Image Processing Using Scilab

Shanmuganathan Raman

Vision and Image Processing Laboratory
Department of Electrical Engineering
Indian Institute of Technology Bombay
shanmuganathan@iitb.ac.in
www.ee.iitb.ac.in/student/~shanmuga

December 2, 2010

Introduction to Digital Images

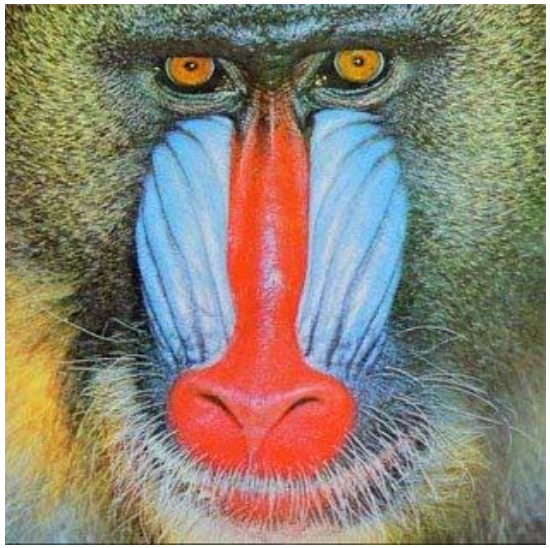
- Images can be Grayscale or Color (RGB)
- Specified as a matrix of size M $\times$ N $\times$ 3 (M $\times$ N for grayscale)

Image Processing in Scilab

- Toolboxes - SIP (Scilab Image Processing) , SIVP (Scilab Image & Video Processing)
- We focus on SIVP Toolbox
- Installation in Ubuntu Linux - SIVP through atoms (5.3beta onwards), SIP through CVS (Both tested on Scilab 5.2.2)
- Installation in Windows - SIVP through atoms (5.3beta onwards), SIP (installed, not working yet)

Grayscale and Color Images

(a) Gray Scale Image



(b) Color Image

Image Read/Write/Show

1. Read Image - $lena = imread('lenagray.jpg')$
2. Show Image - $figure; imshow(lena)$
3. Write Image - $imwrite(lena,' lena.png')$

Image Datatype Conversions

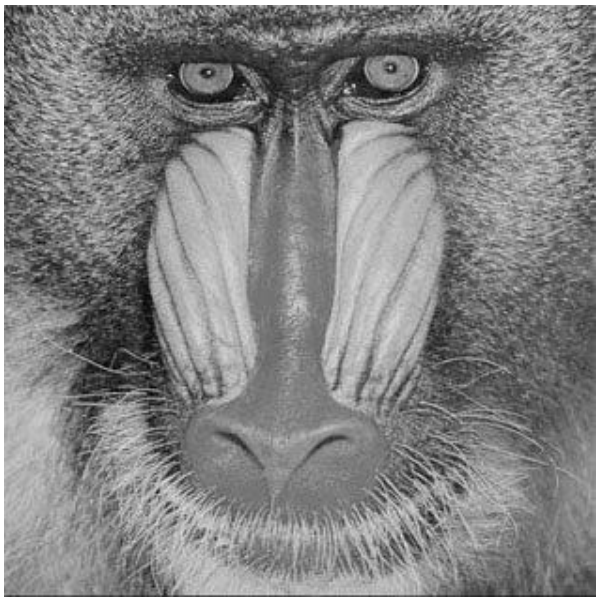
- im2int8
- im2int16
- im2int32
- im2uint8
- im2uint16
- im2double

Image Conversions

$-->babbon = imread('bab.png');$

- RGB to Gray scale - rgb2gray
  $-->babgray = rgb2gray(babbon);$
- RGB to Binary - im2bw
  $-->lenabw = im2bw(lena, 0.5);$
  $-->imwrite(lenabw,'lenabw.png');$
- RGB to HSV format - rgb2hsv
- HSV to RGB format - hsv2rgb
- RGB to YCbCr - rgb2ycbcr
- YCbCr to RGB - ycbcr2rgb

Image Conversions - Results



(a) *rgb2gray*(*babbon*);

(b) *im2bw*(*lena*, 0.5);

Basic Operations

- Crop -
  $--> lenacrop = imcrop(lena, [200, 200, 200, 200]);$
- Complement -
  $--> lenacomp = imcomplement(lena);$
- Resize -
  $--> lenaresize = imresize(lena, 2, 'bicubic');$
  - The last term can be 'nearest', 'bilinear', 'bicubic' or 'area'

Basic Operations - Complement

(a) Original

(b) Complement

Basic Operations - Crop

(a) Original (b) Cropped

Basic Operations - Resize

(a) Original (b) Resize by 2

Add Noise

$-->$ *lenaNgaussian* $=$ *imnoise*(*lena*,' *gaussian*');

The noise can be one of these:

1. salt & Pepper - white/black noise (default probability d=0.05)
2. speckle - multiplicative noise (uniform with mean 0 and variance v=0.04)
3. gaussian - additive noise (with mean 0 and variance v=0.01)

Add Noise - Results

(a) Original

(b) Salt & Pepper

(c) Speckle

(d) Gaussian

LTI Filter Kernels

Derivatives (High-Pass filter) using fspecial

- sobel $--> fspecial('sobel')$

$$\begin{pmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{pmatrix}$$

- prewitt $--> fspecial('prewitt')$

$$\begin{pmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{pmatrix}$$

- laplacian $--> fspecial('laplacian')$

$$\begin{pmatrix} 0.1666667 & 0.6666667 & 0.1666667 \\ 0.6666667 & -3.3333333 & 0.6666667 \\ 0.1666667 & 0.6666667 & 0.1666667 \end{pmatrix}$$

LTI Filter Kernels Contd.

Blur (Low-Pass filter) using fspecial

- gaussian $-->$ *fspecial*('*gaussian*')

$$\begin{pmatrix} 0.0113437 & 0.0838195 & 0.0113437 \\ 0.0838195 & 0.6193470 & 0.0838195 \\ 0.0113437 & 0.0838195 & 0.0113437 \end{pmatrix}$$

- log $-->$ *fspecial*('*log*')

$$\begin{pmatrix} 0.0447924 & 0.0468064 & 0.0564068 & 0.0468064 & 0.0447924 \\ 0.0468064 & 0.3167464 & 0.7146325 & 0.3167464 & 0.0468064 \\ 0.0564068 & 0.7146325 & -4.904764 & 0.7146325 & 0.0564068 \\ 0.0468064 & 0.3167464 & 0.7146325 & 0.3167464 & 0.0468064 \\ 0.0447924 & 0.0468064 & 0.0564068 & 0.0468064 & 0.0447924 \end{pmatrix}$$

- average $-->$ *fspecial*('*average*')

$$\begin{pmatrix} 0.1111111 & 0.1111111 & 0.1111111 \\ 0.1111111 & 0.1111111 & 0.1111111 \\ 0.1111111 & 0.1111111 & 0.1111111 \end{pmatrix}$$

Spatial Domain Processing

- FIR filters using fspecial $-->$ $h = fspecial('sobel')$;
- Convolve 2D image with a 2D kernel imfilter/filter2 $-->$ $lenaSobel = imfilter(lena, h)$;
- Low-pass, High-pass filters realized

Derivative Kernels - Results

(a) Original

(b) Sobel

(c) Prewitt

(d) Laplacian

Blurring Kernels - Results

(a) Original

(b) Gaussian

(c) LoG

(d) Average

Edge Detection

edge - High-pass filtering and thresholding

- Sobel $-->$ *lenaESobel* $=$ *edge*(*lena*,′ *sobel*′);
- Prewitt $-->$ *lenaEPrewitt* $=$ *edge*(*lena*,′ *prewitt*′);
- LoG $-->$ *lenaELog* $=$ *edge*(*lena*,′ *log*′);
- Canny $-->$ *lenaECanny* $=$ *edge*(*lena*,′ *canny*′);

Edge Detection - Results

(a) Original

(b) Sobel

(c) Prewitt

(d) LoG

(e) Canny

Miscellaneous

1. imhist - Histogram of the image$-->$ $[counts, bins] = imhist(lena)$;
2. imfinfo - Image Information $-->$ $imfinfo('lena.jpg')$

Gaussian Pyramid - Work Out

impyramid - Gaussian Smoothing and subsampling

$--> im0 = imread('lena.jpg');$
$--> im1 = impyramid(im0,'reduce');$
$--> im2 = impyramid(im1,'reduce');$
$--> im3 = impyramid(im2,'reduce');$
$--> imshow(im0);$
$--> imshow(im1);$
$--> imshow(im2);$
$--> imshow(im3);$

Image Compression

- Consider grayscale image as a matrix
- Take SVD $I = U\Sigma V^T$
- Drop lowest singular values from diagonal matrix Σ
- Reconstruct Image again

Image Compression in Scilab

$--> A = imread('lena.jpg');$
$--> [u, s, v] = svd(double(A));$
$--> norm(u * s * v' - A)$ // just to check that A = u*s*v'
$--> vdash = v';$
$--> svalues = diag(s);$. // these are ordered increasing to decreasing
$--> n = 100;$// how many singular values of A we want to KEEP.
$--> Alowerrank = u(:, [1 : n]) * diag(svalues(1 : n)) * vdash([1 : n], :);$
$--> imwrite(uint8(Alowerrank), 'lenaSVD100.png');$

Image Compression - Results

(a) Original

(b) Top 20

(c) Top 40

(d) Top 60

(e) Top 80

(f) Top 100

Advanced Topics - To be Explored

- FFT
- Wavelets
- Radon Transform
- Hough Transform

Recap

1. Read/Write/Show Image
2. Basic Operations
3. Noise and Blur
4. LTI Filtering
5. Image as a Matrix
6. Transform Domain Operations

Thanks

Thank You

shanmuganathan@iitb.ac.in