BEng Biomedical Engineering
Computer Programming

Coursework 3 - Take-Home Assignment

Objective

To gain practical experience of computer programming and design using MATLAB.

Introduction

Image-guidance of interventional procedures is increasingly used in clinical practice. For some of these procedures, real-time monitoring of organ motion is paramount for the successful completion of the procedure.

Researchers have designed a novel imaging tool for non-invasive monitoring of organ motion. This tool provides a 1-dimensional signal which shows a sharp signal transition at the interface between organs. Because the imaging probe is fixed during the intervention, the location of this signal transition will depend on the real-time organ location.

In the current application, researchers are interested in monitoring the respiratory motion of the diaphragm which is at the interface between the liver and the lungs. An example of a 1-dimensional signal acquired at two different phases of the breathing cycle (end-inspiration and end-expiration) is shown in Figure 1. The liver/lung interface can be clearly visualised and its position is spatially shifted during breathing. At end-expiration, the interface is further away from the patient's head (i.e. a shift towards the left in Figure 1) whilst at end-inspiration it is further away from the patient's feet (i.e. a shift towards the right).

Instructions

This imaging device was tested in one healthy volunteer breathing normally. Data were acquired with a frequency of 20 Hz for 10 seconds resulting in a total of 200 successive 1-dimensional signals. Each 1-dimensional signal has a spatial resolution of 1 mm and covers a field of view of 100mm. Therefore, each 1-dimensional signal is comprised of 100 values. These data are stored in the file *data1D.csv*. The file has 200 rows of data where each row represents a different 1-dimensional signal measurement. Each row contains 100 comma separated numbers representing a 1-dimensional signal.

You are required to write code to perform the following operations:

1. Read the data: First ask the user for the name of the file. If the file does not exist, the program should flag an error and stop. Otherwise, the program should read in the entire list of 1-dimensional signals from the file and store them in a variable.

2. Calculate the shift (i.e. breathing motion) between the first 1-dimensional signal (which we call the *reference* signal) and all subsequent ones. To calculate the shift between any 1-dimensional signal and the reference signal, you should perform an exhaustive search of all possible shifts (from -10 to 10 in steps of 1) and return the shift that results in the best match between the reference signal and the shifted signal. The best match will be defined as the

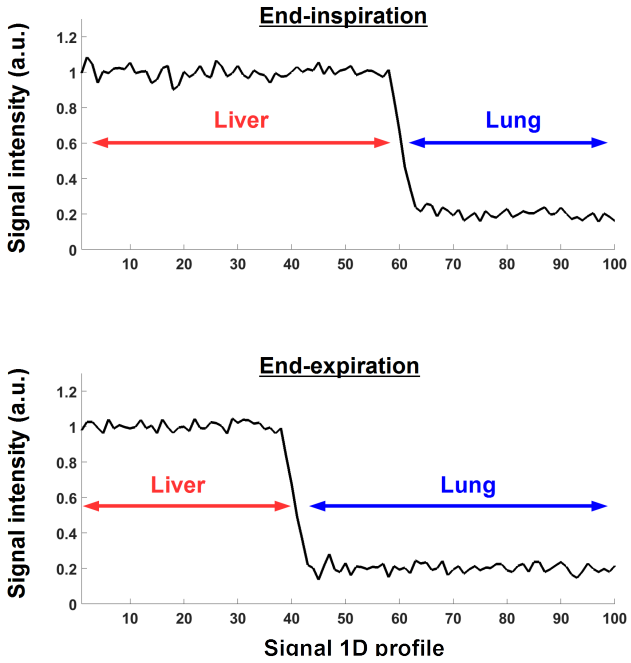

Figure 1: 1-dimensional signal acquired at two different phases of the breathing cycle (end-inspiration and end-expiration).

one with the smallest mean squared error (MSE). The MSE between two signals ($s1$ and $s2$) is defined as follows:

$$MSE = \frac{1}{max - min + 1} \sum_{n=min}^{max} (s1(n) - s2(n))^2 \quad (1)$$

where min and max represent an area of interest in the 1-dimensional signal which contains the liver/lung interface. In our case, min and max should be set to 20 and 80. This enables the beginning and end of the signal (which may contain signal from other organs) to be ignored.
Hint: Read the documentation of the built-in MATLAB function `circshift` *which can be used to apply a known shift to a 1-dimensional signal.*

3. Determine the average shift during the end-expiration position. This should be calculated as the average shift of the 20% most extreme positions towards the feet of the patient.

4. Calculate the number of measurements that were performed during the end-expiration phase, defined as a shift within ±5mm of the end-expiration average shift. We refer to this range as the *end-expiration window*.

5. Save all the results into a text file which has the following format: 200 rows corresponding to each of the 200 measurements. Each row should contain 2 numbers: 1) the calculated shift and 2) a Boolean indicating whether the measurement was acquired within the end-expiration window (=1) or outside (=0). These two numbers should be separated by a semi-colon.

6. Plot the evolution of the shift as a function of time (during the 10s of acquisition). A blue square should be added to the plot for all measurements performed in the end-expiration window. The window should also be displayed using green dotted lines. The plot should be suitably annotated.

7. Display the following information: 1) the average shift during the end-expiration window, and 2) the number of measurements acquired within the end-expiration window.

Expected output

The expected output plot for your program when run on the provided data file is shown in Fig. 2.

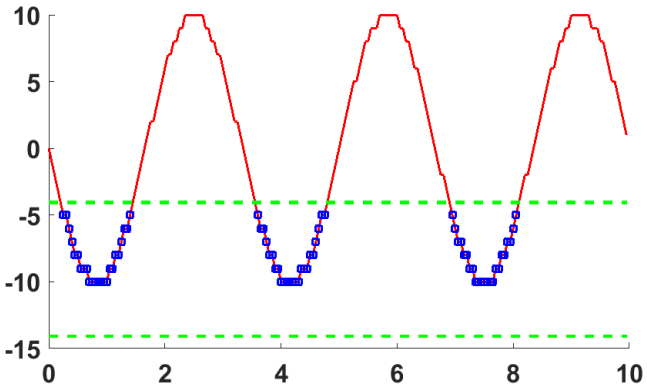

Figure 2: Expected output of the program when run on the provided data file.

The expected output at the command window for your program when run on the provided data file is as follows:

```
Average shift in end-experiatory position: -9.060000
percentage of measurements within gating window: 35.500000%
```

The expected contents of the output text file will be provided to you as part of the coursework details.

Note that you should write your program in such a way that it will work for any input file with the same format as the *data1D.csv* file and with any number of lines. Your code will be tested against a different file from the one provided.

General advice

Before starting any coding, you should produce a structured design for your program, using structure charts and/or pseudocode. You should then apply an incremental development approach to develop your code.

There are many ways that this exercise can be tackled and a good approach is to break the tasks that you need to do down so that they can be implemented in multiple functions that are called by a single main function or script.

Reporting Requirements

You should submit MATLAB files that meet as many of the requirements as possible. Try to use variable/function naming, comments and indentation to make your program as easy to understand as possible.

You should also submit a short written report detailing the design of your program. This report should contain structure charts and/or pseudocode for your program design, together with a short explanation of why you chose to design the program in this way. To help you in producing this report, a sample model report will be made available to you through the KEATS system. The report should not need to be longer than 3 pages of A4, and can be shorter.

Submission will be via the KEATS system. The submission point will only allow you to upload a single file so if you have multiple files you should combine all files into a single *zip* file. Name your file *cw3-YEAR-SURNAME-FIRSTNAME.zip*, replacing *YEAR*, *SURNAME* and *FIRSTNAME* with your personal details (e.g. *cw3-2017-ROUJOL-SEBASTIEN.zip*).

The hand-in date is **29 Nov 2017, 5 pm**. Late submissions (within 24 hours of this deadline) will be accepted but will be capped at the module pass mark (i.e. 40%).

If your program does not meet all requirements then please submit what you have written by the deadline.

Assessment

Your coursework will be marked on a number of factors:

- Does the program work? Does it meet all requirements? (60%)
- Program design, i.e. structure charts and/or pseudocode (20%)
- Appropriate use of MATLAB language features, e.g. control structures, functions, etc. (10%)
- Use of comments, indentation and variable/function names to make code easy to understand (10%)

The overall mark for this coursework will make up 10% of your total mark for this module.

This is an individual assignment. You are not permitted to work together with any other student. Note that general discussions about design decisions and/or coding strategies are permitted, and such discussions can be a useful learning experience for you. But you should not, under any circumstances, share details of designs or code.

If you have any questions about this coursework please contact Sebastien Roujol (sebastien.roujol@kcl.ac.uk).