Vision Systems

Lecture 5

PART 1

Image Sampling and Interpolation

Cost Improvement using Convolution Theorem?

Convolution Theorem

Fourier transform of convolution of two functions is a product of their Fourier transforms:

$$F[g * h] = F[g]F[h]$$

Convolution in the spatial domain can be obtained through **multiplication** in the frequency domain!

$$g * h = F^{-1}[F[g]F[h]]$$

Cost Improvement using Convolution Theorem?

Convolution Theorem

Fourier transform of convolution of two functions is a product of their Fourier transforms:

$$F[g * h] = F[g]F[h]$$

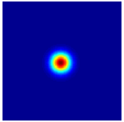
Convolution in the spatial domain can be obtained through **multiplication** in the frequency domain!

$$g * h = F^{-1}[F[g]F[h]]$$

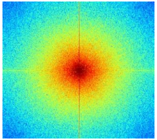
- Image convolution needs $O(N^2 \cdot k^2)$ time, where $N \times N$ is image size, and $k \times k$ is kernel size
- By performing convolution in Fourier domain, cost is: $O(N^2)$ for a single pass over the image + cost of FFT: $O(N^2 \log N^2)$ for the image and $O(k^2 \log k^2)$ for the kernel $\approx O(N^2 \log N^2 + k^2 \log k^2)$, in total (other terms additive)

Exercise: Match spatial domain image to Fourier magnitude image

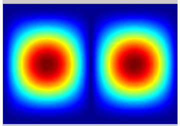
1



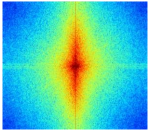
2



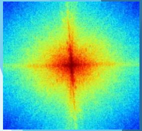
3



4



5

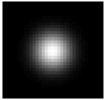


A

B

C

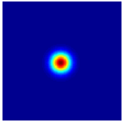
D



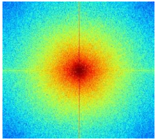
E

Exercise: Match spatial domain image to Fourier magnitude image

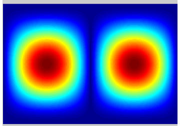
1



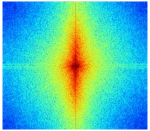
2



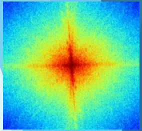
3



4



5

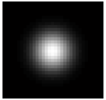


A

B

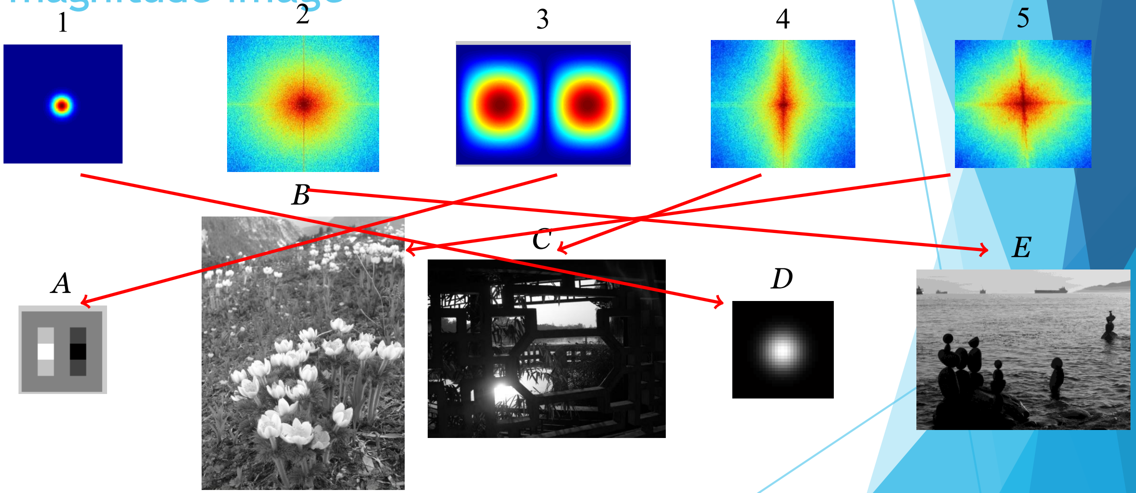
C

D

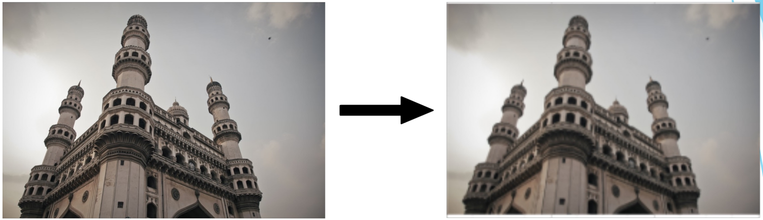


E

Exercise: Match spatial domain image to Fourier magnitude image



What sense does a low-resolution image make to us?



Original   Subsampled & zoomed

Clues from human perception:

- Early processing in human's filters for various orientations and scales of frequency.
- Perceptual cues in mid-high frequencies dominate perception.
- When we see an image from far away, we are effectively **sub-sampling it**.

Credit: Ron Hansen (Unsplash)

What sense does a low-resolution image make to us?

Original

Subsampled & zoomed

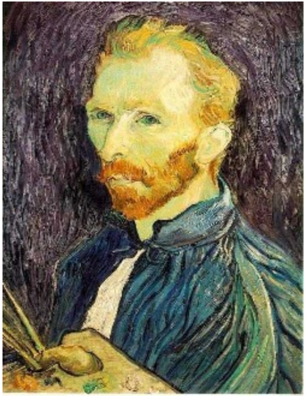
Clues from human perception:

- Early processing in human's filters for various orientations and scales of frequency.
- Perceptual cues in mid-high frequencies dominate perception.
- When we see an image from far away, we are effectively **sub-sampling** it.

Credit: Ron Hansen (Unsplash)

Sub-sampling

Throw away every other row and column to create a 1/2 size image.



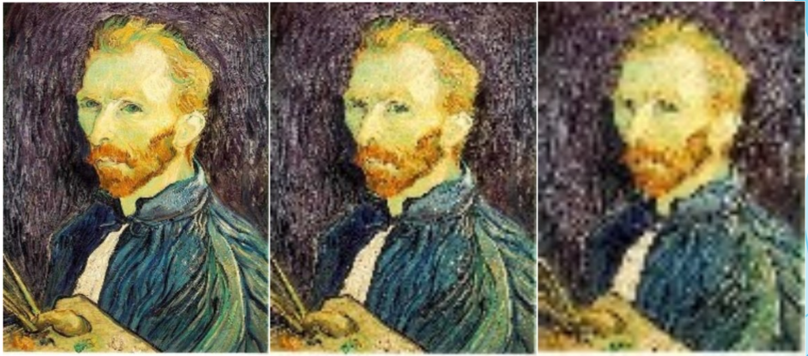
1/4

1/8

Credit: S Seitz, R Urtasun

Sub-sampling

Why does this look so crufty?

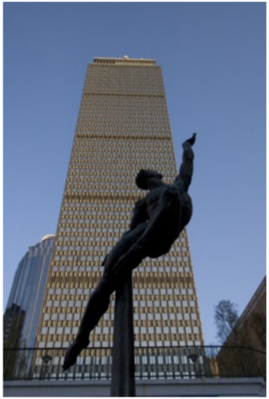


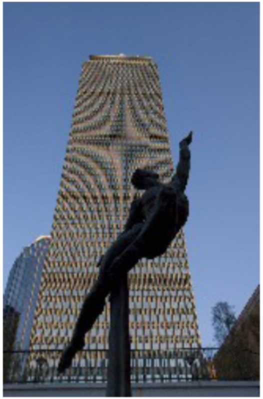
1/2 1/4 (2x zoom) 1/8 (4x zoom)

Credit: S Seitz, R Urtasun

Sub-sampling

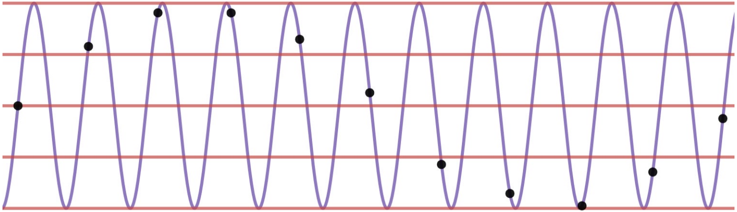
What's happening?



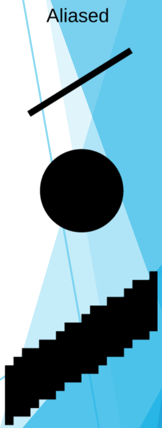


Credit: S Seitz, R Urtasun

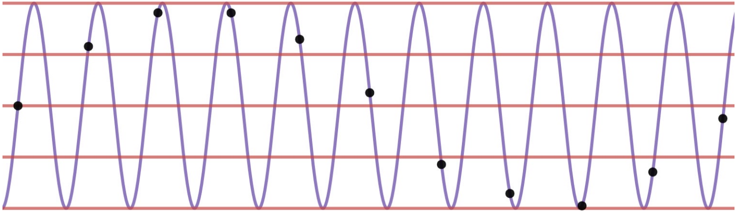
Aliasing



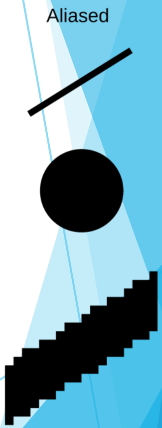
- Occurs when your sampling rate is not high enough to capture the amount of detail in your image.
- To do sampling right, need to understand the structure of your signal/image.
- The minimum sampling rate is called the **Nyquist rate.**



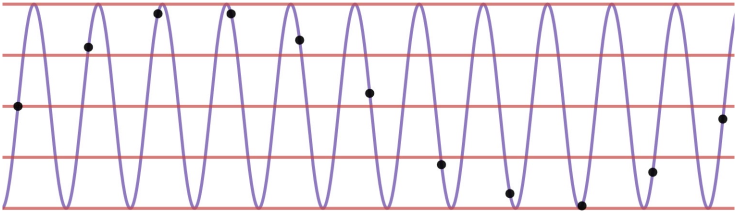
Aliasing

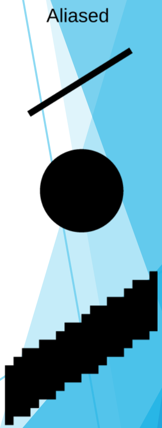


- Occurs when your sampling rate is not high enough to capture the amount of detail in your image.
- To do sampling right, need to understand the structure of your signal/image.
- The minimum sampling rate is called the **Nyquist rate.**



Aliasing



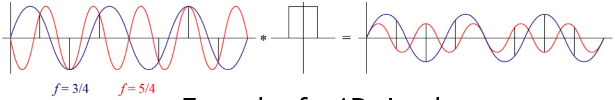


- Occurs when your sampling rate is not high enough to capture the amount of detail in your image.
- To do sampling right, need to understand the structure of your signal/image.
- The minimum sampling rate is called the **Nyquist rate.**

Aliasing: Problems

Shannon's Sampling Theorem shows that the minimum sampling is:

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



Example of a 1D signal

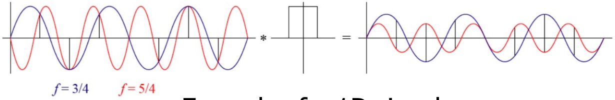
Examples

- Image
  - Striped shirt's pattern look weird on screen.
- Video
  - **Wagon Wheel effect:** Wheels spins in the opposite direction at high speed.
- Graphics
  - Checkerboards disintegrate in ray tracing.

Aliasing: Problems

Shannon's Sampling Theorem shows that the minimum sampling is:

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



Example of a 1D signal

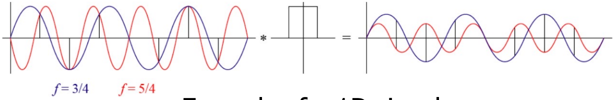
Examples

- Image
  - Striped shirt's pattern look weird on screen.
- Video
  - **Wagon Wheel effect:** Wheels spins in the opposite direction at high speed.
- Graphics
  - Checkerboards disintegrate in ray tracing.

Aliasing: Problems

Shannon's Sampling Theorem shows that the minimum sampling is:

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



Example of a 1D signal

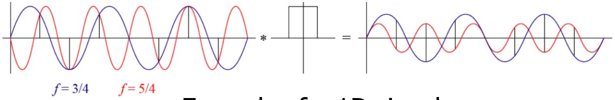
Examples

- Image
  - Striped shirt's pattern look weird on screen.
- Video
  - **Wagon Wheel effect:** Wheels spins in the opposite direction at high speed.
- Graphics
  - Checkerboards disintegrate in ray tracing.

Aliasing: Problems

Shannon's Sampling Theorem shows that the minimum sampling is:

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



Example of a 1D signal

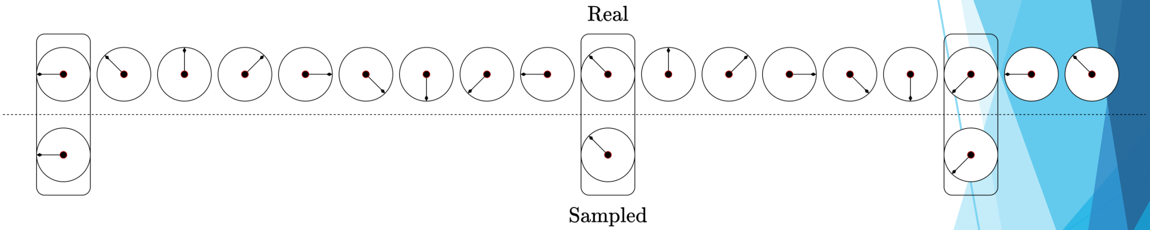
Examples

- Image
  - Striped shirt's pattern look weird on screen.
- Video
  - **Wagon Wheel effect:** Wheels spins in the opposite direction at high speed.
- Graphics
  - Checkerboards disintegrate in ray tracing.

Aliasing: Image

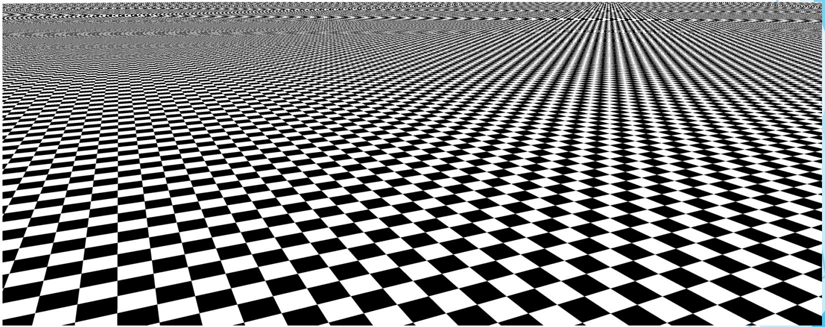
Striped shirt's pattern look weird on screen.

Aliasing : Video



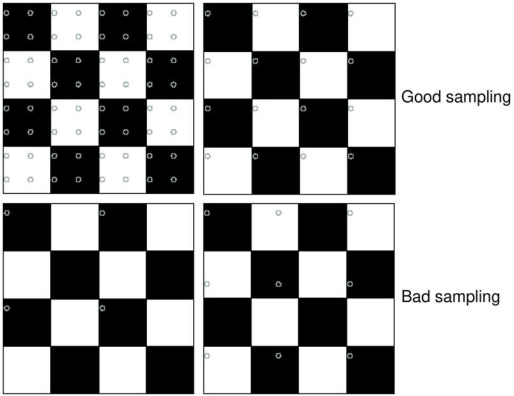
Wagon Wheel effect: Wheels spins in the opposite direction at high speed.

Aliasing: Graphics



Checkerboards disintegrate in ray tracing.

Aliasing: Nyquist Limit 2D example

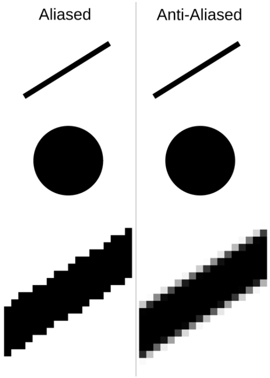


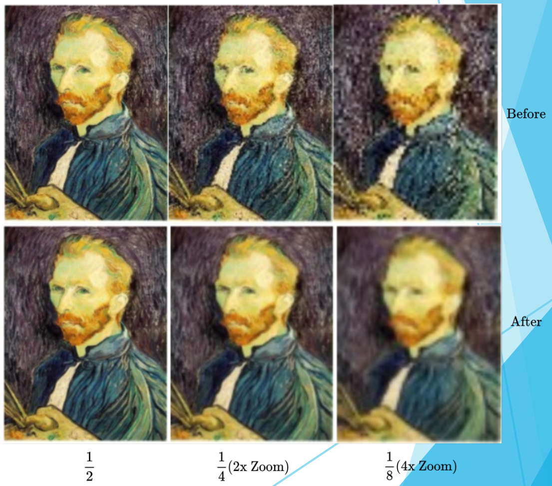
Credit: S Seitz, R Urtasun

Anti-aliasing

Anti-aliasing

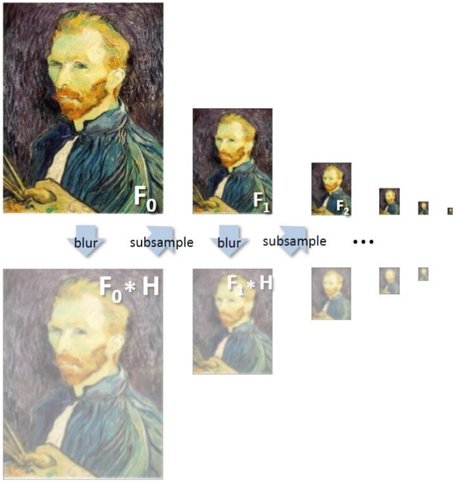
Example: Gaussian Pre-filtering





Credit: N Snavely, R Urtasun

Subsampling with Gaussian Pre-filtering



Credit: N Snavely, R Urtasun

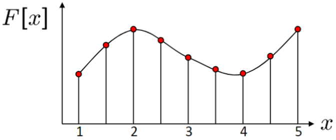
Upsampling

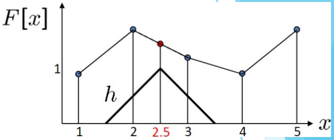
x

$10x$

How to go from left to right? **Interpolation**. Simple method: Repeat each row and column 10 times (Nearest Neighbour Interpolation).

Interpolation





Recall how a digital image is formed,

$$F[x, y] = quantize\{f(xd, yd)\}$$

- It is a discrete point-sampling of a continuous function.
- If we could somehow reconstruct the original function, any new image could be generated, at any resolution and scale.

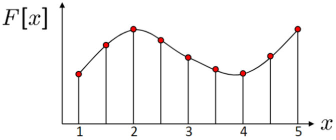
What if we don't know f?

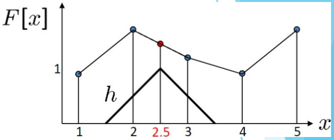
- Guess an approximation: Can be done in a principled way via filtering.
- Convert F to a continuous function:

$$f_F(x) = \begin{cases} F(\frac{x}{d}) & \text{if } \frac{x}{d} \text{ is an integer} \\ 0 & \text{otherwise} \end{cases}$$

- Reconstruct: $\hat{f} = h * f_F$

Interpolation





Recall how a digital image is formed,

$$F[x, y] = quantize\{f(xd, yd)\}$$

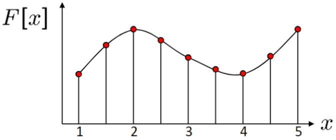
- It is a discrete point-sampling of a continuous function.
- If we could somehow reconstruct the original function, any new image could be generated, at any resolution and scale.

- What if we don't know f?
  - Guess an approximation: Can be done in a principled way via filtering.
  - Convert F to a continuous function:

$$f_F(x) = \begin{cases} F\left(\frac{x}{d}\right) & \text{if } \frac{x}{d} \text{ is an integer} \\ 0 & \text{otherwise} \end{cases}$$

- Reconstruct: $\hat{f} = h * f_F$

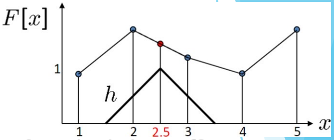
Interpolation



Recall how a digital image is formed,

$$F[x, y] = quantize\{f(xd, yd)\}$$

- It is a discrete point-sampling of a continuous function.
- If we could somehow reconstruct the original function, any new image could be generated, at any resolution and scale.

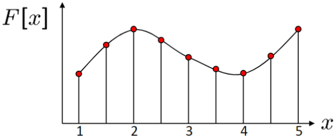


- What if we don't know f?
  - Guess an approximation: Can be done in a principled way via filtering.
  - Convert F to a continuous function:

$$f_F(x) = \begin{cases} F\left(\frac{x}{d}\right) & \text{if } \frac{x}{d} \text{ is an integer} \\ 0 & \text{otherwise} \end{cases}$$

- Reconstruct: $\hat{f} = h * f_F$

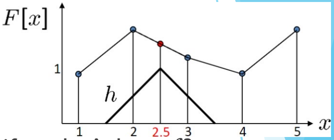
Interpolation



Recall how a digital image is formed,

$$F[x, y] = quantize\{f(xd, yd)\}$$

- It is a discrete point-sampling of a continuous function.
- If we could somehow reconstruct the original function, any new image could be generated, at any resolution and scale.

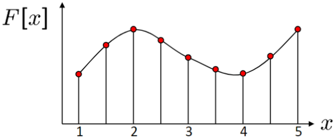


- What if we don't know f?
  - Guess an approximation: Can be done in a principled way via filtering.
  - Convert F to a continuous function:

$$f_F(x) = \begin{cases} F\left(\frac{x}{d}\right) & \text{if } \frac{x}{d} \text{ is an integer} \\ 0 & \text{otherwise} \end{cases}$$

- Reconstruct: $\hat{f} = h * f_F$

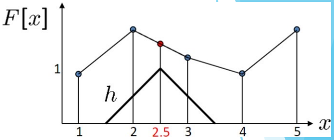
Interpolation



Recall how a digital image is formed,

$$F[x, y] = quantize\{f(xd, yd)\}$$

- It is a discrete point-sampling of a continuous function.
- If we could somehow reconstruct the original function, any new image could be generated, at any resolution and scale.

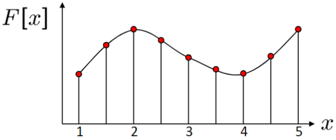


- What if we don't know f?
  - Guess an approximation: Can be done in a principled way via filtering.
  - Convert F to a continuous function:

$$f_F(x) = \begin{cases} F\left(\frac{x}{d}\right) & \text{if } \frac{x}{d} \text{ is an integer} \\ 0 & \text{otherwise} \end{cases}$$

  - Reconstruct: $\hat{f} = h * f_F$

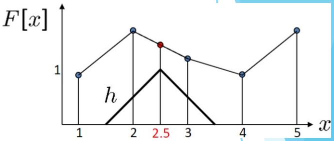
Interpolation



Recall how a digital image is formed,

$$F[x, y] = quantize\{f(xd, yd)\}$$

- It is a discrete point-sampling of a continuous function.
- If we could somehow reconstruct the original function, any new image could be generated, at any resolution and scale.

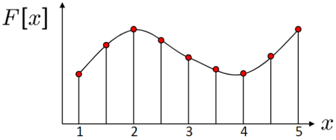


- What if we don't know f?
  - Guess an approximation: Can be done in a principled way via filtering.
  - Convert F to a continuous function:

$$f_F(x) = \begin{cases} F\left(\frac{x}{d}\right) & \text{if } \frac{x}{d} \text{ is an integer} \\ 0 & \text{otherwise} \end{cases}$$

- Reconstruct: $\hat{f} = h * f_F$

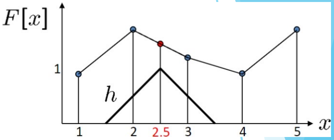
Interpolation



Recall how a digital image is formed,

$$F[x, y] = quantize\{f(xd, yd)\}$$

- It is a discrete point-sampling of a continuous function.
- If we could somehow reconstruct the original function, any new image could be generated, at any resolution and scale.

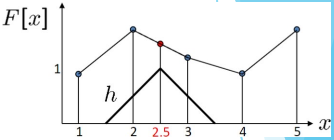


- What if we don't know f?
  - Guess an approximation: Can be done in a principled way via filtering.
  - Convert F to a continuous function:

$$f_F(x) = \begin{cases} F\left(\frac{x}{d}\right) & \text{if } \frac{x}{d} \text{ is an integer} \\ 0 & \text{otherwise} \end{cases}$$

- Reconstruct: $\hat{f} = h * f_F$

Interpolation as Convolution



- To **interpolate** (or upsample) an image to a higher resolution, we need an **interpolation kernel** with which to **convolve** the image:

$$g(i,j) = \sum_{k,L} f(k,l)h(i - rk, j - rl)$$

  Above formula similar to discrete convolution[a], except that we replace k and l in $h(\cdot)$ with rk and rl, where r is the upsampling rate.

- Linear interpolator (corresponding to *tent kernel*) produces interpolating piecewise linear curves.
- More complex kernels e.g., B-splines.

[a] $g = f * h \implies g(i,j) = \sum_{k,l} f(k,L)h(i - k, j - L)$

Interpolation as Convolution

- To **interpolate** (or upsample) an image to a higher resolution, we need an **interpolation kernel** with which to **convolve** the image:

$$g(i,j) = \sum_{k,l} f(k,l)h(i - rk, j - rl)$$

Above formula similar to discrete convolution[a], except that we replace k and l in $h(\cdot)$ with rk and rl, where r is the upsampling rate.

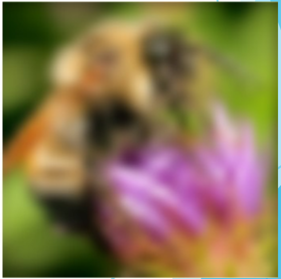
- Linear interpolator (corresponding to *tent kernel*) produces interpolating piecewise linear curves.
- More complex kernels e.g., B-splines.

[a] $g = f * h \implies g(i,j) = \sum_{k,l} f(k,l)h(i - k, j - l)$

Examples

Original Image:

Upsampled Images:



Left to right: Nearest Neighbour Interpolation, Bilinear Interpolation, Bicubic Interpolation.

Interpolation and Decimation

Interpolation

To **interpolate** (or upsample) an image to a higher resolution, we need an **interpolation kernel** with which to convolve the image (r is upsampling rate):

$$g(i,j) = \sum_{k,L} f(k,l)h(i - rk, j - rl)$$

Decimation (Sub-sampling)

To **decimate** (or sub-sample) an image to a lower resolution, we need an **decimation kernel** with which to convolve the image (r is downsampling rate):

$$g(i,j) = \sum_{k,L} f(k,l)h(i - \frac{k}{r}, j - \frac{l}{r})$$

Homework

Readings

Chapter 3 (§3.5.1-3.5.2), Szeliski, *Computer Vision: Algorithms and Applications*

Chapter 7 (§7.4), Forsyth and Ponce, *Computer Vision: A Modern Approach*