Lecture 18-Edge Linking (Chapter 10.2.7)

Yuyao Zhang PhD

zhangyy8@shanghaitech.edu.cn

SIST Building 2 302-F

上海科技大学 ShanghaiTech University

Edge linking

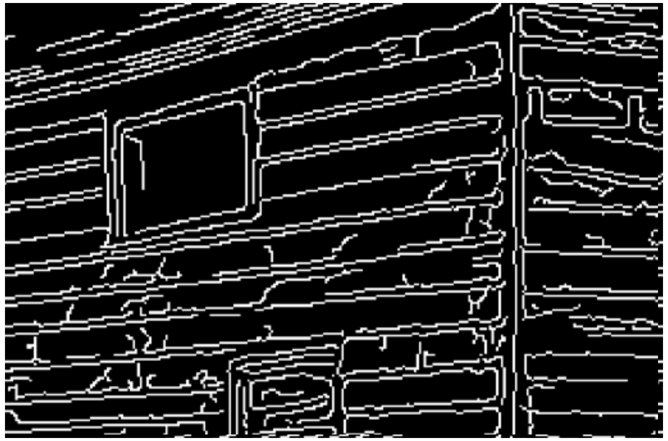
➤ **Previous step: edge detector.**

1. Start with edge pixels and corresponsding $M(x, y)$ and $\alpha(x, y)$;

2. Idea: for each edge pixel (x, y) make a window S_{xy} around that pixel for each $(s, t) \in S_{xy}$, "Link" (x, y) to (s, t) if

$$|M(x, y) - M(s, t)| \leq \tau_1$$

$$|\alpha(x, y) - \alpha(s, t)| \leq \tau_2$$



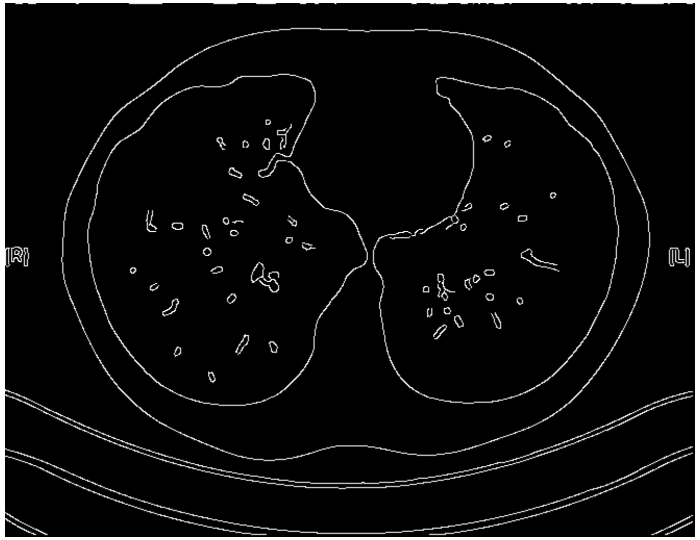
To take out long edges.

Boundary following

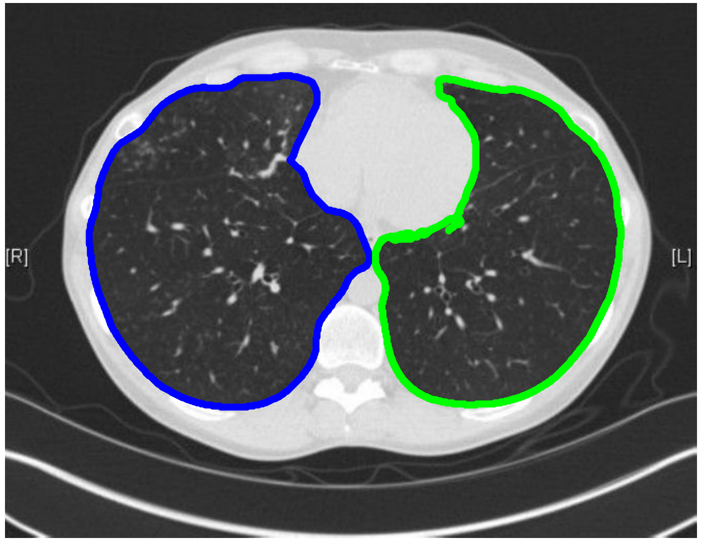
- We have edge point around a closed contour, we want to link/order them in a clock wise direction.

Input image

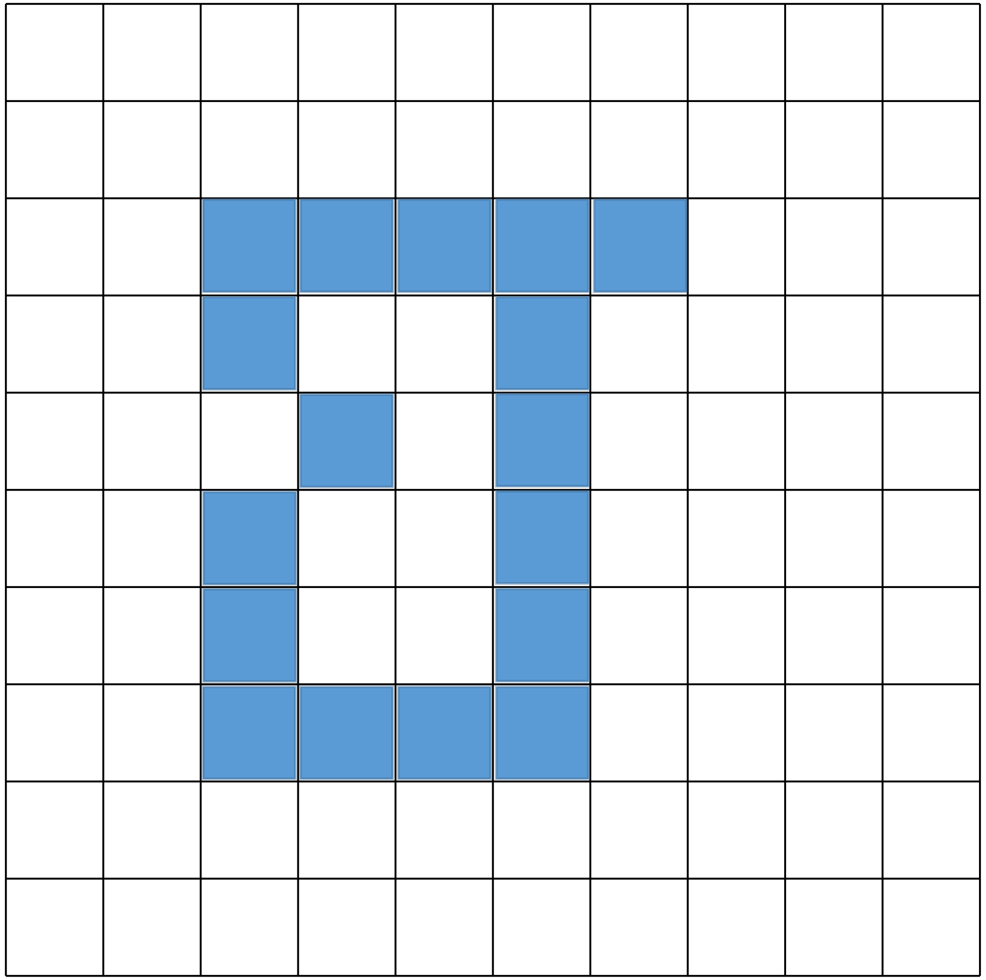
Edge detection



Boundary following



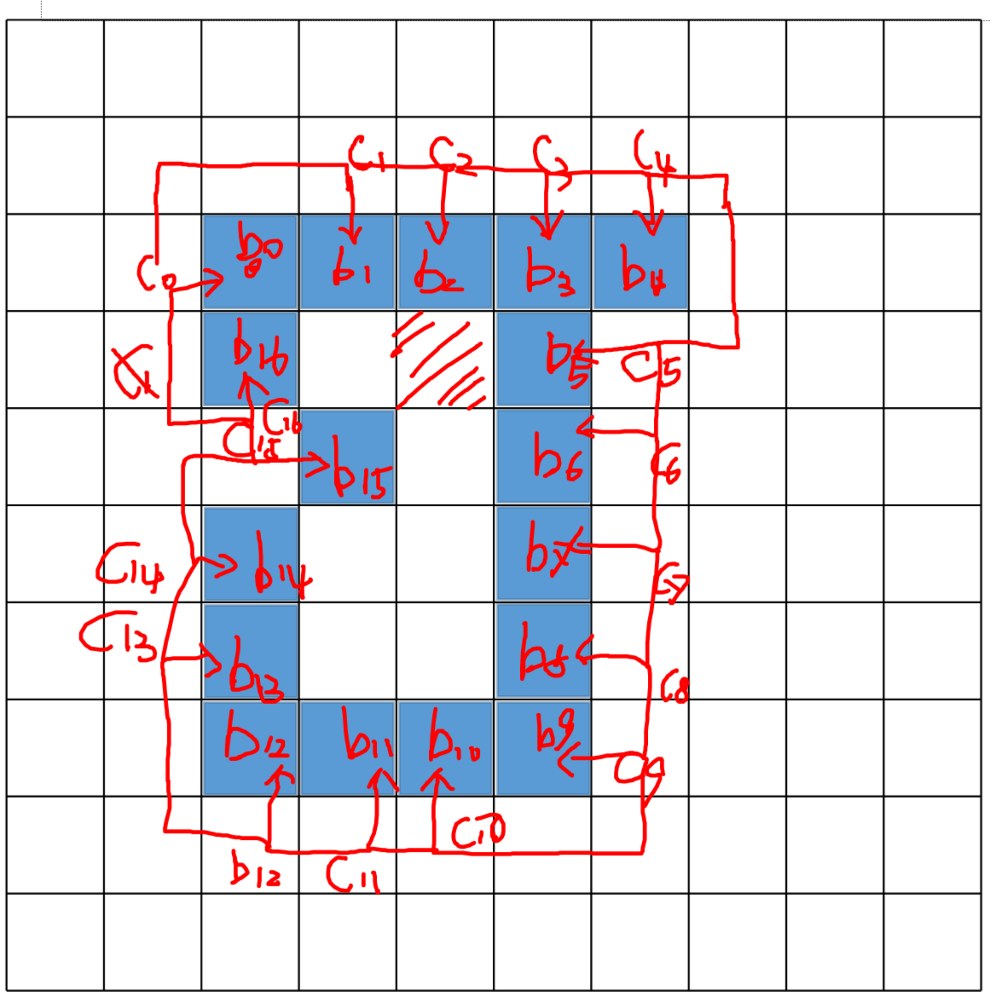
Boundary following

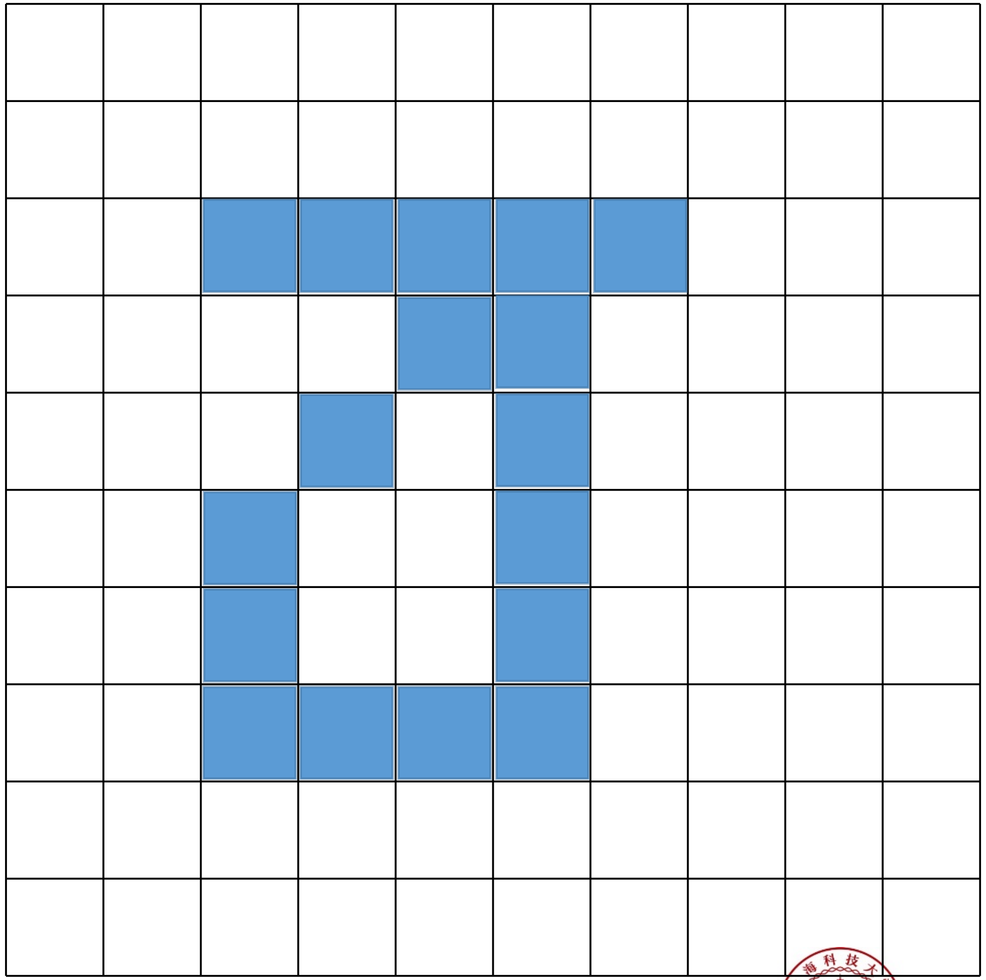


- **Moore's boundary following algorithms:**

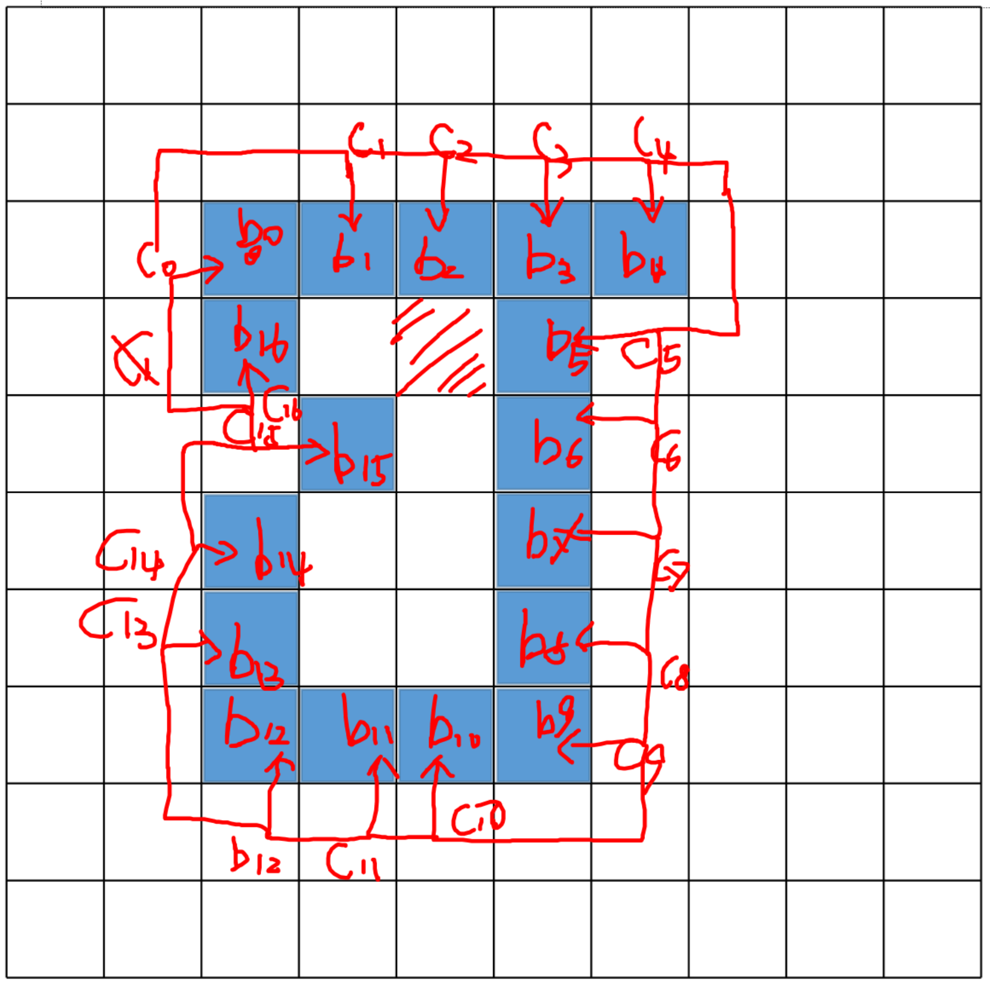
  1. Start with edge maps (binary).
  2. Let starting point b_0 be the uppermost, leftmost point labelled "1". Let c_0 be the left neighbor of b_0.
  3. Examine 8-neighbors of b_0, starting at c_0, and going clock-wise. Let b_1 be the first 1 pixel and c_1 be the preceding 0 pixel.
  4. Let $b = b_1, c = c_1$.
  5. Continue until $b = b_0$, and next bounding point found is b_1. Or until there is no edge point in the 8-neighbor of b.
  6. The opened list of b is the boundary.

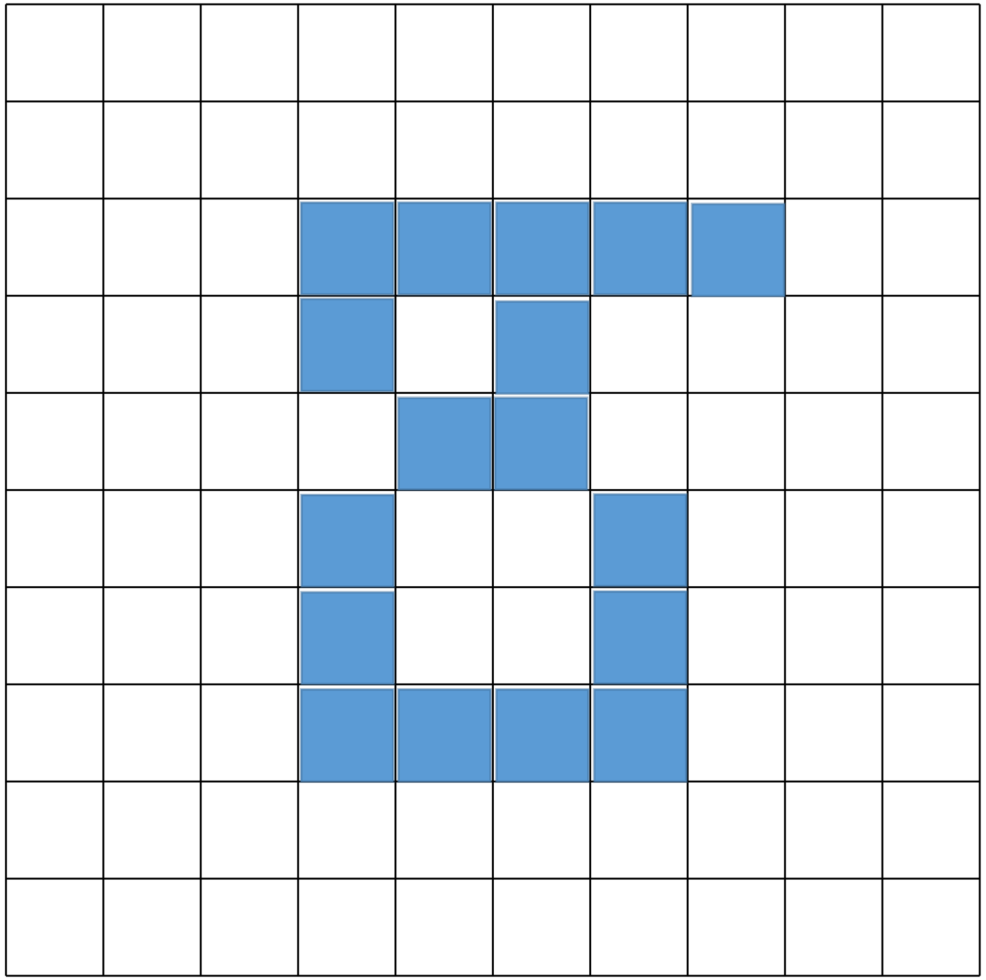
Boundary following





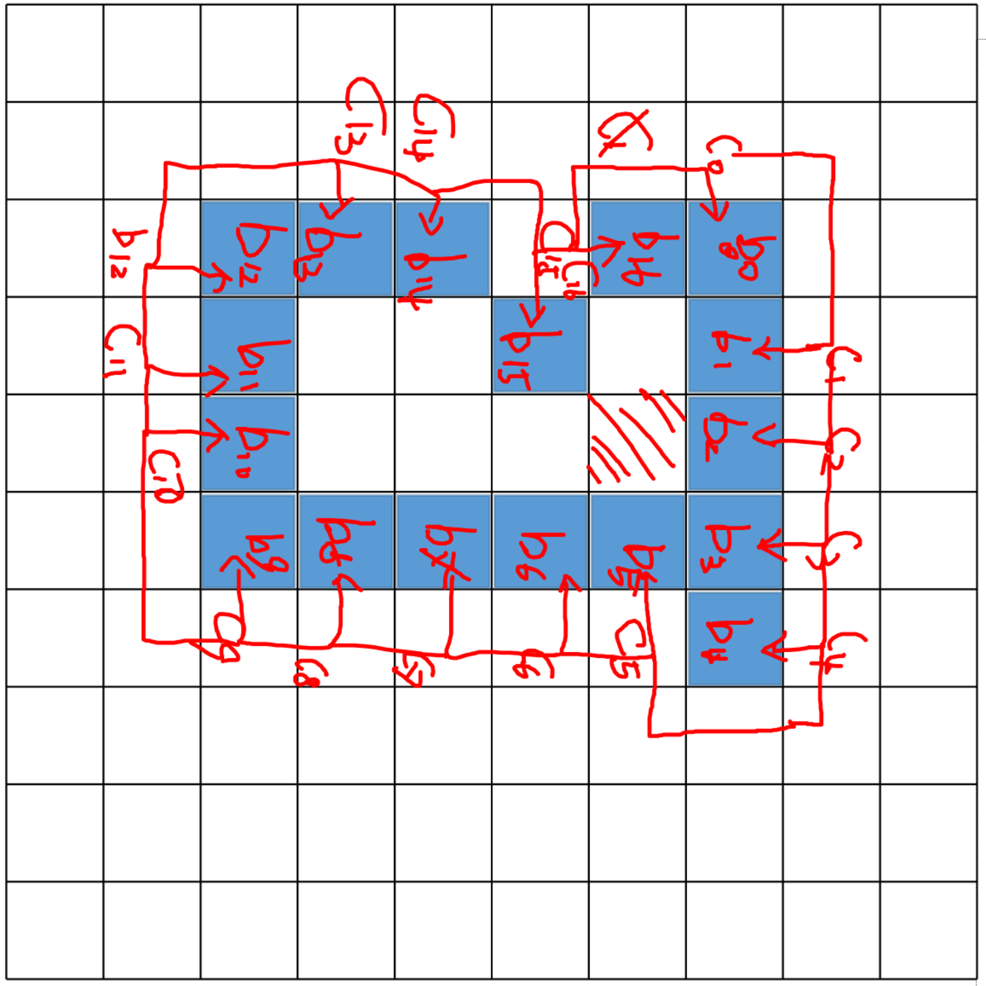
Boundary following





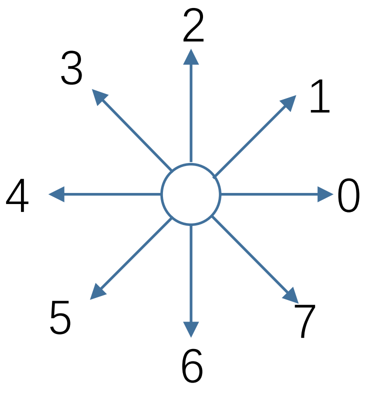
上海科技大学 ShanghaiTech University

Boundary following



➢ Describe the boundary with a chain code:

Define 3-bit direction, corresponding to previous boundary point.

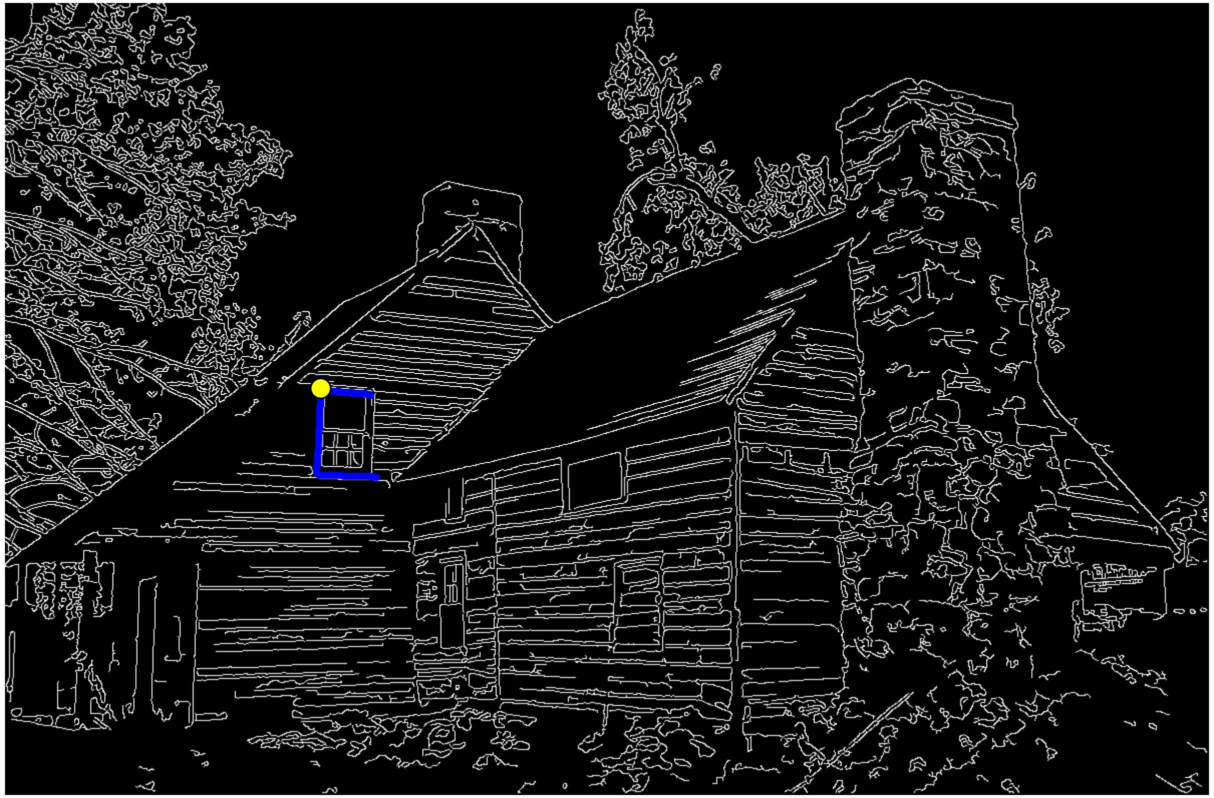


b0	b1	b2	b3	b4	b5	b6	b7	b8	b9
Direction for next P									

b10	b11	b12	b13	b14	b15	b16			
Direction for next P									

Boundary following

- Matlab function: bwtraceboundary



Polygonal fitting

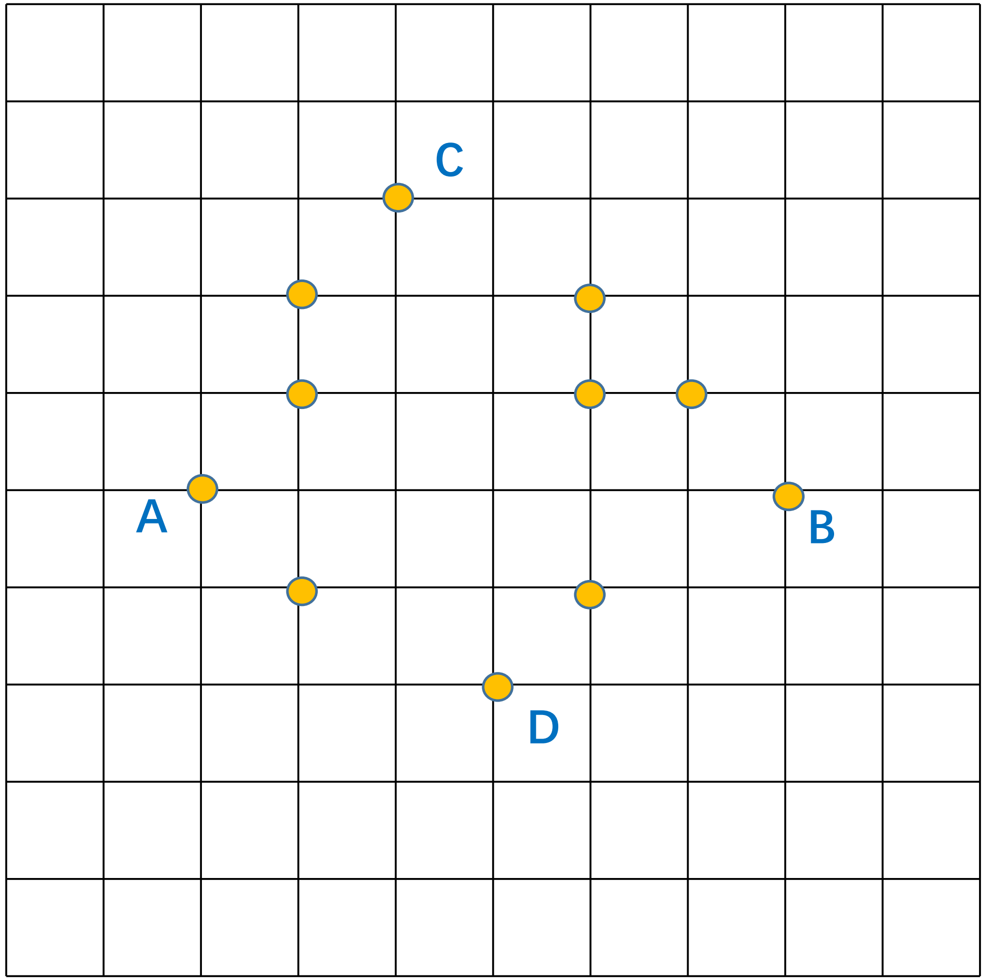
- **Fitting a set of ordered points (find windows/doors)**

  1. Let P be a sequence of ordered, distinct points. (e.g. ordered edges after boundary following).
  2. Specify two starting points A, B.
  3. Specify a threshould T (pixel distance).
  4. Creating the stacks: [final] and [in process].
  5. Compute the distance from this line to all the points between these vertices. Select vertex V_{max} with the max distance D_m.

上海科技大学 ShanghaiTech University

Polygonal fitting

final



In process

1. Let P be a sequence of ordered, distinct points. (e.g. ordered edges after boundary following).

2.Specify two starting points A,B.

3.Specify a threshold T (pixel distance).

4.Creating the stacks: [final] and [in process]. Then connect the vertices on top of each stack.

5. Compute the distance from this line to all the points between these vertices. Select vertex V_{Max} with the max distance D_m.

6. If $D_m > \mathrm{T}$ (a threshold set), put V_{Max} at the end of [in process], and go to step 4.

7. Otherwise, remove the last vertex from [in process] and make it the last vertex in [final].

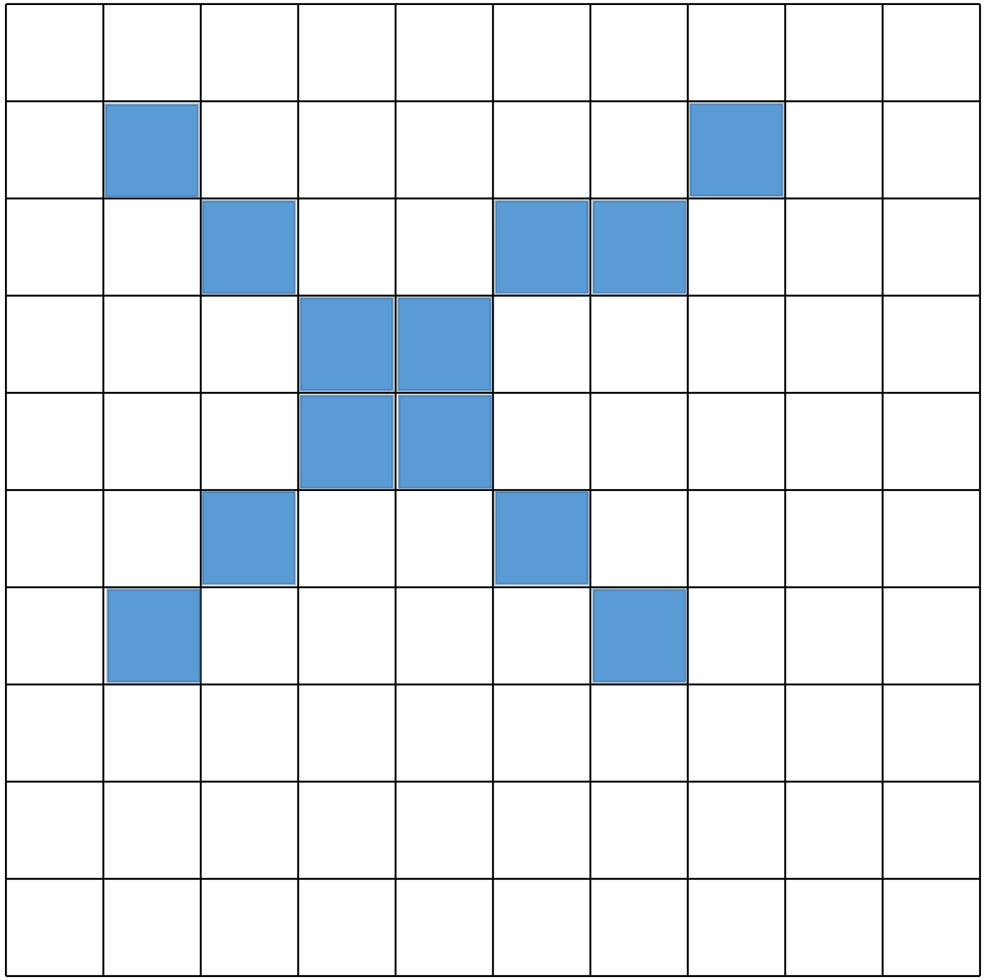
8. If [in process] is not empty, go to step 4.

9. other wise, done. The vertices in are the ordered vertices of a polygonal.

Polygonal fitting

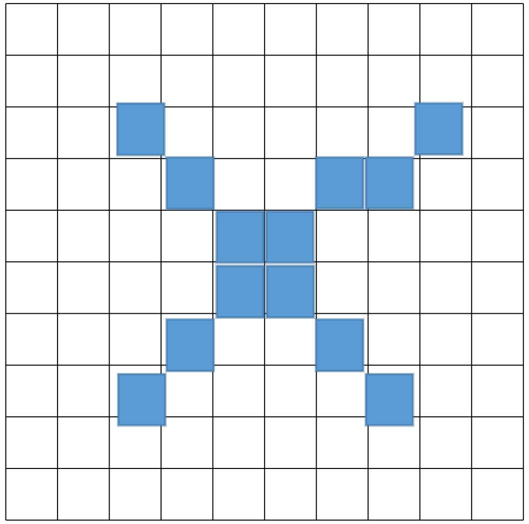
- If the threshold T is small, we will have a polygon with many vertices and smooth fitting.
- Otherwise, a polygon fitting with simple structure and large error.

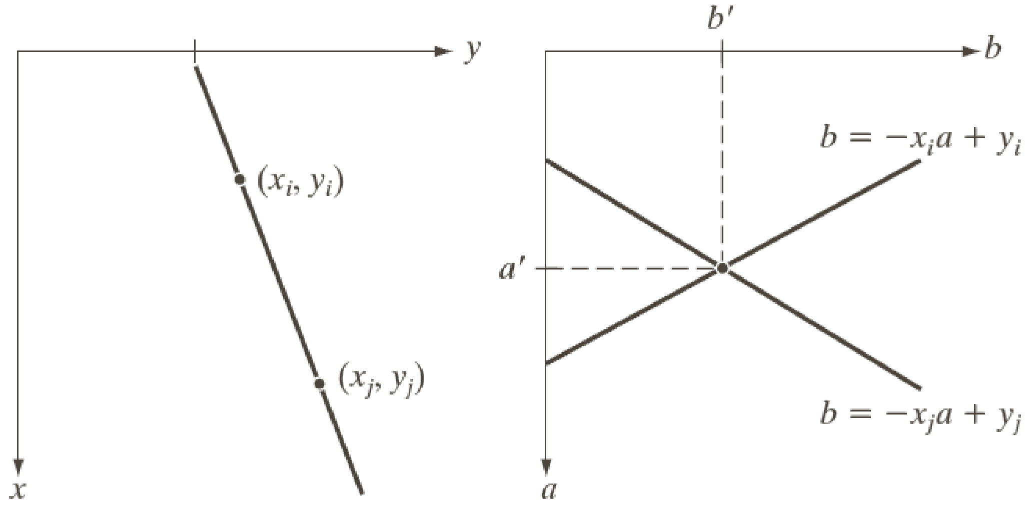
New question

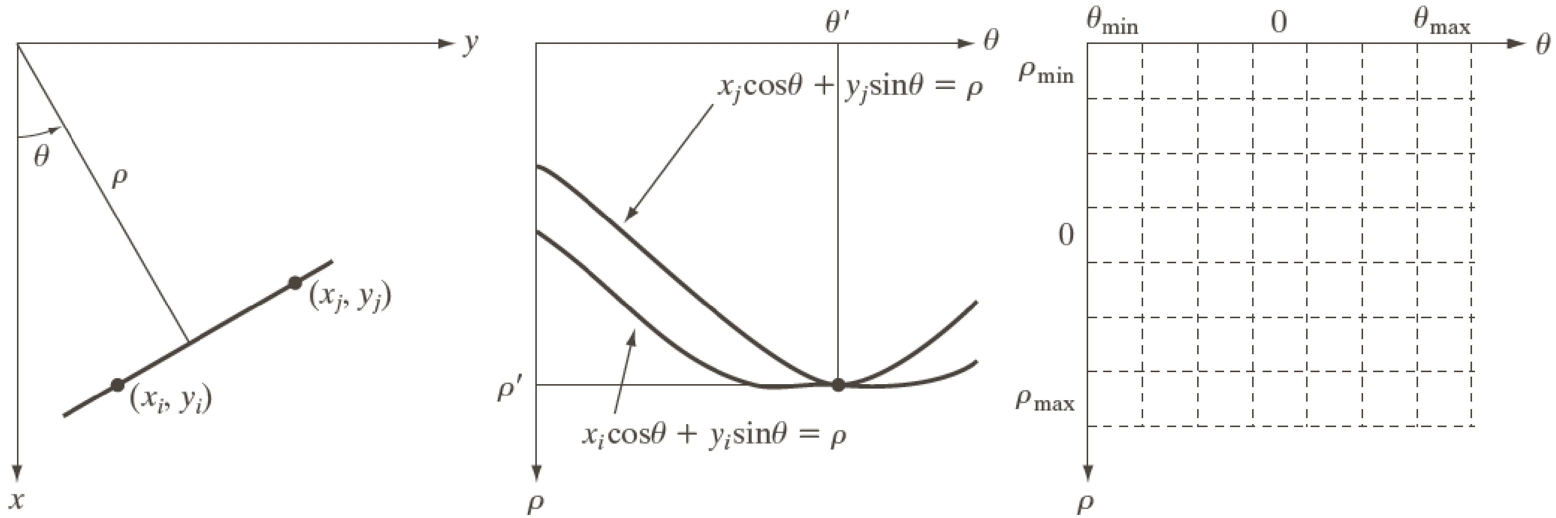


上海科技大学
ShanghaiTech University

Hough Transform (霍夫变换)







Hough Transform (霍夫变换)

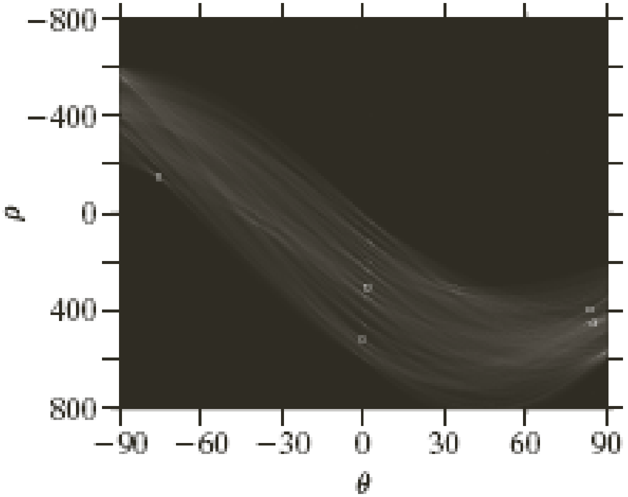
➢ An approach based on Hough Transform

1. Obtain a binary edge image using any edge detector;
2. Specify subdivisions in the $\rho\theta$-plane;
3. Examine the counts of the accumulator cells (累加器单元) for high pixel concentrations;
4. Examine the relationship between pixels in a chosen cell.

➢ Matlab function:

- [H, theta, rho] = hough(f);
- peaks = houghpeaks(H, NumPeaks)
- lines = houghlines(f, theta, rho, peaks)

Hough Transform (霍夫变换)



上海科技大学 ShanghaiTech University

Take home message & Discussion

- Boundary detection & Global structure detection: