# Lecture Notes, Week 3: FFT and Sampling

## 1 Fourier Transform (FFT)

An important mathematical construct to understanding image processing is to realize that images (and any signals really) can be represented in a frequency domain. Mainly, Fourier has proved that any periodic function can always be written as a sum of sinusoidal waves. Specifically, any 1D periodic signal can be written in terms of the sum of the following periodic bases:

$$Asin(\omega x + \phi) \tag{1}$$

where $A$ is the amplitude, $\phi$ is the phase and $\omega$ is the angular frequency. It is customary to look at the amplitude which tells us frequencies that exist in a given signal. This theory can be relatively easily extended to 2D images that would result in 2D Fourier representations when FFT is applied. The reason this is possible is two fold: (1) images often exhibit periodic structure over the rows and columns (*e.g.*, texture tends to be periodic); and (2) one can always think of an image as a 2D periodic signal with "period" equal to the resolution of the image (think of just, possibly infinitely, tiling an image in space).

While we not going to formally study Fourier space and all the involved mathematics in this class, it is useful to know about it and build certain intuitions. Specifically, an amplitude "image" in the Fourier space corresponds to a 2D image where center corresponds to $\omega = 0$ and frequencies increase as one goes away from the center. For example, stepping away from the center along the $x$ axis in Fourier space would correspond to vertical periodic structure in the image (further away to the right/left the higher the frequency and hence the finer the structure detail); stepping away from the center along the $y$ axis in Fourier space would correspond to horizontal periodic structure with increasing frequency. Similarly, stepping in any off-axis directions will correspond to non-axis oriented periodic structures.

The Fourier transform of a constant image is a single value at the origin of the FFT amplitude space. A constant image does not change over space, and hence, does not contain any sinusoidal bases with non-zero frequency. An image will typically have a spectrum of frequencies. Constant patches in the image will correspond to low frequency content. The larger such patches are, the lower the frequency of the content. Rapid changes in greyscale values over the image (*e.g.*, sharp discontinuities/edges) will correspond to high frequencies – representation of a sharp edge, in principle, requires infinite bases with increasing frequency.

Filters applied as convolution can be characterized by how they change the frequency spectra of the image they are applied on. A smoothing filter will generally reduce the sharp edge structure and hence reduce the frequency spectra of the original image. A *low-pass* filter is defined as a filter that would attenuate, or reduce, high frequencies while leaving the low frequencies intact. In contrast, a *high-pass* filter is defined as a filter that attenuates low frequencies and, effectively, enhances sharp changes in the original image (*e.g.*, edges). A perfect low-pass filter only leaves frequencies below a certain cutoff ($\omega < \omega_{\text{cutoff}}$); and a perfect high-pass filter leaves frequencies above a certain cutoff ($\omega > \omega_{\text{cutoff}}$). A *band-pass* filter leaves only frequencies in a certain range, $\omega_{\text{low-cutoff}} < \omega < \omega_{\text{high-cutoff}}$. Finally, an image (or signal) is *bandlimited* if it has no frequency content above some defined maximum – this will become useful in sampling.

It also turns out that the Fourier transform of a Gaussian is equal to a Gaussian, however, the sigma is not preserved; *i.e.*, Gaussian with low sigma value will correspond to Gaussian with large sigma in the Fourier space and vice versa. Hence, a Gaussian serves as a reasonably good approximation to a perfect low-pass filter. Box and pillbox filters not so much as they have sharper defined edges (especially for smaller filters where the number of edge pixels is significant with respect to non-edge pixels) which will result in certain higher frequency content remaining in the image after filtering.

## 2 Sampling

In the beginning of the course we talked about *image formation*, which described how light travels in the world, reflects off objects and ultimately enters the camera and becomes incident on the imaging plane. Note that at this stage the image is *continuous* and can be thought of as a function, $i(x, y, \lambda)$, of two spatial continuous variables $(x, y)$ and a wavelength $\lambda$. The value of this function can also be considered continuous and represents the number of photons of given wavelength at particular position. We then proceeded to talk about *image filtering*, which processes a discrete array of values $I(X, Y)$ that is read out from the camera sensor. We didn't really talk about the process, and implications, of going from *continuous* image to a discrete one with a discrete pixel grid, discretized grayscale values and three discrete color channels (red, green, and blue). Doing so requires us to talk about theory of *sampling*.

It worth mentioning that sampling is also necessary when creating one digital image from another. For example, resizing an image to half the resolution, requires sampling the original (higher resolution) image at locations that correspond to centers of the pixels in the low-resolution counterpart being created. A naive sampling procedure would amount to simply letting $I_{\text{half-res}}(X, Y) = I(2X, 2Y)$, amounting to taking every 2-nd row and column of the original. This is going to be problematic as we will see.

Let's consider a CCD camera. A CCD camera records image values by turning photons that are incident on the CCD element to electrons (more on this later). A CCD sensor consists of an array of tightly packed element, each such element produces a single value proportional to the number of photons incident on it. This is equivalent to sampling a continuous image over a small region of the CCD photoreceptor cell element. While ideally each element is effectively measuring all light incident on its surface (*i.e.*, effectively computing an integral) which is called *area sampling*, in practice this is not the case. In fact, it is easier to think about sampling as happening at the center of each of the CCD's photoreceptor cell elements. This is referred to as *point sampling*, which in the case of CCD can be thought of as happening on a regular grid.

**Sampling the Range.** Discretizing the range of $i(x, y, \lambda)$ could be done relatively easily by simply defining a maximum number of photons (or energy) that can be measured at a cell element, lets call it $M$, then breaking this range into $N$ equal bins which we will call grayscales. Representing this range using 8-bits results in 255 greyscales. For example, first greyscale level will correspond to $[0, M/255]$ and so on. If larger number than M photons is registered at a given cell then it will "overflow" and greyscale value of 255 will be assigned.

**Sampling the Domain.** Discretizing the range amounts to sampling $i(x, y, \lambda)$ at a regular grid defined by the resolution of the sensor. Note that $i(x, y, \lambda)$ will have some frequency spectra which will depend both on the frequency of structure in the physical world and the distance of the camera to this structure. For example, low frequency structure imaged from far away may manifest itself as high frequency structure on the continuous projected image plane $i(x, y, \lambda)$. Vice versa can also be true. An important aspect to consider is how can we reason about the fact whether sampled discrete image is a good representation of the continuous counterpart. One way to formalize this question is to ask when can we reconstruct the original continuous image from the sampled discrete one.

*Nyquist sampling theorem* states that a signal is exactly recoverable from its samples if it is sampled at the Nyquist rate (or higher), where Nyquist rate is defined as twice the highest frequency of the bandlimited signal being sampled. Note the signal must be bandlimited to contain a well defined highest frequency. In other words, the sampling rate $f_s$ must be greater or equal to $2 \times f_{max}$:

$$f_s \geq 2 \times f_{max},$$

otherwise artifacts (named aliasing) can exhibit themselves in the sampled signal. Aliasing can take many forms but typically correspond to certain content of the original signal being lost or distorted. For example,

a black and while checkerboard continuous image sampled below Nyquist rate may appear as a discrete image that is purely white, purely black, or one that does not exhibit a regular pattern of a checkerboard.

Artifacts discussed above also present themselves when resizing an image. For example, re-sizing the image to a lower resolution by simply resampling it, may result in aliasing artifacts as shown in class. To ensure this does not happen one must ensure that original image being resized does not contain frequencies above $f_{max}$ as dictated by the desired resize factor (sapling rate). This can be achieved by filtering an image with low-pass filter with appropriate cut-off. In our case, we will use a Gaussian filter with $\sigma$ related to the subsampling factor. Mainly, the rule of thumb we will use is that $\sigma = \frac{1}{2s}$, where $s$ is a scaling factor, *e.g.*, 0.5 for half resolution.

Another alternative for alleviating aliasing is *oversampling*. In oversampling, the goal is to sample the multiple times for each required sample and to average those results. This is an approximation to area sampling discussed above. For example, when looking for a value of pixel at position $(X, Y)$ one can sample multiple locations and average the values. Consider producing a half-resolution image. Instead of naive sampling: $I_{\text{half-res}}(X, Y) = I(2X, 2Y)$, we can do the following:

$$I_{\text{half-res}}(X, Y) = \frac{I(2X, 2Y) + I(2X - 0.25, 2Y) + I(2X + 0.25, 2Y) + I(2X, 2Y - 0.25) + I(2X, 2Y + 0.25)}{5}$$

to produce an estimate for a pixel. Note that there is a close relationship between *oversampling* and *smoothing* solutions introduced. In oversampling, one samples than averages. In smoothing, one first averages (after all Gaussian is a particular averaging filter) and then sample.

## 3 Design of the color camera

One thing we have not discussed is how color is processed and discretized in cameras. CCD photoreceptor cell element consists of a photodiode, where the photons, through interaction with silicon atoms, are converted to electrons. These electrons are then stored in the potential well before being read off. The percentage of photons that are actually detected is known as the *Quantum Efficiency* (QE). For example, the human eye only has a QE of about 20%, photographic film has a QE of around 10%, and the best CCDs can achieve a QE of over 80%. Quantum efficiency will vary with wavelength, but as a whole, photoreceptor cell elements by themselves can't distinguish wavelengths of photons incident on them. Hence a mechanism that allows to somehow capture and quantize wavelengths (*i.e.*, color) of light is needed. This is achieved by introducing a *Color Filter Array* (CFA) on top of the CCD's photoreceptor cell elements.

A different color filter is placed on top of each photoreceptor. This filter only allows a certain distribution of wavelengths through. This allows the corresponding photoreceptor to only count photons within a range of wavelengths allowed by the corresponding filter. Typically there are three types of color filters used and arranged in certain pattern across photoreceptor elements in CCD. Typically one must make two important design choices: (1) what spectral sensitivity function to encode in each filter, and (2) how to spatially arrange these different color filters – this is called a *mosaic* pattern. These design choices differ from camera to camera. However, the final result is a Bayer image where at each location only one of the R,G,B channels is observed. The missing values for each channel are obtained through interpolation.

Additional steps in camera processing pipeline include *white balancing* and *tone mapping*. White balancing accounts for various lighting sources in the world. The purpose of this step is to ensure that "white" or "grey" looks correct irrespective of the source of light in the scene (be it daylight, sunlight, florescent or any other). Typically this is done automatically by finding the brightest pixels in the image along the three channels and then normalizing the other channels to make sure they have equal value. Tone mapping on the other hand is a process of mapping camera observed color pallet to match the color pallet in the world. This often requires calibration with known colors.