UNIVERSITY OF UTAH
ELECTRICAL AND COMPUTER ENGINEERING DEPARTMENT

ECE1020 **COMPUTING ASSIGNMENT 8**
N. E. COTTER **MATLAB® FUNCTION FILES: COMMUNICATION SYSTEM**

READING
Matlab® Student Version: learning Matlab 6, Ch 6-18 to 6-22
Mastering Matlab® 6, Ch 11

TOPICS
Function files

OVERVIEW
Now that we have completed all the pieces of our communication system, we will improve the structure of our code. In this assignment, we will use function files for major sections of the communication system. We can use these functions as building blocks in other communication systems later on. The advantage of using functions is that we avoid writing code more than once, either in one large program or in different projects. Using functions also makes our code easier to read.

PROCEDURE
In this assignment, you will use function files to improve the structure of the code for your communication system.

+5 pts Command File for calculations
Using a text editor program on your PC, create a command file (a type of function file) called **comm_func.m** containing matlab commands to perform the calculations in this assignment.

+5 pts Start by copying your code from Computing Assignment 7 into comm_func.m. Then write code for the functions described below, and use calls to these functions to replace sections of code from Computing Assignment 7. Where the calls to these functions should be used in comm_func.m should be apparent from the descriptions of the functions.

+5 pts Do **not** use semicolons at the ends of commands in your files.

+5 pts So it can be accessed after comm_func is run, declare as global the variable COMM_STR_RECV containing the final result calculated in comm_func.m. Note that when you run comm_func, you must also type the following in the command window in order to access COMM_STR_RECV:
>>global COMM_STR_RECV

+5 pts Place function definitions at the bottom of comm_func.m.

+10 pts Write code for the function "str2comm_num" in the following line of code:

```
command_num = str2comm_num(command_str)
```

The following comments define and describe the function. (The comments should be placed immediately after the first line of code defining the function.)

```
function comm_num_array = str2comm_num(comm_str_array)
% Convert from string commands for rover to numbers.
% Input:
%   command_str_array  (One string per row)
%      Allowed strings and returned numbers:
%       '0' = 0
%       '1' = 1
%    'left' = 2
%   'right' = 3
% Output:
%   comm_num_array  (Horizontal array of returned #'s in range 0-3.)
```

Note that the variable names in the function definition are slightly different than the names used when calling the function. In particular, they have the "_array" appended to them. These variables in the function definition are a bit like variables in algebra. They are place holders. When calling the function, the user substitutes the names of the actual variable they want to pass into the function and the variable they want to use for the returned answer.

+15 pts Write code for the function in the following line of code:

```
command_code = comm_num2code(codewords, command_num)
```

The following comments define and describe the function. (The comments should be placed immediately after the first line of code defining the function.)

```
function coded_bits = comm_num2code(codewords_matrix, comm_num_array)
% Lookup the codewords for the numbers in command_num_array.
% Inputs:
%   codewords_matrix  (One binary codeword on each row.
%     Used to translate comm_num_array entries into transmitted
%     codewords.  First codeword corresponds to
%     comm_num_array entry = 0.
%   comm_num_array  (Horizontal array of numbers, each in range 0-3.)
%     Command strings and corresponding command numbers:
%       '0' = 0
%       '1' = 1
%    'left' = 2
%   'right' = 3
% Output:
%  coded_bits  (Horizontal array of 0's and 1's for codewords for
%    numbers in comm_num_array
```

+15 pts Write code for the function in the following line of code:

```
command_num_recv = code2comm_num(codewords, command_recv)
```

The following comments define and describe the function. (The comments should be placed immediately after the first line of code defining the function.)

```
function comm_num_array = code2comm_num(codewords_matrix, coded_bits)
% Process codewords into numbers.
% Inputs:
%   codewords_matrix  (One binary codeword on each row.  Used to
%     translate coded_bits (codewords) into codeword numbers 0-3.
%     First codeword corresponds to a comm_num_array entry equal to 0,
```

```
%     and etc.
%   coded_bits  (Horizontal array of 0's and 1's (codewords) to be
%     translated into the numbers in comm_num_array)
% Output:
%   comm_num_array  (Horizontal array of command #'s corresponding to
%     codewords)

% Step through the received bits one codeword (6 bits) at a time.
```

+10 pts Write code for the function in the following line of code:

```
COMM_STR_RECV = comm_num2str(command_num_recv)
```

The following comments define and describe the function. (The comments should be placed immediately after the first line of code defining the function.)

```
function comm_str_array = comm_num2str(comm_num_array)
% Convert from command numbers to string commands for rover.
% Input:
%   comm_num_array  (Horizontal array of command #'s, in range 0-3.)
% Output:
%   comm_str_array  (One string per row)
%      Allowed numbers and returned strings:
%       '0' = 0
%       '1' = 1
%    'left' = 2
%   'right' = 3
```

+20 pts Function calls (5 pts per function)

Use the function calls described above to replace corresponding steps in comm_func.m (and what was previously code from Computing Assignment 7, control_flow.m).

+5 pts Run Script File

Run your script file by typing the name of the file without the .m

>> comm_func

Use a diary file to capture the output of comm_func.

Verify that each function works properly and that the final result of comm_func matches the final result you would obtain from control_flow.m written for Computing Assignment 7.

Note: If you make any changes in your **comm_func.m** file, be sure to run the following Matlab command to insure that Matlab reads your file again the next time you run it:

>> clear all

E-mail your file (comm_func.m) and your diary file to your TA, (as two separate e-mails). In the Subject line of your e-mail, be sure to put Your Name, "ECE1020 Comp8," and the file name, (e.g. comm_func.m). Also, print out the files and hand them in to the TA or to the ECE1020 locker.