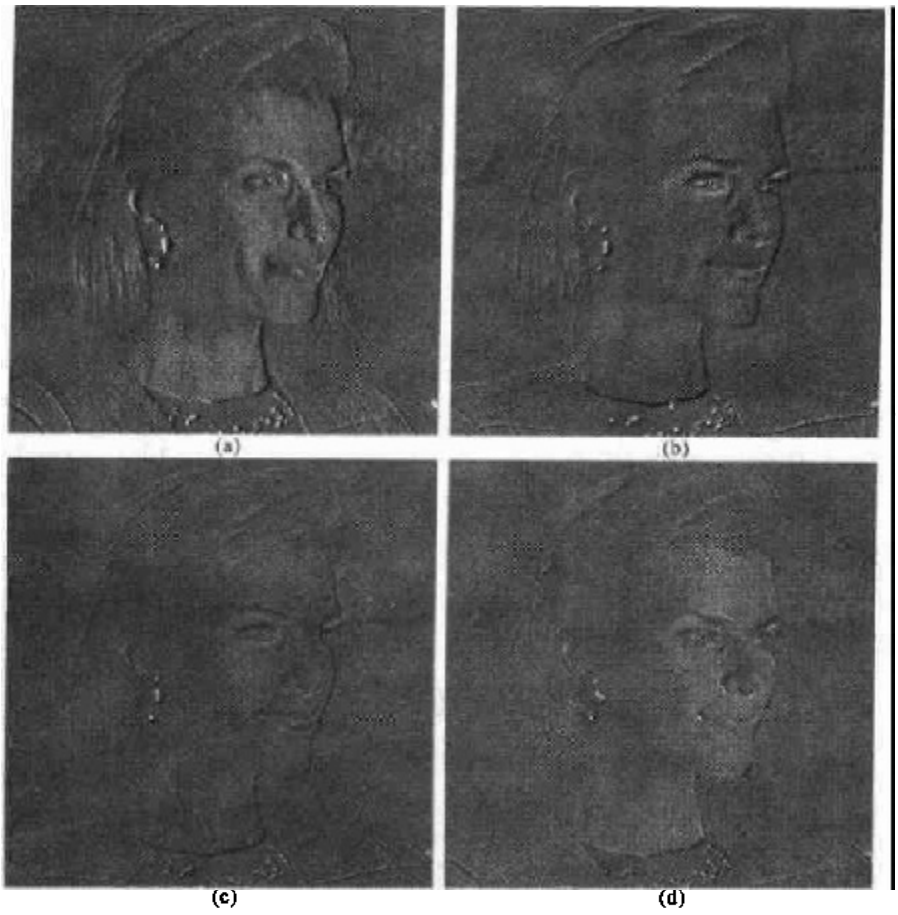

图 8.24 4种线性预测技术的对比

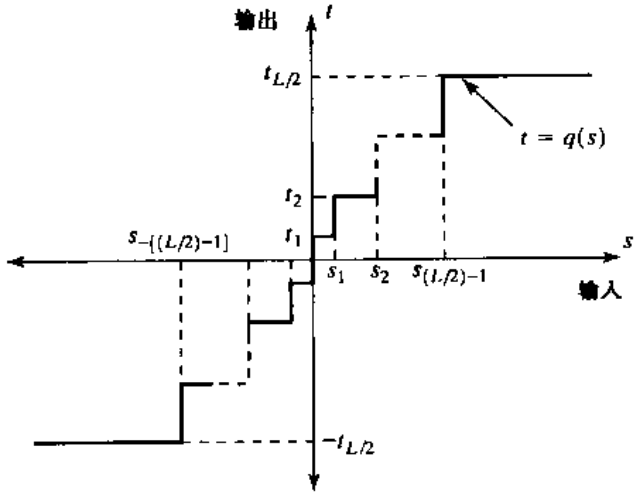

图 8.25 典型的量化函数

量化器的设计问题是,对于特殊的优化准则和输入概率密度函数 $p(s)$选择最佳的 s_i 和 t_i。如果最佳化准则(可以是统计上的或心理视觉量度上的[①])是均方量化误差最小,并且$p(s)$

① 参见 Netravali[1977]及 Limb 和 Rubinstein[1978]可了解更多的关于心理视觉量度的内容。

是偶函数，则对于最小误差(Max[1960])的条件是：

$$\int_{s_{i-1}}^{s_i} (s - t_i)p(s)\mathrm{d}s = 0 \quad i = 1,2,\cdots,\frac{L}{2} \tag{8.5.20}$$

$$s_i = \begin{cases} 0 & i = 0 \\ \dfrac{t_i + t_{i+1}}{2} & i = 1,2,\cdots,\dfrac{L}{2} - 1 \\ \infty & i = \dfrac{L}{2} \end{cases} \tag{8.5.21}$$

和

$$s_{-i} = -s_i \quad t_{-i} = -t_i \tag{8.5.22}$$

式(8.5.20)指明重构层是在概率密度函数 $p(s)$ 下超出特定判定区间的区域的质心，而式(8.5.21)指出判定层在重构层之间的一半处。式(8.5.22)是由于 q 为一个奇函数的结果。对于任意的 L，满足式(8.5.20)到式(8.5.22)的 s_i 和 t_i 在均方误差意义下是最佳的；相应的量化器称为 L 层的劳埃德·马克斯(Lloyd-Max)量化器。

表 8.10 列出了单位方差拉普拉斯概率密度函数[见式(8.4.10)]的 2 层、4 层和 8 层劳埃德·马克斯判定和重构层。因为对大多数非无效的 $p(s)$ 来说，求取式(8.5.20)到式(8.5.22)的明确的或闭合形式的解是困难的，所以这些值都是数字的(Paez 和 Glisson[1972])。所示的这 3 个量化器分别提供了 1,2 和 3 比特/像素的固定的输出速率。由于表 8.10 列出的是单位方差分布，所以 $\sigma \neq 1$ 的情况下，重构和判定层用概率密度函数的标准差乘以列表中的值得到。表中的最后一行列出了步长 θ，步长 θ 同时满足式(8.5.20)到式(8.5.22)和附加的限制条件：

$$t_i - t_{i-1} = s_i - s_{i-1} = \theta \tag{8.5.23}$$

如果使用变长码的符号编码器作为通常的图 8.21(a)的有损预测编码器，则步长为 θ 的最佳均匀量化器提供的编码率(用于拉普拉斯 pdf)，将低于具有相同输出保真度的固定长度编码的劳埃德·马克斯量化器(O'Neil[1971])。

表 8.10　单位方差的拉普拉斯概率密度函数的劳埃德·马克斯量化器

层次数	2		4		8	
i	s_i	t_i	s_i	t_i	s_i	t_i
1	∞	0.707	1.102	0.395	0.504	0.222
2			∞	1.810	1.181	0.785
3					2.285	1.576
4					∞	2.994
θ	1.414		1.087		0.731	

尽管劳埃德·马克斯和最佳均匀量化器不是自适应的，但由基于图像的局部状态调整量化层次可以得到更多的好处。在理论上，缓慢变化的区域的量化比较精细，而快速变化的区域的

量化却比较粗糙。这个方法同时可以减少颗粒状噪声和斜率过载,而在编码率上仅要求最小的增加量。代价是增加了量化器的复杂性。

例 8.18 量化和重构的说明

图 8.26(a),(c)和(e)显示了 DPCM 重构图像,这幅图像是由联合使用表 8.10 中的 2 层、4 层和 8 层劳埃德-马克斯量化器用式(8.5.18)的平面预测器得到的。量化器是由列表中的劳埃德-马克斯判定与重构层和从前面例子中得到的非量化的平面预测误差的标准差(即3.3个灰度级)相乘生成的。注意,由于斜率过载,使解码图像的边缘模糊了。在图 8.26(a)中这一点特别显著,它是由两层量化器产生的,但不如在图 8.26(c)和(e)中(它使用了 4 层和 8 层量化器)明显。图 8.27(a),(c)与(e)显示了这些经过解码的图像和图 8.23的原图像之间标定后的差异。

为了生成图 8.26(b),(d)和(f)中的解码图像,并得到图 8.27(b),(d)和(f)中的误差图像,使用自适应量化方法。在这种方法中,对于每 16 个像素的块可以在 4 个可能的量化器中选择最好的(在均方误差意义上)。这 4 个量化器是前述的最佳劳埃德-马克斯量化器的标定过的版本。标定因子为 0.5,1.0,1.75 和 2.5。因为在每个块上附加了两位编码以便确定选择的量化器,所以与量化器开关相联系的开销为 2/16 或 0.125 比特/像素。注意,由此而导致感觉误差上的显著减少,在编码率上则增加得相对较小。

表 8.11 列出了图 8.27(a)到(f)中差异图像的 rms 误差,连同一些预测器和量化器的组合的数字。注意,在均方误差意义上,两层自适应量化器的表现同四层非自适应量化器相近。另外,四层自适应量化器要强于八层非自适应方法。通常,数字上的结果表明式(8.5.15)、式(8.5.17)和式(8.5.19)的预测器表现出来的所有特点与式(8.5.18)的预测器的特点是一样的。使用每种量化方法得到的压缩效果列在表 8.11 的最后一行。注意,由于使用自适应方法产生的 rms 误差[式(8.1.8)]实质性的减少没有对压缩性能产生明显的影响。

8.5.2 变换编码

8.5.1 节中讨论的预测编码技术直接对图像的像素进行操作,因而是空间域的方法。在这一节中,将考虑基于改进图像变换的压缩技术。在变换编码中,一种可逆线性变换(比如,傅里叶变换)用于将图像映射到变换系数集,然后,这些系数被量化和编码。对大多数自然图像,大量系数的量级很小,可以进行不很精确的量化(或完全丢弃),几乎不会产生多少图像失真。包括第 4 章的离散傅里叶变换(DFT)在内的很多变换可以用于图像数据的变换。

表 8.11 有损 DPCM 均方根误差汇总

	劳埃德-马克斯量化器			自适应量化器		
预测器	2 层	4 层	8 层	2 层	4 层	8 层
式(8.5.16)	30.88	6.86	4.08	7.49	3.22	1.55
式(8.5.17)	14.59	6.94	4.09	7.53	2.49	1.12
式(8.5.18)	9.90	4.30	2.31	4.61	1.70	0.76

（续表）

预测器	劳埃德·马克斯量化器			自适应量化器		
	2 层	4 层	8 层	2 层	4 层	8 层
式(8.5.19)	38.18	9.25	3.36	11.46	2.56	1.14
压缩	8.00:1	4.00:1	2.70:1	7.11:1	3.77:1	2.56:1

图 8.28 显示了一个典型的变换编码系统。解码器执行的步骤（除了量化函数以外）与编码器是相反的。编码器执行 4 种相对简单的操作：子图分解、变换、量化和编码。一幅 $N \times N$ 大小的输入图像首先被分解为大小为 $n \times n$ 的子图像。这些子图进而被变换以生成 $(N/n)^2$ 个子图像变换阵列。每个阵列的大小为 $n \times n$。变换处理的目的是将每幅子图中的像素进行解相关，或用最少量的变换系数包含尽可能多的信息。在量化阶段有选择地消除或更粗略地量化带有最少信息的系数。这些系数对重构子图像质量的影响最小。编码处理在对量化的系数进行编码（通常使用变长码）后就终止了。任何或所有变换编码步骤都可以根据局部图像内容进行适应性调整，这被称为自适应变换编码。而如果这些步骤对所有子图都是固定的，则称为非自适应变换编码。

变换选择

基于各种二维离散变换的变换编码系统已经构建出来了或被广泛研究了。在给定的应用中选择特定的变换取决于可容忍的重构误差的大小和可用的计算资源。在变换系数的量化过程中（不是在变换阶段）实现了压缩。

考虑大小为 $N \times N$ 的图像 $f(x,y)$，该图像的正向离散变换 $T(u,v)$ 可以用通常的关系表示：

$$T(u,v) = \sum_{x=0}^{N-1}\sum_{y=0}^{N-1} f(x,y)g(x,y,u,v) \tag{8.5.24}$$

$u,v = 0,1,2,\cdots,N-1$。给定 $T(u,v)$，$f(x,y)$ 可以同样使用一般的离散反变换计算得到：

$$f(x,y) = \sum_{u=0}^{N-1}\sum_{v=0}^{N-1} T(u,v)h(x,y,u,v) \tag{8.5.25}$$

$x,y = 0,1,2,\cdots,N-1$。在这些公式中，$g(x,y,u,v)$ 和 $h(x,y,u,v)$ 分别称为正向和逆向变换核函数。由于在本节后面将会阐明的原因，这两个函数也被称为基础函数或基础图像。式(8.5.25)中的 $T(u,v)$，$u,v = 0,1,2,\cdots,N-1$，称为变换系数；这些系数可以看做关于基础函数 $h(x,y,u,v)$ 的 $f(x,y)$ 一系列展开的展开系数（见 7.2.1 节）。

如果下列等式成立，则式(8.5.24)中的正向核函数称为是可分离的：

$$g(x,y,u,v) = g_1(x,u)g_2(y,v) \tag{8.5.26}$$

另外，如果 g_1 在函数上等于 g_2，则核函数是对称的。此时，式(8.5.26)可以表示成下列形式：

$$g(x,y,u,v) = g_1(x,u)g_1(y,v) \tag{8.5.27}$$

图 8.26 DPCM结果图像。(a)1.0,(b)1.125,(c)2.0,(d)2.125,(e)3.0,(f)3.125比特/像素

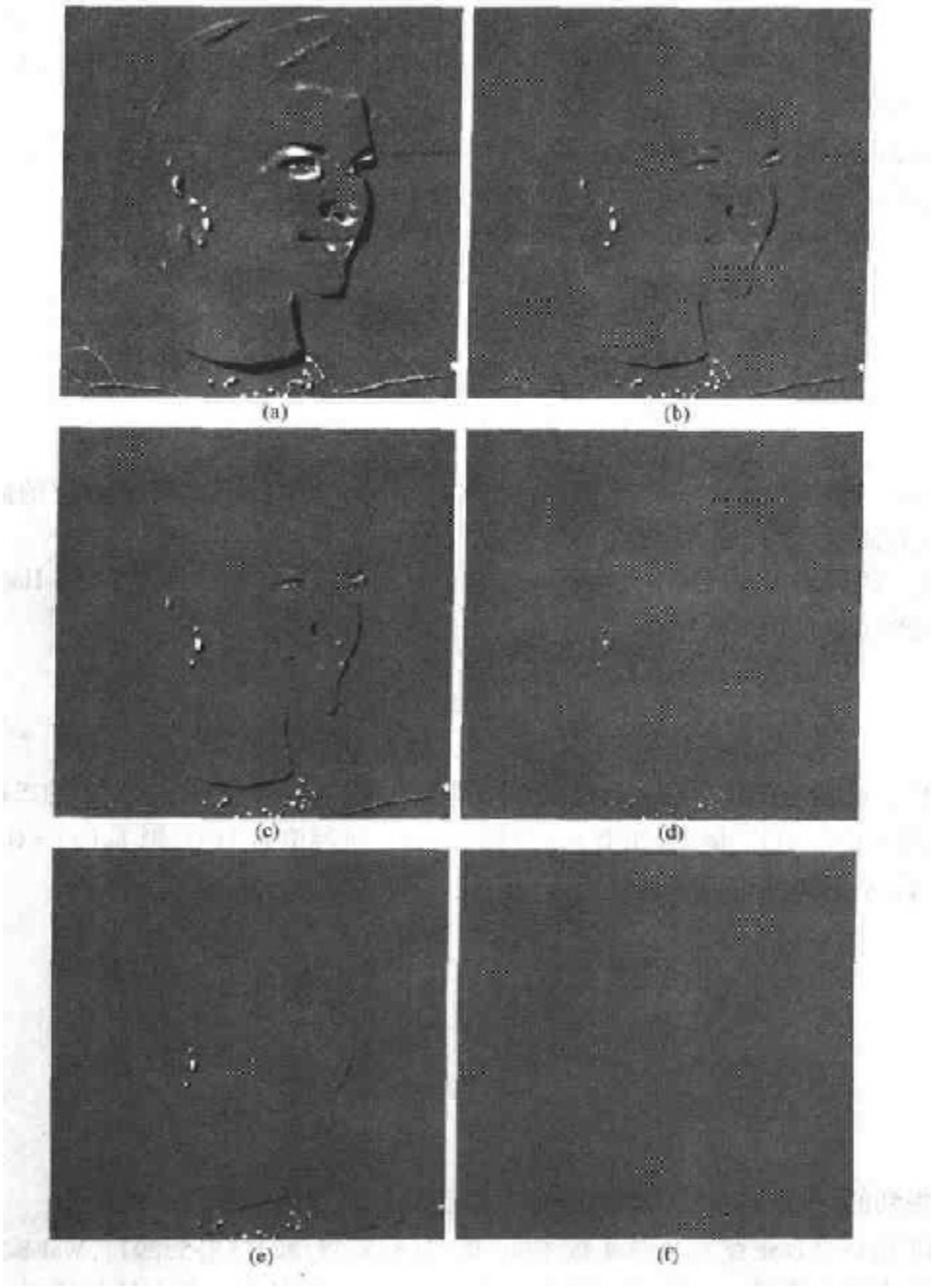

图 8.27　对应于图 8.26(a)到(f)的标定过的(×8)DPCM 误差图像

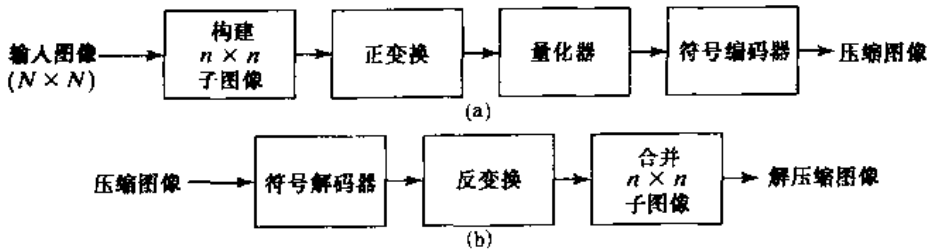

图 8.28　变换编码系统。(a)编码器,(b)解码器

如果 $g(x,y,u,v)$用式(8.5.26)和式(8.5.27)中的 $h(x,y,u,v)$代替,则对逆核也有相同的结论。说明具有可分离核的二维变换可以用对应的一维行-列或列-行变换计算得到,并不困难。4.6.1 节中解释了这种方法。

式(8.5.24)和式(8.5.25)中的正向和逆向变换核决定了所计算的变换类型和总体计算的复杂性以及所应用的变换编码系统的重构误差。最为有名的变换核函数对是:

$$g(x,y,u,v) = \frac{1}{N^2}e^{-j2\pi(ux+vy)/N} \tag{8.5.28}$$

和

$$h(x,y,u,v) = e^{j2\pi(ux+vy)/N} \tag{8.5.29}$$

这里 $j=\sqrt{-1}$。将这些核代入式(8.5.24)和式(8.5.25)得到在 4.2.2 节中介绍的离散傅里叶变换对的简化版本($M=N$)。

在变换编码过程中同样很有用处的一种在计算上更简单的变换称为 Walsh-Hadamard 变换(WHT)。这种变换是根据下列在函数上相同的核推导出来的:

$$g(x,y,u,v) = h(x,y,u,v) = \frac{1}{N}(-1)^{\sum_{i=0}^{m-1}[b_i(x)p_i(u)+b_i(y)p_i(v)]} \tag{8.5.30}$$

这里 $N=2^m$。求表达式的指数之和是以 2 为模的算术运算,并且 $b_k(z)$是 z 的二进制表达式的第 k 位(从右向左的)。例如,如果 $m=3$ 且 $z=6$(二进制值为 110),则 $b_0(z)=0$,$b_1(z)=1$,$b_2(z)=1$。式(8.5.30)中的 $p_i(u)$使用下面公式计算:

$$\begin{aligned}
p_0(u) &= b_{m-1}(u)\\
p_1(u) &= b_{m-1}(u) + b_{m-2}(u)\\
p_2(u) &= b_{m-2}(u) + b_{m-3}(u)\\
&\vdots\\
p_{m-1}(u) &= b_1(u) + b_0(u)
\end{aligned} \tag{8.5.31}$$

如前面所提到的,这里求和是以 2 为模的算术运算。对 $p_i(v)$有相似的表达式。

与 DFT 的核呈现正弦和余弦函数不同[见式(8.5.28)和式(8.5.29)],Walsh-Hadamard 核以棋盘形模式交替排列 +1 和 -1。图 8.29 显示了 $N=4$ 时的核。每个块包括 $4\times4=16$ 个元素(子方块)。白色表示 +1 而黑色表示 -1。为了得到左上角的块,令 $u=v=0$,并标出 $x,y=0,1,2,3$ 时 $g(x,y,0,0)$的值。此时所有的值均为 +1。顶部行中的第 2 个块是 $g(x,y,0,1)$在 $x,y=0,1,2,3$ 时取值的图,等等。正如已经看到的,Walsh-Hadamard 变换的重要意义在于它在实现上的简单性——所有核的值均为 +1 或 -1。

图像压缩中最常用到的一种变换是离散余弦函数变换(DCT)。这种变换是通过将下列(相等的)核代入式(8.5.24)和式(8.5.25)得到的:

$$\begin{aligned}
g(x,y,u,v) &= h(x,y,u,v)\\
&= \alpha(u)\alpha(v)\cos\left[\frac{(2x+1)u\pi}{2N}\right]\cos\left[\frac{(2y+1)v\pi}{2N}\right]
\end{aligned} \tag{8.5.32}$$

这里有：

$$\alpha(u) = \begin{cases} \sqrt{\dfrac{1}{N}} & u = 0 \\ \sqrt{\dfrac{2}{N}} & u = 1,2,\cdots,N-1 \end{cases} \tag{8.5.33}$$

对于 $\alpha(v)$的情况是类似的。图 8.30 显示了 $N=4$ 时的 $g(x,y,u,v)$。在计算方面遵循图 8.29中说明的格式，不同之处是 g 的值不是整数。在图 8.30 中，较浅的灰度级对应于更大的 g 值。

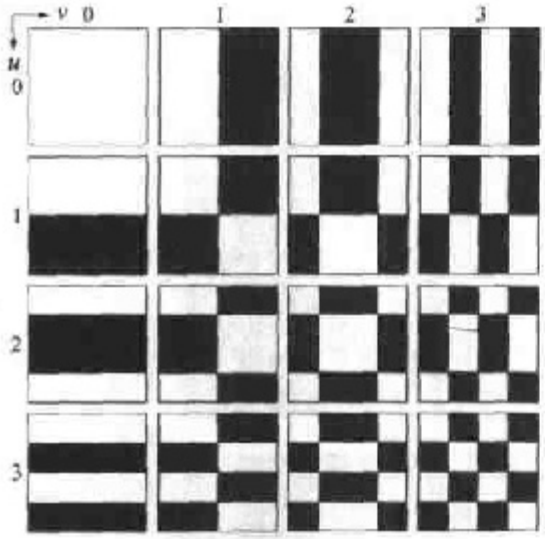

图 8.29　$N=4$ 时的 Walsh-Hadamard 基函数。每个块的原点在左上角

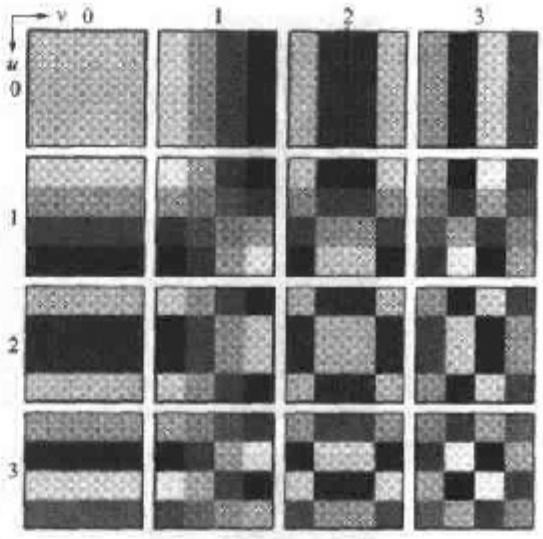

图 8.30　$N=4$ 时的离散余弦基函数。每个块的原点在块的左上角

例 8.19　分别使用 DFT，WHT 和 DCT 进行的变换编码

图 8.31(a)，(c)和(e)显示了对图 8.23 中的 512×512 大小的单色图像的 3 种近似。这些图像是这样得到的：先将原图分割为 8×8 大小的子图像，并使用刚才讲到的变换方法中的一种(即 DFT，WHT 或 DCT 变换)表示每一个子图像，将得到的系数的 50%截去，再对截取的系数阵列进行逆变换。

在每种情况下，保留下来的 32 个系数是根据最大值的量级进行选择的。当不考虑任何量化或编码问题的时候，这个处理过程实际上使用一个等于 2 的因子对原图像进行压缩。注意在所有情况下，被丢弃的 32 个系数对重构图像品质的视觉影响是微乎其微的。然而，这些系数的消去伴随着产生了均方误差，这种均方误差的影响在图 8.31(b)，(d)和(f)的标定后的误差图像中可以见到。实际的 rms 误差分别为 1.28，0.86 和 0.68 个灰度级。

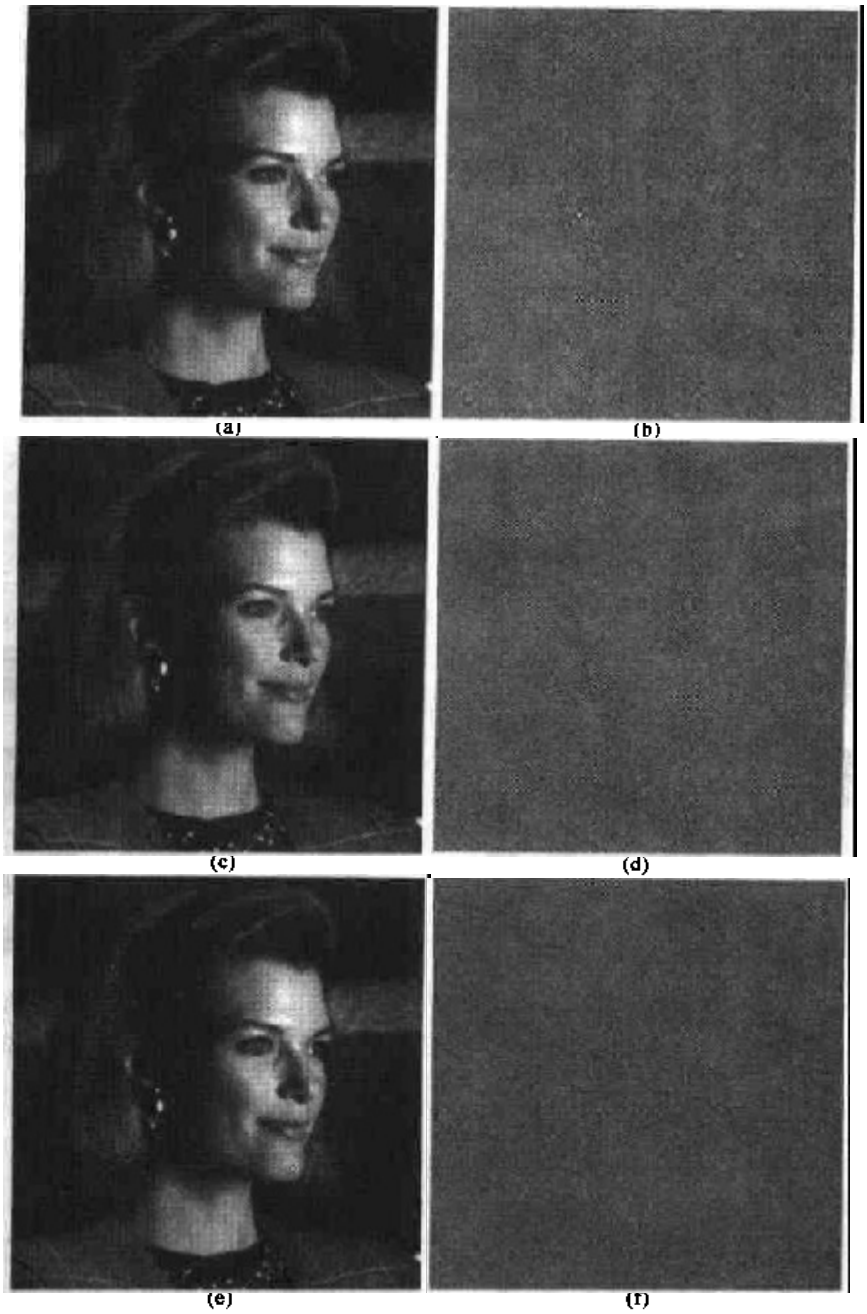

图 8.31 使用(a)傅里叶变换,(c)Hadamard 变换和(e)余弦变换对图8.23的近似,以及对应的标定后的误差图像

在前面的例子中注意到,均方重构误差中很小的差异与所使用的变换方法的信息包的特性或能量有直接关系。根据式(8.5.25),一幅 $n \times n$ 大小的图像 $f(x,y)$ 可以表示为它的二维变换 $T(u,v)$ 的函数:

$$f(x,y) = \sum_{u=0}^{n-1} \sum_{v=0}^{n-1} T(u,v) h(x,y,u,v) \tag{8.5.34}$$

$x,y = 0,1,\cdots,n-1$。注意,仅使用 n 代替了式(8.5.25)中的 N。现在考虑 $f(x,y)$ 如何表示被压缩图像的子图。由于式(8.5.34)中的逆核 $h(x,y,u,v)$ 仅取决于参数 x,y,u,v,并且与

$f(x,y)$或 $T(u,v)$的值无关，所以，这个逆核函数可以看做式(8.5.34)定义的一系列基函数或基础图像集。如果对式(8.5.34)使用的符号进行如下一些修改，这种解释会变得更为清晰：

$$\mathbf{F} = \sum_{u=0}^{n-1}\sum_{v=0}^{n-1} T(u,v)\mathbf{H}_{uv} \tag{8.5.35}$$

这里 $\mathbf{F}$ 是一个包含 $f(x,y)$的像素的 $n \times n$ 矩阵，且有：

$$\mathbf{H}_{uv} = \begin{bmatrix} h(0,0,u,v) & h(0,1,u,v) & \cdots & h(0,n-1,u,v) \\ h(1,0,u,v) & \cdot & \cdots & \cdot \\ \cdot & \cdot & \cdots & \cdot \\ \cdot & & & \\ h(n-1,0,u,v) & h(n-1,1,u,v) & \cdots & h(n-1,n-1,u,v) \end{bmatrix} \tag{8.5.36}$$

而包含输入子图像素的矩阵 $\mathbf{F}$ 明确被定义为 n^2 个大小为 $n \times n$ 的矩阵的线性组合；它们是式(8.5.36)中的 $\mathbf{H}_{uv}$，$u,v = 0,1,\cdots,n-1$。实际上，这些矩阵是式(8.5.35)的系列扩展的基础图像(或函数)；相关的 $T(u,v)$是扩展的系数。图 8.29 和图 8.30 用图形说明了 $n=4$ 时 WHT 和 DCT 的基础图像。

如果现在定义一个变换系数的模板函数：

$$\gamma(u,v) = \begin{cases} 0 & \text{如果 } T(u,v) \text{ 满足指定的截尾准则} \\ 1 & \text{其他} \end{cases} \tag{8.5.37}$$

$u,v = 0,1,\cdots,n-1$，$\mathbf{F}$ 的一种近似可以根据截取后的展开得到：

$$\hat{\mathbf{F}} = \sum_{u=0}^{n-1}\sum_{v=0}^{n-1} \gamma(u,v)T(u,v)\mathbf{H}_{uv} \tag{8.5.38}$$

这里，构造 $\gamma(u,v)$消除了式(8.5.35)中总和的最小分布的基础图像。子图 $\mathbf{F}$ 和它的近似 $\hat{\mathbf{F}}$ 之间的均方误差为：

$$\begin{aligned} e_{ms} &= E\{\| \mathbf{F} - \hat{\mathbf{F}} \|^2\} \\ &= \mathbf{E}\left\{\left\| \sum_{u=0}^{n-1}\sum_{v=0}^{n-1} T(u,v)\mathbf{H}_{uv} - \sum_{u=0}^{n-1}\sum_{v=0}^{n-1} \gamma(u,v)T(u,v)\mathbf{H}_{uv} \right\|^2\right\} \\ &= E\left\{\left\| \sum_{u=0}^{n-1}\sum_{v=0}^{n-1} T(u,v)\mathbf{H}_{uv}[1-\gamma(u,v)] \right\|^2\right\} \\ &= \sum_{u=0}^{n-1}\sum_{v=0}^{n-1} \sigma^2_{T(u,v)}[1-\gamma(u,v)] \end{aligned} \tag{8.5.39}$$

这里，$\| \mathbf{F} - \hat{\mathbf{F}} \|$ 是矩阵$(\mathbf{F} - \hat{\mathbf{F}})$的范数，$\sigma^2_{T(u,v)}$是位置$(u,v)$的变换系数的方差。最终的化简是以基础图像的正交性质和假定 $\mathbf{F}$ 的像素是由零均值随机过程与已知的协方差产生的为基础进行的。因此，总的均方差近似误差是被丢弃的变换系数的方差之和；即，这些系数使 $\gamma(u,v)=0$，因此，式(8.5.39)中的$[1-\gamma(u,v)]$是 1。将最多的信息装入或重新分配到最少的系数中的变换提供了最好的子图近似，因此，也有最小的重构误差。最后，在导出式(8.5.39)的假设条件下，$(N/n)^2$ 个 $N \times N$ 大小的图像的子图像的均方误差是相同的。因此，$N \times N$ 大小的图像的均方误差(是平均误差的一个量度)等于单个子图像的均方误差。

前面的例子显示出 DCT 的信息压缩能力比 DFT 和 WHT 的能力要强。尽管大多数自然图像通常都有这样的条件，但在数据压缩方面最佳的变换方法是 Karhunen-Loève 变换(第 11 章中对 KLT 变换进行了讨论)而不是 DCT。即，对任何输入图像和以任何数量保留下来的系数，