# Matlab图像处理大作业

无58 张硕 2015011186

# 第一章 第三节

2. (a) 测试图像的中心为圆心，图像的长和宽中较小值的一半为半径画一个红颜色的圆

|  |
| --- |
| load('hall.mat');  temp = hall\_color;  [xLen,yLen,~] = size(hall\_color);  r = min(xLen,yLen)/2;  xCen = (1+xLen)/2;  yCen = (1+yLen)/2;  for x = 1:xLen  if abs(x-xCen)>r  continue  else  y1 = round(yCen+sqrt(r^2-(x-xCen)^2));  y2 = round(yCen-sqrt(r^2-(x-xCen)^2));  temp(x,y1,1)=255;  temp(x,y2,1)=255;  temp(x,y1,2)=0;  temp(x,y2,2)=0;  temp(x,y1,3)=0;  temp(x,y2,3)=0;  end  end  for y = 1:yLen  if abs(y-yCen)>r  continue  else  x1 = round(xCen+sqrt(r^2-(y-yCen)^2));  x2 = round(xCen-sqrt(r^2-(y-yCen)^2));  temp(x1,y,1)=255;  temp(x2,y,1)=255;  temp(x1,y,2)=0;  temp(x2,y,2)=0;  temp(x1,y,3)=0;  temp(x2,y,3)=0;  end  end  imshow(temp);  imwrite(temp,'e1\_2\_a.bmp','bmp'); |



2. (b) (b) 将测试图像涂成国际象棋状的"黑白格”的样子，其中"黑”即黑色，"白”则意味着保留原图

|  |
| --- |
| load('hall.mat');  temp = hall\_color;  [xLen,yLen,~] = size(hall\_color);  for x = 1:xLen/8  for y = 1:yLen/8  if mod(x+y,2)==0  temp(8\*(x-1)+1:8\*(x-1)+8,8\*(y-1)+1:8\*(y-1)+8,1)=0;  temp(8\*(x-1)+1:8\*(x-1)+8,8\*(y-1)+1:8\*(y-1)+8,2)=0;  temp(8\*(x-1)+1:8\*(x-1)+8,8\*(y-1)+1:8\*(y-1)+8,3)=0;  end  end  end  imshow(temp);  imwrite(temp,'e1\_2\_b.bmp','bmp'); |



# 第二章 第四节

1. 图像的预处理是将每个像素灰度值减去 128，这个步骤是否可以在变换域进行？请在测试图像中截取一块验证你的结论

可以。每个像素灰度减某值相当于变换域减去相应直流分量。

|  |
| --- |
| load('hall.mat');  load('JpegCoeff.mat');  temp = hall\_gray(1:8,1:8);  res1 = temp - 128  temp = dct2(temp);  temp(1,1) = temp(1,1) - 128\*8;  res2 = idct2(temp) |

结果：

|  |
| --- |
| res1 =  8×8 uint8 矩阵  120 119 120 121 120 119 118 116  120 120 120 121 119 119 120 114  117 119 119 117 118 117 113 110  115 114 113 114 115 113 109 109  115 114 115 114 111 111 111 107  116 116 116 115 114 113 110 107  111 114 117 117 115 117 114 109  105 110 113 113 113 116 116 115  res2 =  120.0000 119.0000 120.0000 121.0000 120.0000 119.0000 118.0000 116.0000  120.0000 120.0000 120.0000 121.0000 119.0000 119.0000 120.0000 114.0000  117.0000 119.0000 119.0000 117.0000 118.0000 117.0000 113.0000 110.0000  115.0000 114.0000 113.0000 114.0000 115.0000 113.0000 109.0000 109.0000  115.0000 114.0000 115.0000 114.0000 111.0000 111.0000 111.0000 107.0000  116.0000 116.0000 116.0000 115.0000 114.0000 113.0000 110.0000 107.0000  111.0000 114.0000 117.0000 117.0000 115.0000 117.0000 114.0000 109.0000  105.0000 110.0000 113.0000 113.0000 113.0000 116.0000 116.0000 115.0000 |

2. 请编程实现二维DCT，并和MATLAB自带的库函数dct2比较是否一致

|  |
| --- |
| load('hall.mat');  load('JpegCoeff.mat');  temp = hall\_gray(1:8,1:8);  res1 = dct2(temp)  res2 = mydct(temp)    function res = mydct(P)  [M,~] = size(P);  D = sqrt(2/M).\*cos(kron([0:M-1]',2.\*[0:M-1]+1).\*pi/2/M);  D(1,:) = 1/sqrt(M);  res = D\*double(P)\*D';  end |

|  |
| --- |
| res1 =  1.0e+03 \*  1.9437 0.0088 -0.0097 0.0021 -0.0040 0.0018 -0.0000 -0.0004  0.0171 0.0058 0.0016 0.0022 0.0024 0.0004 -0.0009 0.0007  0.0108 -0.0097 -0.0010 -0.0020 -0.0017 0.0009 -0.0004 -0.0004  0.0019 0.0055 0.0008 0.0004 0.0010 0.0021 -0.0001 0.0001  -0.0063 -0.0045 0.0023 -0.0012 0.0020 -0.0003 0.0004 -0.0001  -0.0003 0.0011 -0.0025 0.0011 0.0012 -0.0019 0.0012 -0.0004  -0.0012 0.0012 0.0004 -0.0001 0.0004 -0.0023 0.0008 0.0004  0.0007 0.0007 -0.0009 -0.0010 -0.0020 0.0009 0.0003 0.0002  res2 =  1.0e+03 \*  1.9437 0.0088 -0.0097 0.0021 -0.0040 0.0018 -0.0000 -0.0004  0.0171 0.0058 0.0016 0.0022 0.0024 0.0004 -0.0009 0.0007  0.0108 -0.0097 -0.0010 -0.0020 -0.0017 0.0009 -0.0004 -0.0004  0.0019 0.0055 0.0008 0.0004 0.0010 0.0021 -0.0001 0.0001  -0.0062 -0.0045 0.0023 -0.0012 0.0020 -0.0003 0.0004 -0.0001  -0.0003 0.0011 -0.0025 0.0011 0.0012 -0.0019 0.0012 -0.0004  -0.0012 0.0012 0.0004 -0.0001 0.0004 -0.0023 0.0008 0.0004  0.0007 0.0007 -0.0009 -0.0010 -0.0020 0.0009 0.0003 0.0002 |

3. 如果将DCT系数矩阵中右侧四列的系数全部置零，逆变换后的图像会发生什么变化？选取一块图验证你的结论。如果左侧的四列置零呢？

右侧四列置零后高频分量消失，图片变模糊；左侧置零后低频分量消失，图像完全失真。

|  |
| --- |
| clc; clear all; close all;  load('hall.mat');  load('JpegCoeff.mat');  [xLen,yLen] = size(hall\_gray);  %ÈôÓÃzeros³õÊ¼»¯,µ¼ÖÂimshow(double),>1ÔòÊÓÎª1,Ö»ÄÜµÃµ½¶þÖµÍ¼Ïñ  res1 = hall\_gray;  res2 = hall\_gray;  for x = 1:xLen/8  for y = 1:yLen/8  D = dct2(hall\_gray(8\*x-7:8\*x,8\*y-7:8\*y));  temp1 = [D(:,1:4),zeros(8,4)];  temp2 = [zeros(8,4),D(:,5:8)];  res1(8\*x-7:8\*x,8\*y-7:8\*y) = idct2(temp1);  res2(8\*x-7:8\*x,8\*y-7:8\*y) = idct2(temp2);  end  end  figure;  imshow(hall\_gray);  imwrite(hall\_gray,'e2\_ori.bmp','bmp')  figure;  imshow(res1);  imwrite(res1,'e2\_3a.bmp','bmp')  figure;  imshow(res2);  imwrite(res2,'e2\_3b.bmp','bmp') |

4. 若对DCT系数分别做转置，旋转90度和旋转180度操作（rot90），逆变换后恢复的图像有何变化？选取一块图验证你的结论。

转制后每个分块变为了原来的转置；旋转90度后图片出现了水平失真，有明显分行；旋转180度后图片出现了水平和竖直的失真，有明显分块。

|  |
| --- |
| clc; clear all; close all;  load('hall.mat');  load('JpegCoeff.mat');  [xLen,yLen] = size(hall\_gray);  res1 = hall\_gray;  res2 = hall\_gray;  res3 = hall\_gray;  for x = 1:xLen/8  for y = 1:yLen/8  D = dct2(hall\_gray(8\*x-7:8\*x,8\*y-7:8\*y));  temp1 = D';  temp2 = rot90(D);  temp3 = rot90(rot90(D));  res1(8\*x-7:8\*x,8\*y-7:8\*y) = idct2(temp1);  res2(8\*x-7:8\*x,8\*y-7:8\*y) = idct2(temp2);  res3(8\*x-7:8\*x,8\*y-7:8\*y) = idct2(temp3);  end  end  figure;  imshow(res1);  imwrite(res1,'e2\_4a.bmp','bmp')  figure;  imshow(res2);  imwrite(res2,'e2\_4b.bmp','bmp')  figure;  imshow(res3);  imwrite(res3,'e2\_4c.bmp','bmp') |

5. 如果认为差分编码是一个系统，请绘出这个系统的频率响应，说明它是一个**高通**滤波器。DC系数先进行差分编码再进行墒编码，说明DC系数的 **低** 频率分量更多。

|  |
| --- |
| a = 1;  b = [-1,1];  freqz(b,a); |



6. DC预测误差的取值和Category值有何关系？如何利用预测误差计算出其Category？

|  |
| --- |
| function res = Category(CD)  if CD == 0  res = 0;  else  res = floor(log2(abd(CD)))+1;  end  end |

7. 你知道哪些实现Zig-Zag扫描的方法？请利用MATLAB的强大功能设计一种最佳方法。

方法一：（可调整维度）

|  |
| --- |
| function res = ZigZag(m)  res = zeros(64,1);  for x=1:8  for y=1:8  if x+y>9  slide = 17-x-y;  if mod(slide,2)==0  num = slide\*(slide-1)/2+9-x;  else  num = slide\*(slide-1)/2+9-y;  end  res(65-num) = m(x,y);  elseif x+y<9  slide = x+y-1;  if mod(slide,2)==0  num = slide\*(slide-1)/2+x;  else  num = slide\*(slide-1)/2+y;  end  res(num) = m(x,y);  else  res(28+x) = m(x,y);  end  end  end  end |

方法二：建立哈希表（更快）

|  |
| --- |
| function res = ZigZag(m)  res = [m(1,1);m(1,2);m(2,1);m(3,1);m(2,2);m(1,3);m(1,4);m(2,3);...  m(3,2);m(4,1);m(5,1);m(4,2);m(3,3);m(2,4);m(1,5);m(1,6);...  m(2,5);m(3,4);m(4,3);m(5,2);m(6,1);m(7,1);m(6,2);m(5,3);...  m(4,4);m(3,5);m(2,6);m(1,7);m(1,8);m(2,7);m(3,6);m(4,5);...  m(5,4);m(6,3);m(7,2);m(8,1);m(8,2);m(7,3);m(6,4);m(5,5);...  m(4,6);m(3,7);m(2,8);m(3,8);m(4,7);m(5,6);m(6,5);m(7,4);...  m(8,3);m(8,4);m(7,5);m(6,6);m(5,7);m(4,8);m(5,8);m(6,7);...  m(7,6);m(8,5);m(8,6);m(7,7);m(6,8);m(7,8);m(8,7);m(8,8)];  end |

8. 对测试图像分块、DCT和量化，将量化后的系数写成矩阵的形式，其中每一列为一个块的DCT系数Zig-Zag扫描后形成的列矢量，第一行为各个块的DC系数。

|  |
| --- |
| clc; clear all; close all;  load('hall.mat');  load('JpegCoeff.mat');  hall\_gray = hall\_gray-128;  [xLen,yLen] = size(hall\_gray);  res = zeros(64,(xLen/8)\*(yLen/8));  cnt = 1;  for x = 1:xLen/8  for y = 1:yLen/8  D = dct2(hall\_gray(8\*x-7:8\*x,8\*y-7:8\*y));  D = round(D./QTAB);  res(:,cnt) = ZigZag(D);  cnt=cnt+1;  end  end |

9. 请实现本罩介绍的JPEG编码（不包括写JFIF文件），输出为DC系数的码流、AC系数的码流、图像高度和图像宽度，将这四个变量写入jpegcodes.mat文件。

|  |
| --- |
| clc; clear all; close all;  load('hall.mat');  load('JpegCoeff.mat');  hall\_gray = double(hall\_gray)-128;  % hall\_gray = hall\_gray-128;  [xLen,yLen] = size(hall\_gray); %xÐÐÊýyÁÐÊý  zigzag = zeros(64,(xLen/8)\*(yLen/8));  cntBlock = 1;  for x = 1:xLen/8  for y = 1:yLen/8  D = dct2(hall\_gray(8\*x-7:8\*x,8\*y-7:8\*y));  D = round(D./QTAB);  zigzag(:,cntBlock) = ZigZag(D);  cntBlock=cntBlock+1;  end  end  cntBlock = cntBlock-1;  % DC  CD = zigzag(1,:);  CD(2:cntBlock)= CD(1:cntBlock-1)-CD(2:cntBlock);  DC=[];  for i=1:cntBlock  ctg=Category(CD(i));  len=DCTAB(ctg+1,1);  code=fliplr(bitget(abs(CD(i)),1:max(ctg,1)));  if(CD(i)<0)  code=1-code;  end  DC=[DC,DCTAB(ctg+1,2:1+len),code];  end  %AC  AC=[];  EOB=[1,0,1,0];  ZRL=[1,1,1,1,1,1,1,1,0,0,1];  for i=1:cntBlock  cnt0=0;  for j=2:64  if(zigzag(j,i)==0)  if(j==64)  AC=[AC,EOB];  continue;  end  cnt0=cnt0+1;  else %zigzag(j,i)~=0  %Run=cnt0, Size=Category, Amp=xxx  Size=Category(zigzag(j,i));  if(cnt0>15)  while(cnt0>15)  AC=[AC,ZRL];  cnt0=cnt0-16;  end  end  codeLen=ACTAB(10\*cnt0+Size,3);  code=fliplr(bitget(abs(zigzag(j,i)),1:Size));  if(zigzag(j,i)<0)  code=1-code;  end  AC=[AC,ACTAB(10\*cnt0+Size,4:(3+codeLen)),code];  cnt0=0;  end  end  end  save('jpegcodes.mat','DC','AC','xLen','yLen'); |

10.计算压缩比(输入文件长度/输出码流长度)，注意转换为相同进制。

|  |
| --- |
| clc; clear all; close all;  load('hall.mat');  load('jpegcodes.mat');  output = length(DC) + length(AC);  input = length(dec2bin(255)) \* xLen \* yLen;  res = input/output |

|  |
| --- |
| res =  6.4188 |

11. 请实现本章介绍的JPEG解码，输入是你生成的jpegcodes.mat文件。分别用客观 （PSNR）和主观方式评价编解码效果如何。

|  |
| --- |
| clc; clear all; close all;  load('hall.mat');  load('jpegcodes.mat');  load('JpegCoeff.mat');  zigzag=zeros(64,xLen\*yLen/64);  % DC  head=1;  cntBlock=1;  lenDC=length(DC);  while(head<=lenDC)  for category=0:11  len=DCTAB(category+1,1);  if(DC(head:(head+len-1))==DCTAB(category+1,2:1+len))  % right category  head=head+len;  code=DC(head:head+max(category,1)-1);  head=head+max(category,1);  if(category==0)  zigzag(1,cntBlock)=0;  elseif(code(1)==0) % negative  code=1-code;  x=-bin2dec(num2str(code));  zigzag(1,cntBlock)=x;  else % positive  x=bin2dec(num2str(code));  zigzag(1,cntBlock)=x;  end  cntBlock=cntBlock+1;  break;  end  end  end  for cntBlock=2:(xLen\*yLen/64)  zigzag(1,cntBlock)=zigzag(1,cntBlock-1)-zigzag(1,cntBlock);  end  % AC  head=1;  headB=2;  cntBlock=1;  lenAC=length(AC);  EOB=[1,0,1,0];  ZRL=[1,1,1,1,1,1,1,1,0,0,1];  while(head<=lenAC)  if(AC(head:head+3)==EOB) % EOB  head=head+4;  cntBlock=cntBlock+1;  headB=2;  continue;  elseif(AC(head:head+10)==ZRL)  head=head+11;  headB=headB+16;  continue;  else  for RunSize=1:160  len=ACTAB(RunSize,3);  % ½â¾ö½áÎ²Ê±¶Ô±ÈHuffmanÂëÔ½½çÎÊÌâ  if(head+len-1>lenAC)  continue;  end  if(AC(head:(head+len-1))==ACTAB(RunSize,4:(4+len-1))) % right RunSize  head=head+len;  Run=ACTAB(RunSize,1);  headB=headB+Run;  Size=ACTAB(RunSize,2);  code=AC(head:(head+Size-1));  head=head+Size;  if(code(1)==0) % negative  code=1-code;  x=-bin2dec(num2str(code));  zigzag(headB,cntBlock)=x;  else % positive  x=bin2dec(num2str(code));  zigzag(headB,cntBlock)=x;  end  % for\_test=[headB,cntBlock]  headB=headB+1;  break;  end  end  continue;  end  end  myHall=zeros(xLen,yLen);  cntBlock=1;  D=zeros(8,8);  for x = 1:xLen/8  for y = 1:yLen/8  D=antiZigZag(zigzag(:,cntBlock));  D=D.\*QTAB;  myHall(8\*x-7:8\*x,8\*y-7:8\*y)=idct2(D);  cntBlock=cntBlock+1;  end  end  myHall=uint8(round(myHall+128));  figure;  imshow(myHall);  imwrite(myHall,'e2\_9.bmp','bmp')  %PSNR  MSE=sum(sum((double(hall\_gray)-double(myHall)).^2))/xLen/yLen;  PSNR=10\*log10(255\*255/MSE)      function res = antiZigZag(vec)  res=zeros(8,8);  Hash=[1, 2, 6, 7, 15,16,28,29;  3, 5, 8, 14,17,27,30,43;  4, 9, 13,18,26,31,42,44;  10,12,19,25,32,41,45,54;  11,20,24,33,40,46,53,55;  21,23,34,39,47,52,56,61;  22,35,38,48,51,57,60,62;  36,37,49,50,58,59,63,64];  for row=1:8  for col=1:8  res(row,col)=vec(Hash(row,col));  end  end  end |

|  |
| --- |
| PSNR =  31.1874 |

原图/编解码：



客观评价PSNR为31.1874；肉眼主观评价，编解码基本保持了原图包含的信息，不过在门窗等边缘处能看到模糊。

12.将量化步长减小为原来的一半，重做编解码。同标准量化步长的情况比较压缩比和图像质量。

在e2\_9、e2\_10、e2\_11基础上添加代码：

|  |
| --- |
| QTAB=QTAB./2; |

|  |
| --- |
| compressRatio =  4.4081 |

|  |
| --- |
| PSNR =  34.2067 |



量化步长减半后，与原量化步长（压缩比6.4188，峰值信噪比31.1874）相比，压缩比减小，峰值信噪比上升，图像质量变好。这是因为量化过程的舍入误差造成失真，量化步长减小，量化结果数值变大，码流变大，同时舍入误差减小，失真减小。

13. 看电视时偶尔能看到美丽的雪花图像（见snow.mat)，请对其编解码。和测试图像的压缩比和图像质量进行比较，并解释比较结果。

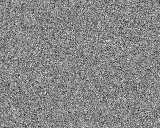
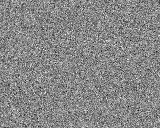
在e2\_9、e2\_10、e2\_11基础上添加代码：

|  |
| --- |
| load('snow.mat');  hall\_gray=snow;  imwrite(snow,'e2\_13ori.bmp','bmp') |

|  |
| --- |
| compressRatio =  3.6407 |

|  |
| --- |
| PSNR =  22.9244 |

原图/编解码：



与测试图像相比，雪花图像压缩比下降，峰值信噪比下降，仔细观察可以发现编解码图像很多细节与原图像不同，图像质量下降。量化矩阵中右下对应高频的量化系数较大，因此高频分量失真严重；二雪花噪音图像含有较多高频分量，失真严重，且DCT系数矩阵中0较少，AC码流较长，因此压缩比下降。

# 第三章 第四节

1. 实现本章介绍的空域隐藏方法和提取方法。验证其抗 JPEG 编码能力。

实现空域隐藏：

|  |
| --- |
| % Hidden: "XinHaoYuXiTongHenYouQv"  clc; clear all; close all;  load('hall.mat');  [xLen,yLen] = size(hall\_gray);  STX='00000010'; % start of text  ETX='00000011'; % end of text  text='XinHaoYuXiTongHenYouQv';  ascii=dec2bin(text);  [nChar,len]=size(ascii);  if(len<8)  ascii=[char(ones(nChar,8-len).\*48),ascii];  end  ascii=[STX;ascii;ETX];  nChar=nChar+2;  xHall=1; yHall=1;  for x=1:nChar  for y=1:8  if(yHall>yLen)  xHall=xHall+1;  yHall=1;  end  temp=dec2bin(hall\_gray(xHall,yHall));  temp(end)=ascii(x,y);  hall\_gray(xHall,yHall)=bin2dec(temp);  yHall=yHall+1;  end  end  save('e3\_1.mat','hall\_gray'); |

实现空域提取：

|  |
| --- |
| clc; clear all; close all;  load('e3\_1.mat');  [xLen,yLen] = size(hall\_gray);  STX='00000010'; % start of text  ETX='00000011'; % end of text  cnt8=1;  cntRes=1;  res='';  ascii='';  flagBreak=false;  for x=1:xLen  for y=1:yLen  b=dec2bin(hall\_gray(x,y));  ascii=[ascii,b(end)];  cnt8=cnt8+1;  if(cnt8>8)  cnt8=1;  res=[res;ascii];  if([x==1,y==8,ascii~=STX])  res='';  '未发现隐藏信息'  flagBreak=true;  break;  end  if(ascii==ETX)  flagBreak=true;  break;  end  ascii='';  end  end  if(flagBreak)  break;  end  end  dec=bin2dec(res);  hidden=char(dec)';  hidden=hidden(2:end-1) |

实现结果：

|  |
| --- |
| hidden =  XinHaoYuXiTongHenYouQv |

验证抗JPEG编码能力：

e3\_1中改为 load('e3\_1JPEG.mat');并在e2\_9、e2\_11基础上添加代码：

|  |
| --- |
| load('e3\_1.mat');  save('e3\_1JPEG.mat','hall\_gray'); |

运行空域提取得到：

|  |
| --- |
| hidden =  Ë pÒ¤Ô™ã$çÿ$F\_5º ¸ô@³q˜$ç$/Ã‚ u%1UA,$$½$šÊ{ó ¿¬ëAÇ®$Ûf$(Ð Ø%wÉ$¸$f$;¢¾ ©\*Ë}$ÿ½$]ÅŸ¨ %?áS1$Z$D~Ý6 ì’ãøÇ$™ÿ$Aÿ}L0' éSðç‹6Q½ ñ f‡ÿª2Õ ¿‰‡[5”É¡ ÂÿZÃªÀ=° æi§Z£lZÿV Zÿ–QÞ~ ÓTÇÉº$ Í ¥<’²ìš ó ”P\_6Qç >ÿZ½ŠtMV mÍ6Û ÿ¾ ¥‡ÿ\*3º§ >¨áN‹”“ •ÿ¥AÃªgyH Dé5òÄ‹6Çç q f3T6ì(0Ì:W~\_Z • l«ôR?æÍN0<ø[Ybf¥ŒÎÂ Ëç"\*gï¢fáþ |

可见空域信息隐藏技术抗JPEG编码能力不强。

2. 依次实现本章介绍的三种变换域信息隐藏方法和提取方法，分析嵌密方法的隐蔽性以及嵌密后JPEG图像的质量变化和压缩比变化。

（1）方法一 DCT域信息隐藏

编码部分：

|  |
| --- |
| % Hidden: "XinHaoYuXiTongHenYouQv"  clc; clear all; close all;  load('hall.mat');  [xLen,yLen] = size(hall\_gray);  STX='00000010'; % start of text  ETX='00000011'; % end of text  text='XinHaoYuXiTongHenYouQv';  ascii=dec2bin(text);  [nChar,len]=size(ascii);  if(len<8)  ascii=[char(ones(nChar,8-len).\*48),ascii];  end  ascii=[STX;ascii;ETX];  nChar=nChar+2;    % JPEG coding  load('JpegCoeff.mat');  hall\_gray = double(hall\_gray)-128;  % hall\_gray = hall\_gray-128;  [xLen,yLen] = size(hall\_gray); %xÐÐÊýyÁÐÊý  zigzag = zeros(64,(xLen/8)\*(yLen/8));  cntBlock = 1;  cntChar=1;  for x = 1:xLen/8  for y = 1:yLen/8  D = dct2(hall\_gray(8\*x-7:8\*x,8\*y-7:8\*y));  D = round(D./QTAB);  for xx=1:8  if(cntChar>nChar)  break;  end  for yy=1:8  if(D(xx,yy)<0)  temp=dec2bin(-D(xx,yy));  temp(end)=ascii(cntChar,yy);  D(xx,yy)=-bin2dec(temp);  else  temp=dec2bin(D(xx,yy));  temp(end)=ascii(cntChar,yy);  D(xx,yy)=bin2dec(temp);  end  end  cntChar=cntChar+1;  if(cntChar>nChar)  break;  end  end  zigzag(:,cntBlock) = ZigZag(D);  cntBlock=cntBlock+1;  end  end  cntBlock = cntBlock-1;  % DC  CD = zigzag(1,:);  CD(2:cntBlock)= CD(1:(cntBlock-1))-CD(2:cntBlock);  DC=[];  for i=1:cntBlock  ctg=Category(CD(i));  len=DCTAB(ctg+1,1);  % code=bitget(abs(CD(i)),max(ctg,1):-1:1);  code=fliplr(bitget(abs(CD(i)),1:max(ctg,1)));  if(CD(i)<0)  code=1-code;  end  DC=[DC,DCTAB(ctg+1,2:1+len),code];  end  %AC  AC=[];  EOB=[1,0,1,0];  ZRL=[1,1,1,1,1,1,1,1,0,0,1];  for i=1:cntBlock  cnt0=0;  for j=2:64  if(zigzag(j,i)==0)  if(j==64)  % AC=[AC,EOB];  continue;  end  cnt0=cnt0+1;  else %zigzag(j,i)~=0  %Run=cnt0, Size=Category, Amp=xxx  Size=Category(zigzag(j,i));  if(cnt0>15)  while(cnt0>15)  AC=[AC,ZRL];  cnt0=cnt0-16;  end  end  codeLen=ACTAB(10\*cnt0+Size,3);  % code=fliplr(bitget(abs(zigzag(j,i)),1:Size));  code=bitget(abs(zigzag(j,i)),Size:-1:1);  if(zigzag(j,i)<0)  code=1-code;  end  AC=[AC,ACTAB(10\*cnt0+Size,4:(3+codeLen)),code];  cnt0=0;  end  end  AC=[AC,EOB];  end  save('jpegcodes4.mat','DC','AC','xLen','yLen');  output = length(DC) + length(AC);  input = length(dec2bin(255)) \* xLen \* yLen;  compressRatio = input/output |

压缩比：

|  |
| --- |
| compressRatio =  6.3409 |

解码部分：

|  |
| --- |
| clc; clear all; close all;  load('hall.mat');  load('jpegcodes4.mat');  load('JpegCoeff.mat');  zigzag=zeros(64,xLen\*yLen/64);  % DC  head=1;  cntBlock=1;  lenDC=length(DC);  while(head<=lenDC)  for category=0:11  len=DCTAB(category+1,1);  if(DC(head:(head+len-1))==DCTAB(category+1,2:1+len))  % right category  head=head+len;  code=DC(head:head+max(category,1)-1);  head=head+max(category,1);  if(category==0)  zigzag(1,cntBlock)=0;  elseif(code(1)==0) % negative  code=1-code;  x=-bin2dec(num2str(code));  zigzag(1,cntBlock)=x;  else % positive  x=bin2dec(num2str(code));  zigzag(1,cntBlock)=x;  end  cntBlock=cntBlock+1;  break;  end  end  end  for cntBlock=2:(xLen\*yLen/64)  zigzag(1,cntBlock)=zigzag(1,cntBlock-1)-zigzag(1,cntBlock);  end  % AC  head=1;  headB=2;  cntBlock=1;  lenAC=length(AC);  EOB=[1,0,1,0];  ZRL=[1,1,1,1,1,1,1,1,0,0,1];  while(head<=lenAC)  if(AC(head:head+3)==EOB) % EOB  head=head+4;  cntBlock=cntBlock+1;  headB=2;  continue;  elseif(AC(head:head+10)==ZRL)  head=head+11;  headB=headB+16;  continue;  else  for RunSize=1:160  len=ACTAB(RunSize,3);  % ½â¾ö½áÎ²Ê±¶Ô±ÈHuffmanÂëÔ½½çÎÊÌâ  if(head+len-1>lenAC)  continue;  end  if(AC(head:(head+len-1))==ACTAB(RunSize,4:(4+len-1))) % right RunSize  head=head+len;  Run=ACTAB(RunSize,1);  headB=headB+Run;  Size=ACTAB(RunSize,2);  code=AC(head:(head+Size-1));  head=head+Size;  if(code(1)==0) % negative  code=1-code;  x=-bin2dec(num2str(code));  zigzag(headB,cntBlock)=x;  else % positive  x=bin2dec(num2str(code));  zigzag(headB,cntBlock)=x;  end  % for\_test=[headB,cntBlock]  headB=headB+1;  break;  end  end  continue;  end  end    STX='00000010'; % start of text  ETX='00000011'; % end of text  res='';  ascii='';  flagNotOver=true;    myHall=zeros(xLen,yLen);  cntBlock=1;  D=zeros(8,8);  for x = 1:xLen/8  for y = 1:yLen/8  D=antiZigZag(zigzag(:,cntBlock));  if(flagNotOver)  for xx=1:8  for yy=1:8  if(D(xx,yy)<0)  b=dec2bin(-D(xx,yy));  else  b=dec2bin(D(xx,yy));  end  ascii=[ascii,b(end)];  end  res=[res;ascii];  if(ascii==ETX)  flagNotOver=false;  break;  end  ascii='';  end  end    D=D.\*QTAB;  myHall(8\*x-7:8\*x,8\*y-7:8\*y)=idct2(D);  cntBlock=cntBlock+1;  end  end  myHall=uint8(round(myHall+128));  dec=bin2dec(res);  hidden=char(dec)';  hidden=hidden(2:end-1)  %PSNR  MSE=sum(sum((double(hall\_gray)-double(myHall)).^2))/xLen/yLen;  PSNR=10\*log10(255\*255/MSE) |

提取结果及PSNR：

|  |
| --- |
| hidden =  XinHaoYuXiTongHenYouQv  PSNR =  30.0970 |

（2）方法二 DCT域信息隐藏，但每个分块隐藏一个字符，仅ZigZag扫描后的前8位隐藏信息。

编码部分：

|  |
| --- |
| % Hidden: "XinHaoYuXiTongHenYouQv"  clc; clear all; close all;  load('hall.mat');  [xLen,yLen] = size(hall\_gray);  STX='00000010'; % start of text  ETX='00000011'; % end of text  text='XinHaoYuXiTongHenYouQv';  ascii=dec2bin(text);  [nChar,len]=size(ascii);  if(len<8)  ascii=[char(ones(nChar,8-len).\*48),ascii];  end  ascii=[STX;ascii;ETX];  nChar=nChar+2;    % JPEG coding  load('JpegCoeff.mat');  hall\_gray = double(hall\_gray)-128;  % hall\_gray = hall\_gray-128;  [xLen,yLen] = size(hall\_gray); %xÐÐÊýyÁÐÊý  zigzag = zeros(64,(xLen/8)\*(yLen/8));  cntBlock = 1;  cntChar=1;  for x = 1:xLen/8  for y = 1:yLen/8  D = dct2(hall\_gray(8\*x-7:8\*x,8\*y-7:8\*y));  D = round(D./QTAB);  zigzag(:,cntBlock) = ZigZag(D);  if(cntChar<=nChar)  for yy=1:8  if(zigzag(yy,cntBlock)<0)  temp=dec2bin(-zigzag(yy,cntBlock));  temp(end)=ascii(cntChar,yy);  zigzag(yy,cntBlock)=-bin2dec(temp);  else  temp=dec2bin(zigzag(yy,cntBlock));  temp(end)=ascii(cntChar,yy);  zigzag(yy,cntBlock)=bin2dec(temp);  end  end  cntChar=cntChar+1;  end  cntBlock=cntBlock+1;  end  end  cntBlock = cntBlock-1;  % DC  CD = zigzag(1,:);  CD(2:cntBlock)= CD(1:(cntBlock-1))-CD(2:cntBlock);  DC=[];  for i=1:cntBlock  ctg=Category(CD(i));  len=DCTAB(ctg+1,1);  % code=bitget(abs(CD(i)),max(ctg,1):-1:1);  code=fliplr(bitget(abs(CD(i)),1:max(ctg,1)));  if(CD(i)<0)  code=1-code;  end  DC=[DC,DCTAB(ctg+1,2:1+len),code];  end  %AC  AC=[];  EOB=[1,0,1,0];  ZRL=[1,1,1,1,1,1,1,1,0,0,1];  for i=1:cntBlock  cnt0=0;  for j=2:64  if(zigzag(j,i)==0)  if(j==64)  % AC=[AC,EOB];  continue;  end  cnt0=cnt0+1;  else %zigzag(j,i)~=0  %Run=cnt0, Size=Category, Amp=xxx  Size=Category(zigzag(j,i));  if(cnt0>15)  while(cnt0>15)  AC=[AC,ZRL];  cnt0=cnt0-16;  end  end  codeLen=ACTAB(10\*cnt0+Size,3);  % code=fliplr(bitget(abs(zigzag(j,i)),1:Size));  code=bitget(abs(zigzag(j,i)),Size:-1:1);  if(zigzag(j,i)<0)  code=1-code;  end  AC=[AC,ACTAB(10\*cnt0+Size,4:(3+codeLen)),code];  cnt0=0;  end  end  AC=[AC,EOB];  end  save('jpegcodes5.mat','DC','AC','xLen','yLen');  output = length(DC) + length(AC);  input = length(dec2bin(255)) \* xLen \* yLen;  compressRatio = input/output |

压缩比：

|  |
| --- |
| compressRatio =  6.3770 |

解码：

|  |
| --- |
| clc; clear all; close all;  load('hall.mat');  load('jpegcodes5.mat');  load('JpegCoeff.mat');  zigzag=zeros(64,xLen\*yLen/64);  % DC  head=1;  cntBlock=1;  lenDC=length(DC);  while(head<=lenDC)  for category=0:11  len=DCTAB(category+1,1);  if(DC(head:(head+len-1))==DCTAB(category+1,2:1+len))  % right category  head=head+len;  code=DC(head:head+max(category,1)-1);  head=head+max(category,1);  if(category==0)  zigzag(1,cntBlock)=0;  elseif(code(1)==0) % negative  code=1-code;  x=-bin2dec(num2str(code));  zigzag(1,cntBlock)=x;  else % positive  x=bin2dec(num2str(code));  zigzag(1,cntBlock)=x;  end  cntBlock=cntBlock+1;  break;  end  end  end  for cntBlock=2:(xLen\*yLen/64)  zigzag(1,cntBlock)=zigzag(1,cntBlock-1)-zigzag(1,cntBlock);  end  % AC  head=1;  headB=2;  cntBlock=1;  lenAC=length(AC);  EOB=[1,0,1,0];  ZRL=[1,1,1,1,1,1,1,1,0,0,1];  while(head<=lenAC)  if(AC(head:head+3)==EOB) % EOB  head=head+4;  cntBlock=cntBlock+1;  headB=2;  continue;  elseif(AC(head:head+10)==ZRL)  head=head+11;  headB=headB+16;  continue;  else  for RunSize=1:160  len=ACTAB(RunSize,3);  % ½â¾ö½áÎ²Ê±¶Ô±ÈHuffmanÂëÔ½½çÎÊÌâ  if(head+len-1>lenAC)  continue;  end  if(AC(head:(head+len-1))==ACTAB(RunSize,4:(4+len-1))) % right RunSize  head=head+len;  Run=ACTAB(RunSize,1);  headB=headB+Run;  Size=ACTAB(RunSize,2);  code=AC(head:(head+Size-1));  head=head+Size;  if(code(1)==0) % negative  code=1-code;  x=-bin2dec(num2str(code));  zigzag(headB,cntBlock)=x;  else % positive  x=bin2dec(num2str(code));  zigzag(headB,cntBlock)=x;  end  % for\_test=[headB,cntBlock]  headB=headB+1;  break;  end  end  continue;  end  end    STX='00000010'; % start of text  ETX='00000011'; % end of text  res='';  ascii='';  flagNotOver=true;    myHall=zeros(xLen,yLen);  cntBlock=1;  D=zeros(8,8);  for x = 1:xLen/8  for y = 1:yLen/8  if(flagNotOver)  for yy=1:8  if(zigzag(yy,cntBlock)<0)  b=dec2bin(-zigzag(yy,cntBlock));  else  b=dec2bin(zigzag(yy,cntBlock));  end  ascii=[ascii,b(end)];  end  res=[res;ascii];  if(ascii==ETX)  flagNotOver=false;  end  ascii='';  end    D=antiZigZag(zigzag(:,cntBlock));  D=D.\*QTAB;  myHall(8\*x-7:8\*x,8\*y-7:8\*y)=idct2(D);  cntBlock=cntBlock+1;  end  end  myHall=uint8(round(myHall+128));  dec=bin2dec(res);  hidden=char(dec)';  hidden=hidden(2:end-1)  %PSNR  MSE=sum(sum((double(hall\_gray)-double(myHall)).^2))/xLen/yLen;  PSNR=10\*log10(255\*255/MSE) |

提取结果及PSNR：

|  |
| --- |
| hidden =  XinHaoYuXiTongHenYouQv  PSNR =  31.1121 |

（3）方法三：DCT域隐藏信息，ASCII码为1将Zig-Zag块最后一个非零位后置1，最后一位非0则最后一位置1，为0则置-1。

编码：

|  |
| --- |
| % Hidden: "XinHaoYuXiTongHenYouQv"  clc; clear all; close all;  load('hall.mat');  [xLen,yLen] = size(hall\_gray);  STX='00000010'; % start of text  ETX='00000011'; % end of text  text='XinHaoYuXiTongHenYouQv';  ascii=dec2bin(text);  [nChar,len]=size(ascii);  if(len<8)  ascii=[char(ones(nChar,8-len).\*48),ascii];  end  ascii=[STX;ascii;ETX];  nChar=nChar+2;    % JPEG coding  load('JpegCoeff.mat');  hall\_gray = double(hall\_gray)-128;  % hall\_gray = hall\_gray-128;  [xLen,yLen] = size(hall\_gray); %xÐÐÊýyÁÐÊý  zigzag = zeros(64,(xLen/8)\*(yLen/8));  cntBlock = 1;  cntChar=1;  cnt8=1;  for x = 1:xLen/8  for y = 1:yLen/8  D = dct2(hall\_gray(8\*x-7:8\*x,8\*y-7:8\*y));  D = round(D./QTAB);  zigzag(:,cntBlock) = ZigZag(D);  if(cntChar<=nChar)  if(ascii(cntChar,cnt8)=='1')  temp=1;  else  temp=-1;  end  yy=find(zigzag(:,cntBlock),1,'last');  if(yy==64)  zigzag(64,cntBlock)=temp;  else  zigzag(yy+1,cntBlock)=temp;  end  if(cnt8==8)  cntChar=cntChar+1;  cnt8=1;  else  cnt8=cnt8+1;  end  end  cntBlock=cntBlock+1;  end  end  cntBlock = cntBlock-1;  % DC  CD = zigzag(1,:);  CD(2:cntBlock)= CD(1:(cntBlock-1))-CD(2:cntBlock);  DC=[];  for i=1:cntBlock  ctg=Category(CD(i));  len=DCTAB(ctg+1,1);  % code=bitget(abs(CD(i)),max(ctg,1):-1:1);  code=fliplr(bitget(abs(CD(i)),1:max(ctg,1)));  if(CD(i)<0)  code=1-code;  end  DC=[DC,DCTAB(ctg+1,2:1+len),code];  end  %AC  AC=[];  EOB=[1,0,1,0];  ZRL=[1,1,1,1,1,1,1,1,0,0,1];  for i=1:cntBlock  cnt0=0;  for j=2:64  if(zigzag(j,i)==0)  if(j==64)  % AC=[AC,EOB];  continue;  end  cnt0=cnt0+1;  else %zigzag(j,i)~=0  %Run=cnt0, Size=Category, Amp=xxx  Size=Category(zigzag(j,i));  if(cnt0>15)  while(cnt0>15)  AC=[AC,ZRL];  cnt0=cnt0-16;  end  end  codeLen=ACTAB(10\*cnt0+Size,3);  % code=fliplr(bitget(abs(zigzag(j,i)),1:Size));  code=bitget(abs(zigzag(j,i)),Size:-1:1);  if(zigzag(j,i)<0)  code=1-code;  end  AC=[AC,ACTAB(10\*cnt0+Size,4:(3+codeLen)),code];  cnt0=0;  end  end  AC=[AC,EOB];  end  save('jpegcodes6.mat','DC','AC','xLen','yLen');  output = length(DC) + length(AC);  input = length(dec2bin(255)) \* xLen \* yLen;  compressRatio = input/output |

压缩比：

|  |
| --- |
| compressRatio =  6.2750 |

解码：

|  |
| --- |
| clc; clear all; close all;  load('hall.mat');  load('jpegcodes6.mat');  load('JpegCoeff.mat');  zigzag=zeros(64,xLen\*yLen/64);  % DC  head=1;  cntBlock=1;  lenDC=length(DC);  while(head<=lenDC)  for category=0:11  len=DCTAB(category+1,1);  if(DC(head:(head+len-1))==DCTAB(category+1,2:1+len))  % right category  head=head+len;  code=DC(head:head+max(category,1)-1);  head=head+max(category,1);  if(category==0)  zigzag(1,cntBlock)=0;  elseif(code(1)==0) % negative  code=1-code;  x=-bin2dec(num2str(code));  zigzag(1,cntBlock)=x;  else % positive  x=bin2dec(num2str(code));  zigzag(1,cntBlock)=x;  end  cntBlock=cntBlock+1;  break;  end  end  end  for cntBlock=2:(xLen\*yLen/64)  zigzag(1,cntBlock)=zigzag(1,cntBlock-1)-zigzag(1,cntBlock);  end  % AC  head=1;  headB=2;  cntBlock=1;  lenAC=length(AC);  EOB=[1,0,1,0];  ZRL=[1,1,1,1,1,1,1,1,0,0,1];  while(head<=lenAC)  if(AC(head:head+3)==EOB) % EOB  head=head+4;  cntBlock=cntBlock+1;  headB=2;  continue;  elseif(AC(head:head+10)==ZRL)  head=head+11;  headB=headB+16;  continue;  else  for RunSize=1:160  len=ACTAB(RunSize,3);  % ½â¾ö½áÎ²Ê±¶Ô±ÈHuffmanÂëÔ½½çÎÊÌâ  if(head+len-1>lenAC)  continue;  end  if(AC(head:(head+len-1))==ACTAB(RunSize,4:(4+len-1))) % right RunSize  head=head+len;  Run=ACTAB(RunSize,1);  headB=headB+Run;  Size=ACTAB(RunSize,2);  code=AC(head:(head+Size-1));  head=head+Size;  if(code(1)==0) % negative  code=1-code;  x=-bin2dec(num2str(code));  zigzag(headB,cntBlock)=x;  else % positive  x=bin2dec(num2str(code));  zigzag(headB,cntBlock)=x;  end  % for\_test=[headB,cntBlock]  headB=headB+1;  break;  end  end  continue;  end  end    STX='00000010'; % start of text  ETX='00000011'; % end of text  cnt8=1;  res='';  ascii='';  flagNotOver=true;    myHall=zeros(xLen,yLen);  cntBlock=1;  D=zeros(8,8);  for x = 1:xLen/8  for y = 1:yLen/8  if(flagNotOver)  yy=find(zigzag(:,cntBlock),1,'last');  if(zigzag(yy,cntBlock)==1)  ascii=[ascii,'1'];  else  ascii=[ascii,'0'];  end  if(cnt8==8)  res=[res;ascii];  if(ascii==ETX)  flagNotOver=false;  end  cnt8=1;  ascii='';  else  cnt8=cnt8+1;  end  end  D=antiZigZag(zigzag(:,cntBlock));  D=D.\*QTAB;  myHall(8\*x-7:8\*x,8\*y-7:8\*y)=idct2(D);  cntBlock=cntBlock+1;  end  end  myHall=uint8(round(myHall+128));  dec=bin2dec(res);  hidden=char(dec)';  hidden=hidden(2:end-1)  %PSNR  MSE=sum(sum((double(hall\_gray)-double(myHall)).^2))/xLen/yLen;  PSNR=10\*log10(255\*255/MSE) |

结果及PSNR：

|  |
| --- |
| hidden =  XinHaoYuXiTongHenYouQv  PSNR =  29.9579 |

结果分析：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 无隐藏信息 | 方法一 | 方法二 | 方法三 |
| 压缩比 | 6.4188 | 6.3409 | 6.3770 | 6.2750 |
| PSNR | 31.1874 | 30.0970 | 31.1121 | 29.9579 |

隐蔽性：方法二最好，方法三其次，方法一最差，因为方法三可以随意约定每个分块中隐藏信息的比特数，且峰值信噪比最大，与原图最接近，不易被发现；而方法三末位通常非零即正负一，从数值上不明显，但峰值信噪比较大，较容易发现与原图的不同而暴露；方法一若隐藏信息较长，则前几个分块充满了非0数值，十分明显有隐藏信息。

压缩比：方法一没有充分利用熵编码压缩0位的优点，压缩比最小；方法三每比特信息对应分块中的一个码字，压缩比变化一般；方法二对码字影响最小，压缩效果最接近正常JPEG编码。

图片质量：方法一和方法三都会对Zig-Zag后位的数值造成改变，而后位数量化步长大，会对图片产生较大变化，方法一的变化更为严重（可能覆盖最大量化步长）且集中。方法二对大量化步长的DCT系数影响最小，图片质量最好。

3. 将第二题方法二中改为仅对每个分块Zig-Zag扫描结果的第六位进行信息隐藏，其量化步长最小即为10，并且可以按照约定的序列仅在固定某几个分块中进行信息隐藏，图片质量和隐蔽性更佳。

|  |
| --- |
| compressRatio =  6.3997  hidden =  XinHaoYuXiTongHenYouQv  PSNR =  31.1496 |

这种方法相比上述三种方法，压缩比和峰值信噪比更佳。

# 第四章 第三节

1.所给资料Faces目录下包含从网图中截取的28张人脸，试以其作为样本训练人脸标准v。

(a) 样本人脸大小不一，是否需要首先将图像调整为相同大小？

不需要，因为以肤色为特征时，特征为一概率密度，与图片大小无关。

(b) 假设 L 分别取 3, 4, 5 ，所得三个 v 之间有何关系？

L=3时，

L=4时

L=5时

设L=3, 4, 5时特征平均值分别为v3, v4, v5，则

V3(R, G, B) = V4(2R, 2G, 2B) + V4(2R, 2G, 2B+1) + V4(2R, 2G+1, 2B) + V4(2R, 2G+1, 2B+1)

+ V4(2R+1, 2G, 2B) + V4(2R+1, 2G, 2B+1) + V4(2R+1, 2G+1, 2B) + V4(2R+1, 2G+1, 2B+1)

V4(R, G, B) = V5(2R, 2G, 2B) + V5(2R, 2G, 2B+1) + V5(2R, 2G+1, 2B) + V5(2R, 2G+1, 2B+1)

+ V5(2R+1, 2G, 2B) + V5(2R+1, 2G, 2B+1) + V5(2R+1, 2G+1, 2B) + V5(2R+1, 2G+1, 2B+1)

计算标准特征v：（保存在’v.mat’中）

|  |
| --- |
| L=5; %L=4,3  shift=8-L;  v=zeros(1,2^(3\*L));  for cntFace=1:33  u=zeros(1,2^(3\*L));  filename=strcat(num2str(cntFace),'.bmp');  face=imread(filename);  [xLen,yLen,~]=size(face);  for x=1:xLen  for y=1:yLen  R=bitshift(face(x,y,1),-shift);  G=bitshift(face(x,y,2),-shift);  B=bitshift(face(x,y,3),-shift);  n=bitshift(R,2\*L)+bitshift(G,L)+B;  u(n+1)=u(n+1)+1; % n 0~255  end  end  u=u./(xLen\*yLen);  v=v+u;  end  v=v./33; |

2. 设计一种从任意大小的图片中检测任意多张人脸的算法并编程实现(输出图像在判定为人脸的位置加上红色的方框)。随意选取一张多人照片(比如支部活动或者足球比赛)，对程序进行测试。尝试L分别取不同的值，评价检测结果有何区别。

计算参考阈值：

|  |
| --- |
| load('v.mat');  L=5;  v=v5;  shift=8-L;  d=[];  for cntFace=1:33  u=zeros(1,2^(3\*L));  filename=strcat(num2str(cntFace),'.bmp');  face=imread(filename);  [xLen,yLen,~]=size(face);  for x=1:xLen  for y=1:yLen  R=bitshift(face(x,y,1),-shift);  G=bitshift(face(x,y,2),-shift);  B=bitshift(face(x,y,3),-shift);  n=bitshift(R,2\*L)+bitshift(G,L)+B;  u(n+1)=u(n+1)+1; % n 0~255  end  end  u=u./(xLen\*yLen);  d=[d,1-sqrt(u)\*(sqrt(v))'];  end  e=max(d); |

检测程序：

|  |
| --- |
| clc; clear; close all;  load('v.mat'); load('e.mat');    L=4; v=v4; e=e4;  block=30;    shift=8-L;  filename='test.bmp';  face=imread(filename);  [xLen,yLen,~]=size(face);  res=zeros(floor(xLen/block),floor(yLen/block));  for x=1:xLen/block  for y=1:yLen/block  u=zeros(1,2^(3\*L));  D1=face((x\*block-block+1):(x\*block),(y\*block-block+1):(y\*block),1);  D2=face((x\*block-block+1):(x\*block),(y\*block-block+1):(y\*block),2);  D3=face((x\*block-block+1):(x\*block),(y\*block-block+1):(y\*block),3);  for xx=1:block  for yy=1:block  R=bitshift(D1(xx,yy),-shift);  G=bitshift(D2(xx,yy),-shift);  B=bitshift(D3(xx,yy),-shift);  n=bitshift(R,2\*L)+bitshift(G,L)+B;  u(n+1)=u(n+1)+1; % n 0~255  end  end  u=u./(block\*block);  d=1-sqrt(u)\*(sqrt(v))';  if(d<=e)  res(x,y)=1;  end  end  end  for x=1:xLen/block  for y=1:yLen/block  if(res(x,y)==1)  if(res(x,y+1)==1 && res(x+1,y)==1)  cntX=1; cntY=1;  while(res(x+cntX+1,y)==1)  cntX=cntX+1;  end  while(res(x,y+cntY+1)==1)  cntY=cntY+1;  end  res(x:(x+cntX),y:(y+cntY))=0;  face(x\*block-block+1,(y\*block-block+1):((y+cntY)\*block),1)=255;  face(x\*block-block+1,(y\*block-block+1):((y+cntY)\*block),2)=0;  face(x\*block-block+1,(y\*block-block+1):((y+cntY)\*block),3)=0;  face((x+cntX)\*block,(y\*block-block+1):((y+cntY)\*block),1)=255;  face((x+cntX)\*block,(y\*block-block+1):((y+cntY)\*block),2)=0;  face((x+cntX)\*block,(y\*block-block+1):((y+cntY)\*block),3)=0;  face((x\*block-block+1):((x+cntX)\*block),y\*block-block+1,1)=255;  face((x\*block-block+1):((x+cntX)\*block),y\*block-block+1,2)=0;  face((x\*block-block+1):((x+cntX)\*block),y\*block-block+1,3)=0;  face((x\*block-block+1):((x+cntX)\*block),(y+cntY)\*block,1)=255;  face((x\*block-block+1):((x+cntX)\*block),(y+cntY)\*block,2)=0;  face((x\*block-block+1):((x+cntX)\*block),(y+cntY)\*block,3)=0;  end  end  end  end  imshow(face);  imwrite(face,'e4\_2.bmp','bmp'); |

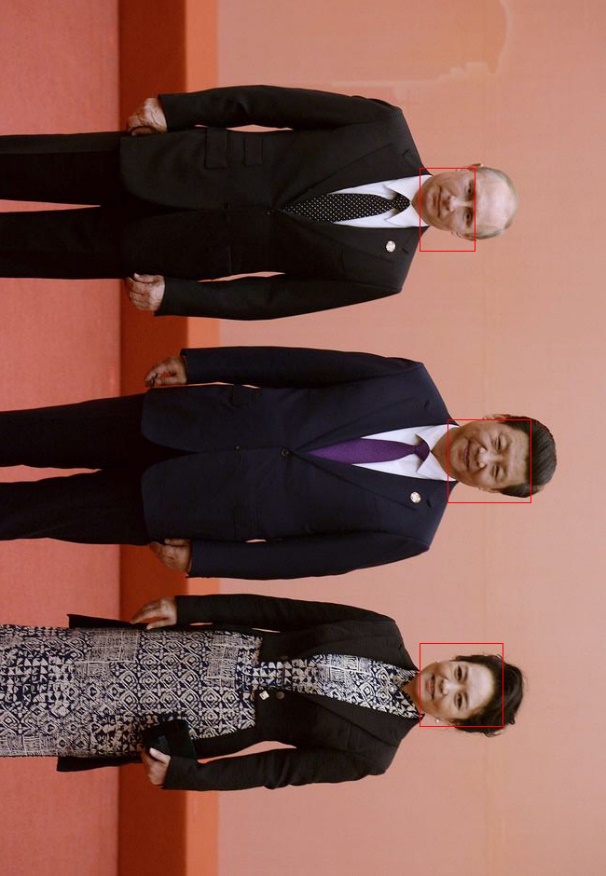
L分别取3、4、5得到：



可以看出，L值越小，每个三元组包含的颜色越多，于是对肤色识别就越包容即识别范围大；相反L越大对颜色越敏感，识别范围越小。

3. 对上述图像分别进行如下处理后

(a) 顺时针旋转90︒ (imrotate)；



(b) 保持高度不变，宽度拉伸为原来的2倍 (imresize) ；



(c) 适当改变颜色 (imadjust) ；



再试试你的算法检测结果如何？并分析所得结果。

旋转和拉伸对颜色分布没有本质改变，对基于肤色的人脸识别影响很小，但改变颜色后会对特征进行映射，难以再与标准特征匹配，故识别效果不好。

4. 如果可以重新选择人脸样本训练标准，你觉得应该如何选取？

我认为选择面部轮廓即面部器官轮廓作为标准会更好，这样光线和颜色接近物体对识别的影响小。