# ECE Department

**Information Theory** 

ECE457



IT Project Report presented by

عبدالرحمن علمي زكي الطوانسي معاذ سعيد حسن مرسي

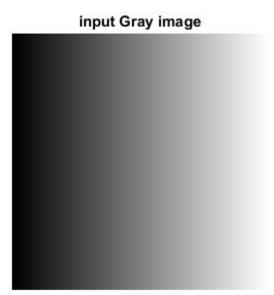
```
Import Image: 2
sorting and combing probs: ......4
Sort all tree elements and applying Huffman Tree:.....4
plot Huffman Tree: ......4
Assign 0 and 1 to symbols in Huffman Tree: ......5
the sizes of the input image and the output one from source coding: ......8
compression ratio of the coded image: .....8
The End .......8
clc;
clear all;
close all;
```

### **Import Image:**

```
%RGB = imread('finallydidit.png'); % for any random image
%imshow(RGB)

%RGB = rgb2gray(RGB);
%figure
%imshow(RGB)

RGB = imread('256_shades_of_grey.png'); % this image has equal probs
imshow(RGB)
title('input Gray image');
```



### Calculate probabilities of each unique symbol:

```
sym=unique(RGB);
sizergb(1:2)=size(RGB);
for s=1:length(sym)
    i=find(RGB==sym(s));
    prob_dist(s)=length(i)/(sizergb(1).*sizergb(2));
end
```

### **Testing Example:**

```
%sym =[1 2 3 4 5];
%prob_dist =[0.2 0.2 0.2 0.2 0.2];
%sym =1:length(prob_dist);
```

# **Sum all probrobilities to check that (Total probs = 1):**

```
total = sum(prob_dist);
display(total)

total =
    1
```

```
for i = 1:length(sym)
    sorted_sym{i} = sym(i);
end
init_sym = sorted_sym;
init_prob = prob_dist;
```

### sorting and combing probs:

```
sorted_prob = prob_dist;
count = 1;
while (length(sorted_prob) > 1)
   % Sort probs
    [sorted_prob,indeces] = sort(sorted_prob, 'ascend');
   % Sort symbol based on indeces
    sorted_sym = sorted_sym(indeces);
   % Create new symbol
    new_node = strcat(sorted_sym(2),sorted_sym(1));
    new_prob = sum(sorted_prob(1:2));
   % Dequeue used symbols from "old" queue
    sorted_sym = sorted_sym(3:length(sorted_sym));
    sorted_prob = sorted_prob(3:length(sorted_prob));
   % Add new symbol back to "old" queue
    sorted_sym = [sorted_sym, new_node];
    sorted_prob = [sorted_prob, new_prob];
   % Add new symbol to "new" queue
    newq_sym(count) = new_node;
    newq_prob(count) = new_prob;
    count = count + 1;
end
```

### Sort all tree elements and applying Huffman Tree:

```
tree = [newq_sym,init_sym];
tree_prob = [newq_prob, init_prob];
[sorted_tree_prob,indeces] = sort(tree_prob,'descend');
sorted_tree = tree(indeces);
parent(1) = 0;
for i = 2:length(sorted_tree)
   % Extract symbol
   me = sorted_tree{i};
   % Find parent's symbol (search until shortest match is found)
    parent_maybe = sorted_tree{i-count};
    different = strfind(parent_maybe,me);
    while (isempty(different))
        count = count + 1;
        parent_maybe = sorted_tree{i-count};
        different = strfind(parent_maybe,me);
    end
    parent(i) = i - count;
end
```

# plot Huffman Tree:

```
figure
treeplot(parent);
%title(strcat('Huffman Coding Tree - "',sym,'"'));
[xs,ys,h,s] = treelayout(parent);
text(xs,ys,sorted_tree);
```

```
69 89 103 121 131 153 167 185 197 217 231 249
           39
                           102 120 130 152 166 184 196 216 230 248
                           101 119 129 151 165 183 195 215 229 247
           37
                   67 87
       22 36 54 66 86 100 118 128 150 164 182 194 214 228 246
0.9
     15213135475363657985959944117272743495963758191920212222232455
     1420303446526264788494981101626264248586274809092081222228384254
8.0
     131929334554616377839397109152525414757617375899208122222324253
     1218283244506062768292961081424244046568072788892021222223242
0.7
     111727314349596175819195107182323394555597177878202021222384251
     1016263042485860748090941061222233445454587076868802081222324250
0.6
    79/$228299447556597398699306$11222362455589767558$8229227322223292
    68142232880658628723888920240020883628668346290922222222624
0.5
    57133222379555655777857910009972583544555667383395296522212232E873
     461222286844536563075846200008$68492449805666728828498222222223
0.4
    0.2
0.1
               36 48 68 82 100 110 132 146 164 176 196 210 228
           18
 0
                           81<sub>4</sub> 99 0.109 131<sub>5</sub> 145 163 175 195 209 227 1
80 98 108 130 144 162 174 194 208 226
                   47<sub>0.3</sub>67
           17
                          80, 98
           16
           15 33 45 65 79 heighto7 829 143 161 173 193 207 225
```

### Assign 0 and 1 to symbols in Huffman Tree:

```
for i = 2:length(sorted_tree)
    % Get my coordinate
    my_x = xs(i);
    my_y = ys(i);
    % Get parent coordinate
    parent_x = xs(parent(i));
    parent_y = ys(parent(i));
    % Calculate weight coordinate (midpoint)
    mid_x = (my_x + parent_x)/2;
    mid_y = (my_y + parent_y)/2;
    % Calculate weight (positive slope = 0, negative = 1)
    slope = (parent_y - my_y)/(parent_x - my_x);
    if (slope < 0)</pre>
        weight(i) = 1; %assign 1
        weight(i) = 0; %assign 0
    text(mid_x,mid_y,num2str(weight(i)));
end
```

```
69 89 103 121 131 153 167 185 197 217 231 249
             39
                              102 120 130 152 166 184 196 216 230 248
                              101 119 129 151 165 183 195 215 229 247
            37 55 67 87
        22 36 54 66 86 100 118 128 150 164 182 194 214 228 246
0.9
      15213135475363657985959911117272743495963758191920712222739455
      14203034465262647884949811016262642485862748090920812222838425
0.8
      1319293345516163778393971091525254147576173798993081222223242
      1218283244506062768292961081424244046568072788892021222223842
0.7
      1117273143495961758191951071823233945585971778782020212223842
      10,16263042485860749090941061222273844545870768682020212223242
0.6
     7945528299475565972798799306511227367455556677586629992732222
0.5
       1332223795556579778899100000725335449555673833652995222222
0.4
          a data a alata a alata
           0.2
        jaminakan kangana kangan k
                         69 83 101 111 133 147 165 177 197 211
           18 36 48 68 82 100 110 132 146 164 176 196 210 228
 0
                              81<sub>4</sub> 99 0.509 131<sub>5</sub> 145 063 175 195 209 227 1
80 98 108 130 144 162 174 194 208 226
        3<sub>0.1</sub> 17
                    47<sub>0.3</sub>67
                             80, 98
             16
             15 33 45 65 79 heightor 929 143 161 173 193 207 225
```

#### **Extract all codewords from Huffman Tree:**

# **Extract the codewords for the probs:**

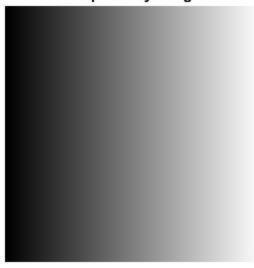
### Mapping the image (convert pixels to bits):

```
for c=1:length(sym)
    for cc=1:sizergb(1)
        for ccc=1:sizergb(2)
        if (sym_codeword{c,1}==RGB(cc,ccc))
            rgb_encoded{cc,ccc}=sym_codeword{c,2};
        end
        end
    end
end
```

# Decoding the encoded image (convert bits to pixels)

```
for c=1:length(sym)
    for cc=1:sizergb(1)
        for ccc=1:sizergb(2)
            if (length(sym_codeword{c,2}) == length(rgb_encoded{cc,ccc}))
                if (sym_codeword{c,2} == rgb_encoded{cc,ccc})
                rgb_decoded{cc,ccc}=sym_codeword{c,1};
                end
            end
        end
    end
end
RGB_decoded=cell2mat(rgb_decoded);
imwrite(RGB_decoded, 'test.png')
Output_RGB = imread('test.png');
imshow(Output_RGB)
title('Output Gray image');
```

#### **Output Gray image**



# the sizes of the input image and the output one from source coding:

# compression ratio of the coded image:

```
compression_ratio=(size_of_encoded_image)./(size_of_image);
display(compression_ratio)
```

```
compression_ratio =
1
```

#### The End

Published with MATLAB® R2016a