maximum likelihood filters for speckle suppression in ultrasonic images have been designed in [KOT94TIPJ]. A representative real ultrasonic image of liver

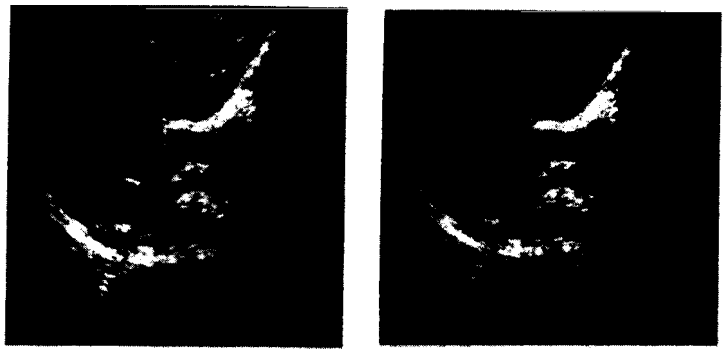

Fig. 3.9.4 (a) Real ultrasonic image of liver recorded using 3 MHz probe; (b) Output of signal-adaptive maximum likelihood filter.

recorded using 3 MHz probe is shown in Figure 3.9.4a. The output of the signal-adaptive maximum likelihood filter that exploits a local signal-to-noise ratio measure to adapt the filter window is shown in Figure 3.9.4b.

3.10 HISTOGRAM AND HISTOGRAM EQUALIZATION TECHNIQUES

A useful approach to digital image processing is to consider image intensities $f(i, j)$ as being raudom variables having probability density function (pdf) $p_f(f)$. Image pdf carries valuable global information about the image content. However, the pdf is generally not available and must be estimated from the image itself by using the empirical pdf usually called the *histogram.* Let us assume that the digital image has L discrete gray levels (usually from 0 to 255) and that *n~, k* = 0, . . . , L - 1, is the number of pixels having intensity k. The histogram ~ *f (f)* is given by the relation:

$$\hat{p}_f(f_k) = \frac{n_k}{n}, \; k=0,1,\ldots,L-1 \qquad (3.10.1)$$

where n is the total number of image pixels. The image histogram can be calculated easily, as can be seen in Figure 3.10.1.

```
int hist,(a, h, Ni,M1,N2,M2)
image a;
vector h;
int N1,Mi,N2,M2;

/* Subroutine to calculate the histogram of an image
```

```
a: image buffers
h: histogram buffer
N1,M1: start coordinates
N2,M2: end coordinates */

{ int i,j; float num;  unsigned long * lh;
lh=(unsigned long *)
malloc((unsigned int)NSIG*sizeof(unsigned long));
for(i=0; i<NSIG; i++) lh[i]=OL;
for(i=N1; i<N2; i++)  for(j=M1; j<M2; j++)
lh[(unsigned int)a[i] [j]]++;
num=(float)(N2-N1)*(float)(M2-M1);  for(i=0; i<NSIG;
i++) h[i]=lh[i]/num;  free(lh);
return(0);
}
```

Fig. 3.10.1 Algorithm for histogram calculation.

The image histogram carries important information about the image content. If its pixel values are concentrated in the low image intensities, as can be seen in Figure 3.10.2a, the image is `dark'. A `bright' image has a histogram that is concentrated in the high image intensities, as seen in Figure 3.10.2b. The histogram of Figure 3.10.2c reveals that the image contains two objects with different intensities (or, possibly, one object clearly distinguished from its background). If the image histogram is concentrated on a small intensity region, the image contrast is poor and the subjective image quality is low. Image quality can be enhanced by modifying its histogram. This can be

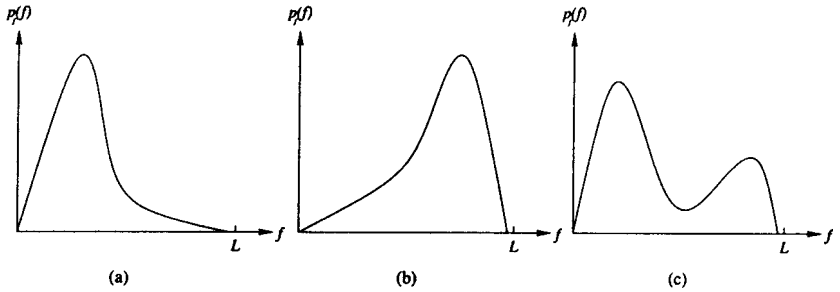

Fig. 3.10.2 (a) Histogram of a dark image; (b) histogram of a bright image; (c) histogram of an image containing two regions with different distributions.

performed by a technique called *histogram equalization.* Let us suppose that an image f is transformed by a nonlinear point-wise function $g = \mathrm{T}(f)$. The pdf $p_g(g)$ of the transformation output is given by [PAP65]:

$$p_g(g) = \frac{p_f(f_1)}{\left|\frac{d\mathrm{T}(f_1)}{df}\right|} \tag{3.10.2}$$

provided that the equation $g = \mathrm{T}(f)$ possesses only one solution f_1. Based on this relation, it can be easily proven [PAP65], [JAI89] that the transformation function:

$$\mathrm{T}(f) = \int_0^1 p_f(w)dw \tag{3.10.3}$$

produces an output image g whose pdf is uniform, $p_g(g) = 1$. Generally, the output image has higher contrast and better subjective quality than the original image f. However, it has to be noted that histogram equalization tends also to amplify noise. The transformation function $\mathrm{T}(f)$ can be approximated by the following relation:

$$g_k = \mathrm{T}(f_k) = \sum_{j=0}^{k} \hat{p}_f(f_j) = \sum_{j=0}^{k} \frac{n_j}{n} \tag{3.10.4}$$

A histogram equalization algorithm that employs (3.10.4) is shown in Figure 3.10.3.

```
int cdfhist(h,t) vector h; vector t;

/*  Subroutine to calculate the cdf of a histogram
    It is used in histogram equalization
    h: histogram buffer
    t: transformation buffer */

{   int i;
    t[0]=h[0] ;
    for(i=1; i<NSIG; i++)
        t[i]=t[i-1]+h[i];
    return(0);
}

int histeq(a,b, h, N1,M1,N2,M2)
image a,b;
vector h;

int N1,M1,N2,M2;
```

```
/* Subroutine for histogram equalization

   a: input buffer
   b: output buffer
   h: histogram cdf function
   N1,M1: start coordinates
   N2,M2: end coordinates */

{ int i,j;
  float * f;

  f=(float*)malloc((unsignedint)NSIG*sizeof(float));
  for(i=0; i<NSIG; i++) f[i]=h[i]*255.0;
  for(i=N1; i<N2; i++)
  for(j=M1; j<M2; j++)
     b[i][j]=(unsigned char)f[{unsigned int)a[i][j]];
  free(f);
  return(0);
}
```

Fig. 3.10.3 Algorithm for the calculation of the transformation function and for histogram equalization.

This algorithm has been applied to the confocal microscopy image of a neuron, shown in Figure 3.10.4a. This image has a very poor contrast. The equalized image has a much better contrast, as tan be Been in Figure 3.10.4b.

In certam cases, histogram modification rather than histogram equalization is desired. The image f having histogram $p_f(f)$ must be transformed to an image g having a predefined histogram $p_g(g)$. This tan be done in a two-step procedure. Let us suppose that we produte an image $S = \mathrm{T}(f)$ by using histogram equalization:

$$s = \mathrm{T}(f) = \int_0^1 p_f(w)dw \tag{3.10.5}$$

Let us also suppose that the desired image is available, is equalized and produces an image $z = G(g)$:

$$z = G(g) = \int_0^1 p_g(w)dw \tag{3.10.6}$$

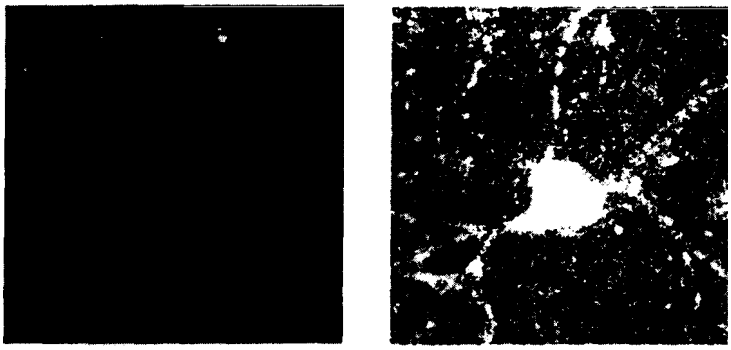

Fig. 3.10.4 (a) Confocal microscopy image of a neuron; (b) equalized image. *(Courtesy of Prof. A. Colchester, Neuromedia Group, University of Canterbury, UK.)*

The intermediate images s, z can be set equal to each other. Thus, the desired image g can be produced in two steps. In the first step, the image s is produced by using (3.10.5). The image g is produced in the second step by using:

$$g = G^{-1}(z) = G^{-1}(s) \tag{3.10.7}$$

The inverse function $G^{-1}(z)$ can be easily obtained from (3.10.6). These two steps can be combined in one transformation function:

$$g = G^{-1}[\mathrm{T}(f)] \tag{3.10.8}$$

Both transformation functions $\mathrm{T}(f), G(g)$ can be easily calculated by using a modification of the algorithm described in Figure 3.10.3.

3.11 PSEUDOCOLORING ALGORITHMS

In many applications, the human eye is more sensitive to color changes rather than to intensity changes for inspection applications (e.g. security inspection, quality control, remote sensing). Therefore, it is natural to encode the intensity of black and white (BW) images by using color information. This process is called *pseudocoloring.* The result of pseudocoloring is usually pleasant to the human eye. Thus, it can be used for artistic applications (e.g. creation of visual effects).

Pseudocoloring is a digital image transformation of the form:

$$c(x, y) = \mathrm{T}(f(x, y)) \tag{3.11.1}$$

where *f (x, y)* is a black and wbite image and *c(x, y)* is a color image. It is usually expressed in terms of its red, green and blue components $c = [c_R, c_G, c_B]^{\mathrm{T}}$. The transformation function T produces a three-channel output. Since the