Object Recognition and Feature Detection in Images Using MATLAB

Sadhana Venkataraman, Farragut High School
Yukai Tomsovic, West High School

Ms. Gangotree Chakma

CURENT Young Scholars Program
18 July 2016
Min Kao Building, University of Tennessee
Topics

➤ What is image processing?
➤ Edge Detection
  • Measuring properties
➤ Shape Detection
➤ Color Segmentation
  • Manually
  • K-Means Clustering
➤ Feature Detection
  • Comparison between images
What is Image Processing?

- Method to perform operations on an image, in order to get an enhanced image or to extract useful information

Basic Steps:

- Acquire and import image
- Analyze and manipulate image
- Output -- image, component of image, or report based on image analysis

---

1. Creating a binary image
   - Otsu’s Method
   - Function `im2bw`
2. Labeling connected components
   - Function `bwlabel`
3. Measuring Properties
   - Area
   - Identify types of objects
triangle = 0;
star = 0;
square = 0;
circle = 0;
for i = 1:num
    area(i) = bwarea(L==i);
    if area(i)>10000; %sets a range of area for each shape
        square = square + 1; %counts no. of objects in this range as a square
    end
    if area(i)>7000 && area(i)<9000;
        triangle = triangle + 1;
    end
    if area(i)<5500;
        star = star + 1;
    end
    if area(i)<6000 && area(i)>5500;
        circle = circle + 1;
    end
end
shapes = [square triangle star circle]; %displays shape count as an array
disp(shapes);

shapes =
    1 1 1 1

figure 1: original image, gray scale image

figure 2: binary image
Shape Detection

➤ Use Code `imdistline` to Measure Radii of Circle(s)
   • Radii of circles are measured by the number of pixels

➤ Set a Range From the Smallest to Largest Radius
   • Use `imfindcircles` code

➤ Color of Shape
   • Adjust object polarity to ‘bright’ or ‘dark’ depending on the background color

```matlab
% objects darker than background
[centers, radii] = imfindcircles(I,[11 15],...'ObjectPolarity','dark',... 'Sensitivity', 0.99,'Method','twostage');

% objects lighter than background
[centersBright, radiiBright] = imfindcircles(I,[11 15],...'ObjectPolarity', ... 'bright','Sensitivity',0.982);
```
clc;
close all;

%% read and show the image
I = imread('SmallerDots.JPG');
imshow(I)

%% measure the radii of the circles
d = imdistline;
delete(d);

%% find circles based on a range of radii
[c, radii] = imfindcircles(I,[1 15], 'ObjectPolarity','dark', ... 'Sensitivity', 0.99, 'Method','twostage');% set sensitivity level
[c, radiiB, metricB] = imfindcircles(I,[1 15], ... 'ObjectPolarity','bright','Sensitivity',0.977,'EdgeThreshold',0.25);
imshow(I);
hB = viscircles(c, radiiB, 'Color','b');
h = viscircles(c, radii);
Practical Uses of Shape Detection

➤ Astronomy
  • Detecting patterns in waveforms
  • Classification of stars and galaxies

➤ Medicine
  • Medical imaging to detect sickness based on cell shape

➤ Archaeology
  • To find missing or similar objects
Color Segmentation

- **Pixel ranges**
  - Using for loop to iterate through each row of the pixel matrix
  - If loop to group centers into colors by the range of pixel

```matlab
rgb = imread('blackdots.jpg');
figure
imshow(rgb);

% find and draw circles
[centersBright, radiiBright] = imfindcircles(rgb,[15 20], 'ObjectPolarity', 'bright', 'Sensitivity',0.96,'EdgeThreshold',0.25);
figure
imshow(rgb);
holdBright = viscircles(centersBright, radiiBright, 'Color','b');

% count circles based on the pixel of their center
yellow = 0;
green = 0;
blue = 0;
red = 0;
r = centersBright(:,2);
c = centersBright(:,1);

% inpixel returns an array with the pixel of CentersBright
pixel = inpixel(rgb,c,r);
for i = 1:1:27
if pixel(i,3) > 81 && pixel(i,3) < 105;
    green = green + 1;
elseif pixel(i,3) > 68 && pixel(i,3) < 81;
    red = red + 1;
elseif pixel(i,3) > 200;
    blue = blue + 1;
elseif pixel(i,3) < 65;
    yellow = yellow + 1;
end
end

colors = [yellow green red blue]; % create array with the no. of each color
```
Counting Colors

>> colors

6  9  5  7
yellow  green  red  blue
Color Segmentation: K-Means Clustering

1. RGB to L*a*b color space
2. Classify colors in ‘a*b*’ space with K-means clustering
3. Label pixels and separate objects
K-Means Clustering

Figure 1. Original Image of watercolor paint

Figure 2. First cluster of color

Figure 3. Second cluster of color

Figure 4. Third cluster of color
Practical Uses of Color Segmentation

➤ Biology
- Distinguish different types of tissue, cells, tumor borders, etc.
- Medical thermography in detecting breast cancer, tendon/ligament injuries, areas of poor circulation
Feature Detection

➤ Distort Image by Rotating

```matlab
% read and distort the image
I = imread('sunsphere.jpg');% read the image
original = rgb2gray(I);% change the image to greyscale
figure;
imshow(original);% show the image
scale = 1.3;% choose the scale of the image
J = imresize(original, scale);% resize the image
theta = 31;% angle by which to rotate
distorted = imrotate(J, theta);% rotate the image
figure
imshow(distorted)% show the distorted/rotated image
```

➤ Locate Features in Each Image

```matlab
% identify and find location of features in each picture
ptsOriginal = detectSURFFeatures(original);% uses the Speeded-Up Robust Feature (SURF) to find algorithms for original shapes
ptsDistorted = detectSURFFeatures(distorted);% uses the SURF feature to find algorithms for distorted shapes
[featuresOriginal, validPtsOriginal] = extractFeatures(original, ptsOriginal);% identifies the features in the original image from pixels surrounding on
[featuresDistorted, validPtsDistorted] = extractFeatures(distorted, ptsDistorted);% identifies the features in the distorted image using the same method.
indexPairs = matchFeatures(featuresOriginal, featuresDistorted);% finds matching features between the two feature sets given
matchedOriginal = validPtsOriginal(indexPairs(:,1));% find location of matching points in original image
matchedDistorted = validPtsDistorted(indexPairs(:,2));% find location of matching points in distorted image
figure
```
Feature Detection, Cont’d

➤ Match the Corresponding Points on Both Images

```matlab
% match corresponding points and compare features
showMatchedFeatures(original, distorted, matchedOriginal, matchedDistorted) % displays an overlay of the original and distorted images
title('Candidate matched points (including outliers)') % title the overlayed picture
[tform, inlierDistorted, inlierOriginal] = estimateGeometricTransform(matchedDistorted, matchedOriginal, 'similarity');
figure
showMatchedFeatures(original, distorted, inlierOriginal, inlierDistorted) % displays a color-coded overlay of the inliers
title('Matching points (inliers only)') % title the overlayed picture
legend('ptsOriginal', 'ptsDistorted')
outputView = imref2d(size(original)); % references 2-D images to world coordinates.
recovered = imwarp(distorted, tform, 'OutputView', outputView); % transforms the distorted image according to the geometric transformation
figure
imshowpair(original, recovered, 'montage') % compare the original and recovered images side-by-side.
numofpairs = max(size(indexPairs)); % determine the largest number of index pairs possible
```

➤ Write an If Statement to Determine Whether the Images are the Same

```matlab
% detect whether the picture are the same
if numofpairs < 25
    disp('Error: Different Pictures. ');
else
    disp('Same Picture! ')
end
```
Feature Detection, Cont’d
Feature Detection, Cont’d
Feature Detection, Cont’d

Command Window
Same Picture!
Feature Detection, Cont’d

Command Window
Error: Different Pictures.
Conclusion

➤ Our research -- foundations for more advanced technology

➤ Further applications of image processing
  • Medical field
    ▪ PET Scans
    ▪ X-Ray Imaging
  • Space Image Processing
    ▪ Hubble Space Telescope Images
  • Automatic Character Recognition
    ▪ Scanning license plates and zip codes
Questions and Answers
Acknowledgements

This work was supported primarily by the ERC Program of the National Science Foundation and DOE under NSF Award Number EEC-1041877.

Other US government and industrial sponsors of CURENT research are also gratefully acknowledged.