Digital Image Processing — Lecture 11

Edge linking and line detection

Last time: Edge Detection

First: EDGE Linking

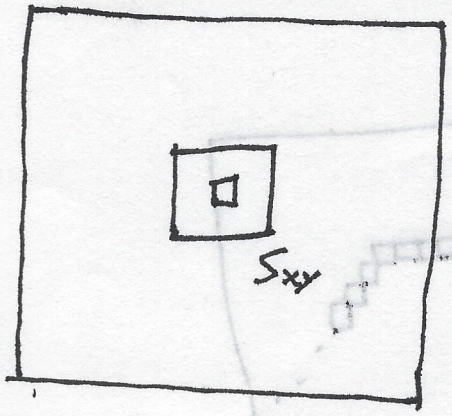
START WITH EDGE PIXELS AND corresponding (EDGE MAP)

$M(x,y)$, $\alpha(x,y)$

Idea: connect edges ~~in~~ with similar orientations and only do that for edges with Large enough magnitudes

— For Each EDGE Pixel (x,y) make a window S_{xy} around That Pixel

[we are only considering binary map of pixels that made it past → made it into Edge map]



(other pixel)

For each $(s,t) \in S_{xy}$ "Link" (x,y) to (s,t) IF

$$|M(x,y) - M(s,t)| \leq T_1$$

$$|\alpha(x,y) - \alpha(s,t)| \leq T_2$$

"Similar magnitude and angle [within a threshold]"

→ Trace out Long Edges

(2) Look at direction of where edge is pointing (we want similar orientations) we only want to connect local edges if the magnitudes are similar

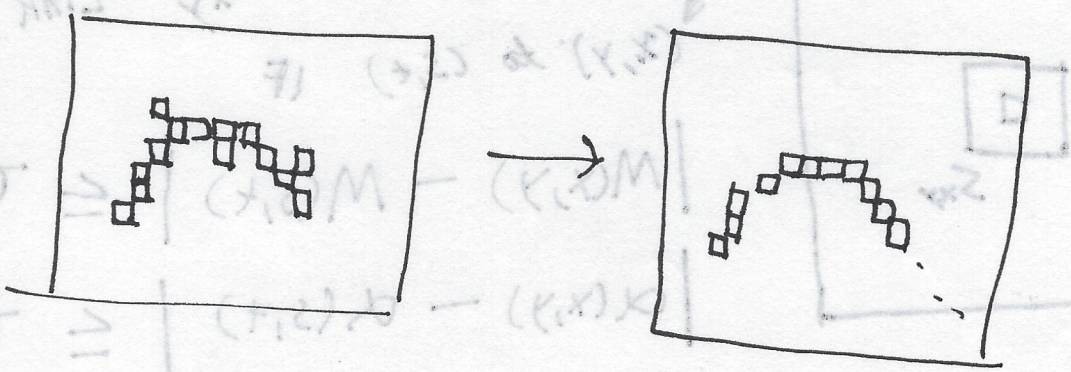
(forget weak magnitudes with similar angles)

→ We can build edges into long strings of pixels

→ New Idea, with these long strings of Edges, we want to describe them. Like if it's a window or a door

Question: How do we follow that boundary around and describe that boundary?

Another Picture of Edge linking via local Processing



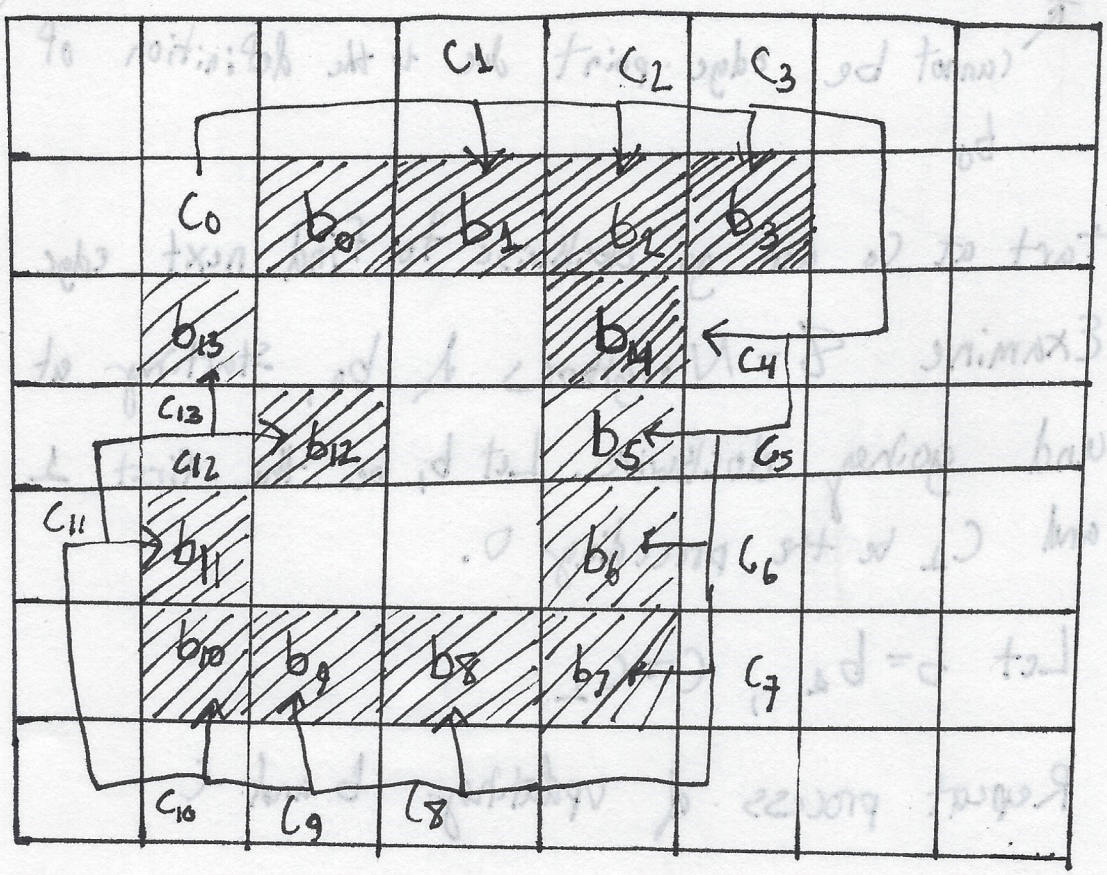
2) Boundary Following

We have EDGE Points around a closed Contour,

Want to Link / ORDER them in a clockwise Direction. → we have an ordered Set.

Moore's Boundary Following Algorithm

Assume Edge Points / Pixels are Labeled "1",
Back Ground Pixels "0"



↑ set of edges acquired describing a closed contour

(4) We want to assign a natural order to the pixels so that we can compare the orders that correspond to different objects → object matching

→ we can find similar object on a conveyor belt by matching up the boundary

we have an edge map 1 = Edge Pixel, 0 = Not

Algorithm States:

1) Let starting point b_0 be the uppermost, Leftmost point Labeled "1"

Let c_0 be the Left Neighbour of b_0.

↖ cannot be edge point due to the definition of b_0

start at c_0 and go clockwise to find next edge point

2) Examine 8-Neighbors of b_0, starting at c_0 and going clockwise. Let b_1 be the first 1 and c_1 be the preceding 0.

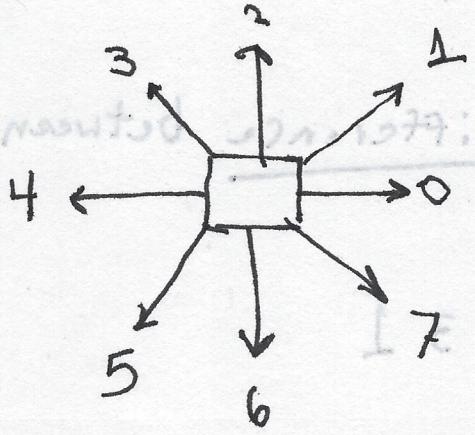
Let $b = b_1$, $c = c_1$

3) Repeat process of updating b and c.

4) Continue until $b = b_0$ and next boundary point found is b_1

5) The ordered list of b's is the boundary

Once we have such a boundary, we can describe it with a chain code



To go from b_0 and b_1
- go in the 0 direction

To go from b_1 to b_2
- go into 0 direction again.

"back at b_0"

So we have

0 0 0 5 6 6 6 4 4 4 2 1 3 1

The order depends on the starting point (by convention, the upper Left hand point)

Question: If we start at 90° rotated version of the same object, we will have a different chain code so How do we identify it as the same object?

(6) To Be able to match shapes at Different orientations, we can

1) order the chain code so it always starts with the minimum magnitude Integer.

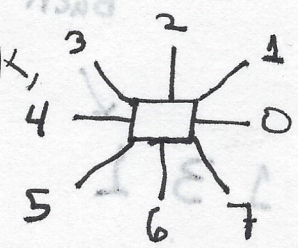
(The chain code defines the same shape of the object no matter the order, so ~~re~~ start at the Lowest number)

2) just Encode the Difference between Directions.

we have

0005666444231

Think,



we only rotate CCW

So we have as a result (Mod) → $(b_{i+1} - b_i) \bmod 8$

00510060067267

five clicks

We have 7, to return to the starting point

If we have 45° rotation intervals of the original object, we would still have the Difference Code.

"Differential CHAIN CODE"

Matlab 21.31 Matlab has this stuff, but we still have to do this ourselves

```
im=imread('shape.png')
imshow(im,[])
```

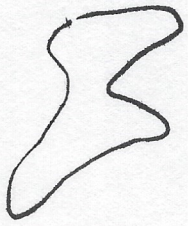
"Red is current Edge point
blue is previous Edge point"

edgelink(im)

bwtraceboundary In MATLAB
→ NOT The Same output In the way defined

```
c=code(1:20)   1st 20 of code
mod(c(2:end)-c(1:end-1),8)
```

Want to focus on Geometry level description,

for  , how do we find the best polygon to distinctly describe this shape?