L314
MPHIL IN ADVANCED COMPUTER SCIENCE

Monday 29 November 2021 16:00 to Tuesday 18 January 2021 16:00

Module L314 – Digital Signal Processing – Assignment 4

This assignment is a project topic. You have a ***choice of two topics****. Prepare a project report on* ***one of either*** *part (a) or part (b).*

Your project report should not exceed 4000 words (excluding tables, figure labels, equations, source code listings and appendices) and 12 A4 pages length and should be submitted as a PDF file. It should include examples of source code written, along with any software output required to demonstrate the effectiveness of the algorithms implemented. You may include a URL to a repository of the full source code you wrote to produce each figure or table.

Submit your work via

`https://www.vle.cam.ac.uk/course/view.php?id=206611`

no later than 16:00 on Tuesday 18 January 2021*.*

Students will be required to sign an undertaking that work submitted will be entirely their own; no collaboration is permitted*.*

4 (*a*) HDMI video cable eavesdropping

At

`https://www.cl.cam.ac.uk/teaching/2021/L314/assignment4a/`

you will find a collection of recordings made with a device that uses IQ down-conversion to capture a 40 MHz wide part of the radio spectrum, with a sampling frequency of 64 MHz. The data format of the stored complex-valued IQ samples in the `*.dat` files there is the same little-endian IEEE float32 IQIQIQ... format as in the FM-radio recordings previously provided in Assignment 3(*b*).

The receiving antenna used to make these recordings was located in the vicinity of a *Raspberry Pi* computer connected with an HDMI cable to an LCD video monitor. Your task is to reconstruct from these IQ recordings the static text displayed by this computer.

The HDMI video interface (similar to earlier DVI and VGA interfaces) transmits a video image as a serialized pixel matrix over the cable, one pixel at a time, using a simple and highly predictable timing pattern. The transmission of the image starts with the top-left pixel, and then proceeds pixel-by-pixel from left to right, until the top-most line is completed. This is then followed by the next line below, as in English reading order, until the last line is transmitted. After this, the transmission starts again with the next image frame at to top-left corner. In addition, after the end of each visible line of image pixels, the computer transmits a fixed number of additional pixels, during a time period known as the *horizontal blanking interval*. These do not contain any image data, but special sync pixel-codes there mark the start of a new line (and in the case of HDMI it may also encode audio data). Likewise, after the bottom-most visible pixel line, during a time period known as the *vertical blanking interval*, a fixed number of additional lines is transmitted to mark the start of a new image frame (and give the display electronics some time to process the data received so far).

The symbol rate at which pixels are transmitted over the cable is known as the pixel clock frequency f_p. Our target computer was configured to use a standard-resolution video timing in which a 640×480 pixel large image frame is transmitted 60 times per second (a video-timing mode known as 640×480@60Hz). The VESA DMT standard specifies that this mode has a pixel-clock frequency of $f_\mathrm{p} = 25.175\ \mathrm{MHz} \pm 0.5\%$. The size of the displayed image matrix in this mode is $x_\mathrm{d} \times y_\mathrm{d} = 640 \times 480$, but when we also include the pixels that make up the horizontal and vertical blanking time periods, the total number of pixels transmitted per frame period is $x_\mathrm{t} \times y_\mathrm{t} = 800 \times 525$ pixels. The line rate $f_\mathrm{h} = f_\mathrm{p}/x_\mathrm{t}$ is the number of lines transmitted per second, i.e. f_h^{-1} is the time that passes between the transmission of one pixel and the one directly below it. The frame rate $f_\mathrm{v} = f_\mathrm{h}/y_\mathrm{t}$ is the number of complete image frames transmitted per second, in our case about 60 Hz. In a static image, the signal on the video cable repeats exactly after a period of f_v^{-1}.

In this project, conduct a number of experiments towards understanding the nature of the radio emissions coming from an HDMI cable, improving the display quality of reconstructions of the displayed information, and automating the estimation of important parameters for that reconstruction. In your project report, demonstrate what you have achieved and learned. The following steps offer some ideas and starting points:

- Start with the data file `scene3-640x480-60-425M-64M-40M.dat` recorded at a centre frequency of $f_\mathrm{c} = 425$ MHz. (You don't have to load all ≈ 60 video frames recorded in that file, a shorter prefix covering just a couple of frames will also get you started.)
- Before you can convert the IQ data into a raster image, you first need to resample it to ensure you have an integer number of samples per line. Use interpolation to change the sampling frequency of the IQ data from $f_\mathrm{s} = 64$ MHz to either f_p or some integer multiple $m \cdot f_\mathrm{p}$ ($m \in \mathbb{N}$). You can experiment with different interpolation techniques (nearest neighbour, linear, windowed sinc, Lanczos, etc.). Once you have the IQ data in a new sampling rate f_r, where each image line is now represented by an integer number $f_\mathrm{r}/f_\mathrm{h}$ of samples, you can now convert the IQ data into a 2D matrix $f_\mathrm{r}/f_\mathrm{h}$ pixels wide, by filling it line-by-line with samples. To display it as a greyscale image, perform amplitude demodulation, i.e. take the absolute value of each complex-valued resampled IQ sample. To preserve the aspect ratio of the displayed information, display the resulting image with pixels that have a height:width ratio of m:1 (MATLAB: `daspect`).
- The image may initially be sheared sideways, which means that your initial guess of f_p or f_h was not accurate. Adjust it manually, until vertical image lines appear vertical. You should now be able to read some text. Try other centre frequencies to find one that results in good contrast between the foreground and background colour of text regions.
- The resulting image is likely not yet aligned correctly. Discard initial samples until the top-left corner of the displayed pixel area roughly coincides with the top-left corner of your reconstructed image matrix. This should leave the blanking intervals at the right and bottom end of the resulting image matrix.
- To adjust f_p further, generate an image where the top half is taken from the first frame and the bottom half is taken from a couple of frames later. Eliminate the jump in vertical lines between these two halves.
- Can you automate these manual adjustments? Try the auto-correlation function of the IQ data. Does it have useful peaks at f_v^{-1} or f_h^{-1}? Why? Does it make a difference in which order you perform the auto-correlation and taking the absolute value? Why? Can you increase the accuracy further by taking a cross-correlation between e.g. the first and the last frame in the recording? You can simulate a larger eavesdropping distance,

either by adding some complex Gaussian white noise to the IQ file data, or by mixing the recorded signal with a noise recording prepared with the target device switched off.

- When your f_v estimate seems accurate enough to maintain horizontal alignment across the entire recording, try to reduce the interference from other radio sources by averaging multiple or all of the frames in the recording. Does this improve the image quality?
- Instead of amplitude demodulation (which discards phase information), try to display the real part (with an offset) in the red channel of the output image, and the imaginary part of your resampled IQ data in the green channel. Alternatively, display the polar coordinates of the complex numbers as brightness and hue (HSV colour coordinates). What do you observe?
- Try to "unrotate" any periodic phase change that you see by multiplying the IQ data with a complex phasor of frequency f_u.
- Plot a spectrogram of your IQ data. You may see a spectral peak near integer multiples of f_p, the emissions of the pixel-clock wire pair in the HDMI cable. Adjust the frequency f_u of your unrotation phasor such that you move one of the $k \cdot f_\mathrm{p} - f_\mathrm{c} - f_\mathrm{u}$ peaks to exactly 0 Hz in the resulting IQ signal.

  Once you have succeeded, the local oscillator of the combined IQ downconversion (from both the major frequency shift $-f_\mathrm{c}$ performed in the IQ recording device and the minor frequency shift $-f_\mathrm{u}$ performed by your unrotating phasor) should now be in the same phase at the start of every pixel. How can you verify this in your colour representation of IQ phase (cartesian red/green or polar HSV)?
- If that unrotation frequency shift is accurate enough to cause pixels to maintain their phase even across multiple frames, you can now try coherent periodic averaging: average the complex numbers, before performing any AM or HSV demodulation for displaying.
- Can you automate the unrotation of the IQ signal, e.g. by FM demodulating the output of a bandpass filter roughly centered on $k \cdot f_\mathrm{p} - f_\mathrm{c}$, to measure the remaining frequency difference with high precision?
- What happens if you try periodic averaging of a complex-valued signal without prior phase unrotation? Why? Compare the numeric RMS amplitudes of the resulting means.
- Also look at the recordings available from other centre frequencies f_c. How do they look differently?
- HDMI represents the 8-bit red, green and blue value of each pixel as a 10-bit data word (using a line encoding known as *Transition Minimized*

Differential Signalling or TMDS). The red, green and blue values of a pixel are transmitted simultaneously, over three wire pairs, at bit rate $10 \times f_\mathrm{p}$. A fourth wire pair carries a pixel-clock signal, which is a square wave of frequency f_p (what's the Fourier transform of that?). The added emissions of the three read-green-blue 10-bit waveforms causes a characteristic radio emission for each pixel colour, and that can be demodulated into some brightness or colour (not directly related to the original brightness). Due to the sampled nature of a raster image, the information about the pixel image content will repeat every f_p throughout the radio spectrum, and can be received at frequencies that are not otherwise occupied by stronger transmitters. The result of demodulating this information will look differently at various multiples of f_p, but should look the same at multiples of $10 \times f_\mathrm{p}$. Why?

- The recording device used can capture only 40 MHz bandwidth at a time, however at least 125 MHz bandwidth would be required to reconstruct a bit pattern transmitted at about 250 Mbaud. Since the recorded signal is periodic, could a series of overlapping 40 MHz recordings be "stitched" together, into an IQ recording of a larger bandwidth? How could resampling, frequency shifting, cross-correlation, and Linkwitz–Riley filters be used to achieve this?

(*b*) **Identification of JPEG recompression history**

Read the articles

- Zhigang Fan, R.L. de Queiroz: Identification of bitmap compression history: JPEG detection and quantizer estimation. IEEE Transactions on Image Processing 12(2), February 2003, DOI: 10.1109/TIP.2002.807361
- R. Neelamani, et al.: JPEG compression history estimation for color images. IEEE Transactions on Image Processing 15(6), June 2006, DOI: 10.1109/TIP.2005.864171
- Andrew B. Lewis, Markus G. Kuhn: Exact JPEG recompression. IS&T/SPIE Electronic Imaging, January 2010, Proc. SPIE 7543, 75430V. DOI: 10.1117/12.838878

Choose and implement a technique (e.g. one described in these references) to determine if a provided uncompressed PNG image

- originated most likely directly from a digital camera (through RAW conversion),
- has been once JPEG compressed and decompressed before, or
- has been more than once been JPEG compressed and decompressed before.

Start with a black-and-white image, apply the DCT in a way that inverts what a JPEG decoder does, and look at histograms of the resulting coefficients.

Optionally extend the technique to colour images. Apply the colour transform in a way that inverts what a JPEG decoder does. Try to determine some of the compression parameters that have been used in the history of the analysed image.

At

`https://www.cl.cam.ac.uk/teaching/2021/L314/assignment4b/`

there are some example images that have been JPEG compressed and decompressed 0–2 times before.

END OF PAPER