Imaging and Data Processing Task Sheet

Chamod Kalupahana Mesthrige

November 18, 2022

1 Simple image processing

1.1 Greyscale enhancement via point processing:

Firstly, an image was captured by a camera shown by Figure 1(a) where the buildings in the image are too dark to see due to the brightness of the sky in the rest of the image.

Then the original image was converted to a greyscale image using the Equ. 1

$$G = 0.333F_R + 0.500F_G + 0.166F_B \tag{1}$$

where F_R, F_G, F_B are the respective brightness of the R, G and B colour bands [4]. Each RGB pixel intensities were replaced with G intensity. Then the greyscale values were normalised such that 0 corresponds to black and 1 corresponds to white.

Then the brightness of pixels were mapped logarithmically using Equ. 2

$$G' = \frac{\ln(1 + \alpha G)}{\ln(1 + \alpha)} \tag{2}$$

(include linear scaling equation, figure of logarithm against different alphas)

where α is a constant. The approach is known as non-linear scaling because the increase of brightness of dark pixels is much bigger than the increase of brightness for bright pixels. This ensures that the only the dark parts of the image become visible while keeping the bright parts of the image at approximately the same brightness.

Each pixel of the greyscale underwent logarithmic mapping and the resulting brightness enhanced image is shown in Figure 1(c).

The value of $\alpha = 20$ was found by inspecting the brightened image for varying values of α. The value of $\alpha < 20$ meant that the dark parts of the building were not visible enough and the value of $\alpha > 20$ meant that the details of the sky were lost.

Looking at Figure 1, we can see that the building is not visible in the original image is visible in the brightness enhanced image. This shows that non-linear scaling of pixel brightness can recover lost information in a otherwise poorly lit image.

The histogram for original and enhanced image are shown in 2.

Figure 1: Image (a) showing the original image, Image (b) showing the greyscale image and Image (c) showing the brightened greyscale image with $\alpha = 20$

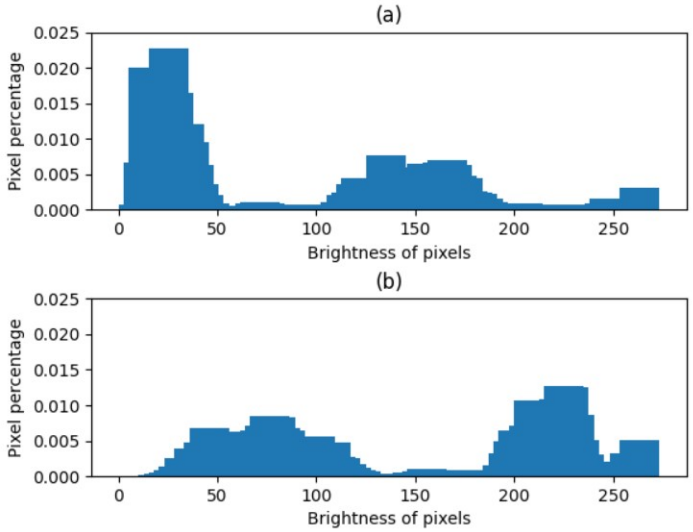


Figure 2: Plot (a) showing the histogram of the original image and Plot (b) showing the histogram of the brightened image

Looking at Figure 2, the percentage of pixels at the 250 brightness peak remains at 0.005 for both histograms whereas we can see that the peak of dark pixels at < 50 brightness has scaled across higher brightness values.

(include figure of images for different alpha values)

1.2 Image thresholding

Similarly to greyscale enhancement, an image was captured by a camera, in this case, the same image and converted to a greyscale image using the same method as before.

Then the threshold equation was applied to each pixel of the image given by Equ. 3

$$G = \begin{cases} 1, & \text{if } x > k. \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

where k is the brightness threshold. The results of thresholding the original image are given by Figure 3.

Figure 3: Image (a) showing the original image, Image (b) showing the greyscale image and Image (c) showing the thresholding image

To threshold to separate the sky from the buildings, a threshold (k) of 60 brightness was used. This is because most of the dark pixels peak at $<$ 55 as seen in Figure 2(a). Therefore, thresholding at 60 would ensure all the bright pixels which correspond to the sky are set to 1.

For a more accurate thresholding of the image, local thresholding could be applied to the image where different blocks of the image are thresholded with different k values.

1.3 Edge detection

Similarly, an image was captured by a camera and converted to a greyscale image using the same method as before.

To detect edges of objects within an image, the Prewitt operator must be applied to the image. This operator is shown in Figure 4.

Kx =

-1	0	1
-1	0	1
-1	0	1

Ky =

1	1	1
0	0	0
-1	-1	-1

Figure 4: Prewitt kernels where Kx is the hoiztonal operator and Ky is the veritcal operator [5]

These 3x3 kernels are applied to each 3x3 block of pixels in the original original images of the lake. This measures the gradient of going from y-1 pixel to y+1 pixel, for the vertical operator, and sets the gradient as the brightness of the resulting image.

Both kernels are applied to the greyscale image and then summed together to produce a image of all the edges within the image. The resulting image for edge detection is given by Figure 5.

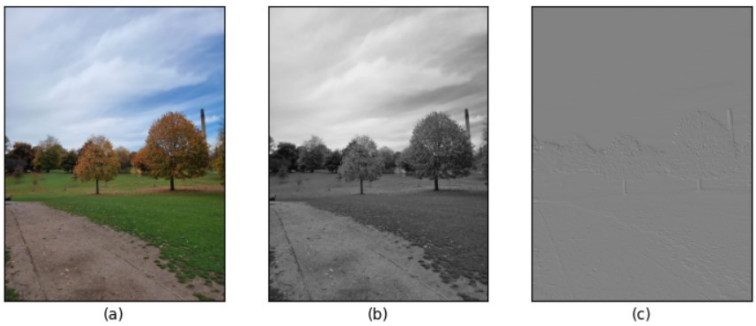


Figure 5: Image (a) showing the original image of trees by the University Park Lake, Image (b) showing the greyscale image and Image (c) showing the edges of the greyscale image

The edges of the tress against the sky are distinct as seen in Figure 5(a) which seen in Figure 5(c). This shows that the Prewett operators show the gradient of moving from one pixel to another in both the vertical and horizontal direction.

2 Affine transformations

Consider data acquired from a 3D motion tracking system. This data includes translation data, rotational data, head coordinates of a healthy control subject. This data also includes room coordinates of the room that the subject was in.

Firstly, the data was processed and the room and head of the control subject was recreated on a 3D axis. This is shown by Figure 6.

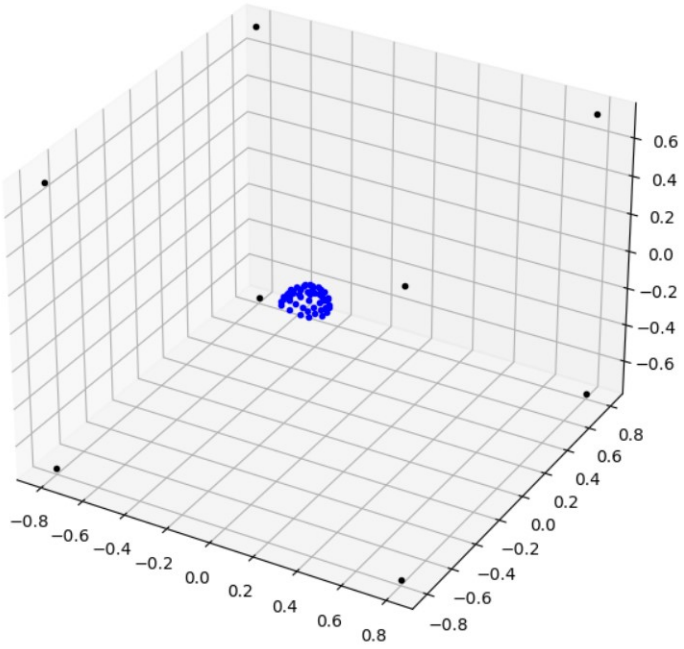


Figure 6: 3D plot of the control subject room. The blue points represent the tracking points on the head of the control subject. The black points represent the corners of the room and provide a reference frame for when the control subject moves

The control subject completed some basic movements which the 3D tracking system recorded 7200 data points in terms of x,y and z translation data and θ, ϕ and α rotational data, where θ gives the rotation about the x axis, ϕ gives the rotation about the y axis and α gives the rotation about the z axis.

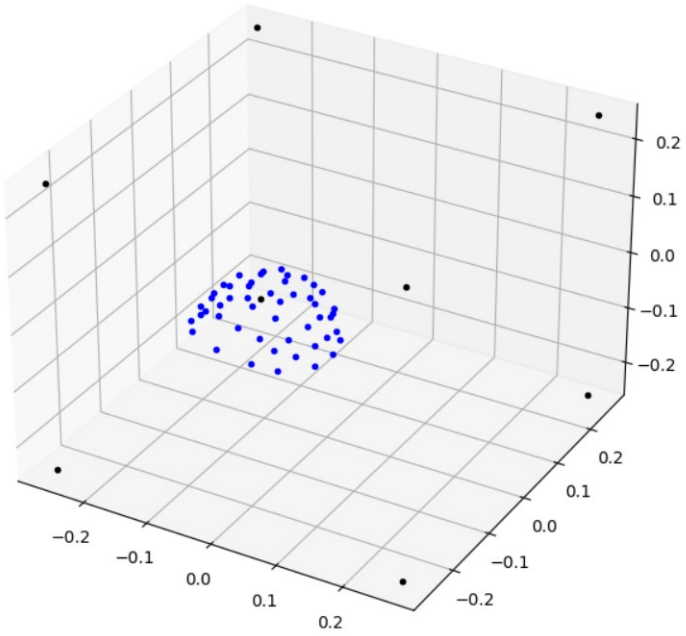


Figure 7: Close of the plot in Figure 6 where the room corner coordinates have been divided by 3

To recreate the movements of the control subject, we must apply affine transformations to the coordinates of the head. Firstly, we apply the rotational affine transformation to the coordinates. We do this by applying the equations in given in Figure 8.

Then we apply the translation matrices to the rotated coordinates given by Figure 9.

The rotation matrices are applied first to the coordinates because the translation matrices and the rotation matrices do not commute. This means that applying the rotation matrices once the translation matrices have been applied will result in different coordinates than if the matrices were applied the other way round.

This is because the rotational matrices rotate the coordinates about the origin and the data captured from the motion tracking included data about the origin. However, for every frame after the initial frame, the head coordinates will not centered about the origin anymore. Therefore, the head

$$\mathbf{R}_x = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{pmatrix},$$
$$\mathbf{R}_y = \begin{pmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{pmatrix},$$
$$\mathbf{R}_z = \begin{pmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

Figure 8: 3D rotational matrices where the θ in each coordinate matrix corresponds to the axis of rotation [3]

$$\mathbf{x}_i' = \mathbf{R}\mathbf{x}_i + \mathbf{t}$$

Figure 9: Translation equation where x_i is each coordinate in the translation matrix, R_{x_i} is the rotation matrix applied to x_i and t is the translation of the coordinate for the given axis

coordinates must be centered about the origin before every rotation.

This was done by calculating the centre of mass of the head coordinates of the control subject which includes summing all the head coordinates. We can do this because we assume that each head coordinate has equal weighting. Then centre of mass coordinates were subtracted from each head coordinate. This ensured that the head coordinates returned to the origin before the rotation matrices were applied. Then the centre of mass coordinates were added back onto the new rotated head coordinates and then finally the translation matrices can be applied to the rotated head coordinates.

The result of this subtracting the centre of mass is shown by Figure 10.

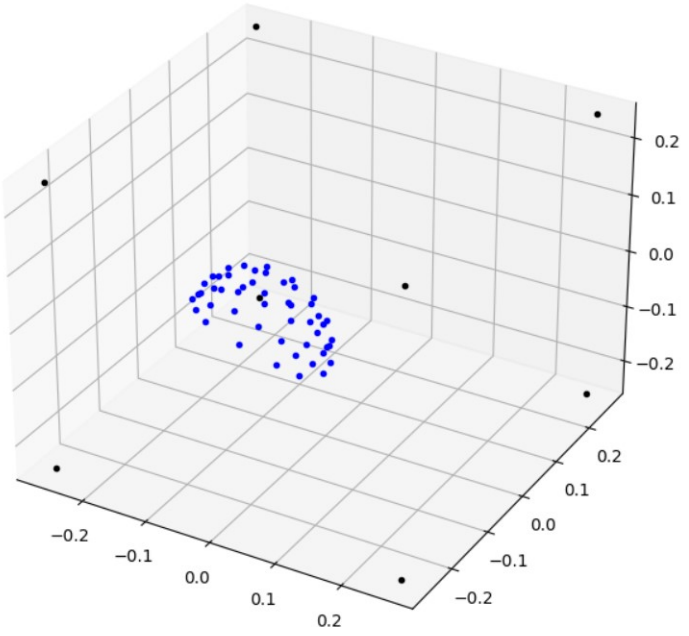


Figure 10: A close up of the head coordinates once R_x, R_y, R_z have been applied to the initial head coordinates

Then these steps of bringing the head coordinates were repeated for all 7200 data points and the resulting animation is shown in the movie included in the .zip file.

3 Time-Frequency decomposition

3.1 Analogue Signal

In this task, we are given an analogue signal, of duration 40 s sampled at 1200 Hz. The signal contains a pure signal with lots of noise. This signal is shown in Figure 11.

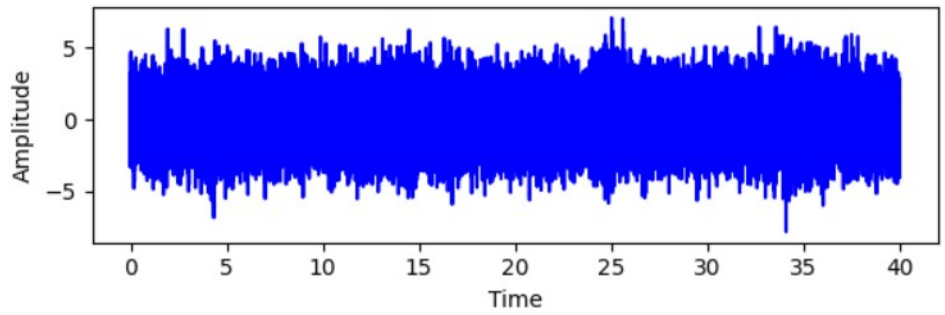


Figure 11: Plot showing the amplitude against time for the unprocessed signal

Producing the time-frequency decomposition for this signal requires taking the Fourier Transform of the signal. This is done using Equ. 4

$$F(\omega) = \int_{-\infty}^{\infty} f(t) e^{-i2\pi\omega t} dt \tag{4}$$

where $F(\omega)$ is the frequency output of the input signal $f(t)$. Calculating the Fourier transform of the signal produces the Figure 12.

From inspection, the signal is composed of many different frequencies ranging from 0 to 580 Hz. There is a large peak at around 5 Hz but there are also several peaks throughout the frequency space.

To obtain the pure signal from the frequency space, we apply a Gaussian filter to this fourier transform of the signal. We do this by multiplying the Gaussian by the fourier transfrom which results in Figure 12.

Now that the peaks of the fourier transform has been separated from the rest of the frequencies, we apply a inverse fourier transform to the filtered fourier transfrom, given by Equ. 5.

$$f(t) = \int_{-\infty}^{\infty} F(\omega) e^{i2\pi\omega t} d\omega \tag{5}$$

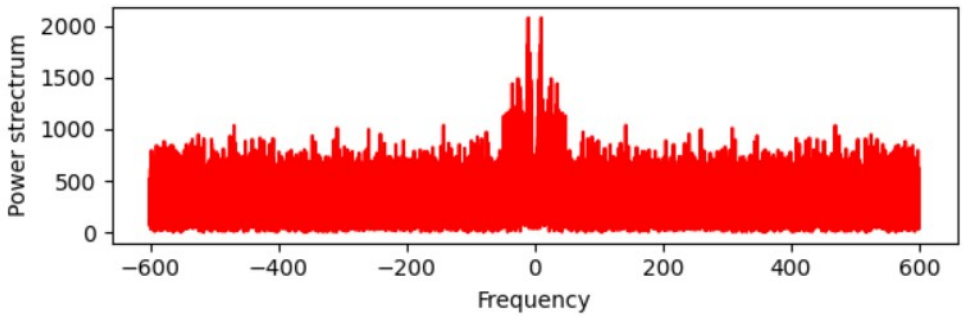


Figure 12: Fourier Transform of the unprocessed signal

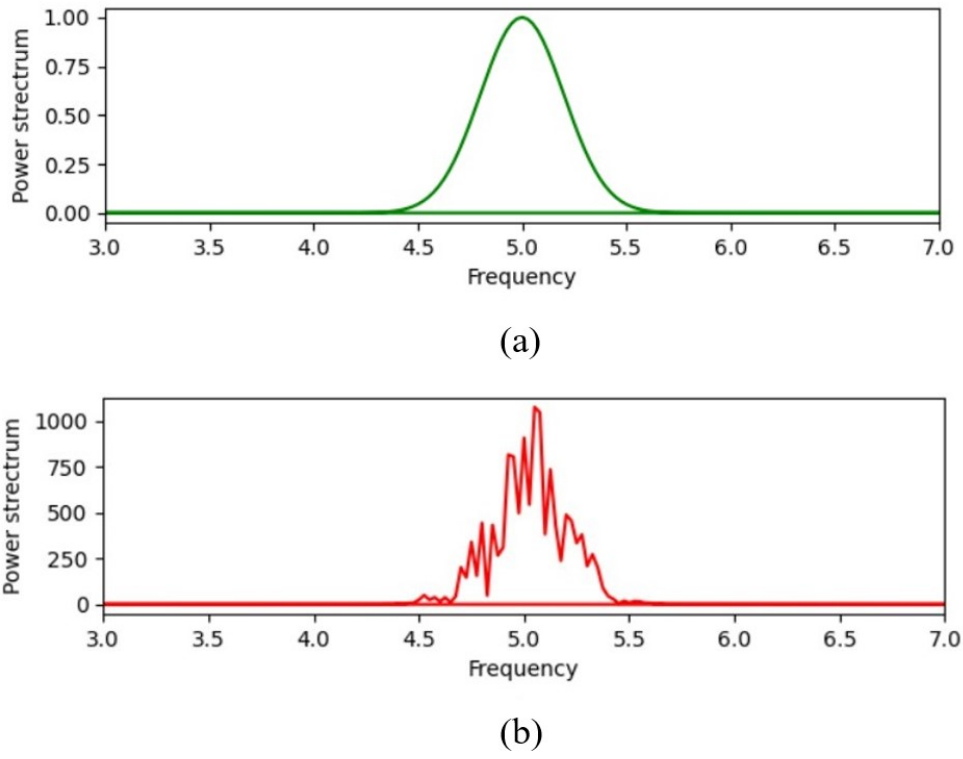
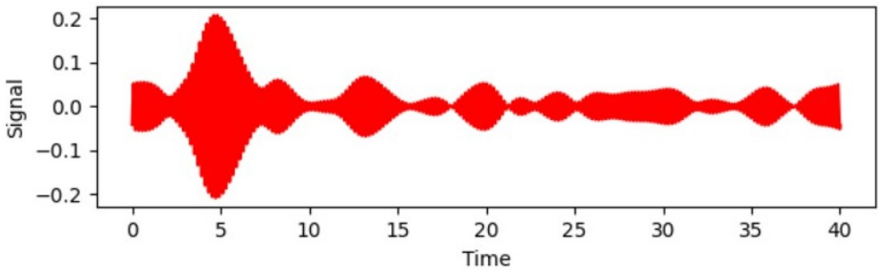
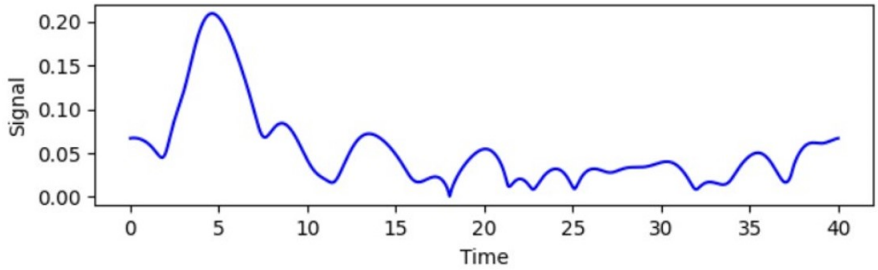


Figure 13: Plot (a) shows a guassian function of $\mu = 5$ and $\sigma = 0.2$ and Plot (b) shows the product of the guassian and the fourier transform

Applying the inverse Fourier transform to the filtered Fourier transform gives the resulting signal in Figure 14.



(a)



(b)

Figure 14: Plot (a) showing the inverse Fourier transfrom and Plot (b) showing the analytical signal of the inverse Fourier transform

Looking at Figure 14(a), there is clearly a pure dominant signal between 0 and 10 seconds in the unprocessed signal. This dominant signal corresponds to 4.5 to 5.5 Hz range because this is the range of the guassian function filtered out the frequencies in the Fourier transform.

The analytical signal shown in Figure 14(b) was produced by taking the Hilbert transform of the pure signal to show the absolute amplitude of the pure signal.

Since there are many peaks in the Fourier transfrom, we will apply a guassian filter for each frequency. We will do this by producing the analytical signal for the pure signal over the full 580 Hz range of frequencies. Then by stacking the analytical signal for 580 frequencies, we can produce a time-frequency decomposition image. This image is shown in Figure 15.

Looking at Figure 15, we can see that there a 4 dominant signals within the unprocessed signal. These dominant signals consist of a 5 Hz signal between 0 and 10s, a 10 Hz signal between 10 and 20s, a 20 Hz between 20 and 30s and a 40 Hz signal between 30 and 40s.

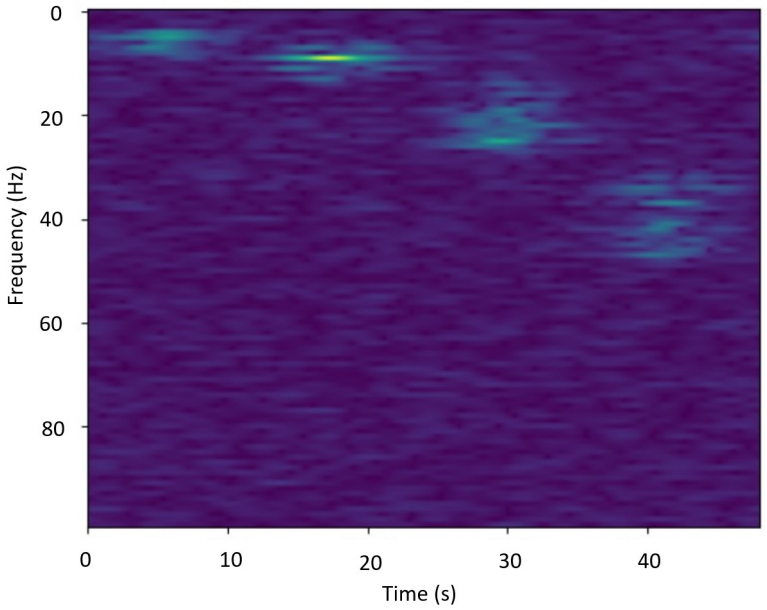


Figure 15: Image showing the intensity of the signal across time and frequency

We can also see that the rest of the image is not made up of dominant signals and seems to be random noise throughout the signal.

For filtering for the signal for each frequency, a $\sigma = 0.2$ was used, this is because the Gaussian function goes to 0 ± 1 from the mean μ frequency. This ensures that the analytical signal produced from each filter corresponds to a single frequency.

Instead of using a Guassian filter, another type of filter that could be used is the smoothed top hat filter which is shown in Figure 16.

Using a smoothed top hat function with a width of 1 would be similar to the Gaussian function used but would include more frequencies since the smoothed top hat function drops off more sharply compared to the Gaussian. However, this may cause ringing artefacts within the time-frequency image.

3.2 Brain Signals

Consider non-invasive electrophysiological imaging data from a human participant given in Figure

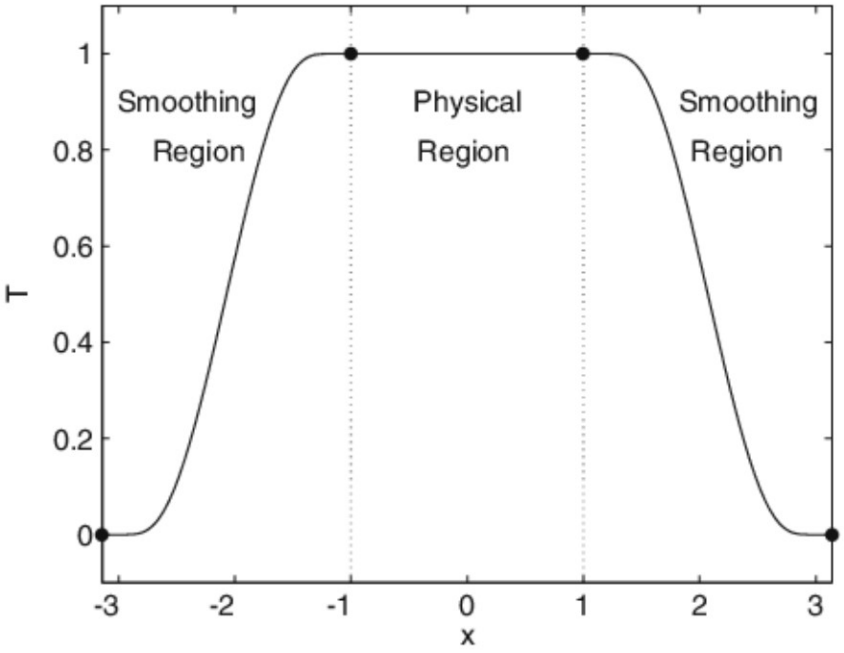


Figure 16: Top hat function with a width of 3 [2]

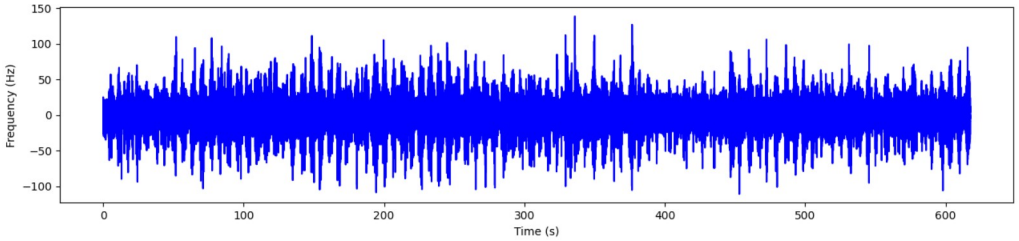


Figure 17: Plot showing electrophysiological imaging data against time

We can repeat the same method as before and take the fourier transform of this signal, as shown by Figure

We can see that there is a very large peak at approximately 10 Hz but the signal also composes of many peaks at different frequencies.

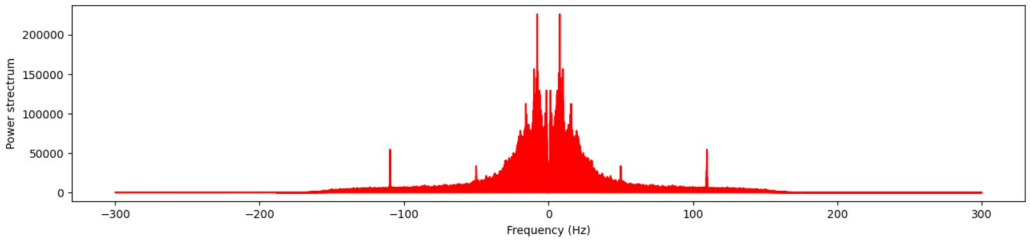


Figure 18: Fouier transfrom of brain signal data

3.3 Other methods of time-frequency analysis

There are other methods of producing time-frequency decomposition such as using wavelet transforms. The wavelet transform improves on the Fourier Transform in that it can analyze a signal by time and frequency simultaneously, thereby easily recovering localized signal information [6].

4 Combination of multiple near-infrared images

Consider astronomical images using ground-based telescopes as shown in Figure 19.

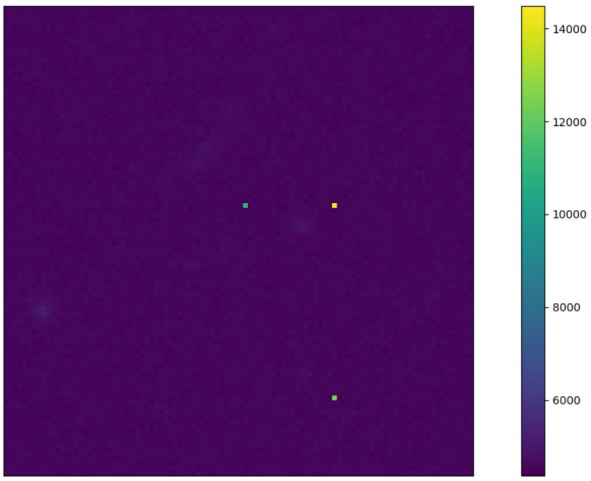


Figure 19: Sample image in the near-infrared (1–2.5µm) wavelength

This sample image contains 3 bright spots of approximate brightness of 14000 where the sensor collected cosmic rays during the exposure. Furthermore, the sensor contains defective pixels that do not detect light properly. Therefore, we must remove these defective pixels and cosmic ray pixels using neighbourhood interpolation. We do this by applying the neighbourhood averaging kernel to each defective pixel in the image, using the kernel given by Figure 20.

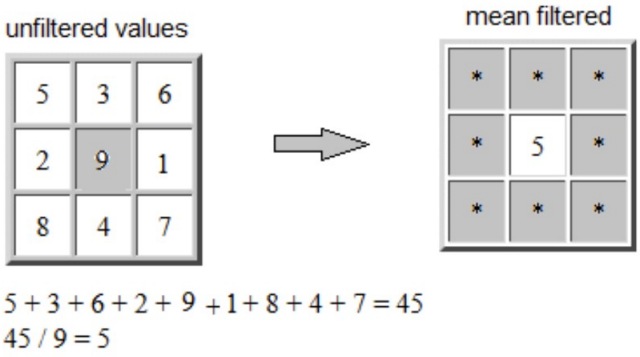


Figure 20: Kernel used to smooth out an image by calculating the mean of all the neighbouring pixels [1]

A list of coordinates for the defective pixels were given and the coordinates of the image were masked and set to NaNs. For the cosmic ray pixels, each pixel of the image was threshold such that, if the pixel brightness was greater than 9000, then it would be masked as a defective pixel and set to NaNs.

Applying this kernel to each defective pixel set to NaN and replacing the NaN with a average pixel brightness produces the image shown in Figure 21.

Looking at Figure 21, we can now see that there are astronomical objects in the image. From the ground-based telescopes, 25 images were captured. We applied this same method of interpolating defective pixels to all 25 images.

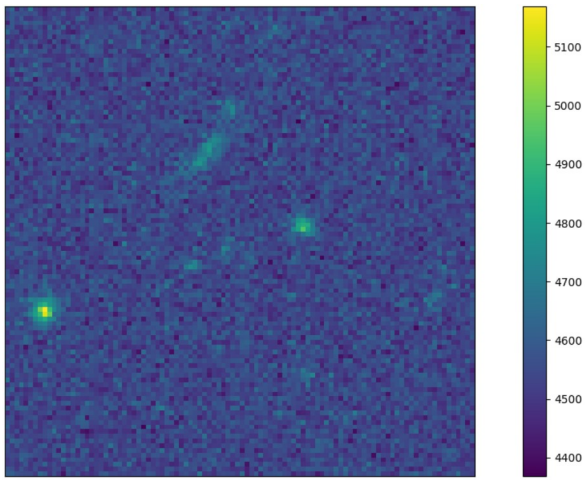


Figure 21: Sample image interpolated using neighbourhood averaging

References

[1] Chandrinos Aristeidis. "The Challenge of Predicting OAG Progression from the Initial Visual Field Test". In: (Jan. 2021). DOI: 10.13140/RG.2.2.20967.27049.

[2] John Boyd. "Asymptotic Fourier Coefficients for a C Bell (Smoothed-"Top-Hat") the Fourier Extension Problem". In: *J. Sci. Comput.* 29 (Sept. 2006). DOI: 10.1007/s10915-005-9010-7.

[3] Philip Evans. "Rotations and rotation matrices". In: *Acta crystallographica. Section D, Biological crystallography* 57 (Nov. 2001), pp. 1355–9. DOI: 10.1107/S0907444901012410.

[4] Tarun Kumar and Karun Verma. "A Theory Based on Conversion of RGB image to Gray image". In: *International Journal of Computer Applications* 7 (Sept. 2010). DOI: 10.5120/1140-1493.

[5] Zalili Musa and Junzo Watada. "Dynamic Tracking System Using Foot Step Direction". In: *BSCHS* 13 (Jan. 2008), pp. 51–57.

[6] Karlton Wirsing. "Time Frequency Analysis of Wavelet and Fourier Transform". In: Nov. 2020. ISBN: 978-1-83881-947-7. DOI: 10.5772/intechopen.94521.