Discrete-time image processing requires the representation of images by a sampled array on a 2-D Lattice. There are several practical methods of sampling. Modern devices, such as charged-coupled devices, contain an array of photodetectors, and a set of electronic switches:

where the CCD array has, in most cases, an array of detectors

Common resolution range from $(256)^2 \leftrightarrow (2000)^2$
Although images are not generally band limited, we can approximately represent them by bandlimited signals. Let \( g(x,y) \) be a 2-D continuous image. Rectangular sampling is modelled as:

\[
g_s(x,y) = \sum_{m} \sum_{n} g(mX,nY) \delta(x-mX,y-nY)
\]

Assuming \( g(x,y) \) is band limited, \( G(u,v) = 0 \) for \( u > B_x \) and \( v > B_y \)
The Fourier transform of the sampled image is

\[ F\{g_s(x, y)\} = F \left\{ g(x, y) \sum_m \sum_n \delta(x - mX, y - uY) \right\} = G(u, v) * \frac{1}{X} \frac{1}{Y} \sum_m \sum_n \delta \left( u - \frac{m}{X}, v - \frac{n}{Y} \right) \]
The Fourier transform of the sampled image is

\[
F\{g_s(x, y)\} = F \left\{ g(x, y) \sum_m \sum_n \delta(x - mX, y - uY) \right\} = G(u, v) * \frac{1}{XY} \sum_m \sum_n \delta \left( u - \frac{m}{X}, v - \frac{n}{Y} \right) = \frac{1}{X} \sum_m \sum_n G \left( u - \frac{m}{X}, v - \frac{n}{Y} \right) = \frac{1}{XY} \left( \text{rep}_\frac{1}{X} \frac{1}{Y} (G(u, v)) \right)
\]
Representing $G(u, v)$ as

then, the spectra of $g_s(x, y)$ is seen as the replication of $G(u, v)$:
To avoid aliasing, we require \( B_x < \frac{1}{2X} \) and \( B_y < \frac{1}{2Y} \).

In order to reconstruct the continuous signal, filter \( G_s(u, v) \) with the LPF (ideal):

\[
H(u, v) = \begin{cases} 
XY & |v| < \frac{1}{2Y}, u < \frac{1}{2X} \\
0 & \text{else}
\end{cases}
\]

The space domain filter is obtained as :

\[
h(x, y) = \text{sinc}\left[\frac{x}{X}, \frac{y}{Y}\right].
\]
Hence,

\[
\hat{g}(x, y) = g_s(x, y) \ast \text{sinc}\left[\frac{x}{X}, \frac{y}{Y}\right] \\
= \left[\sum_m \sum_n g(mX, nY) \delta(x - mX, y - uY)\right] \ast \text{sinc}\left[\frac{x}{X}, \frac{y}{Y}\right] \\
= \sum_m \sum_n g(mX, nY) \text{sinc}\left(\frac{x - mX}{X}, \frac{y - nY}{Y}\right)
\]
Hence,

\[ \hat{g}(x, y) = g_s(x, y) * \text{sinc}\left(\frac{x}{X}, \frac{y}{Y}\right) \]

\[ = \left[ \sum_m \sum_n g(mX, nY) \delta(x - mX, y - uY) \right] * \text{sinc}\left(\frac{x}{X}, \frac{y}{Y}\right) \]

\[ \hat{g}(x, y) = \sum_m \sum_n g(mX, nY) \text{sinc}\left(\frac{x - mX}{X}, \frac{y - nY}{Y}\right) \]

The ideal LPF is however very difficult to obtain, hence, other filters are generally designed. For instance, if we are to obtain a continuous image by projecting into a CRT display, we are effectively replacing the 2-D function by a gaussian function

\[ p(x, y) = \frac{1}{2\pi\sigma^2} \exp\left[\frac{-(x^2 + y^2)}{2\sigma^2}\right] \]

Hence the reconstructed image is:

\[ \hat{g}(x, y) = \sum_m \sum_n g(mX, nY) \left(\frac{1}{2\pi\sigma^2}\right) \exp\left[\frac{-(x - mX)^2 - (y - nY)^2}{2\sigma^2}\right] \]
The effect is the introduction of aliasing. To illustrate consider a slice of $G_s(u,v)$ and $\hat{G}(u,v)$.
The effect is the introduction of aliasing. To illustrate consider a slice of $G_s(u, v)$ and $\hat{G}(u, v)$.

These concepts are further discussed, in the interpolation and decimation of images, where the physical size of the images is varied by keeping the same spatial resolution. These are very important issue in commercial applications.
Aliasing

**FIGURE 4.15**
Two-dimensional Fourier transforms of (a) an oversampled, and (b) under-sampled band-limited function.

Footprint of an ideal lowpass (box) filter
Example of Aliasing

- Example of aliasing error in a sampled image
- Spurious spatial frequency components
- It creates low-spatial-frequency components in the reconstruction
- Known as moiré patterns
Aliasing

FIGURE 4.16 Aliasing in images. In (a) and (b), the lengths of the sides of the squares are 16 and 6 pixels, respectively, and aliasing is visually negligible. In (c) and (d), the sides of the squares are 0.9174 and 0.4798 pixels, respectively, and the results show significant aliasing. Note that (d) masquerades as a “normal” image.
Aliasing

**FIGURE 4.17** Illustration of aliasing on resampled images. (a) A digital image with negligible visual aliasing. (b) Result of resizing the image to 50% of its original size by pixel deletion. Aliasing is clearly visible. (c) Result of blurring the image in (a) with a $3 \times 3$ averaging filter prior to resizing. The image is slightly more blurred than (b), but aliasing is not longer objectionable. (Original image courtesy of the Signal Compression Laboratory, University of California, Santa Barbara.)
Aliasing

Courtesy of Scientific Volume Imaging - http://www.svi.nl/antialiasing
Aliasing

**Figure 4.20**
Examples of the moiré effect. These are ink drawings, not digitized patterns. Superimposing one pattern on the other is equivalent mathematically to multiplying the patterns.
FIGURE 4.21
A newspaper image of size 246 × 168 pixels sampled at 75 dpi showing a moiré pattern. The moiré pattern in this image is the interference pattern created between the ±45° orientation of the halftone dots and the north-south orientation of the sampling grid used to digitize the image.
Aliasing

**FIGURE 4.22**
A newspaper image and an enlargement showing how halftone dots are arranged to render shades of gray.
Moiré Pattern Effect

- Each grate is periodic
- Their superposition breaks the periodicity
- The problem is common in scanning of printed material
- Periodicities do not line up causing aliasing
Aliasing errors can be reduced by low-pass filtering before sampling.
Aliasing error can be reduced by pre-sampling filtering.

However, any attenuation within the passband represents a loss of resolution.

Result: trade-off between sampled image resolution and aliasing error.

Spatial filtering can be done by passing light through a lens with restricted aperture.

Even if perfectly focused lens produce blurring because of the diffraction limit of its aperture.

Example of lens misfocus.
Aliasing

Courtesy of Scientific Volume Imaging - http://www.svi.nl/antialiasing

Chapter IV(b)

Gonzalo R. Arce
Spring, 2013

20 / 66
Aliasing
Implementation Techniques

Image reconstruction by subscanning
Common Pulse Models

- Rectangular

\[ p(x, y) = \begin{cases} \frac{1}{T^2} & |x|, |y| \leq \frac{T}{2} \\ 0 & \text{otherwise} \end{cases} \]

\[ P(u, v) = T \text{sinc}(uT, vT) \]

- Gaussian

\[ p(x, y) = \frac{1}{2\pi\sigma^2} \exp \left\{ -\frac{x^2 + y^2}{2\sigma} \right\} \]

\[ P(u, v) = \exp \left\{ -\frac{((u)^2 + (v)^2)\sigma}{2} \right\} \]
One Dimensional Interpolation Functions

(a) Sinc

(b) Square

(c) Triangle (two squares convolved)

(d) Bell (three squares convolved)

(e) Cubic B-spline (four squares convolved)

(f) Gaussian
One Dimensional Interpolation Waveforms

- Zooming is oversampling
- Shrinking is undersampling
- Both involve resampling
Two Dimensional Linear Interpolation

(a) Piecewise linear interpolation

(b) Bilinear interpolation
Zooming Example

Top row: images zoomed from $128 \times 128$, $64 \times 64$, and $32 \times 32$ pixels to $1024 \times 1024$ pixels, using nearest neighbor gray-level interpolation. Bottom row: same sequence, but using bilinear interpolation.
Image Interpolation

- Given a discrete-space signal, estimate the continuous signal from which it was sampled.
Image Interpolation

- Given a discrete-space signal, estimate the continuous signal from which it was sampled.
- Most common uses in image processing are
  - Resizing an image. For example, zooming by some factor.
Image Interpolation

- Given a discrete-space signal, estimate the continuous signal from which it was sampled.
- Most common uses in image processing are
  - Resizing an image. For example, zooming by some factor.
  - Image rotation. Pixels in rotated coordinates do not usually end up at integral \((n_1, n_2)\).
Given a discrete-space signal, estimate the continuous signal from which it was sampled.

Most common uses in image processing are:
- Resizing an image. For example, zooming by some factor.
- Image rotation. Pixels in rotated coordinates do not usually end up at integral \((n_1, n_2)\).
- Motion compensation for time interpolation of image sequences.
Ideal Interpolation Function

For a signal sampled at the Nyquist frequency (i.e., band-limited)

\[ f_c(x, y) = \sum_{n_1=-\infty}^{\infty} \sum_{n_2=-\infty}^{\infty} f(n_1, n_2) \phi_{n_1, n_2}(x, y) \]

where the interpolating function \( \phi_{n_1, n_2}(x, y) \) is given by

\[ \phi_{n_1, n_2}(x, y) = \text{sinc}(\frac{x - n_1 T_1}{T_1}, \frac{y - n_2 T_2}{T_2}) \]
Nice theoretically, but problems practically.

- Requires every value of $f(n_1, n_2)$ to be used in interpolating $f_c(x, y)$ at a given point. ($\phi(x, y)$ is infinite).

- More practical approaches:
  - Choose spatially limited $\phi(x, y)$.
  - Bilinear interpolation.
  - Cubic spline interpolation.
Zero order interpolation (zero order hold)

You can also do this with mask:

- Take $n_1 \times n_2$ image:

$$\begin{pmatrix}
\_ & \_ & \_ & \_ & \_ \\
\_ & \_ & \_ & \_ & \_ \\
\_ & \_ & \_ & \_ & \_ \\
\_ & \_ & \_ & \_ & \_ \\
\_ & \_ & \_ & \_ & \_ \\
\end{pmatrix}$$

- Interlace with zeros:

$$\begin{pmatrix}
\_ & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\_ & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\_ & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\_ & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\end{pmatrix}$$

- Convolve with

$$H = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}$$
Bilinear Interpolation

First consider 1-D linear interpolation
\[
\frac{f(x_0) - f(n_1)}{x_0 - n_1 T_1} = \frac{f(n_1 + 1) - f(n_1)}{T_1}
\] (1)

\[
f(x_0) = f(n_1) + \left(\frac{x_0 - n_1 T_1}{T_1}\right) [f(n_1 + 1) - f(n_1)]
\] (2)

Define: \(\triangle x \equiv \frac{x_0 - n_1 T_1}{T_1}\)

\[
f(x_0) = f(n_1) + \triangle x [f(n_1 + 1) - f(n_1)]
\] (3)

\[
f(x_0) = (1 - \triangle x)f(n_1) + \triangle x f(n_1 + 1)
\] (4)

For \(T_1 n_1 \leq x_0 \leq T_1 (n_1 + 1)\)
For 2-D case:

You could consider interpolating first in $x$-direction, then in $y$:

- Interpolate along $x$ at $y = n_2T$:

$$f_1(x, n_2T) = (1 - \Delta_x)f(n_1, n_2) + \Delta_xf(n_1 + 1, n_2)$$ (5)

- Interpolate along $x$ at $y = (n_2 + 1)T$:

$$f_1(x, (n_2 + 1)T) = (1 - \Delta_x)f(n_1, n_2 + 1) + \Delta_xf(n_1 + 1, n_2 + 1)$$ (6)
Now interpolate along $y$ using (5) and (6)

$$f_{12}(x, y) = (1 - \triangle y)f_{1}(x, n_2 T) + \triangle y f_{1}(x, (n_2 + 1)T)$$  \hfill (7)
Alternatively, consider first interpolating along $y$, then $x$:

- Interpolate along $y$ at $x = n_1 T$:

$$f_2(x, n_2 T) = (1 - \triangle_y) f(n_1, n_2) + \triangle_y f(n_1, n_2 + 1) \quad (8)$$

- Interpolate along $y$ at $x = (n_1 + 1)T$

$$f_2((n_1 + 1) T, y) = (1 - \triangle_y) f(n_1 + 1, n_2) + \triangle_y f(n_1 + 1, n_2 + 1) \quad (9)$$

- Now interpolate along $x$ using (8) and (9)

$$f_{21}(x, y) = (1 - \triangle_x) f_2(n_1 T, y) + \triangle_x f_2((n_1 + 1)T, y) \quad (10)$$

Both of these give same final result.
\[ f_{12}(x,y) = (1 - \Delta y)(1 - \Delta x)f(n_1, n_2) + (1 - \Delta y)\Delta_x f(n_1 + 1, n_2) \]
\[ \quad + (1 - \Delta x)\Delta_y f(n_1, n_2 + 1) + \Delta_x \Delta_y f(n_1 + 1, n_2 + 1) \]

\[ f_{21}(x,y) = (1 - \Delta y)(1 - \Delta x)f(n_1, n_2) + (1 - \Delta x)\Delta_y f(n_1, n_2 + 1) \]
\[ \quad + (1 - \Delta y)\Delta_x f(n_1 + 1, n_2) + \Delta_x \Delta_y f(n_1 + 1, n_2 + 1) \]

\[ f(x,y) = (1 - \Delta y)(1 - \Delta x)f(n_1, n_2) + (1 - \Delta x)\Delta_y f(n_1, n_2 + 1) \]
\[ \quad + (1 - \Delta y)\Delta_x f(n_1 + 1, n_2) + \Delta_x \Delta_y f(n_1 + 1, n_2 + 1) \]

where \( \Delta_x \equiv (x - n_1 T_1)/T_1 \), \( \Delta_y \equiv (y - n_2 T_2)/T_2 \).
Example: Zoom by 2: \( x = n_1 T_1, n_1 T_1 + T_1 / 2, y = n_2 T_2, n_2 T_2 + T_2 / 2 \).

- when \( x = n_1 T, y = n_2 T_2 \) (on a grid point)
  \( \triangle x = \triangle y = 0 \), then, \( f(n_1 T_1, n_2 T_2) = f(n_1, n_2) \).

- when \( x = n_1 T + T / 2, y = n_2 T_2 \) (between \( x \), on \( y \))
  \( \triangle x = 1 / 2, \triangle y = 0 \), then,
  \[ f(n_1 T_1 + T_1 / 2, n_2 T_2) = 1 / 2 f(n_1, n_2) + 1 / 2 f(n_1 + 1, n_2). \]

\[
\begin{pmatrix}
\cdot & x & \cdot \\
\cdot & & \\
\cdot & x & \cdot \\
\end{pmatrix}
\]
when \( x = n_1 T, y = n_2 T_2 + T_2 / 2 \)
\( \triangle_x = 0, \triangle_y = 1/2, \) then,
\[
f(n_1 T_1, n_2 T_2 + T_2 / 2) = 1/2f(n_1, n_2) + 1/2f(n_1, n_2 + 1).
\]

when \( x = n_1 T + T_1 / 2, y = n_2 T_2 + T_2 / 2 \)
\( \triangle_x = 1/2, \triangle_y = 1/2, \) then,
\[
f(n_1 T_1 + T_1 / 2, n_2 T_2 + T_2 / 2) = 1/4(f(n_1, n_2) + f(n_1 + 1, n_2) + f(n_1, n_2 + 1) + f(n_1 + 1, n_2 + 1)).
\]
Can also do by a 2 zooming (with bilinear interpolation) as convolution.

Create a $2N_1 \times 2N_2$ 0-interlaced image.

$$
\begin{pmatrix}
  \cdot & 0 & \cdot & 0 & \cdot \\
  0 & 0 & 0 & 0 & 0 \\
  \cdot & 0 & \cdot & 0 & \cdot \\
  0 & 0 & 0 & 0 & 0 \\
  \cdot & 0 & \cdot & 0 & \cdot 
\end{pmatrix}
$$

Convolve with the kernel:

$$
H = 
\begin{pmatrix}
  1/4 & 1/2 & 1/4 \\
  1/2 & 1 & 1/2 \\
  1/4 & 1/2 & 1/4 
\end{pmatrix}
$$
Another way to do higher order interpolation:

- Interlace image with $p \, 0's$
  
  i.e. $p = 2$

\[
\begin{pmatrix}
  x & x & x \\
  x & x & x \\
  x & x & x \\
\end{pmatrix}
\]
Convolve new interlaced image with

\[ H = \begin{pmatrix} 1/4 & 1/2 & 1/4 \\ 1/2 & 1 & 1/2 \\ 1/4 & 1/2 & 1/4 \end{pmatrix} \]

\( p \) times.

This gives \( p^{th} \) order interpolation.
Images with a Billion Pixels

Courtesy of O. Cossairt
Images with a Billion Pixels

Why is there no gigapixel camera today?
Is it image sensor resolution?

Assume 1 micron pixels (Fife et al 08)

10 megapixel sensor
Is it image sensor resolution?

Assume 1 micron pixels (Fife et al 08)

1 gigapixel sensor

The limits are due to optics
Defining Optical Resolution

Spatial Resolution
\( \delta \): minimum resolvable spot size
Is resolution limited by diffraction?

\[
\delta_d \approx \frac{\lambda F}{\#}
\]
Compromise: trading off light

Resolution

Scale (M)

Microscope F/1 1mm FL

Wide Angle F/3 27mm FL

SLR Lens F/5 125mm FL

Telephoto F/10 1000mm FL

$R_{\text{tradeoff}}$ (Lohmann '89)

400 mm sensor

$10^9$ pixels
Compromise: trading off light

The F/22 90° FOV Assymagon Lens

Graham Flint courtesy of Wired.com
Proposed solution: computational imaging

Reduce complexity with computations

(Cathey and Dowski '96)
(Robinson et. al '09)
(Guichard et al. '09)
(Cossairt and Nayar '10)
Proposed solution: computational imaging
A ball lens gigapixel camera

15 x 15 Array of 5Mpix ½” Lumenera Sensors
Proof of concept

Ball Lens

Sensor

Pan/Tilt Motor
Proof of concept: image quality

Deblurred
A single element design

Parallel effort by DARPA MOSAIC Program led by D. Brady
(Brady and Hagen ‘09)(Marks and Brady ‘10)
A single element design

- Ball Lens
- Sensor Array
- Lens Array
Still Life (1.7 Gigapixels)

URL: http://gigapan.org/gigapans/0dca576c3a040561b4371cf1d92c93fe/
New York and New Jersey Skyline (1.4 Gigapixels)

- Statue of Liberty (~2.32km)
- Apartments (~860m)
- Person on boat deck (~860m)
- Empire State Building (~6.45km)

URL: http://gigapan.org/gigapans/7173ad0acace87100a3ca728d40a3772/
New York and New Jersey Skyline (1.4 Gigapixels)

- Sailboat (~1.61km)
- People on Boat (~860m)
- Flag on Brooklyn Bridge (~2.19km)
- Cars (~1.5km)

URL: http://gigapan.org/gigapans/7173ad0acace87100a3ca728d40a3772/