CHAPTER 5

Edge Detection Algorithms

Contents

- **INTRODUCTION**
- **EDGE DETECTION**
- **EDGE THRESHOLDING**
- **HOUGH TRANSFORM**
- **EDGE-FOLLOWING ALGORITHMS**

Introduction

An edge can be considered as:

The border between two homogeneous image regions having different illumination intensities.

Edges are useful for image analysis, object identification and image filtering.

Introduction

Edge detectors can be grouped into two classes:

- ***Local techniques***

  Use of operators on local image neighborhoods

- ***Global techniques***

  Use of global information and filtering methods

Edge detection

Local image differentiation techniques can produce edge detector operators :

The image gradient
(local intensity variations)

$$\nabla f(x, y) = [\frac{\partial f}{\partial x} \quad \frac{\partial f}{\partial y}]^T \stackrel{\Delta}{=} [\, f_x \quad f_y \,]^T$$

Its magnitude
(edge detector)

$$e_{(x, y)} = \sqrt{f_x^2(x, y) + f_y^2(x, y)}$$

Alternative to magnitude

$$e(x, y) = |f_x(x, y)| + |f_y(x, y)|$$

Direction angle
(local edge direction)

$$\varphi(x, y) = \arctan(\frac{f_y}{f_x})$$

Edge detection

Gradient estimates can be obtained by using gradient operators of the form:

$$\hat{f}_x = w_1^T x$$

$$\hat{f}_y = w_2^T x$$

where x is the vector containing image pixels in a local image neighborhood. Weight vectors w_1, w_2 are described by gradient masks.

Edge detection

Gradient masks examples:

-1	0	1
-1	0	1
-1	0	1

1	1	1
0	0	0
-1	-1	-1

Prewitt edge detector masks

-1	0	1
-2	0	2
-1	0	1

1	2	1
0	0	0
-1	-2	-1

Sobel edge detector masks

Edge templates are masks that can be used to detect edges along different directions. Such masks of size 3X3 are:

-1	0	1
-1	0	1
-1	0	1

1	1	1
0	0	0
-1	-1	-1

0	1	1
-1	0	1
-1	-1	0

1	1	0
1	0	-1
0	-1	-1

Kirsch edge detector masks

Edge detection

Edge detection can be performed in the following way. All templates are applied to each image pixel. The template that produces the maximal output is the winner:

$$e_{(x,\,y)} = \left|w_i^T x\right| \quad \text{if} \quad \left|w_i^T x\right| \geq \left|w_j^T x\right|, \qquad j = 1,\ldots n$$

where w_i, $i=1,\ldots,n$ is the weight vector associated with each template.

The corresponding output $|w_i^T x|$ is a measure of confidence of the edge detector output. When is close to zero, no edge is present at that pixel location.

Edge detection

Edge detection using the ***Laplace operator***.

$$\nabla^2 f(x, y) = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

Edges - correspond to large image changes

Zero-crossings of second-order derivatives

Maxima or minima of first-order derivatives

Edge detection

- Second-order differentiation tends to enhance image noise.

- The Laplacian operator creates several false edges, especially in areas where the image variance is small.

- One method to reduce its noise sensitivity is to consider zero-crossings only in areas where the local variance $ó^2(i,j)$ is large.

- Local data dispersion measures can be used as edge detector. ***Local variance*** and ***local image range*** can be used as edge detectors.

Edge thresholding

When the edge detector output is large, a local edge is present. This can be found by ***thresholding*** :

$$E(k,l) = \begin{cases} 1 & \text{if} \quad e(k,l) \geq T \\ 0 & \text{otherwise} \end{cases}$$

The threshold T can be chosen from the edge detector output histogram, so that only a small percentage of the pixels are above it.

Hough transform

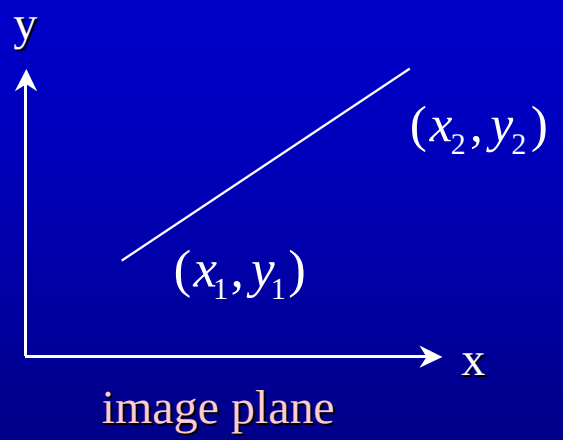
Edge detector output thresholding produces a binary image having 1s at edge locations. This image can be further processed to produce more useful information that can be used in the detection of shapes.

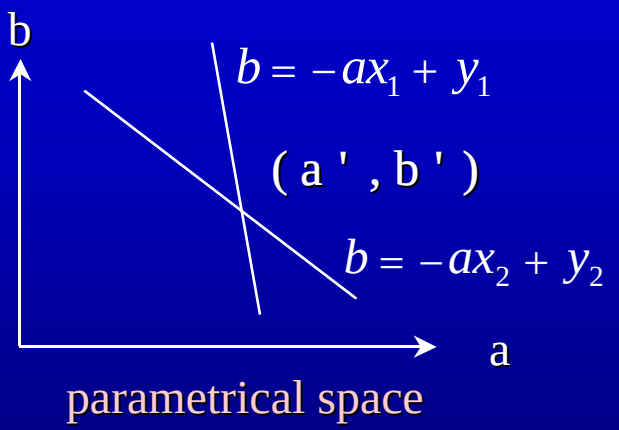
The ***Hough Transform*** uses a parametric description of simple geometrical shapes (curves) in order to reduce the computational complexity of the search space.

Hough transform

The parametric description of straight lines is a linear equation:

$$y=ax+b.$$





Hough transform

A simple procedure for straight line detection is the following. The parameter space is discretized and a parameter matrix P(á,b), $á_1<á<á_K$, $b1<b<b_L$ is formed. For every pixel (x_i,y_i) that possesses value 1 at the binary edge detector output, the equation $b=-áx_i+y_i$ is formed. For every parameter value á, $á_1<á<á_K$, the corresponding parameter b is calculated and the appropriate parameter matrix element P(a,b) is increased by 1:

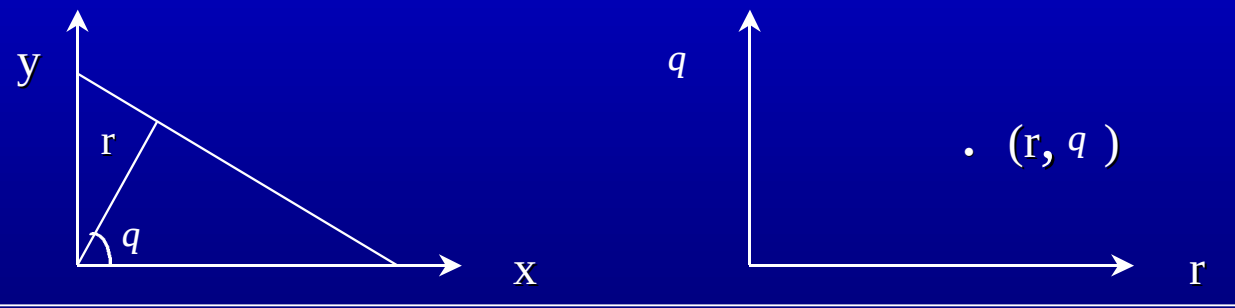
$$P(á,b) = P(á,b) + 1$$

This process is repeated until the entire binary image is scanned.

Hough transform

The parametric model has some difficulties in the representation of vertical straight lines, because the parameter *a* must tend to infinity. A polar representation of a straight line given by: $r = x\cos\theta + y\sin\theta$ can be used instead. It describes a line having the orientation θ at the distance *r* from the origin, as can be seen in the following figure .

Therefore, an algorithm can be used by employing the above model instead of the equation $b = ax + y$.



Hough transform

The directional information of edge detectors can be used in the Hough Transform calculation by reducing the two-dimensional search to a one-dimensional search. If both sides of r=xcosè+ysinè are differentiated with respect to x, the following equation gives the line gradient:

$$\frac{\P y}{\P x} = -\cos q = \tan(\frac{p}{2} + q)$$

The use of the edge gradient reduces the computational complexity of the Hough Transform to the order O(N). However, care must be taken to avoid using noisy edge direction estimates that will reduce the performance of the algorithm.

Hough transform

The Hough Transform can be generalized to detect any parametric curves of the form *f(x,a)=0*, where *a* is the parameter vector. The memory required for the parametric matrix *P(a)* increases as K^p where *p* is the parameter number.Thus, this method is practical only for curves having a small number of parameters. Such a curve is the circle:

$$(x-a)^2+(y-b)^2=r^2$$

Its parameters are the radius *r* and the centre coordinates *(a,b).* A three-dimensional parameter matrix *P(r,a,b)* is needed.

Hough transform

- **Implementation of Hough Transform.**

Let (x_i,y_i) be a candidate binary edge image pixel. Then:

$$x_i = a + R\cos q$$

$$y_i = b + R\sin q$$

where (a,b) the centre coordinates of a circle having radius $r = R$

- For any radius r, $0<r<r_{max,}$ the coordinates (a,b) are calculated.
- These points belong to a cone surface.
- This process is repeated for any eligible pixel of the binary edge detector output.

Edge-following algorithms

• Boundary following: perform edge detection and follow the local edge elements.

• $e(x) = e(x,y)$: the edge magnitude.

• $\ddot{o}(x)=\ddot{o}(x,y)$: the direction produced by the edge detector output at the location $x=(x,y)$.

• $|e(x_i)-e(x_j)|$: similarity measure for neighboring edge elements.

• $|\ddot{o}(x_i)-\ddot{o}(x_j)|$: direction difference similarity measure.

Edge-following algorithms

- Two neighboring edge elements can be linked if:

$$\left| e(x_i) - e(x_j) \right| \leq T_1$$

$$\left| \Phi(x_i) - \Phi(x_j) \right| \bmod 2p \leq T_2$$

$$\left| e(x_i) \right| \geq T, \left| e(x_j) \right| \geq T$$

According to the third relationship small perturbations will not be mistaken as edge elements

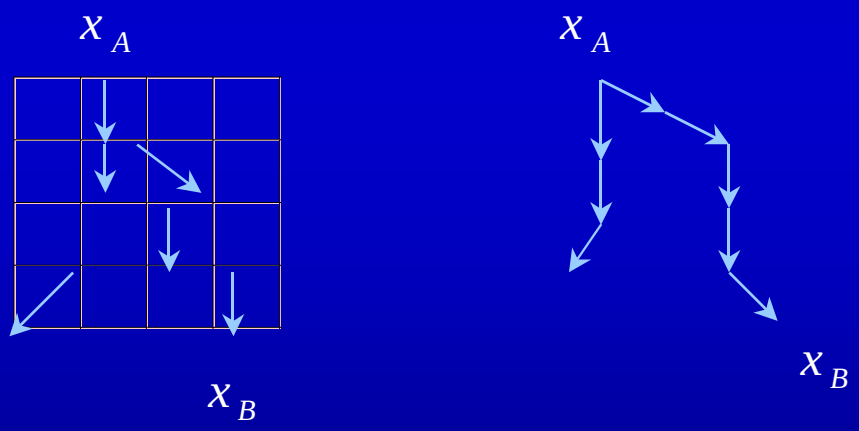
Edge-following algorithms

A simple algorithm for edge following can use the previous equations. Let us suppose that we start from a point x_A satisfying $|e(x_A)|>T$. If there is no neighboring edge element satisfying all the equations, then the algorithm stops. If more than one neighbor satisfy them, the element x_N that possesses the minimal differences $|e(x_N)-e(x_A)|$, $|\ddot{o}(x_N)-\ddot{o}(x_A)|$ is chosen.

The procedure continues recursively, with the new edge element x_N as a starting element.

Edge-following algorithms

A more comprehensive approach to edge following is based on graph searching. Edge elements at position x_i can be considered as graph nodes. The nodes are connected to each other if local edge linking rules are satisfied. Such a graph is:



Edge-following algorithms

A heuristic search algorithm has been proposed for edge following. Let us suppose that we form a cost function $C(x_1,x_2,\ldots,x_N)$ for a path connecting nodes $x_1=x_A$ to $x_N=x_B$:

The heuristic search algorithm tries to produce a minimum cost path from x_A to x_B . The algorithm is based on the cost function and on the choice of the successors of a node x_i by using edge linking criteria.

Basic disadvantages of the heuristic search algorithm:

1. The need to keep track of all current best paths.
2. Short paths (close to the origin) may have smaller cost than longer paths that are more likely to be the final winners.

Edge-following algorithms

Another useful approach to edge following is based on dynamic programming. The fundamental observation underlying this method is that any optimal path between two nodes of a graph has optimal subpaths for any node lying on it. Thus, the optimal path between two nodes x_A, x_B can be split into two optimal subpaths $x_A x_i$ and $x_i x_B$ for any x_i lying on the optimal path $x_A x_B$.