Part VI

images as matrices

eval function

IMAGES (1)

The philosophy:

Any two dimensional matrix can be easily transformed into an image.

If matrix is 40 x 40
--> 40 x 40 picture.

matrix(i,j) --> pixel(y,x)

where y=ith pixel on y scale
and x=jth pixel on x scale

IMAGES (1)

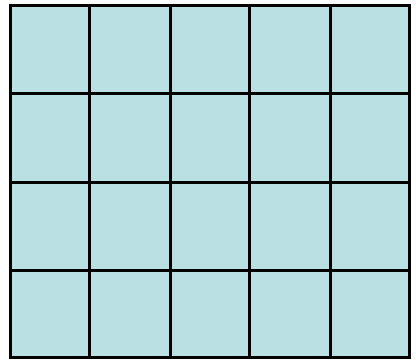
Example:

```
br = [ 0 0 0 0 0
       1 1 1 1 1
       0 0 0 0 0
       1 1 1 1 1]
```

IMAGES (1)

Example:

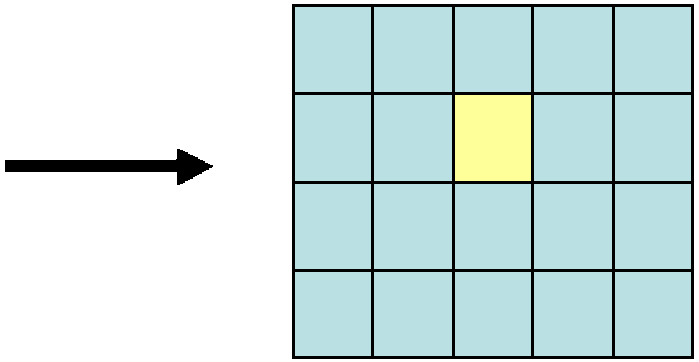
```
br = [ 0 0 0 0 0
       1 1 1 1 1
       0 0 0 0 0
       1 1 1 1 1]
```



IMAGES (1)

Example:

```
br = [ 0 0 0 0 0
       1 1 1 1 1
       0 0 0 0 0
       1 1 1 1 1]
```



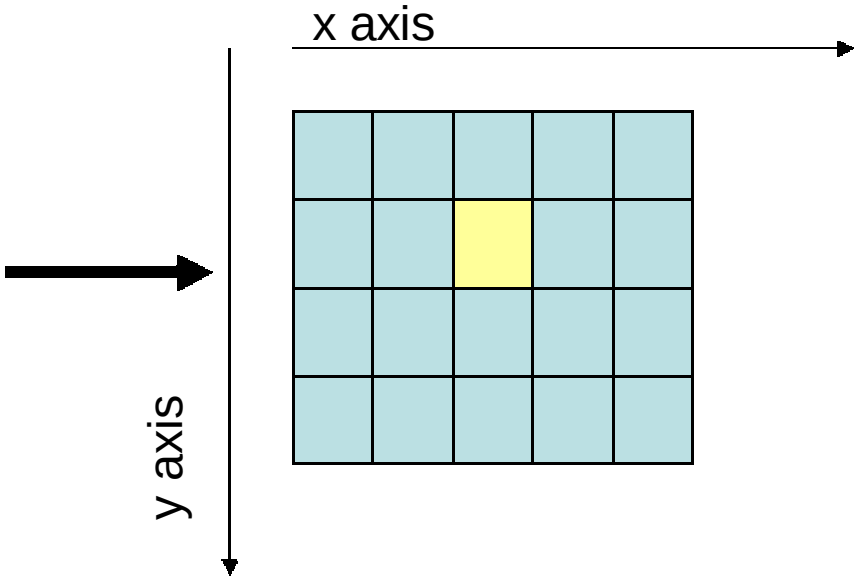
elements of br:

br(2,3) = 1

IMAGES (1)

Example:

```
br = [ 0 0 0 0 0
       1 1 1 1 1
       0 0 0 0 0
       1 1 1 1 1]
```



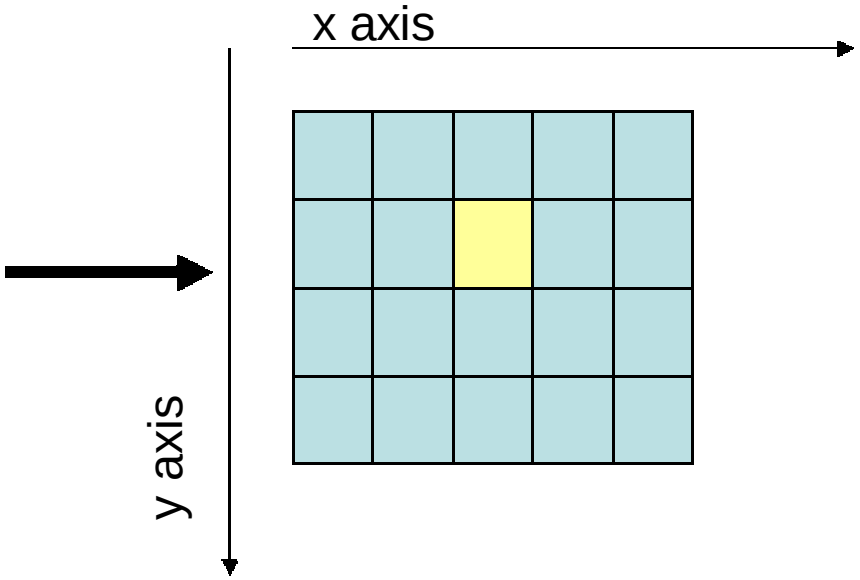
elements of br:

br(2,3) = 1

IMAGES (1)

Example:

```
br = [ 0 0 0 0 0
       1 1 1 1 1
       0 0 0 0 0
       1 1 1 1 1]
```



elements of br:

br(2,3) = 1

br(2,3) has coordinates

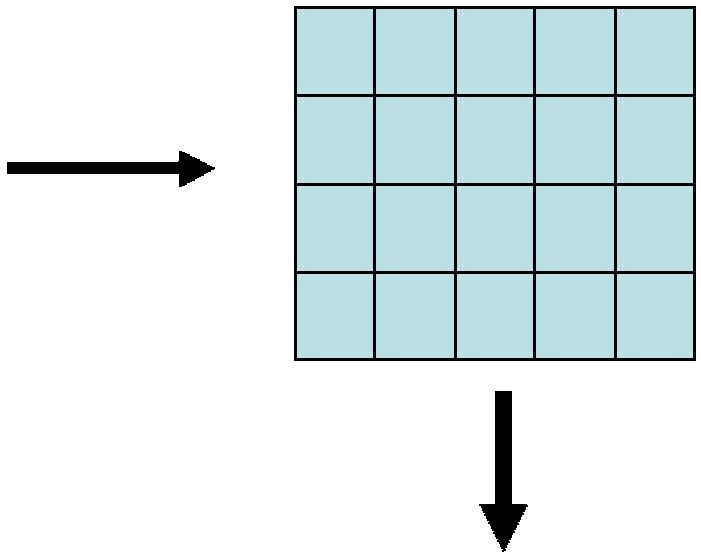
x=3

y=2

IMAGES (1)

Example:

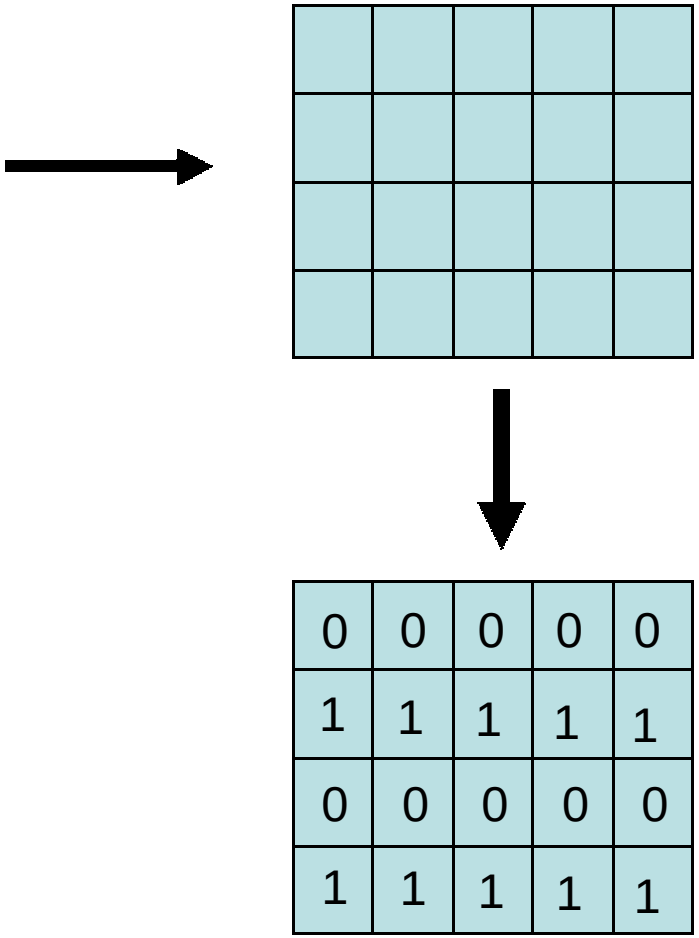
```
br = [ 0 0 0 0 0
       1 1 1 1 1
       0 0 0 0 0
       1 1 1 1 1]
```



IMAGES (1)

Example:

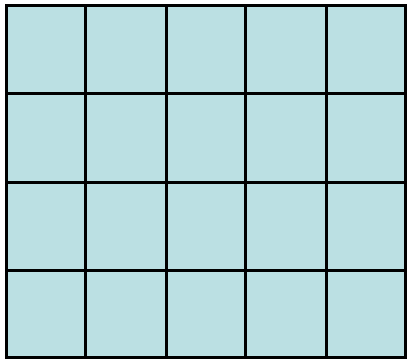
```
br = [ 0 0 0 0 0
       1 1 1 1 1
       0 0 0 0 0
       1 1 1 1 1]
```

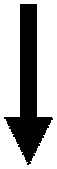


IMAGES (1)

Example:

```
br = [ 0 0 0 0 0
       1 1 1 1 1
       0 0 0 0 0
       1 1 1 1 1]
```





0	0	0	0	0
1	1	1	1	1
0	0	0	0	0
1	1	1	1	1

say…
0 = black

1 = red

IMAGES (1) p. 22

- **Let's play with Matlab's demo:**
- **Type:**

```
> clear all
> load durer                %check workspace.
> image(X)                  %what happened?
> colormap(map)             %ruminate on this…
> axis equal
> axis image   %same as equal, but image fits tightly
> axis off     %turns off tick marks
```

IMAGES (1) p. 22

- **Try:**

```
> X(1,1)
> max(max(X))     %returns largest value in matrix
> min(min(X))     %returns smallest value in matrix
```

- **so we have values between 1 and 128, each translated into a color through a colormap (aka Color Look Up Table).**
- **what are the dimensions of the variable map?**

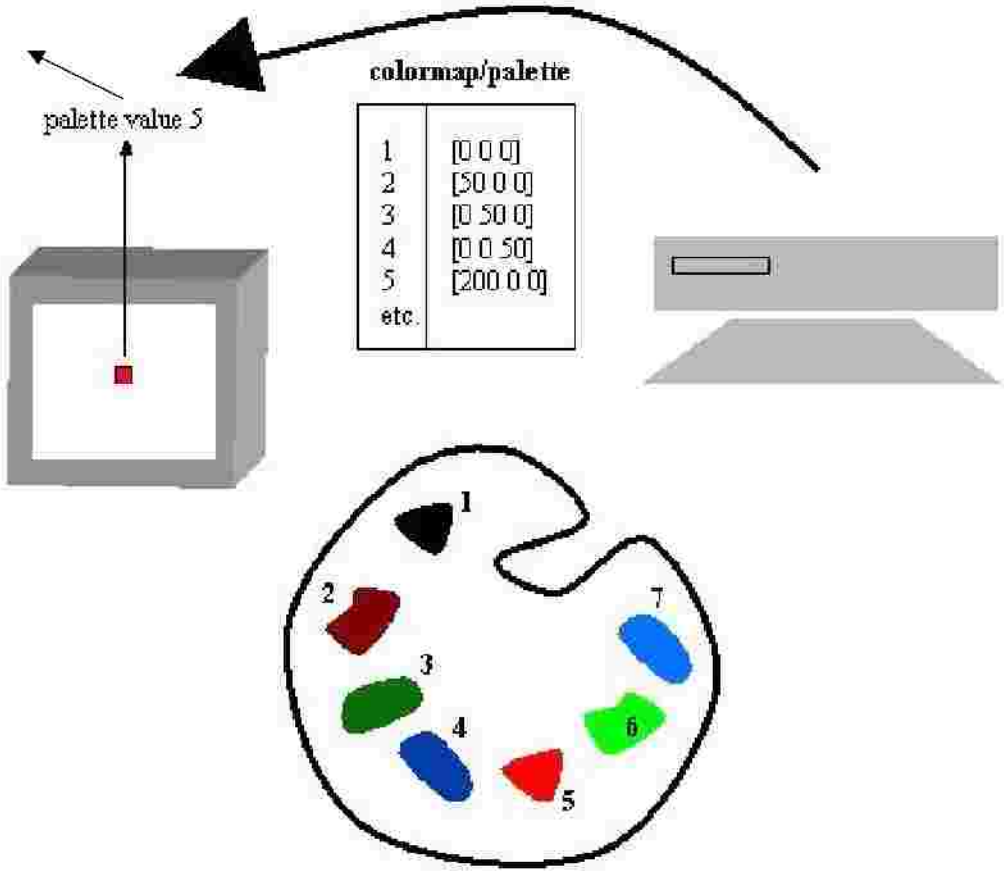
COLORMAPS p. 22

“A colormap consists of a set of entries defining color values. The colormap associated with a window is used to display the contents of the window; each pixel value indexes the colormap to produce an RGB value that drives the guns of a monitor. Depending on hardware limitations, one or more colormaps can be installed at one time so that windows associated with those maps display with true colors.”

COLORMAPS p. 22

RGB value

[200 0 0]



Two ways of changing color on the screen

1) Change the palette value of that pixel
2) Change the rgb triplet referenced by that palette value.

COLORMAPS p. 22

= SHORTCUTS to your 'kinda colors'.

How many colors in my computer?

- 8 bit color = 256 colors.
  (GIFs)
- 24 bit color = 8 bits/gun.
  256^3 = 16.7 million
- **30 bit integer color = 10 bits/gun.**
  (2^10)^3 = 1 billion colors.
- **32 bit: true color + alpha channel (transparency)**
- 48 bit integer color: 281 trillion

CLUTs

Matlab native colormaps (you can make your own):

- hsv - Hue-saturation-value color map.
- hot - Black-red-yellow-white color map.
- gray - Linear gray-scale color map.
- bone - Gray-scale with tinge of blue color map.
- copper - Linear copper-tone color map.
- pink - Pastel shades of pink color map.
- white - All white color map.
- flag - Alternating red, white, blue, and black color map.
- lines - Color map with the line colors.
- colorcube - Enhanced color-cube color map.
- vga - Windows colormap for 16 colors.
- jet - Variant of HSV.
- prism - Prism color map.
- cool - Shades of cyan and magenta color map.
- autumn - Shades of red and yellow color map.
- spring - Shades of magenta and yellow color map.
- winter - Shades of blue and green color map.
- summer - Shades of green and yellow color map.

Exercise

Look at map. What do you notice?

Display Durer's etching in all shades of red.

```
>map(:,2)=0;
>map(:,3)=0;
>colormap(map)
```

Other types of images

You can load TIFF, JPEG, BMP… with imread

[X,map] = imread(filename,ext);

for indexed images. (like GIF)
MxN matrices of color indeces

or

X = imread(filename,ext);

for non-indexed images (like JPG)
MxNx3 matrices with rgb triplets at m x n

for more options do 'help imread'

demos…

write a script named classdemo.m

Other types of images

Examples:
go to my webpage:
www.psych.uiuc.edu/~alleras/courseImages.htm

indexed image: type this on classdemo

```
[x,map]=imread('VisionLab','gif')
image(x);
axis off;
axis equal;
input('click key when ready');
colormap(map);
input('click key when ready');
colormap(hot);
```

CHECK OUT 'map'. These are **customized** color values needed to reproduce THIS image.

Other types of images

non-indexed image:

```
[x,map]=imread('VisionLab','jpg')
what's map?
          %what are the dimensions of x?
image(x);
axis off;
axis equal;
input('click key when ready');
colormap(cold);
```

.

Write images to files

Let's make and save a random b/w mask image.

imwrite(matrix,'nameoffile','ext')

%imwrite(matrix,colormap,'nof','ext') for indexed images

```
> mask = rand(400,400);
> imwrite(mask,'mask','bmp');
> clear all
> input('click key when ready');
> [X,map] = imread('mask','bmp');
> image(X);
> input('click key when ready');
> colormap(map);    %default map is grayscale
```

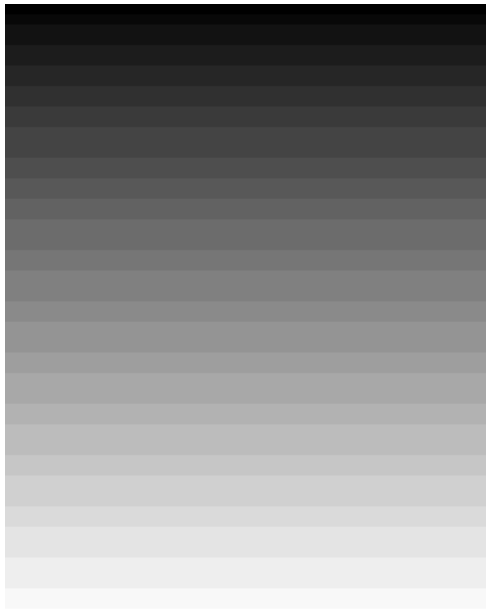
Write images to files

Let's make and save a random color-noise mask image -unindexed

```
> mask2 = rand(400,400,3); %why 3?
> imwrite(mask2,'mask2','jpg');
> clear all;
> input('click key when ready');
> X = imread('mask2','jpg');
> image(X);
> colormap(map);
> input('click key when ready');
> colormap(hot);
```

Exercise

Display to the Figure window an image of a horizontal grayscale (size 256 x 200) (all colors of grayscale(256))



```
grate = zeros(256,200);
for n=1:256
  grate(n,:) = n;
end;
colormap(gray(256));
image(grate);
axis off;
axis equal;
```

Other demo

Display to the Figure window an image of a grayscale horizontal sinusoidal grating (size 200 x 200)

```
grate = zeros(200,200);
for n=1:200
  grate(n,:) = 256 * (abs(sin(2*pi*n/200));
end;
colormap(gray(256));
image(grate);
axis off;
axis equal;
```

Question:

How can I easily reduce brightness by half in Durer Drawing?

```
> clear all
> load durer;
> image(X);
> colormap(gray(256));
```

or

```
> X2=X./2;
> image(X2);
```

eval function!!

eval

eval is a powerful command that calculates a string and **then** evaluates it.

syntax: eval(string)

ex: creates ten matrices named M1, M2, … M10, of increasing size.

```
for n=1:10
  eval(['M' num2str(n) ' = zeros(n,n)'])
end;
```

The parenthesis OPEN the function

The brackets CONCATENATE the strings.

eval

For instance, open file names with changing values in them.

Download the vowels from the website. Save them to your working directory.

DRAW A BANNER THAT CONTAINS ALL THE VOWELS!

eval function!

```
banner=[];
for vowel=['a' 'e' 'i' 'o' 'u']
   string = ['''let' vowel ''','''gif'' '];
   %concatenates file name string
   eval(['[letter,map]=imread(' string ');']);
   banner = [banner letter];


end;
image(banner);
colormap(map);
axis off;
axis equal;
```

Notice the ; in the command!!

Similar command to open subject output files!!

Exercise: eval

Download the vowels from my website:

www.psych.uiuc.edu/~alleras/courseImages.htm

Write a program that displays the user's name using the gif letters.

Try at home

Display the durer image on a background of white and red stripes. Each background stripe should be 24 pixels.

```
newmap1 = colormap(gray(128));
newmap2 = newmap1;
newmap2(:,2) = 0;
newmap2(:,3) = 0;
newmap = [newmap1;newmap2];
```

Try at home

with a for loop, every 24 lines, you add 128 to color index

```
image has 27 'bands'
create a matrix with these bands
zer = zeros(24,width_of_X)
one = ones(24,width_of_X)
conc = [zer;one]
newX = repmat(conc,13,1);
newX = [newX;zer];
newX = newX .*128;
newX = newX + X2;  %done!
image(newX);
colormap(newmap);
```