

T-61.5070 COMPUTER VISION, Exercise 8/08

1.

Hough transform is described in Sec. 5.2.6, pp. 163–173. First, the edges in the image are detected, e.g., using gradient operators. Algorithm 5.15, curve detection with the Hough transform (p. 167), may then be applied to the edge image. The arbitrary curve equation, $f(\mathbf{x}, \mathbf{a}) = 0$, is now a line equation

$$y = kx + q. \quad (1)$$

1. Quantize the parameter space (k', q') within the limits of parameters k and q using, for instance, the definitions

$$k' = \min k, \min k + \Delta k', \dots, \max k - \Delta k', \max k;$$

$$q' = \min q, \min q + \Delta q', \dots, \max q - \Delta q', \max q.$$

2. Form a two-dimensional accumulator array $A(k', q')$ whose size is the number of elements in k' multiplied by the number of elements in q' : $\# \{A(k', q')\} = \# \{k'\} \times \# \{q'\}$. Set all elements of $A(k', q')$ to zero.
3. For each point (x, y) in the edge image, form the parameter line equation

$$q = -xk + y, \quad (2)$$

and increase accumulator cells $A(k', q')$ for all k' and q' that are hit by the line of Eq. 2 (see Fig. 1):

$$A(k', q') = A(k', q') + 1. \quad (3)$$

4. Local maxima in the array $A(k', q')$ correspond to lines that are present in the original image.

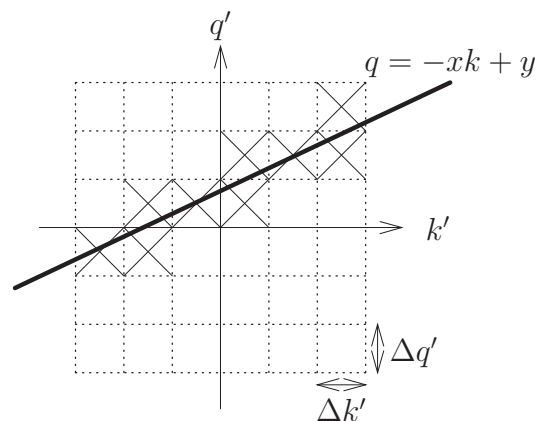


Figure 1: Quantized parameter space.

2.

The Hough transform for the detection of circles is explained in the textbook on p. 168. The general curve equation, $f(\mathbf{x}, \mathbf{a}) = 0$, becomes the circle equation

$$(x - x_0)^2 + (y - y_0)^2 = r^2. \quad (4)$$

1. Form a quantized parameter space (x'_0, y'_0, r') .
2. Form a three-dimensional accumulator array $A(x'_0, y'_0, r')$. Set all elements of $A(x'_0, y'_0, r')$ to zero.
3. For each point (x, y) in the edge image, update those accumulator cells $A(x'_0, y'_0, r')$ for which the point (x'_0, y'_0) is at the distance r' from the point (x, y) .
4. The local maxima in the array $A(x'_0, y'_0, r')$ correspond to the circles in the original image.

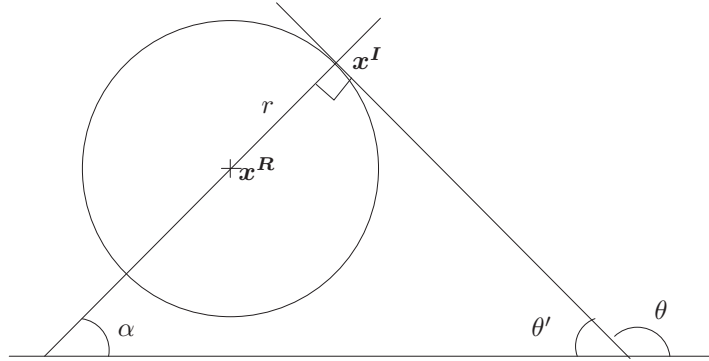


Figure 2: Geometry for the description of a circle with an R-table

The generalized Hough transform is explained on pp. 171–172. Follow the Algorithm 5.16:

1. An R-table for a sample circle with radius r should be constructed. A natural reference point for a circle is the center

$$\mathbf{x}^R = [x_0 \ y_0]^T. \quad (5)$$

A line that goes through the center in direction α is a normal of the circle, Fig. 2. The tangent at the intersection point, \mathbf{x}^I , gives the edge direction θ . The directions α and θ are related by the equation

$$\alpha = 90^\circ - \theta' = 90^\circ - (180^\circ - \theta) = \theta - 90^\circ. \quad (6)$$

Because α is a simple function of θ and r is constant for all θ they need not be tabulated. The R-table would thus contain only one element: the radius of the sample circle. Therefore, the description of a circle does not require an R-table.

2. To detect circles with different radii, the radius r is replaced by a set of quantized radii $r_i \in]0, R]$. Form an array $A(x_0, y_0, r_i)$ which represents the potential reference points. Set the values in the array A to zero.
3. Determine the edge direction θ for each edge pixel (x, y) in the thresholded gradient image. The potential reference points \mathbf{x}^R are on the line in the direction $\theta - 90^\circ$ at the distances r_i ; there are two potential reference points for each r_i . Increase all $A(x_0, y_0, r_i)$, where

$$x_0 = x + r_i \cos(\theta - 90^\circ), \quad y_0 = y + r_i \sin(\theta - 90^\circ) \quad (7)$$

or

$$x_0 = x - r_i \cos(\theta - 90^\circ), \quad y_0 = y - r_i \sin(\theta - 90^\circ). \quad (8)$$

4. The centers and radii of the circles are given by the local maxima in the array A .

Compare this result with Eq. (5.28) in the textbook. The procedure was reduced into the Hough transform which takes the edge direction into account. The Hough transform is a special case of the generalized Hough transform.

3.

The chain-number refers to the normalized chain code which is explained in Sec. 6.2.1, p. 236 in the textbook. Fig. 3 represents a spatially quantized circle and octagon. The chain-number

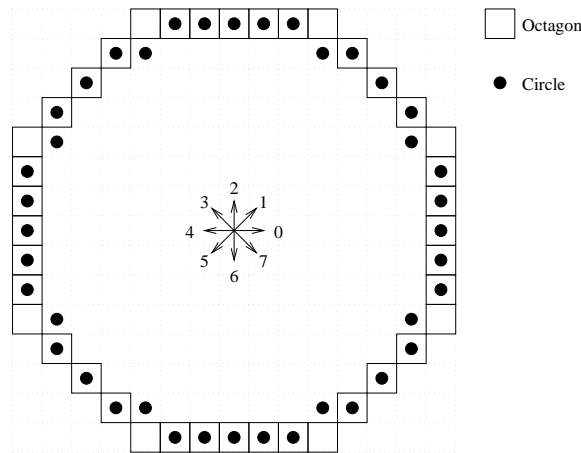


Figure 3: Circle and octagon for chain code formation.

of the circle is

1000070776766665655454444343323222212110.

The chain-number of the octagon is

10000007777666665555444444333322222111.

Levenshtein distance is defined as the smallest number of deletions, insertions, and substitutions necessary to convert one string into the other. Eight substitutions are needed to convert the chain-number of the circle to that of the octagon.

4.

Fourier transforms of boundaries are explained in Sec. 6.2.3, pp. 240–242. Two rotated and translated versions of the same irregular shape are depicted in Fig. 4.

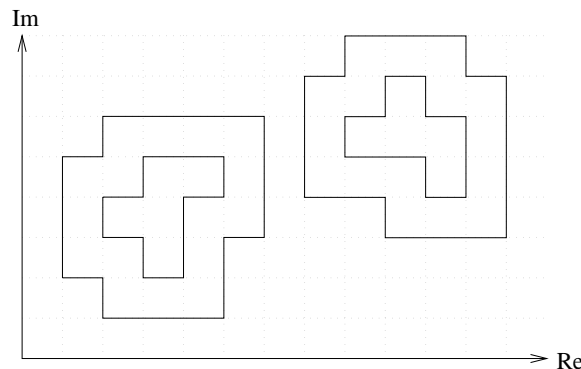


Figure 4: A shape with two different rotations and translations.

The boundary of the image on the left is given by the coordinates

$$B_A = (3, 6), (4, 6), (5, 6), (6, 6), (6, 5), (6, 4), (5, 4), (5, 3), \dots$$

$$(5, 2), (4, 2), (3, 2), (3, 3), (2, 3), (2, 4), (2, 5), (3, 5). \tag{9}$$

Similarly, the boundary of the object on the right is given by the coordinates

$$B_B = (9, 8), (10, 8), (11, 8), (11, 7), (12, 7), (12, 6), (12, 5), (12, 4), \dots \\ (11, 4), (10, 4), (10, 5), (9, 5), (8, 5), (8, 6), (8, 7), (9, 7). \quad (10)$$

The coordinates define for both images a set of sixteen complex numbers. The Fourier descriptors of the boundaries are computed from the discrete Fourier transform of the complex numbers. Absolute values of the Fourier descriptors are computed in *Mathematica*:

```
In[21]:= zA = {3 + 6 I, 4 + 6 I, 5 + 6 I, 6 + 6 I,
6 + 5 I, 6 + 4 I, 5 + 4 I, 5 + 3 I,
5 + 2 I, 4 + 2 I, 3 + 2 I, 3 + 3 I,
2 + 3 I, 2 + 4 I, 2 + 5 I, 3 + 5 I}
```

```
In[22]:= zB = {9 + 8 I, 10 + 8 I, 11 + 8 I, 11 + 7 I,
12 + 7 I, 12 + 6 I, 12 + 5 I, 12 + 4 I,
11 + 4 I, 10 + 4 I, 10 + 5 I, 9 + 5 I,
8 + 5 I, 8 + 6 I, 8 + 7 I, 9 + 7 I}
```

```
In[23]:= Abs[Fourier[zA]] // N
```

```
Out[23]= {22.9837, 7.90317, 0.270598, 0.587938,
> 0.707107, 0.527311, 0.270598, 0.13795,
> 0.5, 0.390655, 0.653281, 0.392847,
> 0., 0.330522, 0.653281, 0.69352}
```

```
In[24]:= Abs[Fourier[zB]] // N
```

```
Out[24]= {47.0771, 7.90317, 0.270598, 0.587938,
> 0.707107, 0.527311, 0.270598, 0.13795,
> 0.5, 0.390655, 0.653281, 0.392847,
> 0., 0.330522, 0.653281, 0.69352}
```

Excluding the DC-components, the Fourier descriptors for the two image boundaries are identical. The example demonstrates that Fourier descriptors are suitable for rotation and translation invariant object recognition.