Image Processing

Chapter 2. Digital Image Fundamentals

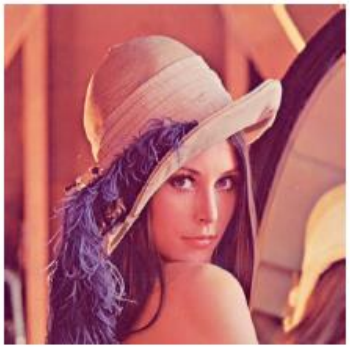
Yeong Jun Koh

Department of Computer Science & Engineering

Chungnam National University

Image?

- 2-dimentional function $f(x, y)$
- (x, y): 2-D spatial coordinate
- f: 1-D scalar (gray image), 3-D vector (color image: RGB)
  - Each value is in (0. 255)

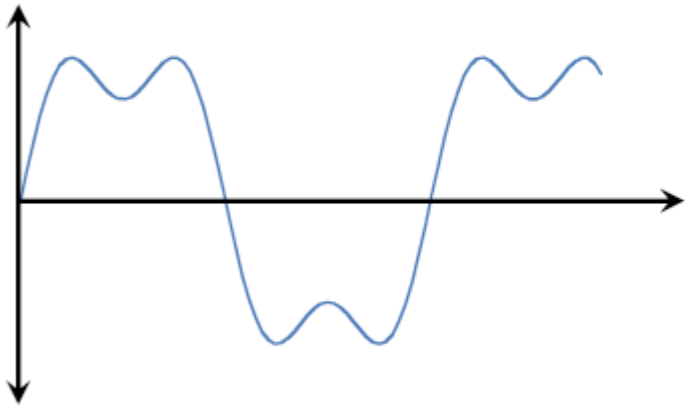


- $f(x, y, t)$: video sequence
- $f(x, y, z)$: 3-D object
- $f(x, y, z, t)$: moving 3-D object

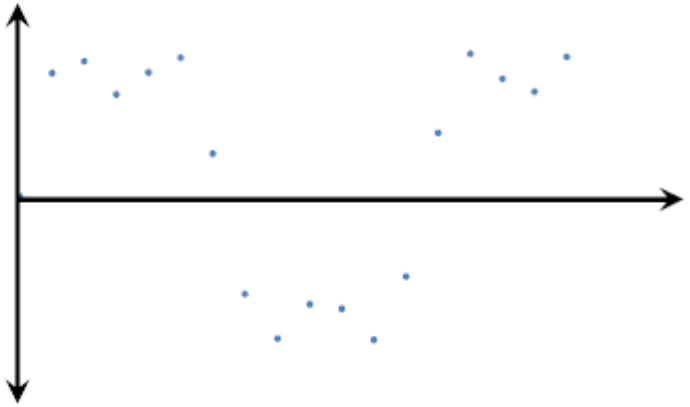
Image Sampling

Continuous 1D signal

$$y = \sin(x) + \frac{1}{3}\sin(3x)$$



Under-sampling: Not perfect reconstruction



Over-sampling: Nyquist theory

$sampling\ freq. > 2 * signal's\ freq.$

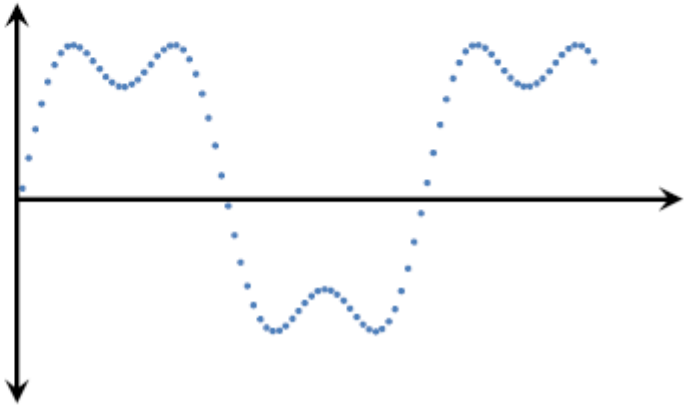


Image Sampling

Continuous signal from real world

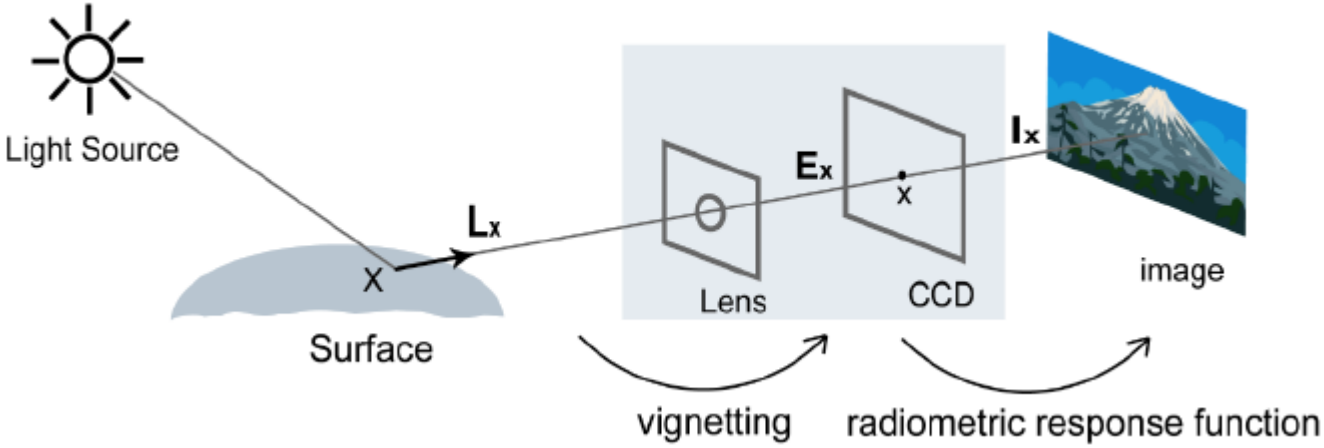
Discrete signal using sampling

When human interprets an image, it is considered as a **continuous** signal!

Note that all imaging devices capture 2D image in a **discrete** manner, but the human being interprets it in a **continuous** manner.

Image Acquisition

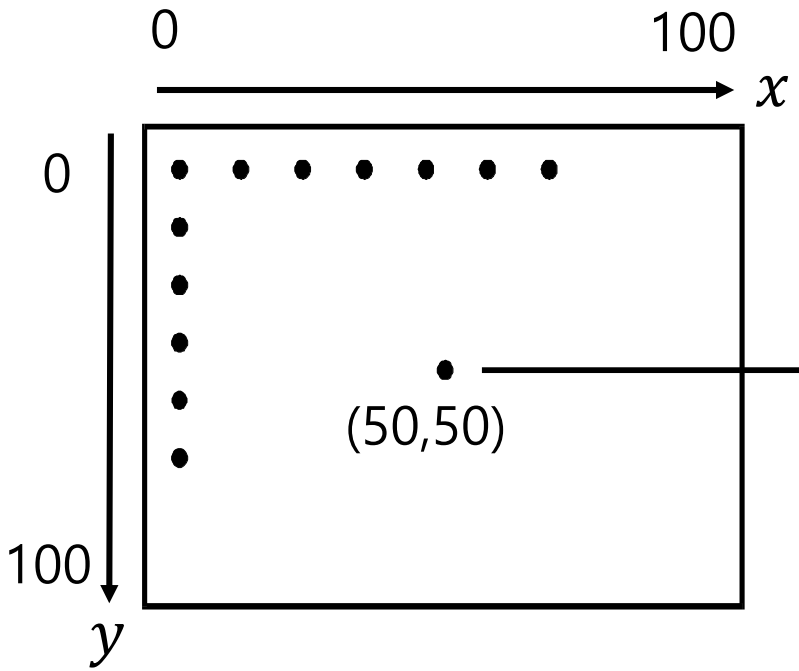
- Image model (from light to image)



  - CCD (charge coupled device) camera
    - High-end DSLR camera
  - CMOS (Complementary metal-oxide semiconductor) camera
    - Smartphone

Image Acquisition

- Digital Image
  - Pixel means
    - pixel coordinate
    - pixel value
  - Both coordinates and value are discrete

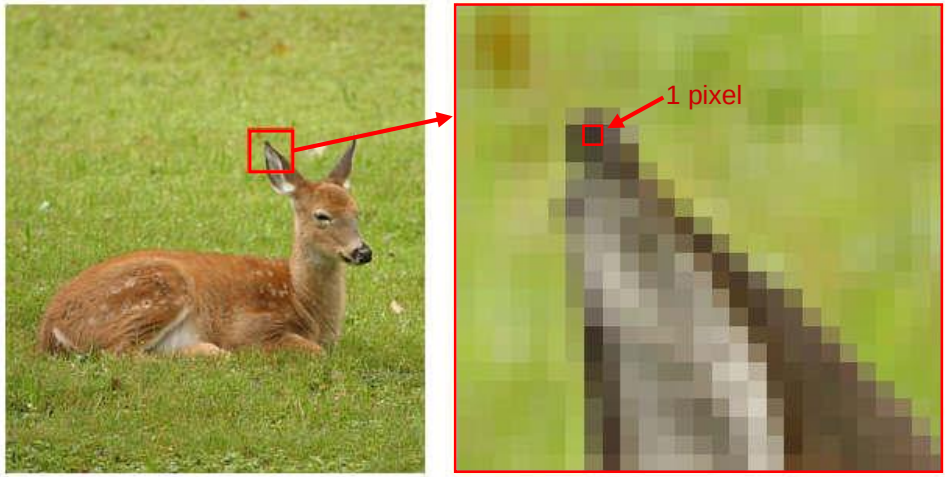


(x, y): pixel coordinate
$f(x, y)$: intensity value at (x, y)

0 (black) ..... 255 (white)

Image Acquisition

- Digital Image
  - Pixel means
    - pixel coordinate
    - pixel value
  - Both coordinates and value are discrete



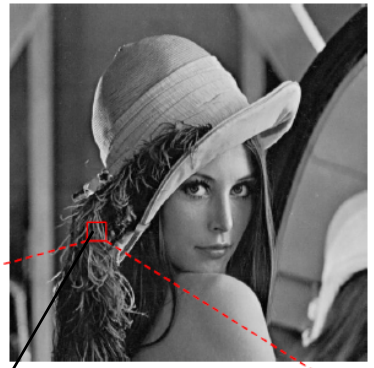
Quantization (or Sampling) on 2D Image

- Quantization happens twice!

$f(x, y)$

1. Spatial 2D coordinate: 2D grid (x, y)

2. Intensity value f: 0~255

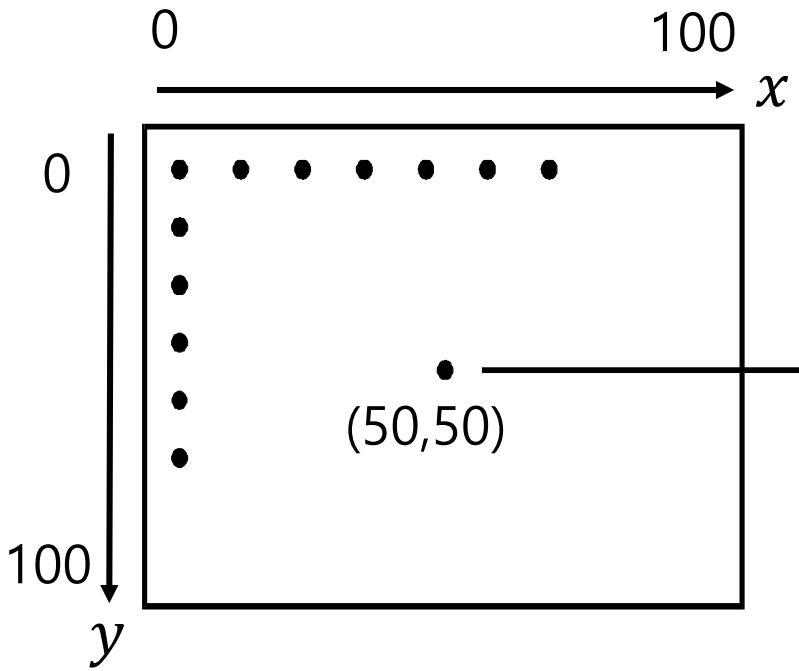


37	54	29	17	22	72	112	84
42	67	33	19	38	60	108	86
59	84	36	22	53	46	102	85
78	97	41	24	50	42	101	93
85	95	46	**28**	33	46	101	106
81	86	48	32	21	47	97	111
52	65	45	36	25	42	97	125
42	54	52	55	31	49	93	10

3 × 5 neighborhood

Image Acquisition

- Digital Image
  - Pixel means
    - pixel coordinate
    - pixel value
  - Both coordinates and value are discrete

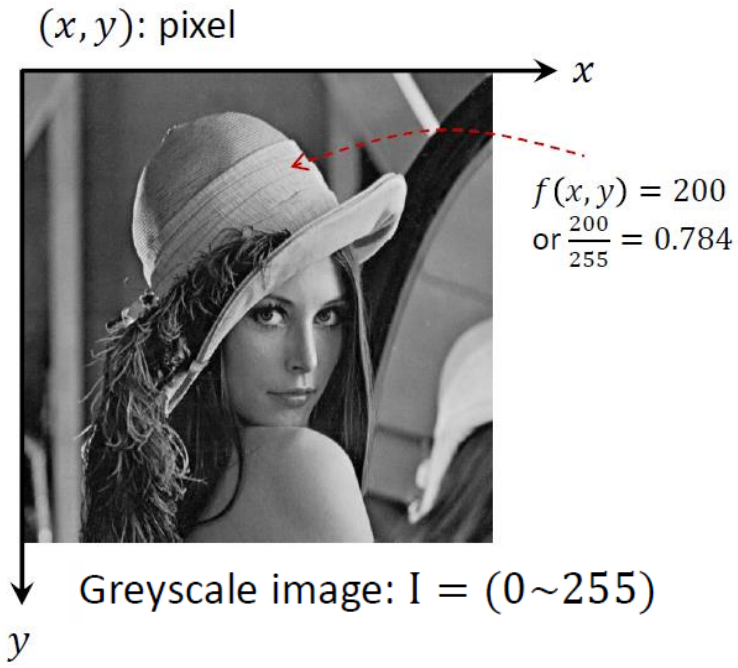


(x, y): pixel coordinate
$f(x, y)$: intensity value at (x, y)

0 (black) ..... 255 (white)

Grayscale Image

- Grayscale image



Color Image

- Color image – **RGB**

(97,37,86)

Color image: RGB = (0~255, 0~255, 0~255)

R

97	117	90	86	87	113	145	138
103	123	102	79	91	110	141	139
106	140	109	84	95	104	139	137
124	137	111	81	102	99	128	139
125	133	119	92	87	112	134	141
124	126	116	102	89	96	137	143
120	119	109	108	93	102	136	148
107	116	109	121	102	99	129	144

G

37	56	27	17	26	68	115	88
39	70	35	17	33	61	104	87
55	88	37	19	56	50	103	88
85	89	42	21	61	44	92	98
91	92	57	21	22	52	102	100
78	79	55	31	24	39	104	115
65	57	50	37	26	47	103	116
43	53	46	62	33	39	90	110

B

86	126	89	87	83	120	176	148
92	139	97	81	94	116	160	145
110	151	105	78	105	104	152	156
141	158	105	82	104	96	148	161
136	144	122	92	80	100	152	171
123	135	114	106	80	94	154	167
110	117	106	114	87	99	147	169
96	113	99	134	96	94	141	154

Binary Image

- Binary image
  - 0 or 255
  - Usually used to show the result of edge detection

Goal

- Image reading and writing
  - Grayscale, RGB color, indexed color
- Data types and Conversions
  - Uint8, double, and so on
- Image Files and Formats
  - JPEG, TIFF, GIF, BMP, PNG, HDF
- Image Display

Grayscale Images

- MATLAB is a software package with powerful support for matrices and matrix operations
  - Command window

```
>>
```

  - Reads pixel values from an image and saves it to *w*

```
>> w=imread('wombats.tif');
```

    - ;->Means not displaying results of the command to the screen
    - ' ' ->Used to indicate the filename
    - An image file is assumed to be in the current directory, or you can define the path

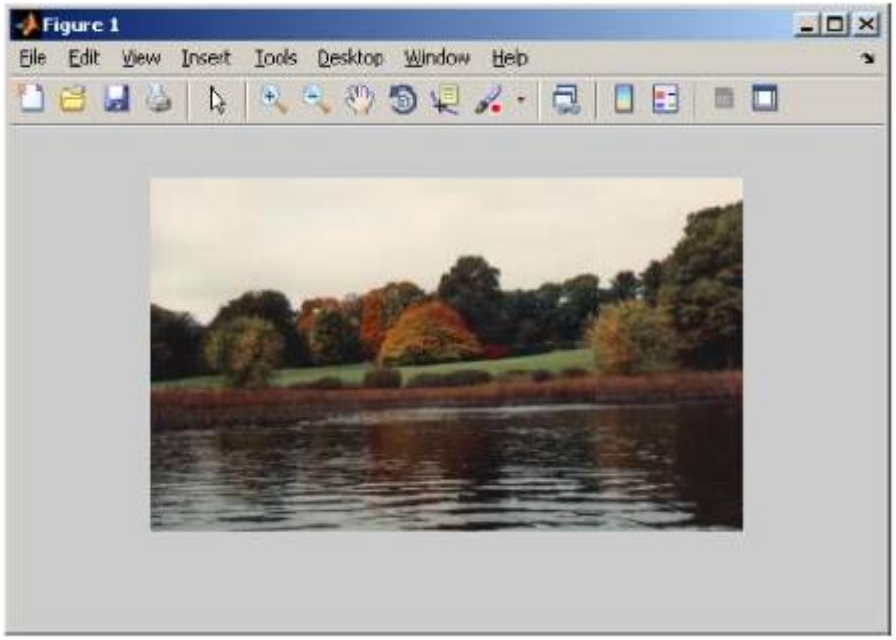
Grayscale Images

- Display the matrix w as an image

```
>> figure, imshow(w)
```

RGB Images

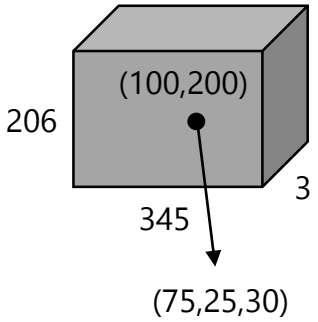
```
>> a=imread('autumn.tif');
>> figure,imshow(a)
```



Note that the command `pixval on` is not used in a later MATLAB version

RGB Images

- Multidimensional array
  - **Q:**
    ```
    >> size(a)
    ```
  - **A:** 206 345 3 :rows columns channels



  - **Q:**
    ```
    >> a(100,200,2)
    ```
  - **A:** 25

  - **Q:**
    ```
    >> a(100,200,1:3)
    ```
    ```
    >> a(100,200,:)
    ```
    ```
    >> impixel(a,200,100)
    ```
  - **A:** 75 25 30

a(100,200,2:3)=(25, 30)

Indexed Color Images

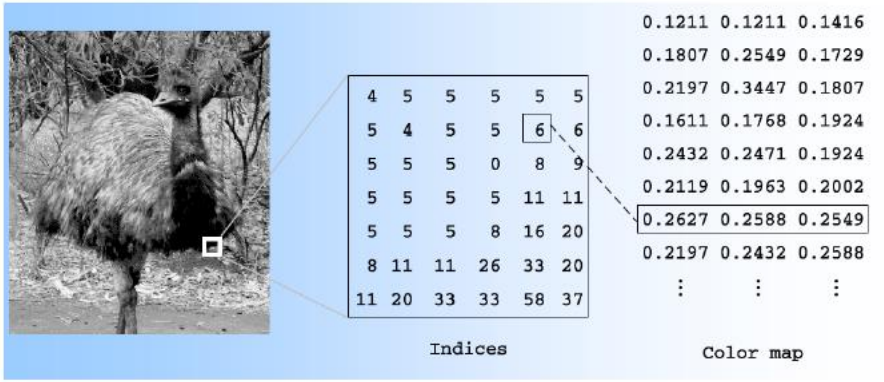
```
>> figure, imshow('trees.tif')
```

```
>> em = imread('trees.tif');
>> figure, imshow(em)
```

Why?

Indexed Color Images

- Index data is saved in double format




```
>> [em, emap] = imread('trees.tif');
>> figure, imshow(em, emap)
```

- How to know information of an image?
  - imfinfo function

Indexed Color Images

```
>> imfinfo('emu.tif')

ans =

                    Filename: 'emu.tif'
                  FileModDate: '26-Nov-2002 14:23:01'
                    FileSize: 119804
                      Format: 'tif'
                FormatVersion: []
                       Width: 331
                      Height: 384
                    BitDepth: 8
                   ColorType: 'indexed'
              FormatSignature: [73 73 42 0]
                   ByteOrder: 'little-endian'
              NewSubfileType: 0
                BitsPerSample: 8
                  Compression: 'PackBits'
    PhotometricInterpretation: 'RGB Palette'
                 StripOffsets: [16x1 double]
              SamplesPerPixel: 1
                 RowsPerStrip: 24
              StripByteCounts: [16x1 double]
                  XResolution: 72
                  YResolution: 72
               ResolutionUnit: 'Inch'
                     Colormap: [256x3 double]
          PlanarConfiguration: 'Chunky'
                   TileWidth: []
                  TileLength: []
                 TileOffsets: []
              TileByteCounts: []
                 Orientation: 1
                   FillOrder: 1
             GrayResponseUnit: 0.0100
                MaxSampleValue: 255
                MinSampleValue: 0
                  Thresholding: 1
```

Data Types and Conversions

```
>>  a=23;
>> b=uint8(a);
>> b

b =

    23

>> whos a b
  Name        Size                Bytes  Class

  a           1x1                     8  double array
  b           1x1                     1  uint8 array
```

Data Types and Conversions

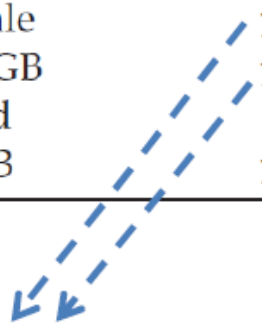
TABLE 2.1 *Data types in MATLAB*

Data Type	Description	Range
`int8`	8-bit integer	−128–127
`uint8`	8-bit unsigned integer	0–255
`int16`	16-bit integer	−32768–32767
`uint16`	16-bit unsigned integer	0–65535
`double`	Double precision real number	Machine specific

Data Types and Conversions

TABLE 2.2 *Converting images in MATLAB*

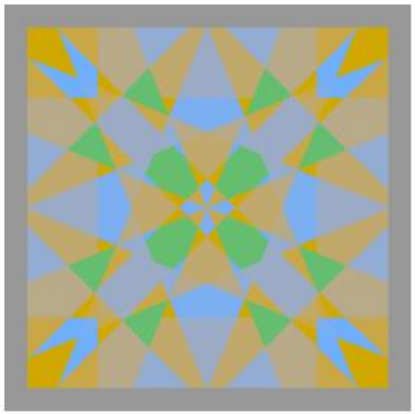
Function	Use	Format
`ind2gray`	Indexed to grayscale	`y=ind2gray(x,map);`
`gray2ind`	Grayscale to indexed	`[y,map]=gray2ind(x);`
`rgb2gray`	RGB to grayscale	`y=rgb2gray(x);`
`gray2rgb`	Grayscale to RGB	`y=gray2rgb(x);`
`rgb2ind`	RGB to indexed	`[y,map]=rgb2ind(x,n);`
`ind2rgb`	Indexed to RGB	`y=ind2rgb(x,map);`



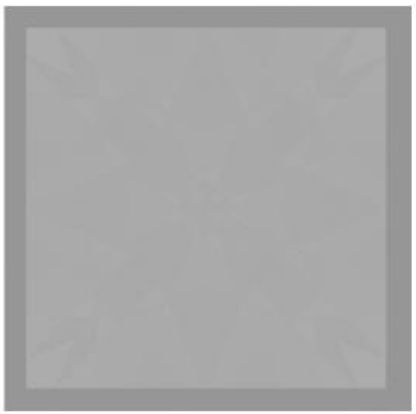
Q: How?

Some interesting topics

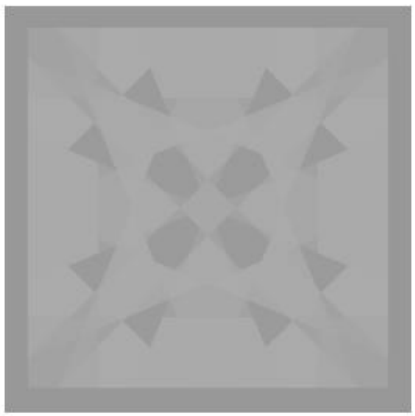
- Converting RGB to Gray with a minimum perceptual loss



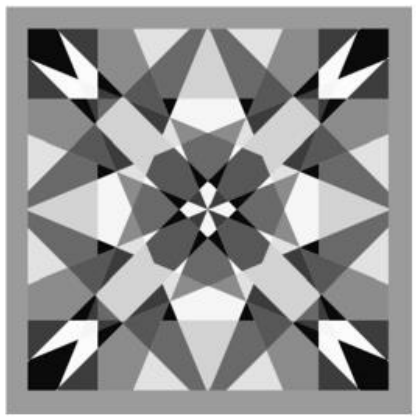
Input Picture



Photoshop CS5



IrfanView



Proposed method

From: "Contrast Preserving Decolorization with Perception-Based Quality Metrics", Int. Journal of Computer Vision, 2014

Some interesting topics

- Image colorization (gray2rgb)
  - Grayscale image + color scribble (given by user)-> color image

Some interesting topics

- Image colorization

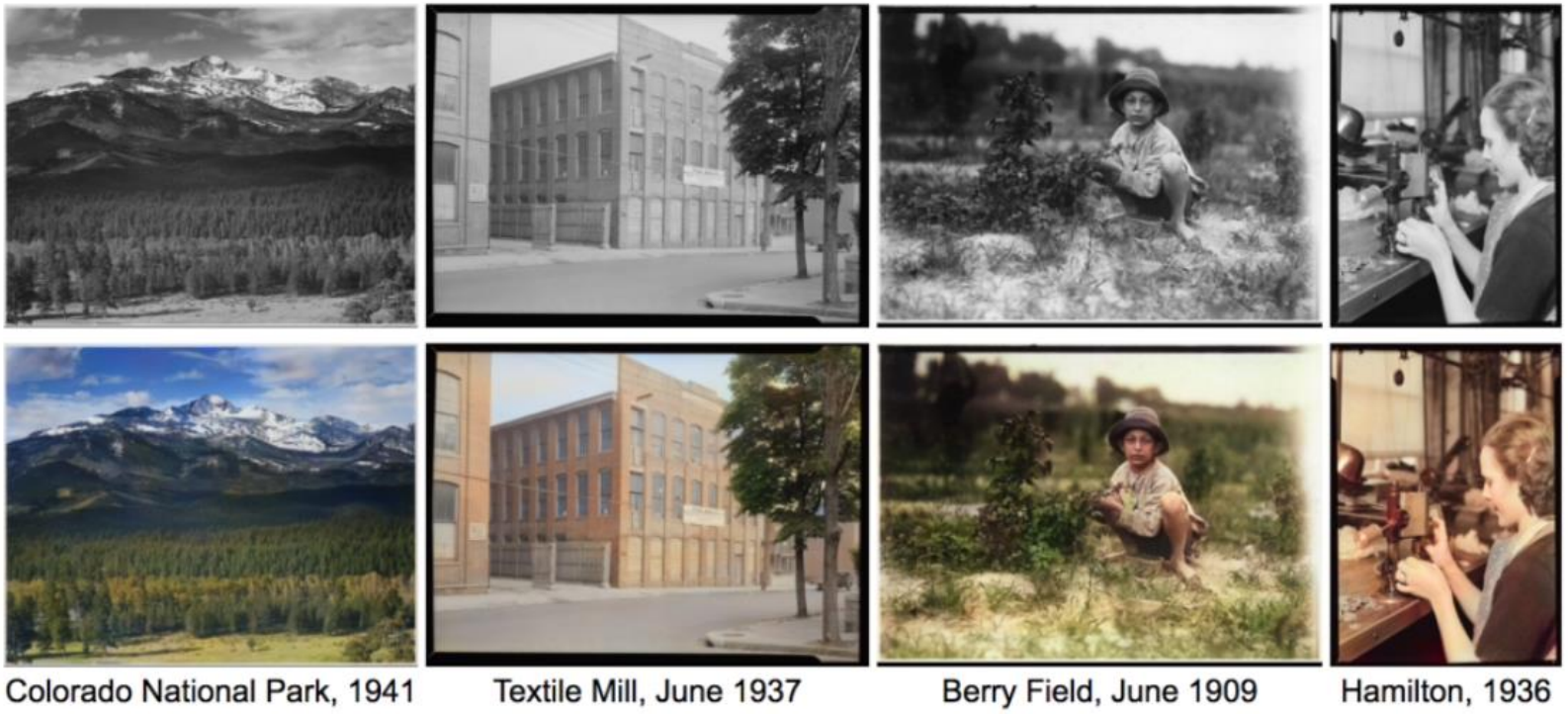


Image Files and Formats

- imread and imwrite functions of MATLAB support the following formats
  - JPEG, TIFF, GIF, PNG, BMP

- imwrite(X,′ filename′,′ fmt′)

```
a=imread('autumn.tif');
imwrite(a,'autumn.png','png');
```

Image Files and Formats

- GIF
  - Colors are stored using a color map
  - Allows multiple images per file: animated GIFs
- PNG
  - Supports true color
  - Lossless compression
- JPEG
  - Lossy compression

The imshow Function

- If x is a matrix of type uint8

```
imshow(x)
```

  - uint8 restricts values to be integers between 0 and 255

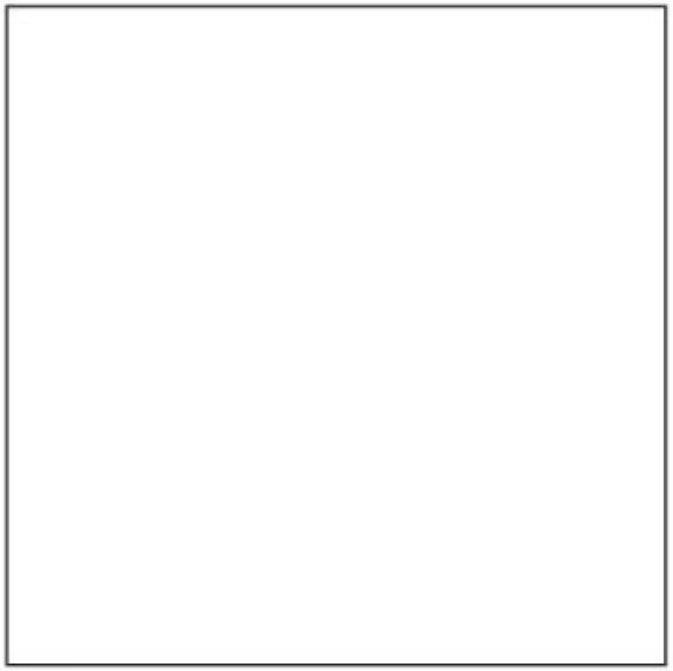
- Two choices with a matrix of type double
  - Convert to type uint8 and then display
  - Display the matrix directly

The imshow Function

```
>> c=imread('caribou.tif');
>> cd=double(c);
>> imshow(c),figure,imshow(cd)
```

Capt. Budd Christman/NOAA Corps.

(a)



(b)

FIGURE 3.1 *An attempt at data type conversion. (a) The original image. (b) After conversion to type* `double`.

The imshow Function

```
>> c=imread('caribou.tif');
>> cd=double(c);
>> imshow(c),figure,imshow(cd)
```

```
>> imshow(cd/255)
```

Capt. Budd Christman/NOAA

(a) (b)

FIGURE 3.1 *An attempt at data type conversion. (a) The original image. (b) After conversion to type* `double`.

The imshow Function

```
>> imshow(cd/512)
>> imshow(cd/128)
```

FIGURE 3.2 *Scaling by dividing an image matrix by a scalar. (a) The matrix* `cd` *divided by 512. (b) The matrix* `cd` *divided by 128.*

The imshow Function

- double - type change only
- double – type change, value scaling

```
>> cd=im2double(c);
```

- Convert back to an image of type uint8 in two ways

```
>> c2=uint8(255*cd);
>> c3=im2uint8(cd);
```

The imshow Function

- Binary image
  - logical flag

```
>> c1=c>120;
```

  - Check c1 with whos

```
Name        Size              Bytes  Class     Attributes
c1        256x256             65536  logical
```

The imshow Function

- Binary image

FIGURE 3.3 *Making the image binary. (a) The caribou image turned binary. (b) After conversion to type* `uint8`.