Summer of Code

Computer Vision

Midterm report 2025

By prem kumar lodhi(24B1847)
Under yashvardhan and krunal

Content

1.Sampling in SP

Sampling a **cosine function** in signal processing means evaluating the function at discrete time intervals, rather than continuously. This is a foundational concept in digital signal processing (DSP).Sampling involves taking this continuous signal and evaluating it at discrete time instants:

Where: $$x[n] = x(nT_s) = A\cos(2\pi f n T_s + \phi)$$

Ts=1/fs is the **sampling interval** fs is the **sampling frequency** n∈Z (0, 1, 2, ...) is the sample index

So the discrete-time version becomes:

$$x[n] = A\cos\left(2\pi \frac{f}{f_s} n + \phi\right)$$

1.1.Normalized Frequency

We define **normalized frequency**: $$f_{\text{norm}} = \frac{f}{f_s}$$

Then: $$x[n] = A\cos(2\pi f_{\text{norm}} n + \phi)$$

This is a **periodic sequence** in discrete-time if f(norm) is rational.

To **perfectly reconstruct** the original signal from its samples:

fs> >2f

Otherwise, **aliasing** occurs — frequencies get misrepresented.

1.2.Aliasing(undersampling)

Aliasing is a fundamental concept in signal processing where high-frequency signals appear as lower frequencies after sampling — a kind of **"identity theft"** of frequencies. It's one of the most important pitfalls when digitizing analog signals.

Aliasing occurs when you sample a continuous-time signal at a rate that is too low to capture all of its frequency content.Imagine you spin a wheel very fast and take photos at slow intervals — the wheel may appear to spin backward or slower than it actually is. That illusion is aliasing.

Consequences of aliasing :

A signal of frequency fff can be mistaken for a signal of another frequency after sampling.

Continuous-Time Cosine: $x(t) = \cos(2\pi f t)$

Sampled Discrete-Time Version: $x[n] = x(nT_s) = \cos(2\pi f n T_s) = \cos\left(2\pi \frac{f}{f_s} n\right)$

The problem arises when:

2f>fs

In that case, the normalized frequency fnormf_{\text{norm}}fnorm will "wrap around" and look like a different lower frequency.To **avoid aliasing**, you must sample at **at least twice the highest frequency** present in the signal

2. Quantization

Digital systems (like computers, ADCs, microcontrollers) cannot handle infinite precision. They can only store numbers with a finite number of bits — so we **approximate** real-valued amplitudes using a limited set of levels.Quantization is the process of mapping a large (often infinite) set of amplitude values to a smaller finite set.

HOW QUANTIZATION WORKS

imagine a signal with values ranging from -1 to 1. If we want to store it using **3 bits**, we only have 23=82^3 = 823=8 levels.

So we divide the range [−1,1][-1, 1][−1,1] into 8 equal intervals:

Each incoming value is then **rounded** (or "snapped") to the nearest quantization level.

$$\text{Step size } \Delta = \frac{\max - \min}{2^n} = \frac{2}{8} = 0.25$$

2.1.Quantization Error

Because we round real values to discrete ones, we introduce **quantization error**:

$$e[n] = x[n] - x_Q[n]$$

Original sample quantized version

If we use **uniform quantization**, this error is bounded:

$$|e[n]| \leq \frac{\Delta}{2}$$

Ps: Ive studied a lot more things but for the time being i was able to make a presentation of these two topics only.

Learning is ok but presenting it is a totaly different game

thankyou