Yanling Qu, Yuexian Hou, Handong Zhao, and Chenyang Zhang. Detecting and extracting the photo composites using planar homography and graph cut. *IEEE Transactions on Information Forensics and Security*, 5(3):544–555, 2010.

[497] Wei Zhang, Xiaochun Cao, Jiawan Zhang, Jigui Zhu, and Ping Wang. Detecting photographic composites using shadows. In *IEEE International Conference on Multimedia and Expo*, pages 1042–1045, 2009.

[498] Zhen Zhang, Jiquan Kang, and Yuan Ren. An effective algorithm of image splicing detection. In *International Conference on Computer Science and Software Engineering*, pages 1035–1039, 2008.

[499] Zhen Zhang, Yuan Ren, Xi-Jian Ping, Zhi-Yong He, and Shan-Zhong Zhang. A survey on passive-blind image forgery by doctor method detection. In *International Conference on Machine Learning and Cybernetics*, 2008.

[500] Zhen Zhang, Ying Zhou, Jiquan Kang, and Yuan Ren. Study of image splicing detection. In *International Conference on Intelligent Computing*, pages 1103–1110, 2008.

[501] Xudong Zhao, Jianhua Li, Shenghong Li, and Shilin Wang. Detecting digital image splicing in chroma spaces. In *International Workshop on Digital Watermarking*, pages 12–22, 2010.

[502] Yu Qian Zhao, Frank Y. Shih, and Yun Qing Shi. Passive detection of paint-doctored JPEG images. In *International Workshop on Digital Watermarking*, pages 1–11, 2010.

[503] Jiangbin Zheng and Miao Liu. A digital forgery image detection algorithm based on wavelet homomorphic filtering. In *International Workshop on Digital Watermarking*, pages 152–160, 2008.

[504] Linna Zhou, Dongming Wang, Yunbiao Guo, and Jingfei Zhang. Blur detection of digital forgery using mathematical morphology. In *International Symposium on Agent and Multi-Agent Systems: Technologies and Applications*, pages 990–998, 2007.

[505] Xiu Ming Zhu, Guo Rong Xuan, Qiu Ming Yao, Xue Feng Tong, and Yun Qing Shi. Resampling detection in information forensics. *Journal of Computer Applications*, 26(11):2596–2597, 2006.