Digital Image

- A grid of squares, each of which contains a single color.
- Each square is called a pixel (for picture element).

Color images have 3 values per pixel; monochrome images have 1 value per pixel.
Pixels

- A digital image, $I$, is a mapping from a 2D grid of uniformly spaced discrete points, $\{p = (r,c)\}$, into a set of positive integer values, $\{I(p)\}$, or a set of vector values, e.g., $\{[R\ G\ B]^T(p)\}$.
- At each column location in each row of $I$ there is a value.
- The pair $(p, I(p))$ is called a “pixel” (for picture element).
Pixels

- \( p = (r,c) \) is the pixel location indexed by row, \( r \), and column, \( c \).
- \( I(p) = I(r,c) \) is the value of the pixel at location \( p \).
- If \( I(p) \) is a single number then \( I \) is monochrome.
- If \( I(p) \) is a vector (ordered list of numbers) then \( I \) has multiple bands (e.g., a color image).
Pixels

Pixel Location: \( p = (r, c) \)
Pixel Value: \( I(p) = I(r, c) \)
Pixels

Pixel: \([ p, I(p) ]\)

\[ p = (r, c) \]
\[ = (\text{row #, col #}) \]
\[ = (272, 277) \]

\[ I(p) = \begin{bmatrix} \text{red} \\ \text{green} \\ \text{blue} \end{bmatrix} = \begin{bmatrix} 12 \\ 43 \\ 61 \end{bmatrix} \]
Sampling and Quantization

real image  sampled  quantized  sampled & quantized
Sampling and Quantization

- Real image
- Sampled
- Quantized
- Sampled & Quantized
Sampling

$I_C(\rho, \chi)$

continuous image

$I_S(r,c)$

sampled image

\[ I_S(r,c) = \frac{1}{\Delta^2} \int_{r\Delta}^{(r+1)\Delta} \int_{c\Delta}^{(c+1)\Delta} I_C(\rho, \chi) \delta \rho \delta \chi \]

Take the average within each square.
Sampling

Take the average within each square.

\[ I_C(\rho, \chi) \]

\[ I_S(r, c) = \frac{1}{\Delta^2} \int_{r\Delta}^{(r+1)\Delta} \int_{c\Delta}^{(c+1)\Delta} I_C(\rho, \chi) \delta\rho \delta\chi \]

Continuous image

Sampled image
Sampling

$I_C(\rho, \chi)$

continuous image

$I_S(r, c)$

sampled image

\[ I_S(r, c) = \frac{1}{\Delta^2} \int_{r\Delta}^{(r+1)\Delta} \int_{c\Delta}^{(c+1)\Delta} I_C(\rho, \chi) \delta \rho \delta \chi \]
Sampling

\[ I_C(\rho, \chi) \]

continuous image

\[ I_S(r, c) = \frac{1}{\Delta^2} \int_{r\Delta}^{(r+1)\Delta} \int_{c\Delta}^{(c+1)\Delta} I_C(\rho, \chi) \delta\rho \delta\chi \]

sampled image

Take the average within each square.
Read a Truecolor Image into Matlab

```matlab
>> cd 'E:\images\Animals\People\Famous'
>> I = imread('Les_Boineoisie.jpg','jpg');
>> class(I)
ans =
uint8
>> size(I)
ans =
    600    1200     3
>> figure
>>
```
Read a Truecolor Image into Matlab

To get started, select MATLAB Help or Demos from the Help menu.

```matlab
>> cd 'E:\images\Animals\People\Famous'
>> I = imread('Les_Boingoisie.jpg','jpg');
>> class(I)
ans =
    uint8
>> size(I)
ans =
    600  1200  3
>> figure
>> image(I)
>> title('Les Boingoisie: The Boing-Boing Bloggers')
>> xlabel('Photo: Bart Nagel, 2006, www.bartnagel.com')
>>
```
To get started:

```matlab
>> cd 'E:\image
>> I = imread('lesboing.jpg');
>> class(I)
ans =
    uint8
>> size(I)
ans =
    600
>> figure
>> image(I)
>> title('Les Boingeoisie: The Boing-Boing Bloggers')
>> xlabel('Photo: Bart Nagel, 2006, www.bartnagel.com')
```

To get started:

```
>> cd 'E:\image'
>> I = imread('foo.png')
>> class(I)
ans =
    uint8
>> size(I)
ans =
    600
>> figure
>> image(I)
>> title('Les Boingeoisie: The Boing-Boing Bloggers')
>> xlabel('Photo: Bart Nagel, 2006, www.bartnagel.com')
```
Crop the Image

First, select a region using the magnifier.

left click here and hold

drag to here and release

Cut out a region from the image
Crop the Image

From this close-up we can estimate the coordinates of the region:

rows: about 125 to 425
cols: about 700 to 1050
To get started, select MATLAB Help or Demos from the Help menu.

```matlab
>> cd 'E:\images\Animals\People\Famous'
>> I = imread('Les_Boinggeoisie.jpg','jpg');
>> class(I)
ans =
uint8
>> size(I)
ans =
     600     1200     3
>> figure
>> image(I)
>> title('Les Boinggeoisie: The Boing-Boing Bloggers')
>> xlabel('Photo: Bart Nagel, 2006, www.bartnagel.com')
>> truecolor
>> J = I(125:425,700:1050,:);
>> figure
>> image(J)
>> truecolor
```
Crop the Image

To get started:
```matlab
>> cd 'E:\image
>> I = imread('Les Boingeoisie.jpg');
>> class(I)
ans =
uint8
>> size(I)
ans =
600
>> figure
>> image(I)
>> title('Les Boingeoisie: The Boing-Boing Bloggers')
>> truesize
>> J = I(125:42,150:625);
>> figure
>> image(J)
>> truesize
>> figure(1)
```

Bring it to the front using the figure command,
Crop the Image

To get started, select MATLAB Help or Demos from the Help menu.

```matlab
cd 'E:\images\Animals\People\Famous'
I = imread('Les_Boingeoisie.jpg','jpg');
class(I)
ans =
uint8
>> size(I)
ans =
   600   1200     3
>> figure
>> image(I)
>> title('Les Boingeoisie: The Boing-Boing Bloggers')
>> xlabel('Photo: Bart Nagel, 2006, www.bartnagel.com')
>> trueimage
>> J = I(125:425,700:1050,:);
>> figure
>> image(J)
>> trueimage
>> figure(1)
>> close
```
Read a Colormapped Image into Matlab

To get started, select MATLAB Help or Demos from the Help menu.

```matlab
>> cd 'D:\classes\EECE253\Fall 2006\graphics\matlab intro'
>> [I,cmap] = imread('Jim Woodring - PlusMinus.gif','gif');
>> figure
>> image(I)
>> class(I)
ans =
   uint8
>> size(I)
ans =
   383   533
>>
```
Read a Colormapped Image into Matlab

To get started, select MATLAB Help or Demos from the Help menu.

```
>> cd 'D:\classes\ECE253\Fall 2006\graphics\matlab intro'
>> [I,cmap] = imread('Jim Woodring - PlusMinus.gif','gif');
>> figure
>> image(I)
>> class(I)
ans =
  uint8
>> size(I)
ans =
  383   533
>> colormap(cmap)
>> title('Plus Minus');
>> xlabel('Jim Woodring (http://www.jimwoodring.com)
>> truesize
>>
```
Colormapped vs. Truecolor in Matlab

To get started, select MATLAB Help or Demos from the Help menu.

```matlab
>> cd 'D:\classes\EECE253\Fall 2006\graphics\matlab intro'
>> [I,cmap] = imread('Jim Woodring - PlusMinus.gif','gif');
>> figure
>> image(I)
>> class(I)
ans =
uint8
>> size(I)
ans =
383   533
>> colormap(cmap)
>> title('Plus Minus');
>> xlabel('Jim Woodring (http://www.jimwoodring.com')
>> truecolor
>> T = imread('Jim Woodring - PlusMinus.jpg','jpg');
>> figure
>> image(T)
>> truesize
>>
```
Colormap vs. Truecolor in Matlab

To get started,

```matlab
>> cd 'D:\classes\'nn>> [I,cmap] = imread('Jim Woodring - PlusMinus.jpg');nn>> figure
>> image(I)
>> class(I)
ans =
    uint8
>> size(I)
ans =
    383   533
>> colormap(cmap)
>> title('Plus Minus');nn>> xlabel('Jim Woodring (http://www.jimwoodring.com)'
>> truesize
>> T = imread('Jim Woodring - PlusMinus.jpg','jpg');nn>> figure
>> image(T)
>> truesize
>>
```
Colormapped vs. Truecolor in Matlab

Intensity values are integers between 0 and 255.

T(231,326,:) =

227
222
96

image class: uint8
image type: truecolor

row: 231

Intensity values are integers between 0 and 255.
Colormapped vs. Truecolor in Matlab

Intensity values are integers between 0 and 1.

Image class: double
Image type: truecolor

T(231,326,:) =

0.89
0.87
0.38

Row: 231
Col: 326
Number at pixel location is an index into a colormap.

Intensity values are integers between 0 and 1.

Intensity values are integers between 0 and 1.

Number at pixel location is an index into a colormap.

Image class: uint8
Image type: colormapped
How to Make Colormaps

This code, \((0:255)\), generates a 1 row by 256 element vector of class double that contains numbers 0 through 255 inclusive.

\[
\begin{bmatrix}
0 & 0.0039 & 0.0078 & 0.0118 & 0.0157 \\
0.9843 & 0.9882 & 0.9922 & 0.9961 & 1.0000
\end{bmatrix}
\]

This, \((0:255)\)' , has the same contents and class but is a 256 row by 1 column vector. The apostrophe (') is the matrix transpose operator.

This code, \(\text{ramp} = (0:255)'/255;\)

\[
\text{kcm} = \begin{bmatrix}
\text{ramp} & \text{ramp} & \text{ramp}
\end{bmatrix};
\]

\[
\text{rcm} = \begin{bmatrix}
\text{ramp} & \text{zeros}(256,2)
\end{bmatrix};
\]

\[
\text{gcm} = \begin{bmatrix}
\text{zeros}(256,1) & \text{ramp} & \text{zeros}(256,1)
\end{bmatrix};
\]

\[
\text{rcm} = \begin{bmatrix}
\text{zeros}(256,2) & \text{ramp}
\end{bmatrix};
\]

\[
\% \text{ apply one by selecting the figure}
\% \text{ then entering:}
\% \text{ colormap(kcm)}
\]

256 \times 3 matrix

red colormap: \(R = B = 0\);

green colormap: \(R = B = 0\);

blue colormap: \(R = G = 0\);

gray colormap: \(R(k)=G(k)=B(k)\)
R, G, & B bands of a truecolor image displayed with grayscale colormaps.

```matlab
>> I = imread('blue_grapes_sm.jpg','jpg');
>> Rd = I(:,:,1);
>> colormap(kcm);
>> Gn = I(:,:,2);
>> colormap(kcm);
>> Bl = I(:,:,3);
>> colormap(kcm);
```

R, G, & B bands of a truecolor image displayed with grayscale colormaps.
R, G, & B bands of a truecolor image displayed with grayscale colormaps.
R, G, & B bands of a truecolor image displayed with tinted colormaps

>> I = imread('blue_grapes_sm.jpg','jpg');
>> Rd = I(:,:,1);
>> colormap(rcm);

R, G, & B bands of a truecolor image displayed with tinted colormaps

>> Gn = I(:,:,2);
>> colormap(gcm);

>> Bl = I(:,:,3);
>> colormap(bcm);
R, G, & B bands of a truecolor image displayed with tinted colormaps.
R, G, & B bands of a truecolor image displayed with grayscale colormaps.
Saving Images as Files

Assuming that 'I' contains the image of the correct class, that 'cmap' is a colormap, and that 'image_name' is the file-name that you want.
Double Exposure: Adding Two Images

Jim Woodring - Bumperillo

Mark Rayden - The Ecstasy of Cecelia

Rayden Woodring - The Ecstasy of Bumperillo (?)
Double Exposure: Adding Two Images

>> cd 'D:\Classes\EECE253\Fall 2006\Graphics\matlab intro'
>> JW = imread('Jim Woodring - Bumperillo.jpg','jpg');
>> figure
>> image(JW)
>> truesize
>> title('Bumperillo')
>> xlabel('Jim Woodring')
>> MR = imread('Mark Ryden - The Ecstasy of Cecelia.jpg','jpg');
>> figure
>> image(MR)
>> truesize
>> title('The Ecstasy of Cecelia')
>> xlabel('Mark Ryden')
>> [RMR,CMR,DMR] = size(MR);
>> [RJW,CJW,DJW] = size(JW);
>> rb = round((RJW-RMR)/2);
>> cb = round((CJW-CMR)/2);
>> JWplusMR = uint8((double(JW(rb:(rb+RMR-1),cb:(cb+CMR-1),:))+double(MR))/2);
>> figure
>> image(JWplusMR)
>> truesize
>> title('The Ecstasy of Bumperillo')
>> xlabel('Jim Woodring + Mark Ryden')
Double Exposure: Adding Two Images

```matlab
>> cd 'D:\Classes\EECE253\Fall 2006\Graphics\matlab intro'
>> JW = imread('Jim Woodring - Bumperillo.jpg','jpg');
>> figure
>> image(JW)
>> trueSize
>> title('Bumperillo')
>> xlabel('Jim Woodring')
>> MR = imread('Mark Ryden - The Ecstasy of Cecelia.jpg','jpg');
>> figure
>> image(MR)
>> trueSize
>> title('The Ecstasy of Cecelia')
>> xlabel('Mark Ryden')
>> [RMR,CMR,DMR] = size(MR);
>> [RJW,CJW,DJW] = size(JW);
>> rb = round((RJW-RMR)/2);
>> cb = round((CJW-CMR)/2);
>> JWplusMR = uint8((double(JW(rb:(rb+RMR-1),cb:(cb+CMR-1),:))+double(MR))/2);
>> figure
>> image(JWplusMR)
>> trueSize
>> title('The Ecstasy of Bumperillo')
>> xlabel('Jim Woodring + Mark Ryden')
```

**Example Matlab Code**

Cut a section out of the middle of the larger image the same size as the smaller image.
Double Exposure: Adding Two Images

```matlab
>> cd 'D:\Classes\EECE253\Fall 2006\Graphics\matlab intro'
>> JW = imread('Jim Woodring - Bumperillo.jpg','jpg');
>> figure
>> image(JW)
>> truesize
>> title('Bumperillo')
>> xlabel('Jim Woodring')
>> MR = imread('Mark Ryden - The Ecstasy of Cecelia.jpg','jpg');
>> figure
>> image(MR)
>> truesize
>> title('The Ecstasy of Cecelia')
>> xlabel('Mark Ryden')
>> [RMR,CMR,DMR] = size(MR);
>> [RJW,CJW,DJW] = size(JW);
>> rb = round((RJW-RMR)/2);
>> cb = round((CJW-CMR)/2);
>> JWplusMR = uint8((double(JW(rb:(rb+RMR-1),cb:(cb+CMR-1),:))+double(MR))/2);
>> figure
>> image(JWplusMR)
>> truesize
>> title('The Ecstasy of Bumperillo')
>> xlabel('Jim Woodring + Mark Ryden')
```

Note that the images are averaged, pixelwise.
Double Exposure: Adding Two Images

```matlab
>> cd 'D:\Classes\EECE253\Fall 2006\Graphics\matlab intro'
>> JW = imread('Jim Woodring - Bumperillo.jpg','jpg');
>> figure
>> image(JW)
>> truesize
>> title('Bumperillo')
>> xlabel('Jim Woodring')
>> MR = imread('Mark Ryden - The Ecstasy of Cecelia.jpg','jpg');
>> figure
>> image(MR)
>> truesize
>> title('The Ecstasy of Cecelia')
>> xlabel('Mark Ryden')
>> [RMR,CMR,DMR] = size(MR);
>> [RJW,CJW,DJW] = size(JW);
>> rb = round((RJW-RMR)/2);
>> cb = round((CJW-CMR)/2);
>> JWplusMR = uint8((double(JW(rb:(rb+RMR-1),cb:(cb+CMR-1),:)) + double(MR))/2);
>> figure
>> image(JWplusMR)
>> truesize
>> title('The Ecstasy of Bumperillo')
>> xlabel('Jim Woodring + Mark Ryden')
```
Intensity Masking: Multiplying Two Images

Jim Woodring - Bumperillo

Mark Rayden - The Ecstasy of Cecelia

Rayden Woodring - Bumperillo Ecstasy (?)
Intensity Masking: Multiplying Two Images

```matlab
>> JW = imread('Jim Woodring - Bumperillo.jpg','jpg');
>> MR = imread('Mark Ryden - The Ecstasy of Cecelia.jpg','jpg');
>> [RMR,CMR,DMR] = size(MR);
>> [RJW,CJW,DJW] = size(JW);
>> rb = round((RJW-RMR)/2);
>> cb = round((CJW-CMR)/2);
>> JWplusMR = uint8((double(JW(rb:(rb+RMR-1),cb:(cb+CMR-1),:))+double(MR))/2);
>> figure
>> image(JWplusMR)
>> truesize
>> title('The Extacy of Bumperillo')
>> xlabel('Jim Woodring + Mark Ryden')
>> JWtimesMR = double(JW(rb:(rb+RMR-1),cb:(cb+CMR-1),:)).*double(MR);
>> M = max(JWtimesMR(:));
>> m = min(JWtimesMR(:));
>> JWtimesMR = uint8(255*(double(JWtimesMR)-m)/(M-m));
>> figure
>> image(JWtimesMR)
>> truesize
>> title('EcstasyBumperillo')
```
Intensity Masking: Multiplying Two Images

```matlab
>> JW = imread('Jim Woodring - Bumperillo.jpg','jpg');
>> MR = imread('Mark Ryden - The Ecstasy of Cecelia.jpg','jpg');
>> [RMR,CMR,DMR] = size(MR);
>> [RJW,CJW,DJW] = size(JW);
>> rb = round((RJW-RMR)/2);
>> cb = round((CJW-CMR)/2);
>> JWplusMR = uint8((double(JW(rb :(rb+RMR-1),cb:(cb+CMR-1),:)) + double(MR))/2);
>> figure
>> image(JWplusMR)
>> truesize
>> title('The Extacsy of Bumperillo')
>> xlabel('Jim Woodring + Mark Ryden')
```

```
>> JWtimesMR = double(JW(rb:(rb+RMR-1),cb:(cb+CMR-1),:)).*double(MR);
>> M = max(JWtimesMR(:));
>> m = min(JWtimesMR(:));
>> JWtimesMR = uint8(255*(double(JWtimesMR)-m)/(M-m));
>> figure
>> image(JWtimesMR)
>> truesize
>> title('EcstasyBumperillo')
```

Note that the images are multiplied, pixelwise.

Note how the image intensities are scaled back into the range 0-255.
Pixel Indexing in Matlab

“For” loops in Matlab are inefficient, whereas Matlab’s native indexing procedures are very fast.

Rather than

```matlab
for r = 1:R
    for c = 1:C
        J(r,c,:) = IP_Function(I(r,c,:));
    end
end
```

use, if possible

```matlab
J = IP_Function(I);
```

But, sometimes that is not possible. For example, if the output, \(J\), is decimated with respect to the input, \(I\), the above will not work (unless, of course, it is done within IP_function).
Pixel Indexing in Matlab

To decimate the above image by a factor of $n$, create a vector, $r$, that contains the index of every $n$th row, and a similar vector, $c$. 

$$r = 1:n:R;$$

$$c = 1:n:C;$$

$$I(r,:, :)$$

$$I(:,c,:)$$

Here, $n=3$ 

$$r = [1 4 7 10 13 16 19 22 25 28 31]$$

$$c = [1 4 7 10 13 16 19 22 25 28 31]$$
Pixel Indexing in Matlab

Then, vectors $r$ and $c$ used as index arguments for image $I$ select every $n$th column in every $n$th row.

This is called, 'vectorizing'.

Take the pixels indexed by both $r$ and $c$.

Here, $n=3$

$I(r,c)$
Pixel Indexing in Matlab

Here, \( n = 3 \)

\[
J = I(r,c,:);
\]

\[
r = 1:n:R;
\]
\[
c = 1:n:C;
\]

image, \( I \)
Pixel Indexing in Matlab

Indexing in Matlab is fully general.

If $I$ is $R \times C \times B$, vectors $r$ and $c$ can contain any numbers $1 \leq r_k \leq R$ and $1 \leq c_k \leq C$.

The numbers can be in any order and can be repeated within $r$ and $c$.

The result of $I(r,c)$ is an ordinal shuffling of the pixels from $I$ as indexed by $r$ and $c$.

Whenever possible, avoid using 'for' loops; vectorize instead.
Pixel Indexing in Matlab

Indexing in Matlab is fully general.

If $I$ is $R \times C \times B$, vectors $r$ and $c$ can contain any numbers $1 \leq r_k \leq R$ and $1 \leq c_k \leq C$.

The numbers can be in any order and can be repeated within $r$ and $c$.

The result of $I(r,c)$ is an ordinal shuffling of the pixels from $I$ as indexed by $r$ and $c$.

Whenever possible, avoid using 'for' loops; vectorize instead.
Pixel Indexing in Matlab

Indexing in Matlab is fully general.

If \( I \) is \( R \times C \times B \), vectors \( r \) and \( c \) can contain any numbers \( 1 \leq r_k \leq R \) and \( 1 \leq c_k \leq C \).

The numbers can be in any order and can be repeated within \( r \) and \( c \).

The result of \( I(r, c) \) is an ordinal shuffling of the pixels from \( I \) as indexed by \( r \) and \( c \).

Whenever possible, avoid using ‘for’ loops; vectorize instead.
Pixel Indexing in Matlab

Indexing in Matlab is fully general.

If \( I \) is \( R \times C \times B \), vectors \( r \) and \( c \) can contain any numbers \( 1 \leq r_k \leq R \) and \( 1 \leq c_k \leq C \).

The numbers can be in any order and can be repeated within \( r \) and \( c \).

The result of \( I(r, c) \) is an ordinal shuffling of the pixels from \( I \) as indexed by \( r \) and \( c \).

Whenever possible, avoid using 'for' loops; vectorize instead.
Pixel Indexing in Matlab

Indexing in Matlab is fully general.

If \( I \) is \( R \times C \times B \), vectors \( r \) and \( c \) can contain any numbers \( 1 \leq r_k \leq R \) and \( 1 \leq c_k \leq C \).

The numbers can be in any order and can be repeated within \( r \) and \( c \).

The result of \( I(r, c) \) is an ordinal shuffling of the pixels from \( I \) as indexed by \( r \) and \( c \).

Whenever possible, avoid using ‘for’ loops: vectorize instead.
Pixel Indexing in Matlab

Indexing in Matlab is fully general.

If $I$ is $R \times C \times B$, vectors $r$ and $c$ can contain any numbers $1 \leq r_k \leq R$ and $1 \leq c_k \leq C$.

The numbers can be in any order and can be repeated within $r$ and $c$.

The result of $I(r,c)$ is an ordinal shuffling of the pixels from $I$ as indexed by $r$ and $c$.

Whenever possible, avoid using ‘for’ loops; vectorize instead.