| QF | RMS | C |
|---|---|---|
| 1 | 5.4684 | 0.9588 |
| 5 | 2.7347 | 0.8819 |
| 10 | 1.9734 | 0.8382 |

Image7401

QF:1

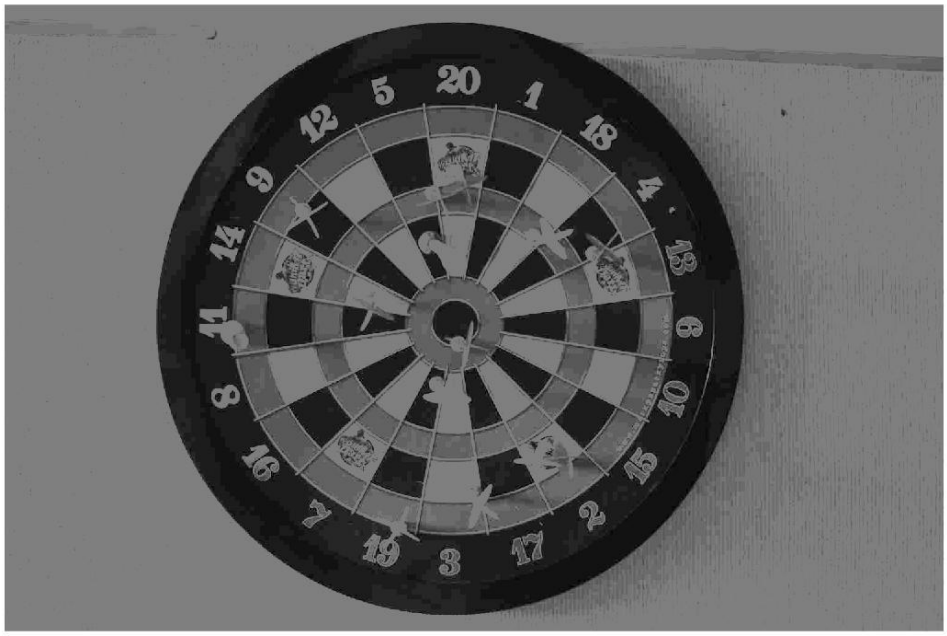
QF:5

QF:10

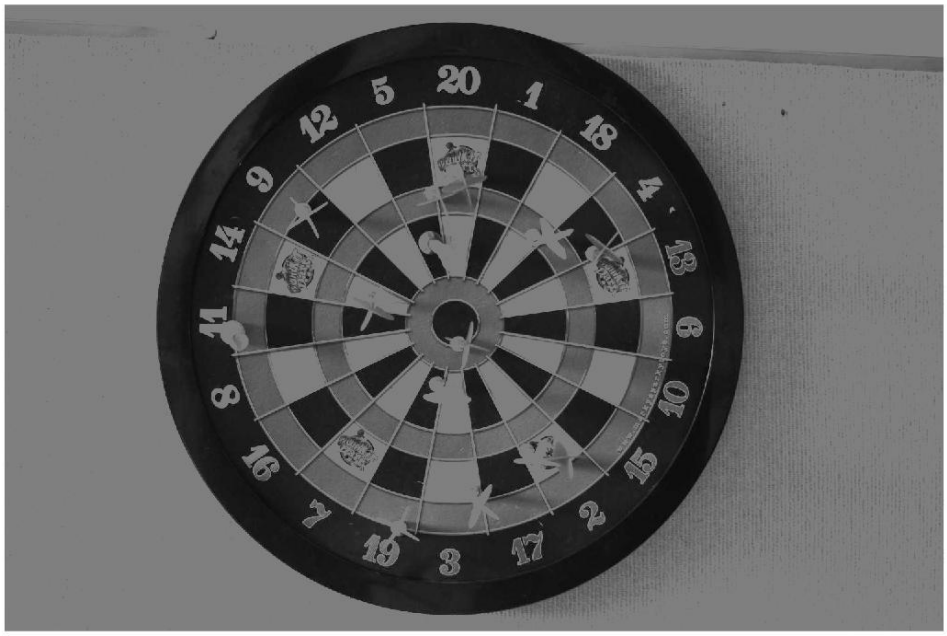
| QF | RMS | C |
|---|---|---|
| 1 | 4.1590 | 0.9476 |
| 5 | 2.1626 | 0.8435 |
| 10 | 1.5590 | 0.7869 |

Image:7405

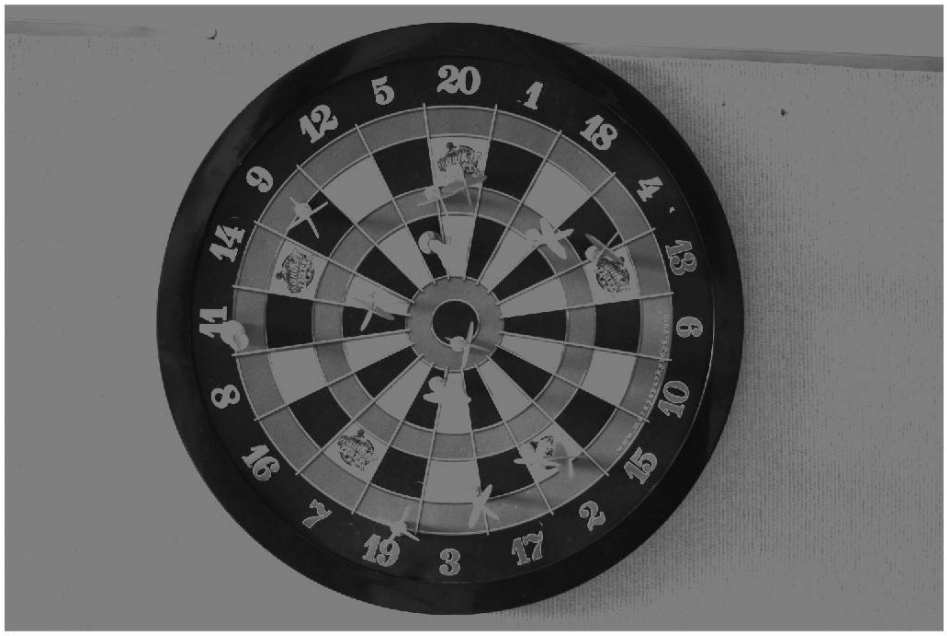
QF:1




QF:5



QF:10



Levelshift Function:

```
%level shift the matrix by 128 and then do dct2
tranformation
function [J] = levelshift(I)
I = I - 128;
J = dct2(I);
end
```

MyJpeg Function:

```
function [J,C,rms] = myJpeg(InputImage,QF)
if nargin == 1
  QF = 5;
%set default QF to 5 if no QF is specified
InputImage = double(InputImage);
[M,N] = size(InputImage);
I = InputImage;
%partition the image to 8x8 blocks and do subtracting 128
and computing the
%dct value, the blockproc function by default zero pads
the blocks if the
%block is not 8x8 dimension.step (a),(b),(c)
fun = @(block_struct)levelshift(block_struct.data);
B = blockproc(I,[8 8],fun);
%compute the q matrix with quality factor, and divide by
q-matrix with 8x8
%block processing, again by default blkproc function
zeropad the block that
%is not 8x8 dimension.step(d)
q_mtx =     [16 11 10 16 24 40 51 61;
          12 12 14 19 26 58 60 55;
          14 13 16 24 40 57 69 56;
          14 17 22 29 51 87 80 62;
          18 22 37 56 68 109 103 77;
          24 35 55 64 81 104 113 92;
          49 64 78 87 103 121 120 101;
          72 92 95 98 112 100 103 99];
q_mtx = q_mtx.*(5/QF);
quantize = @(x)x./q_mtx;
B2 = blkproc(B,[8 8],quantize);
B2 = round(B2);
% calculate the amount of zeros and then compute the
compressing
% rate. step(e)
C = 0;
for i = 1:M
   for j = 1:N
       if(B2(i,j)==0)
           C = C+1;
       end
    end
end
C = C/(M*N);
```

```
% inverse part,8x8 block processing multiplied by the q
matrix and then do idct2 tranformation, then add 128 back
step (f)
iquantize = @(x)(x).*q_mtx;
B3 = blkproc(B2,[8 8],iquantize);
ifun =
@(block_struct)(int8(idct2(block_struct.data)));
J = blockproc(B3,[8 8],ifun);
J = J+128;
J = uint8(J);
% compute the rms step(g)
sum = 0;
for i = 1:M
    for j = 1:N
      sum = sum
+double((J(i,j))-(InputImage(i,j)))*double((J(i,j))-(
InputImage(i,j)));
   end
end
rms = double(sqrt(1/(M*N)*double(sum)));
```

Script code example for image 7405 with QF = 10:

```
disp("load image");
I = load("IMG_7405.mat");
O_I=I.I;
QF = 10;
[J,C,rms] = myJpeg(O_I,QF);
figure(1);
imshow(O_I);
figure(2);
imshow(J);
disp("Cmpression rate is:")
disp(C);
disp("root mean square is:")
disp(rms);
```