

Scilab Manual for
Image Processing and Computer Vision
Laboratory
by Dr Majharoddin
Others
Dr. G. Y. Pathrikar College Of CS And IT,
Aurangabad¹

Solutions provided by
Dr Majharoddin
Others
Dr. G. Y. Pathrikar College Of CS And IT, Aurangabad

September 7, 2026

¹Funded by a grant from the National Mission on Education through ICT, <http://spoken-tutorial.org/NMEICT-Intro>. This Scilab Manual and Scilab codes written in it can be downloaded from the "Migrated Labs" section at the website <http://scilab.in>

Contents

List of Scilab Solutions	3
1 Basic Statistical Analysis of Images	5
2 Different Filtering Operations on Images	8
3 Morphological Operations on Images	11
4 Basic Image Segmentation	16
5 Image Enhancement for Performance	20
6 Hough Transform Implementation on Images	25
7 SIFT Implementation for Images	28
8 Different Distance Measures implementations	31
9 Basic Shape Analysis Implementation	34
10 Feature Extraction from Images	42
11 Basics of OCR	45
12 Face Detection Algorithm Implementation	50

List of Experiments

Solution 1.1	Example 1	5
Solution 2.1	Example 1	8
Solution 3.1	Basic Morphological Operations on Image	11
Solution 4.1	Example 1	16
Solution 5.1	Example 1	20
Solution 6.1	Example 1	25
Solution 7.1	Example 1	28
Solution 8.1	Example 1	31
Solution 9.1	Example 1	34
Solution 10.1	Example 1	42
Solution 11.1	Example 1	45
Solution 12.1	Example 1	50
AP 1	groupimage	56
AP 2	kalpanachawla	56
AP 3	randomimage	57
AP 4	numberimage	57
AP 5	regionimage	58
AP 6	textimage	58
AP 7	shape	59
AP 8	lennaimage	59
AP 9	circleimage	60
AP 10	harewood	61
AP 11	Text Image	62
AP 12	inputimage	63

List of Figures

2.1	Example 1	10
3.1	Basic Morphological Operations on Image	14
3.2	Basic Morphological Operations on Image	15
4.1	Example 1	19
5.1	Example 1	22
5.2	Example 1	23
5.3	Example 1	24
6.1	Example 1	27
7.1	Example 1	30
9.1	Example 1	37
9.2	Example 1	38
9.3	Example 1	39
9.4	Example 1	40
9.5	Example 1	41
10.1	Example 1	44
11.1	Example 1	48
11.2	Example 1	49
11.3	Example 1	49
12.1	Example 1	53
12.2	Example 1	54
12.3	Example 1	54

Experiment: 1

Basic Statistical Analysis of Images

check Appendix [AP 12](#) for dependency:

`cameraman.png`

Scilab code Solution 1.1 Example 1

```
1 // This scilab code is to calculate basic image
  statistics
2 //such as sum, average, standard deviation, min and
  max
3
4 //The scilab environment for this is : Scilab 5.5.2
5
6 //Toolbox used: SIVP 0.5.3.2
7
8 //OS used :Windows 10 64 bit
9 //
10 //Reference book name : Digital Image Processing
11 //book author: Rafael C. Gonzalez and Richard E.
  Woods
12
```

```

13 clc //to clear command window.
14 clear all //to kill previously defined variables.
15 xdel(winsid())//to close all currently open figure(s
    ).
16
17 //This code uses cameraman.tif file for processing
18
19 img = imread('cameraman.tif'); // Reads the
    cameraman image
20 imshow(img); // Show original image.
21 if(size(img,3)>1) then
22     Gray_img= rgb2gray(img); //Converts color image
        to grayscale
23 else
24     Gray_img=img;           // if image is 2D image
25 end
26 imshow(Gray_img);           // Show gray scale image
    .
27 imgsum=sum(Gray_img);       // Calculates sum of
    image pixels
28 imgmean=mean2(Gray_img);    // Calculates mean /
    average of image pixels
29 imgstd=std2(Gray_img);     // Calculates standard
    deviation of image
30 imgmin=min(Gray_img);       // Calculates minimum of
    image pixels
31 imgmax=max(Gray_img);       // Calculates maximum of
    image pixels.
32 disp("Size of Image =");
33 disp(size(Gray_img));       // Displays size of
    image
34 disp("Sum = ");
35 disp(imgsum);               // Displays Sum of image
    pixels
36 disp("Mean = ");
37 disp(imgmean);             // Displays mean /
    average of image pixels
38 disp("Standard Deviation = ");

```

```

39 disp(imgstd);           // Displays standard
    deviation of image pixels
40 disp("Minimum of Image = ");
41 disp(imgmin);           // Displays minimum of
    image
42 disp("Maximum of Image =");
43 disp(imgmax);           // Displays maximum of
    image
44
45 //////////// OUTPUT ////////////
46 //
47 //Size of Image =
48 //
49 //    256.    256.
50 //
51 // Sum =
52 //
53 //    120
54 //
55 // Mean =
56 //
57 //    118.72449
58 //
59 // Standard Deviation =
60 //
61 //    62.341715
62 //
63 // Minimum of Image =
64 //
65 //    7
66 //
67 // Maximum of Image =
68 //
69 //    253

```

Experiment: 2

Different Filtering Operations on Images

check Appendix [AP 12](#) for dependency:

`cameraman.png`

Scilab code Solution 2.1 Example 1

```
1 // This scilab code is to calculate basic image
  statistics
2 //such as sum, average, standard deviation, min and
  max
3
4 //The scilab environment for this is : Scilab 5.5.2
5
6 //Toolbox used: SIVP 0.5.3.2
7
8 //OS used : Windows 10 64 bit
9 //
10 //Reference book name : Digital Image Processing
11 //book author: Rafael C. Gonzalez and Richard E.
  Woods
12
```

```

13 clc //to clear command window.
14 clear all //to kill previously defined variables.
15 xdel(winsid())//to close all currently open figure(s
    ).
16
17 //This code uses cameraman.tif file for processing
18 //SCI + '/contrib/SIVP_0.5.3.2/images/lena.png'
19 img = imread('cameraman.png'); // Reads the
    cameraman image
20 //title("Original Image");
21 imshow(img); // Show original
    image.
22 if(size(img,3)>1) then
23     Gray_img= rgb2gray(img); //Converts color
        image to grayscale
24 else
25     Gray_img=img; // if image is 2D
        image
26 end
27
28
29 filter = fspecial('sobel'); // Creates 2D
    special filter 'sobel'
30 imf = imfilter(Gray_img, filter); // Apply filter on
    2D image
31 //pause;
32 //title("Filtered Image");
33 figure, imshow(imf); // Show
    Resultant Image

```



Figure 2.1: Example 1

Experiment: 3

Morphological Operations on Images

check Appendix [AP 11](#) for dependency:

Fig0419.png

Scilab code Solution 3.1 Basic Morphological Operations on Image

```
1
2 // This scilab code is to perform basic
   morphological operation on image
3 //such as erosion and dilation , opening and closing
   an image
4
5 //The scilab environment for this is : Scilab 5.5.2
6
7 //Toolbox used: Image Processing Desing (IPD)
   Toolbox ver. 8.3.3
8 //           SIVP 0.5.3.2
9
10 //OS used : Windows 10 64 bit
11 //
12 //Reference book name : Digital Image Processing
```

```

13 //book author: Rafael C. Gonzalez and Richard E.
    Woods
14
15 clc //to clear command window.
16 clear all //to kill previously defined variables.
17 xdel(winsid())//to close all currently open figure(s
    ).
18
19 //This code uses camaraman.tif file for processing
20 //SCI + '/contrib/SIVP_0.5.3.2/images/lena.png'
21 img = imread('Fig0419.png'); // Reads the text
    image (Image from References)
22 //title(" Original Image");
23 imshow(img); // Show original
    image.
24 if(size(img,3)>1) then
25     Gray_img= rgb2gray(img); //Converts color
        image to grayscale
26 else
27     Gray_img=img; // if image is 2D
        image
28 end
29 SE=CreateStructureElement('square',3); // create
    structuring element 3x3 square matrix
30
31 Eimg=ErodeImage(Gray_img,SE); // erosion of the
    image
32 pause; // type 'resume'
    command on command prompt
33 imshow(Eimg); // show Resultant
    image
34 //se=[1 1 1;1 1 1;1 1 1];
35 Dimg=DilateImage(Gray_img,SE); // dilation of the
    image
36 pause; // type 'resume'
    command on command prompt
37 imshow(Dimg); // Show Resultant
    Image

```

```
38
39 Oimg=OpenImage(Gray_img,SE); // opening operation
    on the image
40 pause; // type 'resume'
    command on command prompt
41 imshow(Oimg); // Show Resultant
    Image
42
43
44 Cimg=CloseImage(Gray_img,SE); // closing operation
    on the image
45 pause; // type 'resume'
    command on command prompt
46 imshow(Cimg); // Show Resultant
    Image
```

Historically, certain computer programs were written using only two digits rather than four to define the applicable year. Accordingly, the company's software may recognize a date using "00" as 1900 rather than the year 2000.

Figure 3.1: Basic Morphological Operations on Image

Historically, certain computer programs were written using only two digits rather than four to define the applicable year. Accordingly, the company's software may recognize a date using "00" as 1900 rather than the year 2000.

Figure 3.2: Basic Morphological Operations on Image

Experiment: 4

Basic Image Segmentation

check Appendix [AP 10](#) for dependency:

harewood.jpg

Scilab code Solution 4.1 Example 1

```
1
2 // This scilab code is used to perform image
   segmentation using Otsu Thresholding method
3
4 //The scilab environment for this is : Scilab 6.0.1
5
6 //Toolbox used: Image Processing and Computer Vision
   Toolbox ver. 1.1
7
8
9 //OS used : Windows 10 64 bit
10 //
11 //Reference book name : Digital Image Processing
12 //book author: Rafael C. Gonzalez and Richard E.
   Woods
13
14 //clc //to clear command window.
```

```

15 //clear all //to kill previously defined variables.
16 //xdel(winsid())//to close all currently open figure
   (s).
17
18 //This code uses harewood.jpg image file for
   processing
19 //clc;
20 //clear all;
21 //I=imread('nutsBolts.jpg');
22 I=imread('harewood.jpg');           // Reads
   input image
23 //I=rgb2gray(I);
24
25 Ihist=imhist(I);                     //
   Calculate histogram
26 Ihist=Ihist';                       //
   Transpose the histogram output
27 s=length(I);                       //
   Calculates Total number of pixels
28 t=0:255;                           // set
   counter t as 0 to 255
29 sumA=0.0;                          //
   Initialize the sum variable
30 for t=1:256                         //
   Calculate sum of pixels in histogram output
31     sumA = sumA+ t*Ihist(t);
32 end
33 sumB = 0;                          //
   Initialize another variable as summation
34 wB = 0;                            //
   Initialize weight background variable
35 wF = 0;                            //
   Initialize weight foreground variable
36 varMax = 0;                        //
   Initialize maximum value for threshold
37 threshold = 0;                     //
   Initialize threshold to zero
38

```

```

39
40 // This for block calculates threshold using Otsu
    Thresholding method
41 for t=1:256
42     wB = wB + Ihist(t);                // Weight
        Background
43     if (wB == 0)
44         continue;
45     end
46     wF = s - wB;                        // Weight Foreground
47     if (wF == 0)
48         break;
49     end
50     sumB = sumB + (t * Ihist(t));
51     mB = sumB / wB;                    // Mean Background
52     mF = (sumA - sumB) / wF;          // Mean Foreground
53     // Calculate Between Class Variance
54     varBetween = wB * wF * (mB - mF) * (mB - mF);
55     // Check if new maximum found
56     if (varBetween > varMax)
57         varMax = varBetween;
58         threshold = t;
59     end
60 end
61 //n=(threshold - max(I))/ (max(I)-min(I));
62 R=im2bw(I,threshold/256);              //
    Binarize image with given threshold
63 imshow(R);                            //
    Show resultant binarized image
64 //

```

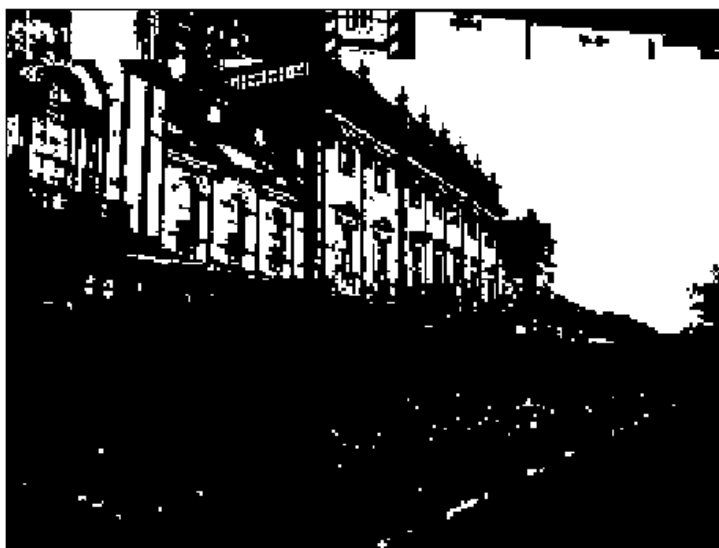


Figure 4.1: Example 1

Experiment: 5

Image Enhancement for Performance

check Appendix [AP 8](#) for dependency:

`lenna1.png`

Scilab code Solution 5.1 Example 1

```
1
2 // This scilab code is to perform image enhancement
   by adjusting the intensity values in the image
3 //which also called as histogram equalization.
4
5 //The scilab environment for this is : Scilab 6.0.1
6
7 //Toolbox used: Image Processing and Computer Vision
   Toolbox ver. 1.1
8
9
10 //OS used : Windows 10 64 bit
11 //
12 //Reference book name : Digital Image Processing
```

```

13 //book author: Rafael C. Gonzalez and Richard E.
    Woods
14
15 clc //to clear command window.
16 clear all //to kill previously defined variables.
17 xdel(winsid())//to close all currently open figure(s
    ).
18
19 //This code uses lenna.png image file for processing
20
21
22 img=imread('lenna.png'); // Reads
    original lenna (color) image
23 grayImg=rgb2gray(img); //
    Converts color image to gray scale image
24 histequalImg=imhistequal(grayImg); //
    Perform image enhancement using intensity
    adjustment i.e. histogram equalization
25 imshow(img); //
    Displays original lenna image
26 figure;imshow(grayImg); //
    Display resultant gray scale image
27 figure;imshow(histequalImg); //
    Displyas resultant histogram equalized image

```



Figure 5.1: Example 1



Figure 5.2: Example 1

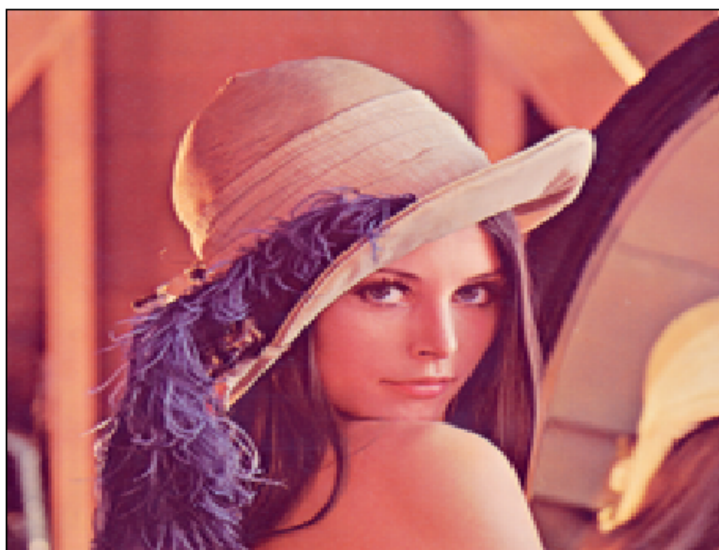


Figure 5.3: Example 1

Experiment: 6

Hough Transform Implementation on Images

check Appendix [AP 9](#) for dependency:

pic26.png

Scilab code Solution 6.1 Example 1

```
1
2 // This scilab code is used to perform Hough
   Transform operation on the image
3 //to extract the features from image.
4
5 //The scilab environment for this is : Scilab 6.0.1
6
7 //Toolbox used: Image Processing and Computer Vision
   Toolbox ver. 1.1
8
9
10 //OS used : Windows 10 64 bit
11 //
12 //Reference book name : Digital Image Processing
```

```

13 //book author: Rafael C. Gonzalez and Richard E.
    Woods
14
15 //clc //to clear command window.
16 //clear all //to kill previously defined variables.
17 //xdel(winsid())//to close all currently open figure
    (s).
18
19 //This code uses an circle image pic26.png file for
    processing
20
21 S = imread('pic26.png'); // Read
    input image using imread function
22 [houghmat, dist, theta] = imhough(S); // Apply
    hough transformation
23 // This
    returns
    hough
    matrix
    houghmat
24 //
    distance
    from
    center
    to
    the
    point
25 // and
    angle
    of
    point
    to
    the

```

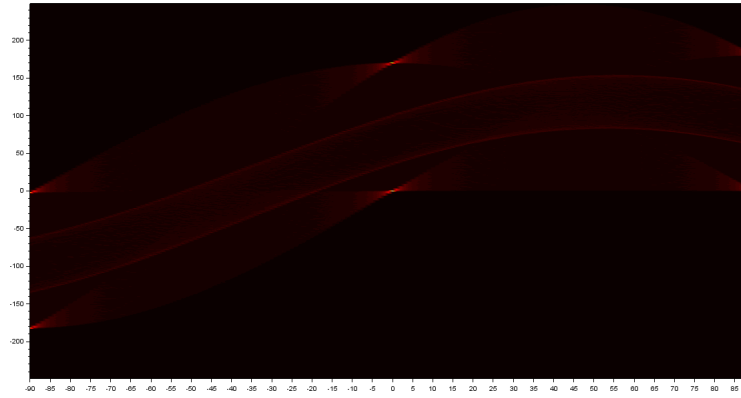


Figure 6.1: Example 1

center

```

26
27 f=scf(); //
    Assign current graphics window handle to f
28 f.color_map=hotcolormap(64); //
    Change colormap for current graphics window
29 Sgrayplot(theta,dist,houghmat',strf="021"); // Plots
    smooth 2D surface

```

Experiment: 7

SIFT Implementation for Images

check Appendix [AP 8](#) for dependency:

`lenna1.png`

Scilab code Solution 7.1 Example 1

```
1
2 // This scilab code is to perform SIFT Transform on
   the image
3 //to extract the features from image.
4
5 //The scilab environment for this is : Scilab 6.0.1
6
7 //Toolbox used: Image Processing and Computer Vision
   Toolbox ver. 1.1
8
9
10 //OS used : Windows 10 64 bit
11 //
12 //Reference book name : Digital Image Processing
```

```

13 //book author: Rafael C. Gonzalez and Richard E.
    Woods
14
15 //clc //to clear command window.
16 //clear all //to kill previously defined variables.
17 //xdel(winsid())//to close all currently open figure
    (s).
18
19 //This code uses lenna.png image file for processing
20
21
22 img=imread('lenna1.png');           // Reads
    original lenna (color) image
23 SIFTfeatures=imdetect_SIFT(img);     //
    Extract features from image using SIFT algorithm
24 imshow(img);                       //
    Displays original lenna image
25 plotfeature(SIFTfeatures);          // Plot
    the features extracted on displayed image

```

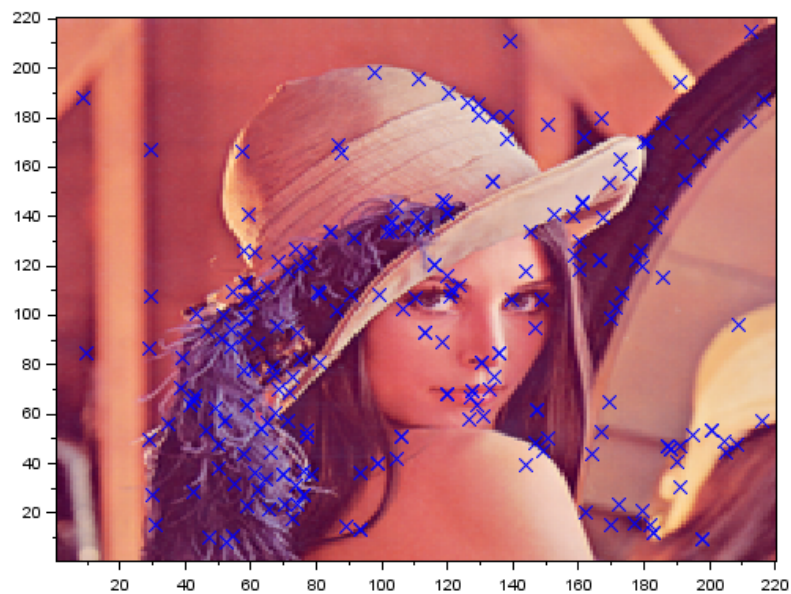


Figure 7.1: Example 1

Experiment: 8

Different Distance Measures implementations

Scilab code Solution 8.1 Example 1

```
1
2 // This scilab code is used to calculate distance
   measure between different vectors
3 //such as euclidean distance , checkerboard distance
   and mahalanobis distance
4
5 //The scilab environment for this is : Scilab 6.0.1
6
7 //OS used : Windows 10 64 bit
8 //
9 //Reference book name : Digital Image Processing
10 //book author: Rafael C. Gonzalez and Richard E.
   Woods
11
12 //clc //to clear command window.
13 clear all //to kill previously defined variables.
14 //xdel(winsid())//to close all currently open figure
   (s).
15
```

```

16 //A = rand(1, 100); //
    Create a vector of random values
17 //B = rand(1, 100); //
    Creates another vector of random values
18
19 A = [5.76 43 34;6.7 32 5;3 3 5;34 12 6]; //
    Create a vector A
20 B = [1.55 5 32;4.5 9 8;2 4 3;3.2 6 7];
    // Create a vector B
21
22 // Euclidean Distance
23 D = sqrt(sum((A - B).^2)) //
    Calculates euclidean distance between A and B
24 Da = sqrt(sum((A - A).^2)) //
    Calculates euclidean distance between A itself
25 Db = sqrt(sum((B - B).^2)) //
    Calculates euclidean distance between B itself
26
27
28 // Checkerboard Distance
29 Dc1 = sum(sum((A-B).^2,2)) //
    Calculates Checkerboard distance between A and B
30 Dc2 = sum(sum((A-A).^2,2)) //
    Calculates Checkerboard distance between A and B
31 Dc3 = sum(sum((B-B).^2,2)) //
    Calculates Checkerboard distance between A itself
32
33
34 // Mahalanobis Distance
35 A = [5.76 43 34;6.7 32 5;3 3 5;34 12 6]; //
    Create a vector A
36 B = [1.55 5 32]; //
    Create a vector B
37 cv = cov(A); //
    Calculate covariance matrix of A
38 mu = mean(A,1); //
    Calucated mean value of A
39 Dm = (B-mu)*inv(cv)*(B-mu)'; //

```

```

    Calculates Mahalanobis distance between A and B
40 //out=[D Da Db Dc1 Dc2 Dc3 Dm]
41 disp([D Da Db Dc1 Dc2 Dc3 Dm]);           //
    Displays output on Scilab console
42
43 //////////////////////////////////// OUTPUT
    ////////////////////////////////////
44
45 //54.774119    0.    0.    3000.2041    0.    0.
    11.17056

```

Experiment: 9

Basic Shape Analysis Implementation

check Appendix [AP 7](#) for dependency:

`shapes.jpg`

Scilab code Solution 9.1 Example 1

```
1
2 // This scilab code is used to perform different
   edge detection methods on images
3 //which can be used in shape analysis.
4
5 //The scilab environment for this is : Scilab 6.0.1
6
7 //Toolbox used: Image Processing and Computer Vision
   Toolbox ver. 1.1
8
9
10 //OS used : Windows 10 64 bit
11 //
12 //Reference book name : Digital Image Processing
```

```

13 //book author: Rafael C. Gonzalez and Richard E.
    Woods
14
15 //clc //to clear command window.
16 //clear all //to kill previously defined variables.
17 //xdel(winsid())//to close all currently open figure
    (s).
18
19 //This code uses shapes.jpg image file for
    processing
20
21
22 img = imread('shapes.jpg');
    // Reads input image shapes.jpg
23 img = rgb2gray(img);
    // Converts input image to gray scale
24 clf
    // Clears figure handle
25 imshow(img);
    // Show gray scale image (Result file
    exp_9_1_result_1.png)
26
27 e = edge(img);
    // This performs edge detection operation with
    sobel, thresh = 0.5
28 figure(2);
    // Opens new figure
29 imshow(e)
    // Show result image (Result file
    exp_9_1_result_2.png)
30
31 e = edge(img, 'prewitt'); // thresh = 0.5
    // Applied prewitt edge detection method
32 figure(3);
33 imshow(e)
    // Show result image (Result file
    exp_9_1_result_3.png)
34

```

```
35 e = edge(img, 'canny', [0.06 0.2]);  
    // Applies canny edge detection method  
36 figure(4);  
37 imshow(e)  
    // Show result image (Result file  
    exp_9_1_result_4.png)  
38  
39 e = edge(img, 'fftderiv', 0.4);  
    // Applies FFT gradient method; 0.4 threshold  
40 figure(5);  
41 imshow(e)  
    // Show result image (Result file  
    exp_9_1_result_5.png)
```

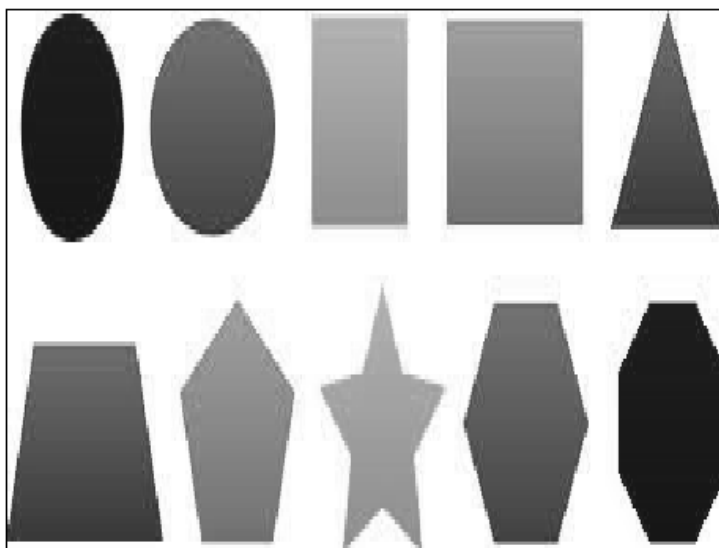


Figure 9.1: Example 1

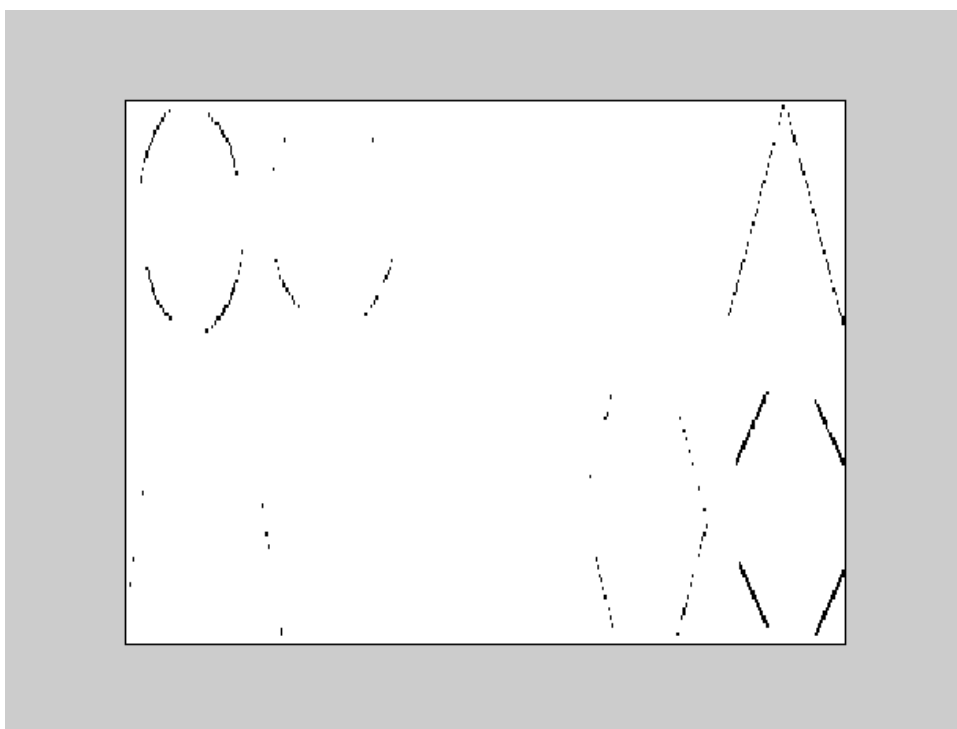


Figure 9.2: Example 1

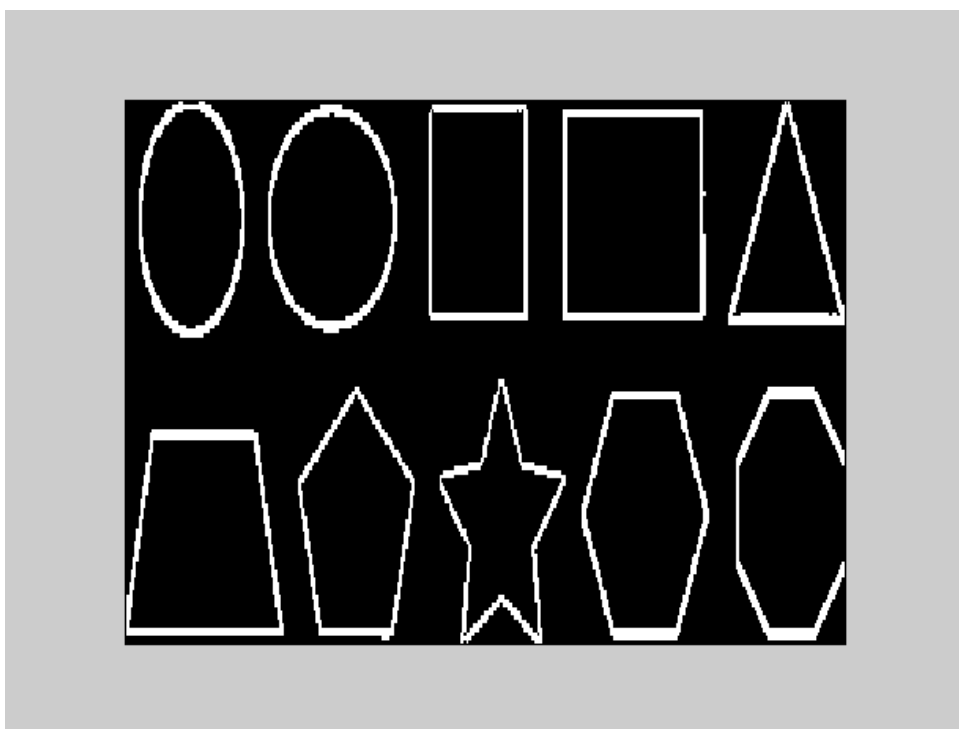


Figure 9.3: Example 1

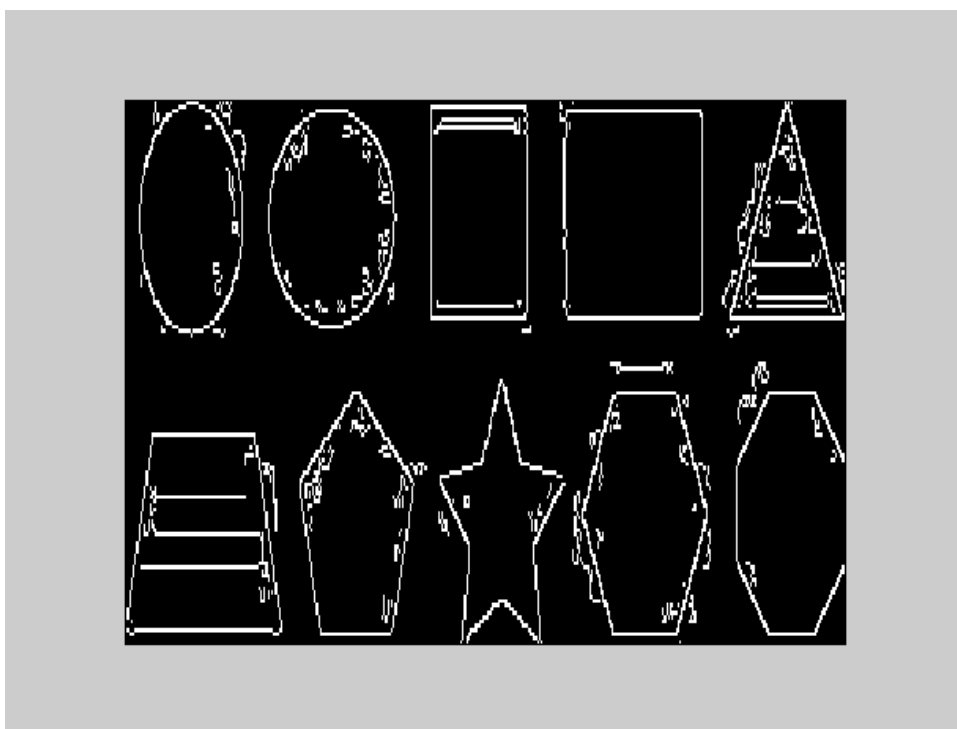


Figure 9.4: Example 1

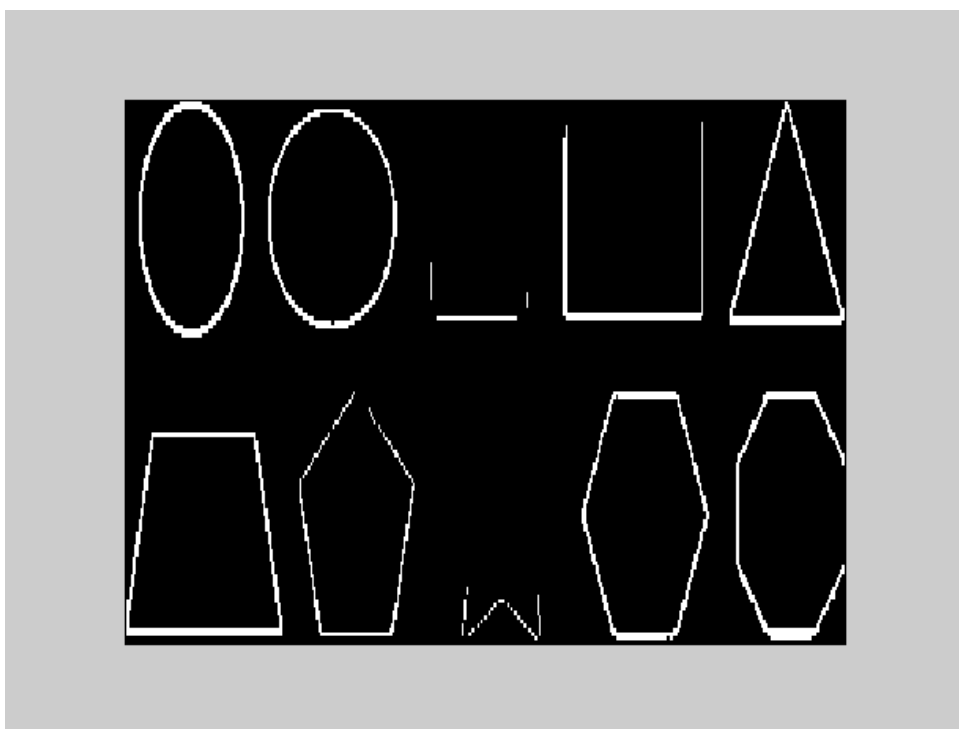


Figure 9.5: Example 1

Experiment: 10

Feature Extraction from Images

check Appendix [AP 7](#) for dependency:

`shapes.jpg`

Scilab code Solution 10.1 Example 1

```
1
2 // This scilab code is used to extract features from
   images
3 //using FAST algorithm (this algorithm is best
   suitable for corner detection)
4
5 //The scilab environment for this is : Scilab 6.0.1
6
7 //Toolbox used: Image Processing and Computer Vision
   Toolbox ver. 1.1
8
9
10 //OS used : Windows 10 64 bit
11 //
12 //Reference book name : Digital Image Processing
13 //book author: Rafael C. Gonzalez and Richard E.
   Woods
```

```

14 // Rosten. Machine Learning for High-speed Corner
    Detection , 2006.
15
16 //clc //to clear command window.
17 //clear all //to kill previously defined variables.
18 //xdel(winsid())//to close all currently open figure
    (s).
19
20 //This code uses shapes.jpg image file for
    processing
21
22
23 //S = imcreatechecker(8,8,[1 0.5]);
24 S=imread('shapes.jpg');
                                     // Reads an input
    image
25 fobj = imdetect_FAST(S);
                                     // Extract
    features from image using FAST algorithm
26 imshow(S);
                                     //
    Displays original image
27 plotfeature(fobj);
                                     // Plots the
    extracted features on the image shown in figure

```

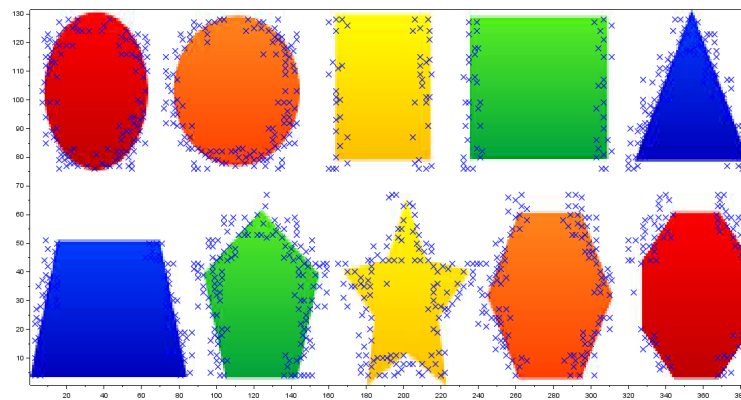


Figure 10.1: Example 1

Experiment: 11

Basics of OCR

check Appendix [AP 4](#) for dependency:

`number.png`

check Appendix [AP 5](#) for dependency:

`region.png`

check Appendix [AP 6](#) for dependency:

`text.png`

Scilab code Solution 11.1 Example 1

```
1
2 // This scilab code is used to perform basic
   operation such as binarization of images
3 //that contains text with random value of threshold
4
5 //The scilab environment for this is : Scilab 6.0.1
6
7 //Toolbox used: Image Processing and Computer Vision
   Toolbox ver. 1.1
8
```

```

9
10 //OS used : Windows 10 64 bit
11 //
12 //Reference book name : Digital Image Processing
13 //book author: Rafael C. Gonzalez and Richard E.
    Woods
14
15 //clc //to clear command window.
16 clear all //to kill previously defined variables.
17 //xdel(winsid())//to close all currently open figure
    (s).
18
19 //This code uses text.png, region.png and number.png
    image files for processing
20
21 img = imread('text.png');           // Reads
    input image
22 threshold=0.93;                     // set a
    random threshold value
23 imagen =~im2bw(img,threshold);       // Convert
    input image to binarized image with given
    threshold
24                                     // and also
                                         negate
                                         the
                                         output
25 imshow(imagen);                     // Shows the
    resultant image (exp_11_1_result_1.png)
26
27 img = imread('number.png');         // Reads
    input image
28 threshold=0.93;                     // set a
    random threshold value
29 imagen =~im2bw(img,threshold);       // Convert
    input image to binarized image with given
    threshold
30                                     // and also
                                         negate

```

```

31 imshow(imagen);                                the
                                                    output
                                                    // Shows the
    resultant image (exp_11_1_result_2.png)
32
33 img = imread('region.png');                    // Reads
    input image
34 threshold=0.93;                                // set a
    random threshold value
35 imagen =~im2bw(img,threshold);                  // Convert
    input image to binarized image with given
    threshold
36                                                    // and also
                                                    negate
                                                    the
                                                    output
37 imshow(imagen);                                // Shows the
    resultant image (exp_11_1_result_3.png)

```

YES, it's true M Ivan WINKLER! R
guaranteed chances to win a shar
Lump-Sum, TAX-FREE CASH. Ph
Australian Lotto Multi-Millionaire,
granted you WIN-PLUS PRIORITY.

Figure 11.1: Example 1

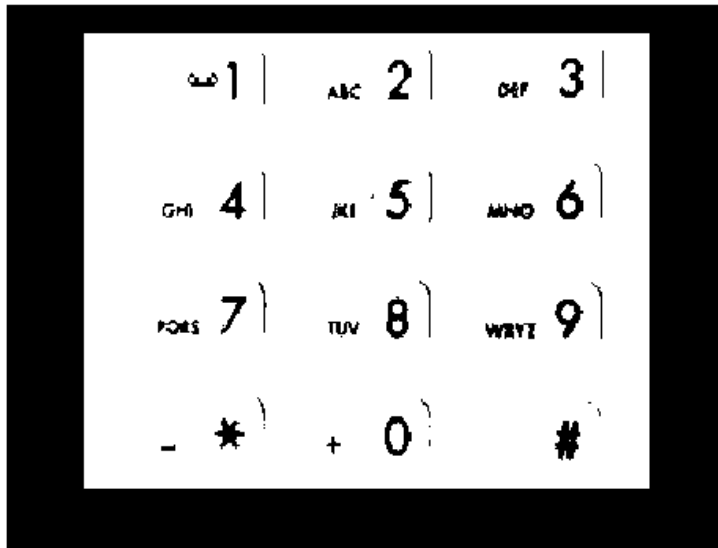


Figure 11.2: Example 1

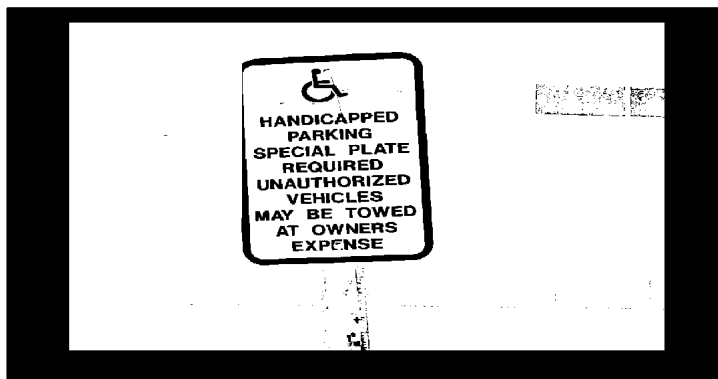


Figure 11.3: Example 1

Experiment: 12

Face Detection Algorithm Implementation

check Appendix [AP 1](#) for dependency:

`group.png`

check Appendix [AP 2](#) for dependency:

`kalpana.jpg`

check Appendix [AP 3](#) for dependency:

`random.jpg`

Scilab code Solution 12.1 Example 1

```
1
2 // This scilab code is used to detect the human face
   in the image
3
4 //The scilab environment for this is : Scilab 5.5.2
5
6 //Toolbox used: SIVP 0.5.3.2
7
```

```

8
9 //OS used : Windows 7 32 bit
10 //
11 //Reference book name : Digital Image Processing
12 //book author: Rafael C. Gonzalez and Richard E.
    Woods
13
14 //clc //to clear command window.
15 //clear all //to kill previously defined variables.
16 //xdel(winsid())//to close all currently open figure
    (s).
17
18 //This code uses different images having human faces
    for processing
19 // Images used:
20 //   kalpana.jpg
21 //   random.jpg
22 //   group.png
23
24 img = imread('kalpana.jpg');
                                // Reads input image
25
26 faces = detectfaces(img);
    // Uses function in the toolbox to detect faces
27 [m,n] = size(faces);
    // Calculates number of faces in image
28
29 for i=1:m,
    // This block draws rectangles for each face
    images detected
30     img = rectangle(img, faces(i,:), [0,255,0]);
31 end;
32
33 imshow(img);
    // Shows resultant images with the rectangle
    around face images
34
35 pause;

```

```

        // Pause till user enters resume command at
        command window
36 img = imread('random.jpg');
    // Reads input image
37
38 faces = detectfaces(img);
    // Uses function in the toolbox to detect faces
39 [m,n] = size(faces);
    // Calculates number of faces in image
40
41 for i=1:m,
    // This block draws rectangles for each face
    images detected
42     img = rectangle(img, faces(i,:), [0,255,0]);
43 end;
44
45 imshow(img);
    // Shows resultant images with the rectangle
    around face images
46 pause;
    // Pause till user enters resume command at
    command window
47 img = imread('group.png');
    // Reads input image
48
49 faces = detectfaces(img);
    // Uses function in the toolbox to detect faces
50 [m,n] = size(faces);
    // Calculates number of faces in image
51
52 for i=1:m,
    // This block draws rectangles for each face
    images detected
53     img = rectangle(img, faces(i,:), [0,255,0]);
54 end;
55
56 imshow(img);
    // Shows resultant images with the rectangle

```



Figure 12.1: Example 1

around face images

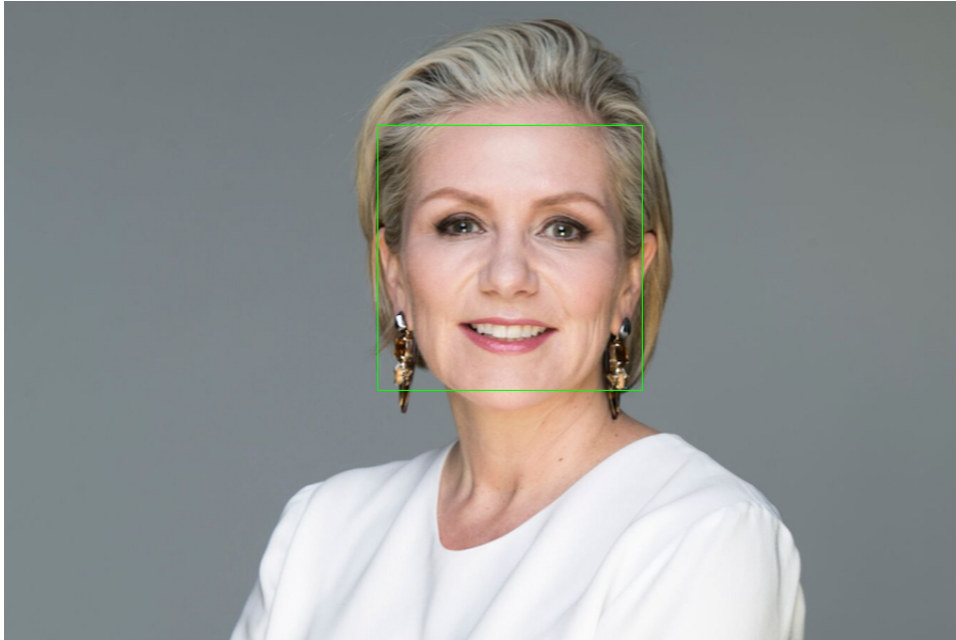


Figure 12.2: Example 1

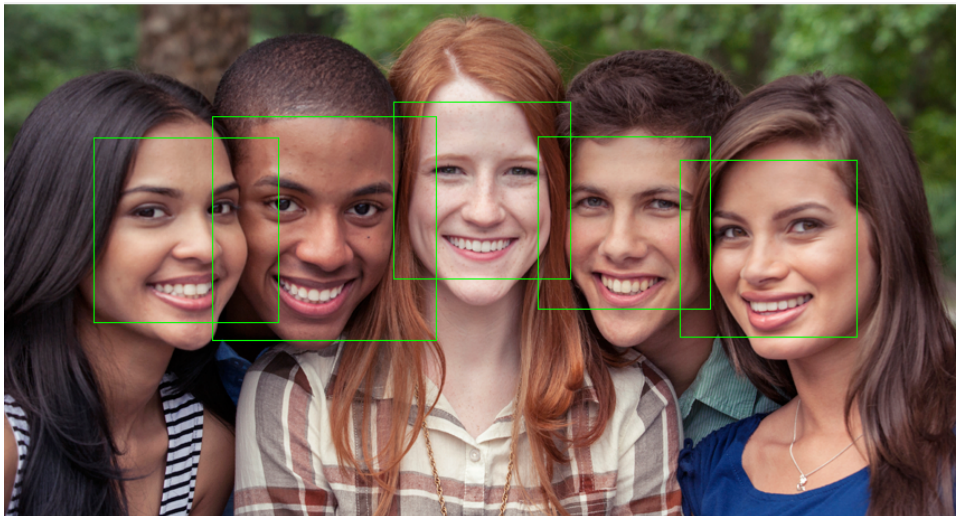


Figure 12.3: Example 1

Appendix



groupimage



kalpanachawla



domimage

ran-



numberimage



regionimage

YES, it's true M Ivan WINKLER! R
guaranteed chances to win a shar
Lump-Sum, TAX-FREE CASH. Plu
Australian Lotto Multi-Millionaire,
granted you WIN-PLUS PRIORITY.

tex-

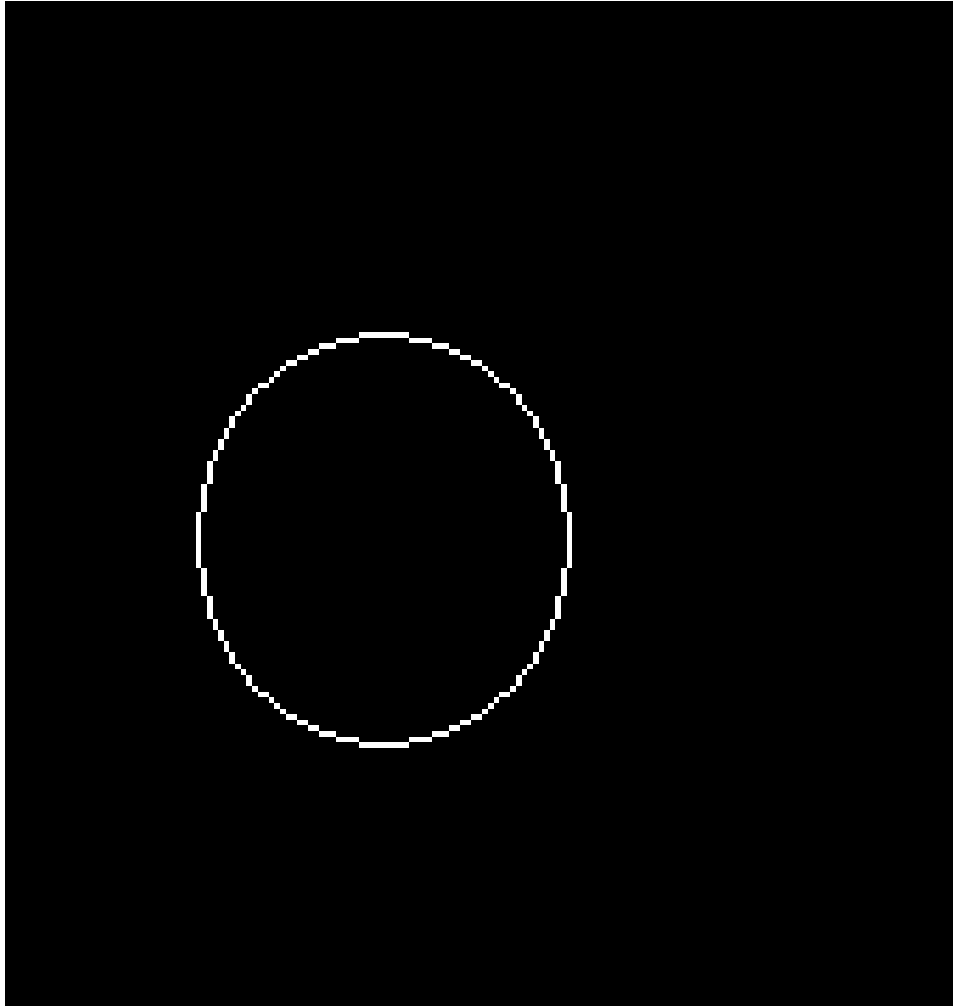
timage



shape



lennaimage



cleimage

cir-



harewood

Historically, certain computer programs were written using only two digits rather than four to define the applicable year. Accordingly, the company's software may recognize a date using "00" as 1900 rather than the year 2000.

Text

Image



inputimage