Image Sampling

Michael Kazhdan

(601.457/657)

HB Ch. 4.8, 16.5
FvDFH Ch. 14.10, 17.6
Sampling Questions

• How should we sample an image:
  ◦ Nearest Point Sampling?
  ◦ Bilinear Sampling?
  ◦ Gaussian Sampling?
  ◦ Something Else?
Image Representation

What is an image?

An image is a discrete collection of pixels, each representing a sample of a continuous function.

Continuous image

Digital image
Sampling

Let’s look at a 1D example:

Continuous Function  Discrete Samples
Sampling

At in-between positions, values are undefined.

How do we determine the value of a sample?

We need to reconstruct a continuous function, turning a collection of discrete samples into a 1D function that we can sample at arbitrary locations.
Nearest Point Sampling

The value at a point is the value of the closest discrete sample.
Nearest Point Sampling

The value at a point is the value of the closest discrete sample.

The reconstruction:

- Interpolates the samples
- Is not continuous
(Bi)linear Sampling

The value at a point is the (bi)linear interpolation of the two surrounding samples.
(Bi)linear Sampling

The value at a point is the (bi)linear interpolation of the two surrounding samples.

The reconstruction:

- Interpolates the samples
- Is not smooth
Gaussian Sampling

The value at a point is the Gaussian average of the surrounding samples.
Gaussian Sampling

The value at a point is the Gaussian average of the surrounding samples.

The reconstruction:
✗ Does not interpolate
✓ Is smooth
Image Sampling

Conceptually, this is done in two steps:
1. Reconstruct a continuous function from input samples.
2. Sample a continuous function at a fixed resolution.

Challenge:
Reconstruction is an under-constrained problem.
⇒ Need to define what makes a good reconstruction.
Conceptually this is done in two steps:

1. Reconstruct a continuous function from input samples.
2. Sample a continuous function at a fixed resolution.

**Challenge:**

Reconstruction is an under-constrained problem. ⇒ Need to define what makes a good reconstruction.

**Key Ideas:**

1. Of all possible reconstructions, we want the one that is smoothest (has lowest frequencies).
2. How we reconstruct should also depend on how we will sample.

Signal processing helps us formulate this precisely.
Fourier Analysis

- Fourier analysis provides a way for expressing (or approximating) any signal as a sum of scaled and shifted cosine functions.

The Building Blocks for the Fourier Decomposition
Fourier Analysis

- Fourier analysis provides a way for expressing (or approximating) any signal as a sum of scaled and shifted cosine functions.

Recall:
In the expression $\cos(k\theta)$, the value $k$ is the frequency of the function.
Fourier Analysis

- As higher frequency components are added to the approximation, finer details are captured.

\[ f_0(\theta) = a_0 \cdot \cos(0 \cdot (\theta + \phi_0)) \]

Initial Function

\[ f(\theta) \]

0th Order Approximation

0th Order Component
Fourier Analysis

• As higher frequency components are added to the approximation, finer details are captured.

Initial Function

1\textsuperscript{st} Order Approximation

\( f_1(\theta) = a_1 \cdot \cos(1 \cdot (\theta + \phi_1)) \)

1\textsuperscript{st} Order Component
Fourier Analysis

- As higher frequency components are added to the approximation, finer details are captured.

\[ f_2(\theta) = a_2 \cdot \cos(2 \cdot (\theta + \phi_2)) \]
Fourier Analysis

• As higher frequency components are added to the approximation, finer details are captured.

\[ f_3(\theta) = a_3 \cdot \cos(3 \cdot (\theta + \phi_3)) \]
Fourier Analysis

- As higher frequency components are added to the approximation, finer details are captured.

Initial Function

$\mathcal{F}(\theta)$

4th Order Approximation

$f_4(\theta) = a_4 \cdot \cos(4 \cdot (\theta + \phi_4))$

3rd Order Approximation

4th Order Component
Fourier Analysis

- As higher frequency components are added to the approximation, finer details are captured.

\[
f_5(\theta) = a_5 \cdot \cos(5 \cdot (\theta + \phi_5))
\]

5th Order Component
Fourier Analysis

- As higher frequency components are added to the approximation, finer details are captured.

\[
f_6(\theta) = a_6 \cdot \cos(6 \cdot (\theta + \phi_6))
\]

6th Order Component
Fourier Analysis

- As higher frequency components are added to the approximation, finer details are captured.

\[ f_7(\theta) = a_7 \cdot \cos(7 \cdot (\theta + \phi_8)) \]

7th Order Component
Fourier Analysis

• As higher frequency components are added to the approximation, finer details are captured.

\[ f_8(\theta) = a_8 \cdot \cos(8 \cdot (\theta + \phi_8)) \]
Fourier Analysis

- As higher frequency components are added to the approximation, finer details are captured.

\[ f_9(\theta) = a_9 \cdot \cos(9 \cdot (\theta + \phi_9)) \]

9th Order Component
Fourier Analysis

- As higher frequency components are added to the approximation, finer details are captured.

\[ f_{10}(\theta) = a_{10} \cdot \cos(10 \cdot (\theta + \phi_{10})) \]

10th Order Component
Fourier Analysis

• As higher frequency components are added to the approximation, finer details are captured.

\[ f_{11}(\theta) = a_{11} \cdot \cos(11 \cdot (\theta + \phi_{11})) \]

Initial Function

10th Order Approximation

11th Order Approximation

11th Order Component
Fourier Analysis

• As higher frequency components are added to the approximation, finer details are captured.

\[ f_{12}(\theta) = a_{12} \cdot \cos(12 \cdot (\theta + \phi_{12})) \]

12th Order Component
Fourier Analysis

- As higher frequency components are added to the approximation, finer details are captured.

Initial Function

13th Order Approximation

12th Order Approximation

13th Order Component

\[ f_{13}(\theta) = a_{13} \cdot \cos(13 \cdot (\theta + \phi_{13})) \]
Fourier Analysis

• As higher frequency components are added to the approximation, finer details are captured.

\[ f_{14}(\theta) = a_{14} \cdot \cos(14 \cdot (\theta + \phi_{14})) \]

Initial Function  

14th Order Approximation

13th Order Approximation

14th Order Component
Fourier Analysis

- As higher frequency components are added to the approximation, finer details are captured.

\[ f_{15}(\theta) = a_{15} \cdot \cos(15 \cdot (\theta + \phi_{15})) \]

15\textsuperscript{th} Order Component
Fourier Analysis

- As higher frequency components are added to the approximation, finer details are captured.

\[
f_{16}(\theta) = a_{16} \cdot \cos(16 \cdot (\theta + \phi_{16}))
\]

16th Order Component
Fourier Analysis

- Combining all of the frequency components together, we get the initial function:

\[ f(\theta) = \sum_{k=0}^{\infty} a_k \cdot \cos(k(\theta + \phi_k)) \]

\(a_k\): amplitude of the \(k^{th}\) frequency component

\(\phi_k\): shift of the \(k^{th}\) frequency component
Question

Goal:

• Fit a continuous signals to the samples using only the lowest frequencies.

• Given \( m \) samples, how many frequencies can we reconstruct?

\[
\text{Initial Function } f(\theta) = f_0(\theta) + f_1(\theta) + f_2(\theta) + f_3(\theta) + f_4(\theta) + f_5(\theta) + f_6(\theta) + f_7(\theta) + f_8(\theta) + \ldots
\]
Question

Goal:

• Fit a continuous signals to the samples using only the lowest frequencies.

• Given $m$ samples, how many frequencies can we reconstruct?

Each frequency component has two degrees of freedom:

• Amplitude
• Shift

With $m$ samples we can reconstruct $m/2$ frequencies.
Shannon’s Theorem:

A signal can be reconstructed from its samples, if the original signal has no frequencies above 1/2 the sampling rate -- a.k.a. the *Nyquist Frequency*.

**Terminology:**

- A signal is *band-limited* if its highest non-zero frequency is bounded.
- The frequency is called the *bandwidth*.
- The minimum sampling rate for band-limited function is called the *Nyquist rate* (twice the bandwidth).
To reconstruct the continuous function from $m$ input samples, we can find the unique function of frequency $m/2$ that interpolates the values.

Q: Why don’t we just evaluate this function at the $n$ output sample positions?

A: If $n < m$ we sample below the Nyquist rate!
Aliasing

- When a high-frequency signal is sampled with insufficiently many samples, it can be perceived as a lower-frequency signal. This masking of higher frequencies as lower ones is referred to as aliasing.
Aliasing

- When a high-frequency signal is sampled with insufficiently many samples, it can be perceived as a lower-frequency signal. This masking of higher frequencies as lower ones is referred to as **aliasing**.
Aliasing

• When a high-frequency signal is sampled with insufficiently many samples, it can be perceived as a lower-frequency signal. This masking of higher frequencies as lower ones is referred to as **aliasing**.
Aliasing

When a high-frequency signal is sampled with insufficiently many samples, it can be perceived as a lower-frequency signal. This masking of higher frequencies as lower ones is referred to as aliasing.
Temporal Aliasing

- Artifacts due to limited temporal resolution
Temporal Aliasing

- Artifacts due to limited temporal resolution
Temporal Aliasing

- Artifacts due to limited temporal resolution
Temporal Aliasing

- Artifacts due to limited temporal resolution
Temporal Aliasing

- Artifacts due to limited temporal resolution
Temporal Aliasing

• Artifacts due to limited temporal resolution
Sampling

- There are two problems:
  - You don’t have enough samples to correctly reconstruct/represent the high-frequency information.
  - You corrupt the low-frequency information because the high-frequencies mask themselves as lower ones.
Anti-Aliasing

Two possible ways to address aliasing:

• Sample at higher rate

• Pre-filter to form band-limited signal
Anti-Aliasing

Two possible ways to address aliasing:

• Sample at higher rate
  - Not always possible
  - Still rendering to fixed resolution

• Pre-filter to form band-limited signal
Anti-Aliasing

Two possible ways to address aliasing:

- Sample at higher rate
- Pre-filter to form a band-limited signal
  - You still don’t get your high frequencies, but at least the low frequencies are uncorrupted.
Fourier Analysis

- If we look at the amplitude at each frequency, we obtain the power spectrum of the signal:

\[
f(\theta) = \sum_{k=0}^{\infty} a_k \cos(k(\theta + \phi_k))\]
Fourier Analysis

• If we look at the amplitude at each frequency, we obtain the power spectrum of the signal:

\[ f(\theta) = \sum_{k=0}^{\infty} a_k \cos(k(\theta + \phi_k)) \]
Pre-Filtering

• Band-limit by discarding the high-frequency (greater than $n/2$) components that can’t be represented by the output sampling resolution.

Initial Spectrum

Band-Limited Spectrum
Pre-Filtering

- Band-limit by discarding the high-frequency (greater than $n/2$) components that can’t be represented by the output sampling resolution.

- We could do this if we could multiply the frequency components by a 0/1 function:
Pre-Filtering

- Band-limit by discarding the high-frequency (greater than $n/2$) components that can’t be represented by the output sampling resolution.

- We could consider the frequency function:

$$f(\theta) = \sum_{k=0}^{\infty} a_k \cos(k(\theta + \phi_k))$$

$$\downarrow$$

$$f(\theta) = \sum_{k=0}^{n/2} a_k \cos(k(\theta + \phi_k))$$

Initial Spectrum  $\overset{\text{Frequency Filter}}{\rightarrow}$  Band-Limited Spectrum
Fourier Theory

• A fundamental fact from Fourier theory is that multiplication in the frequency domain is equivalent to convolution in the spatial domain.
Convolution

- The convolution of two functions $f$ and $g$, denoted $f \ast g$, is obtained by sampling the function $f$ using the weights given by $g$. 

$$f(\theta)$$

$$g(\theta)$$

1
Convolution

- The convolution of two functions $f$ and $g$, denoted $f \ast g$, is obtained by sampling the function $f$ using the weights given by $g$. 

$$f(\theta) \ast g(\theta)$$

$$(f \ast g)(\theta)$$
Convolution

- The convolution of two functions $f$ and $g$, denoted $f * g$, is obtained by sampling the function $f$ using the weights given by $g$. 

\[
(f * g)(\theta) = \int f(\theta - t) g(t) \, dt
\]
Convolution

• The convolution of two functions $f$ and $g$, denoted $f \ast g$, is obtained by sampling the function $f$ using the weights given by $g$. 

$$f(\theta) \ast g(\theta) = (f \ast g)(\theta)$$
Convolution

- The convolution of two functions $f$ and $g$, denoted $f \ast g$, is obtained by sampling the function $f$ using the weights given by $g$. 

\[
(f \ast g)(\theta) = \int f(\theta') \cdot g(\theta - \theta') \, d\theta'
\]
Convolution

• The convolution of two functions $f$ and $g$, denoted $f * g$, is obtained by sampling the function $f$ using the weights given by $g$. 

$$f(	heta)$$

$$g(	heta)$$

$$(f* g)(\theta)$$
Convolution

- The convolution of two functions $f$ and $g$, denoted $f * g$, is obtained by sampling the function $f$ using the weights given by $g$. 

\[
(f * g)(\theta) = \int f(\theta - \tau) g(\tau) \, d\tau
\]
Convolution

- The convolution of two functions $f$ and $g$, denoted $f \ast g$, is obtained by sampling the function $f$ using the weights given by $g$. 
Convolution

- The convolution of two functions $f$ and $g$, denoted $f * g$, is obtained by sampling the function $f$ using the weights given by $g$. 
Convolution

- The convolution of two functions \( f \) and \( g \), denoted \( f \ast g \), is obtained by sampling the function \( f \) using the weights given by \( g \).
Convolution

- The convolution of two functions \( f \) and \( g \), denoted \( f \ast g \), is obtained by sampling the function \( f \) using the weights given by \( g \).
Convolution

• To convolve two functions $f$ and $g$, we resample the function $f$ using the weights given by $g$.

• Nearest, (bi)linear, and Gaussian interpolation are just convolutions with different filters.
Convolution

- Recall that convolution in the spatial domain is equal to multiplication in the frequency domain.
- In order to avoid aliasing, we need to convolve with a filter whose power spectrum has value:
  - 1 at low frequencies
  - 0 at high frequencies

![Initial Spectrum](image1.png) ![Frequency Filter](image2.png) ![Band-Limited Spectrum](image3.png)
Nearest Point Convolution

Note: The spectrum does not really fall off at high frequencies.

Also: The nearest-point filter does not provide a way for controlling the cut-off frequency.
(Bi)Linear Convolution

**Note:**
The spectrum does a better job of falling off at high frequencies, but still doesn’t get to zero.

**Also:**
The (bi)linear filter does not provide a way for controlling the cutoff frequency.
Gaussian Convolution

The Gaussian filter does provide a way for controlling the cut-off frequency.

Note: The spectrum quickly decays to zero at high frequencies, (falling off like a Gaussian).

Also: The Gaussian filter does provide a way for controlling the cut-off frequency.
Convolution

• The ideal filter for avoiding aliasing should have a power spectrum with values:
  ◦ 1 at low frequencies
  ◦ 0 at high frequencies

• The sinc function has such a power spectrum and is referred to as the ideal reconstruction filter:

\[
sinc(\theta) = \begin{cases} 
\frac{\sin(\theta)}{\theta} & \text{if } \theta \neq 0 \\
1 & \text{if } \theta = 0 
\end{cases}
\]
The Sinc Filter

• The ideal filter for avoiding aliasing should have a power spectrum with values:
  ○ 1 at low frequencies
  ○ 0 at high frequencies

• The sinc function has such a power spectrum and is referred to as the *ideal reconstruction filter*:

![Reconstruction Filter](image)

![Filter Spectrum](image)
The Sinc Filter

- Limitations:
  - Has negative values, giving rise to negative weights in the interpolation → can extrapolate values.
The Sinc Filter

- Limitations:
  - Has negative values, giving rise to negative weights in the interpolation → can extrapolate values.
  - Discontinuity in the frequency domain causes ringing near spatial discontinuities (Gibbs Phenomenon).
The Sinc Filter

- Limitations:
  - Has negative values, giving rise to negative weights in the interpolation $\rightarrow$ can extrapolate values.
  - Discontinuity in the frequency domain causes ringing near spatial discontinuities (Gibbs Phenomenon).
  - The filter has large support so evaluation is slow.
Summary

There are different ways to sample an image:

- Nearest Point Sampling
- Linear Sampling
- Gaussian Sampling
- Sinc Sampling
Summary – Nearest

✓ Can be implemented efficiently because the filter is non-zero in a very small region.

? Interpolates the samples.

× Is discontinuous.

× Does not address aliasing, giving bad results when a high-frequency signal is under-sampled.
Summary – (Bi)linear

✓ Can be implemented efficiently because the filter is non-zero in a very small region.

? Interpolates the samples.

✗ Is not smooth.

✗ Partially addresses aliasing, but stills give bad results when a high-frequency signal is undersampled.

Discrete Samples \* Reconstruction Filter = Reconstructed Function
Summary – Gaussian

✗ Is slow to implement because the filter is non-zero in a large region.
?

Does not interpolate the samples.
✓ Is smooth.
✓ Addresses aliasing by killing off high frequencies.
Summary – Sinc

✗ Is really slow to implement because the filter is non-zero, and large, in a large region.
?< Interpolates the samples.
✗ Assigns negative weights.
✗ Ringing at discontinuities.
✓ Addresses aliasing by killing off high frequencies.
Image Sampling (Theory)

Given a source signal sampled at $m$ positions, to get a destination image sampled at $n$ positions we:

1. **Reconstruct:**
   Generate a source function with maximum non-zero frequency equal to $m/2$.

2. **Band-limit:**
   Filter the source function to have frequency no larger than $n/2$.

3. **Sample:**
   Evaluate the filtered function at the $n$ positions.
Image Sampling

Example ($m = 25 \rightarrow n = 25/10$):

Sampled $m = 25$

Sampled $m = 25$
Image Sampling

Example \((m = 25 \rightarrow n = 25/10)\):

Sampled \(m = 25\)

Reconstruction

Sampled \(m = 25\)

Reconstruction
Image Sampling

Example \((m = 25 \rightarrow n = 25/10)\):

- **Sampled** \(m = 25\)
- **Sampled** \(n = 25\)
- **Sampled** \(n = 10\)
- **Reconstruction**
Image Sampling (Practice)

Given a source signal sampled at $m$ positions, to get a destination image sampled at $n$ positions we:

- Resample the source image using a (Gaussian) filter whose width is determined by the number of input/output samples.
- This simultaneously:
  1. Constructs a continuous function from the input samples
  2. Band-limits the continuous function based to account for the output sampling resolution
  3. Samples the band-limited function
Gaussian Sampling

Recall:

To avoid aliasing, we kill off the high-frequency components by convolving with a Gaussian because its power spectrum is:

- (approximately) one at low frequencies
- (approximately) zero at high frequencies
Gaussian Sampling (Rule of Thumb)

**Q:** What standard deviation should we use?

**A:** The standard deviation should be between 0.5 and 1.0 times the maximum distance between samples in the input and output.

---

Gaussians used for reconstructing and sampling a function with 20 samples

$\sigma = \frac{1}{20}$

$\sigma = \frac{5}{20}$
Gaussian Sampling (Rule of Thumb)

Q: What standard deviation should we use?

A: The standard deviation should be between 0.5 and 1.0 times the maximum distance between samples in the input and output.

Power spectra of the Gaussians used for reconstructing and sampling a function with 20 samples
Gaussian Sampling

Scaling Example:

Q: If we have data represented by 20 samples that we would like to down-sample to 5 samples. What standard deviation should we use?

A: Distance between adjacent input samples: 1
   Distance between adjacent output samples: 4
   ⇒ The standard deviation of the Gaussian used to sample the input should be between 2.0 and 4.0 input sampling units.
Gaussian Sampling

Scaling Example:

Q: If we have data represented by 5 samples that we would like to up-sample to 20 samples. What standard deviation should we use?

A: Distance between adjacent input samples: 1
   Distance between adjacent output samples: 0.25

⇒ The standard deviation of the Gaussian used to sample the input should be between 0.5 and 1.0 input sampling units.