Cairo University
Faculty of Engineering - EECE dept

DSP Applications
4th year – first semester

Assignment #2

1) A file consists of the following symbols with the following frequencies

Symbol	'a'	'b'	'c'	'd'	'e'
Frequency	0.1	0.3	0.1	0.05	.45

a) find the Huffman coding dictionary (show steps)

b) calculate the average symbol length

L = 0.1*3 + 0.3*2+0.1*4+0.05*4+1*0.45 = 1.95 bits/symbol

c) write a MATLAB/octave code to find the dictionary (use built-in functions)

```
huffmandict ([1:5], [0.1 0.3 0.1 0.05 .45 ])

ans = {
  [1,1] =

     0   0   0   0

  [1,2] =

     0   1

  [1,3] =

     0   0   1

  [1,4] =

     0   0   0   1

  [1,5] =  1
}
```

d) compare the average length of (c) and (a)

they are the same

e) define : prefix code – instantaneously decodable code. Is it necessary to be a prefix code to be instantaneously decodeable? (you can check online!)
https://en.wikipedia.org/wiki/Prefix_code
A **prefix code** is a type of code system (typically a variable-length code) distinguished by its possession of the "prefix property", which requires that there is no code word in the system that is a prefix (initial segment) of any other code word in the system

Instantaneously decodable code: a code that does NOT need a lookahead (waiting) before decoding the codeword. Check https://www.ics.uci.edu/~dan/pubs/DC-Sec1.html

Yes, it should be a prefix to be instantaneously decodable.

f) If a file consists of only two symbols: 'a' with prob 0.8 and 'b' with prob 0.2. What is the average symbol length?

1 bit per symbol

g) A technique that can be used to decrease the average length even more is to encode more than one symbols together, i.e. to encode vectors. Construct a new Huffman table using symbols (aa), (ab),(ba), (bb) and calculate the new average symbol length

New symbol	aa	ab	ba	bb
probability	0.8*0.8 = 0.64	0.8*0.2 = 0.16	0.2*0.8=0.16	0.2*0.2=0.04

2) The DCT output of the an 8x8 block in an image that is encoded by a JPEG encoder is

219.50	8.90	-2.58	1.76	2.50	2.91	1.61	-0.72
6.17	-4.01	-2.66	-0.82	-0.92	1.41	3.91	2.79
-4.61	-2.05	0.53	-0.26	2.34	-2.00	-2.84	1.11
0.67	2.09	0.25	0.84	-1.58	0.82	0.59	-0.90
-1.50	-2.61	-0.27	1.09	0.00	-2.69	0.65	-0.17
0.99	-1.45	-0.29	-0.73	0.13	1.89	-0.95	1.97
-4.59	-1.51	-0.34	0.70	1.74	-3.85	-0.53	0.62
-2.28	-6.27	-0.45	2.71	-1.70	-2.47	-0.16	1.78

a) It is required to encode the image with JPEG quality = 50. Assume that this block belongs to the Y component of the color space. Use the appropriate quantization table to quantize the image.

Use quantization table of slide 13 in the jpeg lecture

Table 4.1 (a) The default quantization table for luminance (Y component); (b) the default quantization table for chrominance (C_bC_r components)

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

and the equation of quantization is

$$F^Q(u,v) = IntegerRound\left(\frac{F(u,v)}{Q(u,v)}\right)$$

Hence, the quantized values are:

14	1	0	0	0	0	0	0
1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

b) Use ZigZag and run length encoding to encode the quantized block
{DC,1,1,EOB} ==> DC,(0,1),(0,1),EOB

c) Use the entropy coding tables to encode the AC components
The value +1 belongs to category (1) : slide 20. Category 1 represents (-1, +1)
-1 will be represented by '0', +1 will be represented by '1'
run 0 with category 1 is represented by a code : 00 : slide 21, first row after EOB
hence (0,1) is represented by 001
Hence, AC component si represented as 001/001/1010, the las 1010 is for EOB

d) assume the quantized DC component of the previous block = 12. Quantize the current DC components. Use the DC quantization table

current DC coefficient = 14
previous DC coefficient = 12
Difference = 14-12 = 2
From the Entropy coding – DC difference table, +2 belongs to category #2
From slide 18 category#2 is represented by a base code = 011
This category has the following elements -3,-2,2,3
we will represent them as -3:00, -2:01, 2:10, 3:11
hence, the DC component will be represented as **01110**

e) read this article about the story of inventing DCT and how you should work hard and believe that your ideas can make a difference :)
http://www.cse.iitd.ernet.in/~pkalra/siv864/assignment2/DCT-History.pdf
where was the algorithm first published?

3) It is required to calculate the DCT coefficients of the following vector $x = [+1\ -1\ -1\ +1]$.

a) write the forward and inverse DCT transform in case of $N=4$
b) write the basis functions of the DCT transform in case of N=4
c) find the DCT transform of vector x by observing the 4 basis functions
d) write a short MATLAB/octave code to calculate the coefficients. Compare your results with the

results in (c)

4) Consider the following video frames

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	**1**	**5**	**0**	**0**	0	0	0	0
0	0	**0**	**2**	**6**	**0**	0	0	0	0
0	0	**0**	**0**	**3**	**7**	0	0	0	0
0	0	**0**	**0**	**0**	**4**	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	**1**	**5**	**0**	**0**	0	0
0	0	0	0	**0**	**2**	**6**	**0**	0	0
0	0	0	0	**0**	**0**	**6**	**7**	0	0
0	0	0	0	**0**	**0**	**0**	**4**	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	**1**	**5**	**0**	**0**
0	0	0	0	0	0	**0**	**2**	**6**	**0**
0	0	0	0	0	0	**0**	**1**	**6**	**7**
0	0	0	0	0	0	**0**	**0**	**0**	**4**
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

--> Assume a simple image compression algorithm consisting of finding the motion vector, quantizing the motion compensated difference image values, and finally sending the quantized values. The decoder acts by decoding the (quantized) first frame, and the rest of frames are obtained from the decoded first frame, the motion vectors, and the difference images.

Consider only the bold block:

a) Assume a quantization step of (1) is used. Find the quantized first frame, the motion vectors and the motion compensated difference images for the 2nd and 3rd frames. Show the reconstructed blocks at the decoder

Solution

In case quantization step = 1, there is no difference between the image and its quantized version

I1q (quantized first frame) : first frame

–

M12 (motion vector from 1st to 2nd frame) = [2 pixels to the right, 2 pixels down]

Motion compensated image =

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	**1**	**5**	**0**	**0**	0
0	0	0	0	0	**0**	**2**	**6**	**0**	0
0	0	0	0	0	**0**	**0**	**3**	**7**	0
0	0	0	0	0	**0**	**0**	**0**	**4**	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

D12 (difference image between 2nd frame and motion compensated image) = frame2 – motion compensated frame frame 1

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	**0**	**0**	**0**	**0**	0
0	0	0	0	0	**0**	**0**	**0**	**0**	0
0	0	0	0	0	**0**	**0**	**3**	**0**	0
0	0	0	0	0	**0**	**0**	**0**	**0**	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

Decoded frame 2 = frame 2 (because there is no quantization noise)
–
M23 (motion vector from $2^{nd\ t}$ to 3^{rd} frame) = [2 pixels to the right, 1 pixels up]
Motion compensated image =

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	**1**	**5**	**0**	**0**
0	0	0	0	0	0	**0**	**2**	**6**	**0**
0	0	0	0	0	0	**0**	**0**	**6**	**7**
0	0	0	0	0	0	**0**	**0**	**0**	**4**
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

D23 (difference image between 3^{rd} frame and motion compensated image) = frame3 – motion compensated frame frame 2

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	**0**	**0**	**0**	**0**
0	0	0	0	0	0	**0**	**0**	**0**	**0**
0	0	0	0	0	0	**0**	**1**	**0**	**0**
0	0	0	0	0	0	**0**	**0**	**0**	**0**
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

Decoded frame 3 = frame 3 (because there is no quantization noise)
–

b) repeat (a) but using a step of (2)
In case quantization step = 2, the difference image, and hence the decoded images will be reconstructed
During quantization, we will assume that the rounding is to the closest step. If there is a tie (the value to be quantized is in the middle of quantization step, the value will be approximated to the upper level

I1q (quantized first frame) :

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	**2**	**6**	**0**	**0**	0	0	0	0
0	0	**0**	**2**	**6**	**0**	0	0	0	0
0	0	**0**	**0**	**4**	**8**	0	0	0	0
0	0	**0**	**0**	**0**	**4**	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

The highlighted points will be different than the pixel values at the original frame
The decoded first frame will equal to quantized first frame
–
M12 (motion vector from DECODED 1st frame to 2nd frame) = [2 pixels to the right, 2 pixels down]
D12 (difference image between 2nd frame and motion compensated DECODED first frame) = frame2 – motion compensated DECODED frame frame 1

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	**1**	**5**	**0**	**0**	0	0
0	0	0	0	**0**	**2**	**6**	**0**	0	0
0	0	0	0	**0**	**0**	**6**	**7**	0	0
0	0	0	0	**0**	**0**	**0**	**4**	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

−

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	**2**	**6**	**0**	**0**	0	0
0	0	0	0	**0**	**2**	**6**	**0**	0	0
0	0	0	0	**0**	**0**	**4**	**8**	0	0
0	0	0	0	**0**	**0**	**0**	**4**	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

=

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	**-1**	**-1**	**0**	**0**	0	0
0	0	0	0	**0**	**0**	**0**	**0**	0	0
0	0	0	0	**0**	**0**	**2**	**-1**	0	0
0	0	0	0	**0**	**0**	**0**	**0**	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

Quantized difference image (will be sent to decoder)

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	**0**	**0**	**0**	**0**	0	0
0	0	0	0	**0**	**0**	**0**	**0**	0	0
0	0	0	0	**0**	**0**	**2**	**0**	0	0
0	0	0	0	**0**	**0**	**0**	**0**	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

Decoded frame 2 = Quantized difference image + motion compensated frame 1 =

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	**2**	**6**	**0**	**0**	0	0
0	0	0	0	**0**	**2**	**6**	**0**	0	0
0	0	0	0	**0**	**0**	**6**	**8**	0	0
0	0	0	0	**0**	**0**	**0**	**4**	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

–

M23 (motion vector from 2nd t to 3rd frame) = [2 pixels to the right, 1 pixels up]
D23 (difference image between 3rd frame and motion compensated DECODED second frame) = frame3 – motion compensated DECODED frame frame 2

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	**1**	**5**	**0**	**0**
0	0	0	0	0	0	**0**	**2**	**6**	**0**
0	0	0	0	0	0	**0**	**1**	**6**	**7**
0	0	0	0	0	0	**0**	**0**	**0**	**4**
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

–

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	**2**	**6**	**0**	**0**
0	0	0	0	0	0	**0**	**2**	**6**	**0**
0	0	0	0	0	0	**0**	**0**	**6**	**8**
0	0	0	0	0	0	**0**	**0**	**0**	**4**
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

=

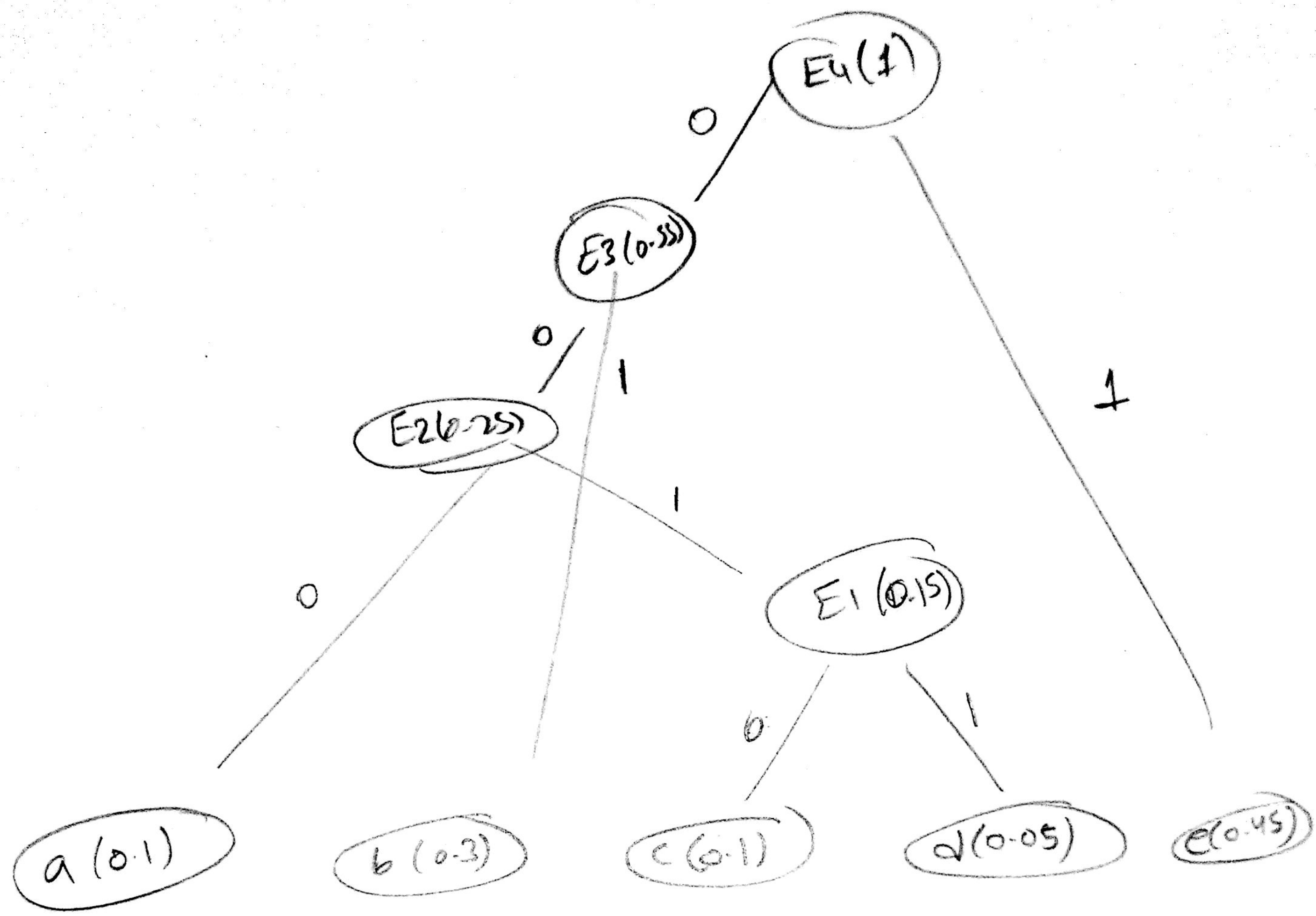
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	**-1**	**-1**	**0**	**0**
0	0	0	0	0	0	**0**	**0**	**0**	**0**
0	0	0	0	0	0	**0**	**1**	**0**	**-1**
0	0	0	0	0	0	**0**	**0**	**0**	**0**
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

Quantized difference image (will be sent to decoder)

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	**0**	**0**	**0**	**0**
0	0	0	0	0	0	**0**	**0**	**0**	**0**
0	0	0	0	0	0	**0**	**2**	**0**	**0**
0	0	0	0	0	0	**0**	**0**	**0**	**0**
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

Decoded frame 3 = Quantized difference image + motion compensated frame 2 =

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	**2**	**6**	**0**	**0**
0	0	0	0	0	0	**0**	**2**	**6**	**0**
0	0	0	0	0	0	**0**	**2**	**6**	**8**
0	0	0	0	0	0	**0**	**0**	**0**	**4**
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0



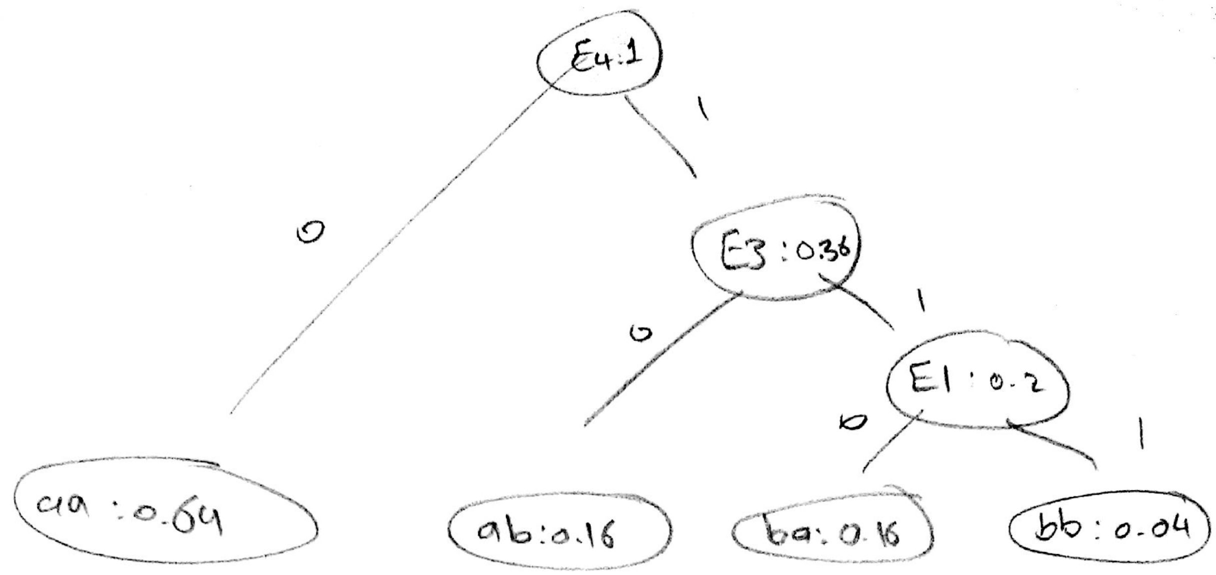
a : 0 0 0

b : 01

c : 0010

d : 0011

e : 1



aa : 0

ab : 10

ba : 110

bb : 111

average length = $1\times0.64 + 2\times0.16 + 3\times0.16 + 3\times0.04$

$= 1.56$ bits/ (Big symbol)

$= 0.78$ bits/symbol

a Big symbol consists of two small symbols "a, b"

Scanned by CamScanner

$\boxed{3}$ a) $C(u) = \alpha(u) \sum_{x=0}^{3} f(x) \cos\left(\frac{(2x+1)u\pi}{8}\right)$, $\alpha(u) = \begin{cases} \sqrt{\frac{1}{N}}, & u=0 \\ \sqrt{\frac{2}{N}}, & u \neq 0 \end{cases}$

$f(x) = \sum_{u=0}^{3} \alpha(u) C(u) \cos\left(\frac{(2x+1)u\pi}{8}\right)$

b) Basis functions : set one coefficient to (1) and rest to (0)

$u=0 \rightarrow f(x) = \alpha(0) \times C(0) \times 1 = \sqrt{\frac{1}{4}} \times 1 \times 1 = \frac{1}{2}$

$[\frac{1}{2} \quad \frac{1}{2} \quad \frac{1}{2} \quad \frac{1}{2}]$

$u=1 \rightarrow f(x) = \frac{1}{\sqrt{2}} \times \cos\left(\frac{(2x+1) \times 1 \times \pi}{8}\right)$, $x = 0, 1, 2, 3$

$= [0.653 \quad 0.27 \quad -0.27 \quad -0.653]$

$u=2 \rightarrow f(x) = \frac{1}{\sqrt{2}} \times \cos\left(\frac{(2x+1) \times 2 \times \pi}{8}\right)$, $x = 0, 1, 2, 3$

$= [\frac{1}{2} \quad -\frac{1}{2} \quad \frac{1}{2} \quad -\frac{1}{2}]$

$u=3 \rightarrow f(x) = \frac{1}{\sqrt{2}} \times \cos\left(\frac{(2x+1) \times 3 \times \pi}{8}\right)$

$= [0.27 \quad -0.653 \quad 0.653 \quad -0.27]$

c) $DCT\{x\} = [0, 0, 2, 0]$ because x is scaled version of the basis function at $u=2$

d) $C = dct([1 \quad -1 \quad 1 \quad -1])$