- Pre-course Materials
- Topic 1: Course Overview
- **Topic 2: Lossless Source Coding: Hamming Codes**
  - **2.1 Source Coding** Week 1 Quiz due Nov 02, 2015 at 15:30 UTC
  - **2.2 Sequence of Yes/No Questions** Week 1 Quiz due Nov 02, 2015 at 15:30 UTC
  - **2.3 Entropy of a Bit** Week 1 Quiz due Nov 02, 2015 at 15:30 UTC
  - **2.4 Entropy of a Discrete Random Variable** Week 1 Quiz due Nov 02, 2015 at 15:30 UTC
  - **2.5 Average Code Length** Week 1 Quiz due Nov 02, 2015 at 15:30 UTC
  - **2.6 Huffman Code** Week 1 Quiz due Nov 02, 2015 at 15:30 UTC
  - **2.7 Lab 1 - Source Coding** Lab due Nov 02, 2015 at 15:30 UTC
- MATLAB download and tutorials
- MATLAB Sandbox

LAB 1 - TASK 4 (3 points possible)

In this task, you create separate dictionaries for black and white runs based upon their run-length distributions using the Huffman code algorithm.

```
58
59 % recreate 500 by 500 image from the length code
60 img_new = runlength_decode(new_length_code);
61 figure(4);
62 imshow(img_new);
63 title('Recreated image');
64
65 % Put figure  1, 2 and 3 on top by calling them again
66 figure(3);
67 figure(2);
68 figure(1);
69
70 % compare the lengths
71 size_raw_data = length(lorem_img(:))
72 size_huffman = length(huffman)
73
```

Unanswered

```
% % % % Revise the following code  % % % %

% separate the black and white runs
len_runs = length(runs);
white_runs = runs(1:2:len_runs);
black_runs = runs(2:2:len_runs);

% compute the distributions and show the probabilities
figure(2)
hist = histogram(white_runs,rlen_list,'Normalization','probability','Visible','o
ff');
white_prob = hist.Values
hist = histogram(black_runs,rlen_list,'Normalization','probability','Visible','o
ff');
black_prob = hist.Values

%show the distributions using the function bar
bar([0:10],white_prob); xlabel('white runs'); ylabel('Probability');
set(gca,'XTickLabel',{'0','1','2','3','4','5','6','7','8','9','>=10'});
title('Histogram of the white runs from 0 to 9 and >= 10');

figure(3); bar([0:10],black_prob); xlabel('black runs'); ylabel('Probability');
set(gca,'XTickLabel',{'0','1','2','3','4','5','6','7','8','9','>=10'});
title('Histogram of the black runs from 0 to 9 and >= 10');

% Use the information in white_prob and black_prob to create
% separate dictionaries for the white and black run lengths.
white_dict = {[0,0,1,1,1,1,1];[0,0,1,1,0];[0,1];[0,0,0];...
    [1,1];[0,0,1,0];[1,0,1,0];[1,0,1,1];...
    [0,0,1,1,1,0];[0,0,1,1,1,1,0];[1,0,0]};
black_dict = {[1,0,0,0,1];0;[1,1];[1,0,1];...
    [1,0,0,1];[1,0,0,0,0,0];[1,0,0,0,0,1,1,0,1,1];[1,0,0,0,0,1,0];...
    [1,0,0,0,0,1,1,1];[1,0,0,0,0,1,1,0,0];[1,0,0,0,0,1,1,0,1,0]};

% % % % Do not change the code below % % % %
```

Figure 1

Figure 2

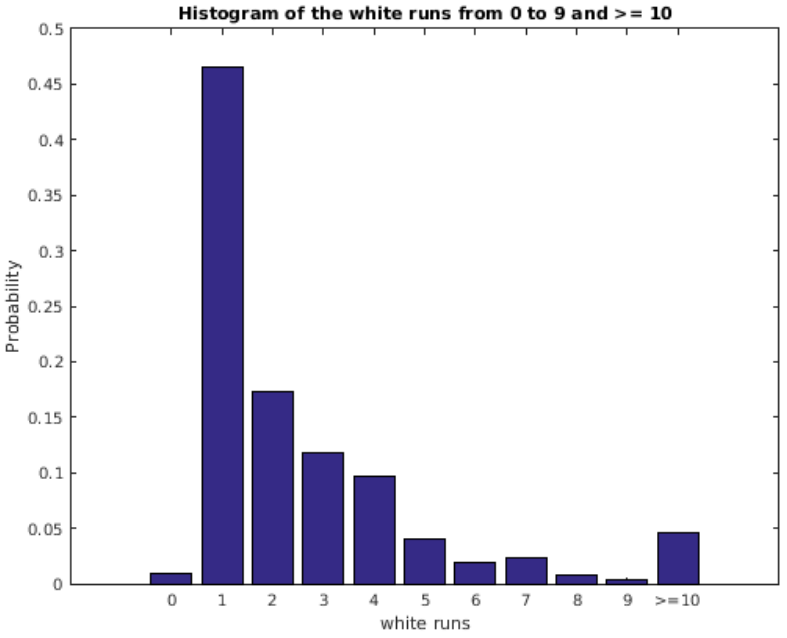

Figure 3

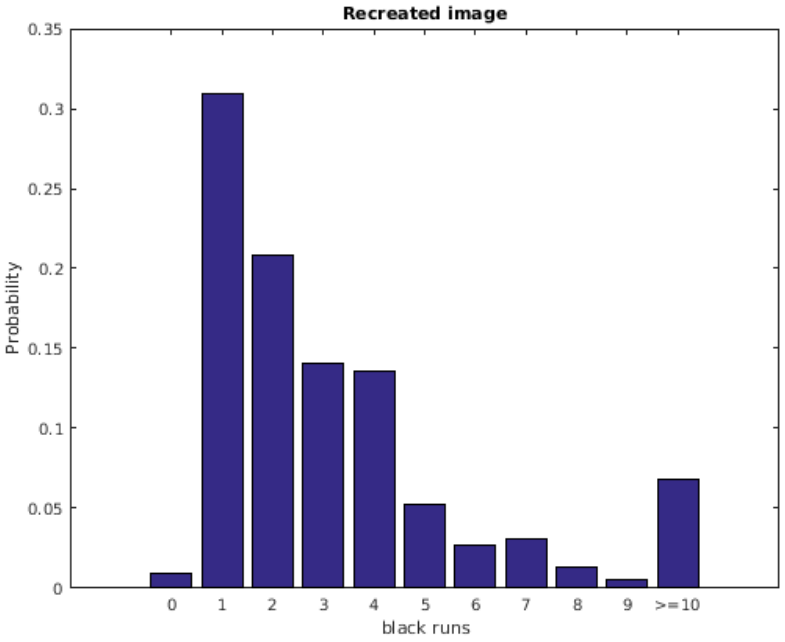

Figure 4

white_prob =

0.0092 0.4648 0.1735 0.1178 0.0963 0.0401 0.0191 0.0239 0.0072 0.0029 0

black_prob =

```
    0.0092    0.3093    0.2088    0.1410    0.1362    0.0520    0.0273    0.0309    0.0127    0.0050    0
```

[Warning: runlength_decode: Invalid Run Length]

size_raw_data =

```
    250000
```

size_huffman =

```
    128893
```

Run Code

You have used 0 of 10 submissions

INSTRUCTIONS

The initial MATLAB code in the above window has the same function as the initial code in Task 3. In this task, your goal will be to find two separate dictionaries for encoding the run lengths for black and white pixels.

The key difference between the code here and that of Task 3 is that we have created two new variables below the comment "% separate the black and white runs": **white_runs** and **black_runs**. These two variables should contain the run lengths corresponding to *only* the white pixels or *only* the black pixels. However, in the initial code, these are erroneously both set equal to the vector **runs**, which contain the run lengths for *both* white and black pixels.

The following code runs the same code as in Task 3, but twice: once for the **white_runs** and once for the **black_runs**. In other words, it returns histograms of the run length distributions both graphically in Figures 2 and 3, and as variables **white_prob** and **black_prob**. If you run the initial code, these are identical because of the error above, but they should not be.

The code then defines two dictionaries, **white_dict** and **black_dict**, for encoding the run lengths of white pixels and black pixels respectively. These are both initialized so that the 11 symbols are encoded using the standard 4 bit binary representation of the numbers from 0 to 10. However, these dictionaries are not optimal.

The final part of the code uses these dictionaries to encode the run lengths, measure the length of the resulting bit stream, and to check whether the dictionaries are valid by reconstructing the image from the run lengths encoded by the dictionaries and showing the reconstructed image in Figure 4.

Your task is to modify the code between the lines

% % % % Revise the following code % % % %

and

% % % % Do not change the code below % % % %

There are two modifications you should make.

First, you should correct the definitions of **white_runs** and **black_runs**. If you do this correctly, the two histograms and vectors of probabilities will show the distribution of runs for the white and black pixels separately.

Second, you should use these two distributions and the Huffman algorithm to find two different dictionaries that optimally encode run lengths of white and black pixels, and replace the initial dictionary definitions.

If you do these correctly, then when you run the code, the variable **size_huffman** should be the same as you found in Task 1 for two separate dictionaries.

Do not change the other parts of the code.