Lecture 14-2-Edge Linking (Chapter 10.2.7)

Yuyao Zhang, Xiran Cai PhD

zhangyy8@shanghaitech.edu.cn caixr@shanghaitech.edu.cn

SIST Building 2 302-F/302-C

Course piazza link: piazza.com/shanghaitech.edu.cn/spring2021/cs270spring2021

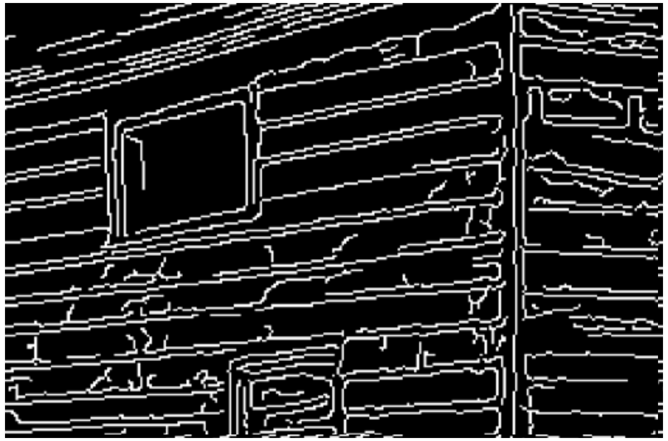
Edge linking

- **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

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

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

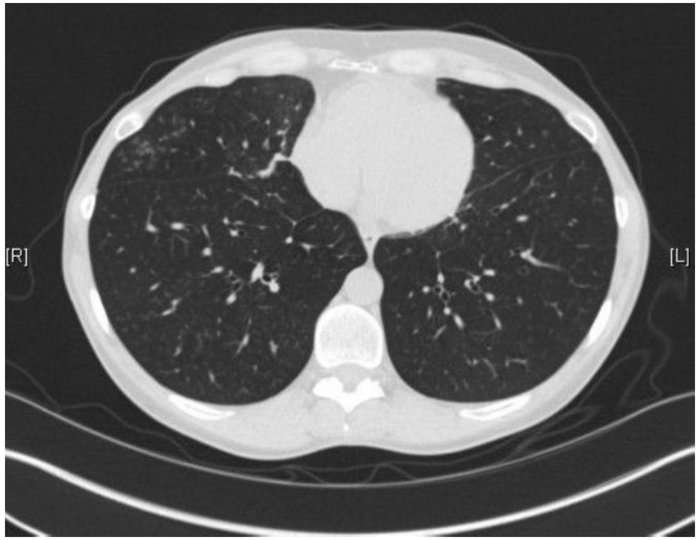


To take out long edges.

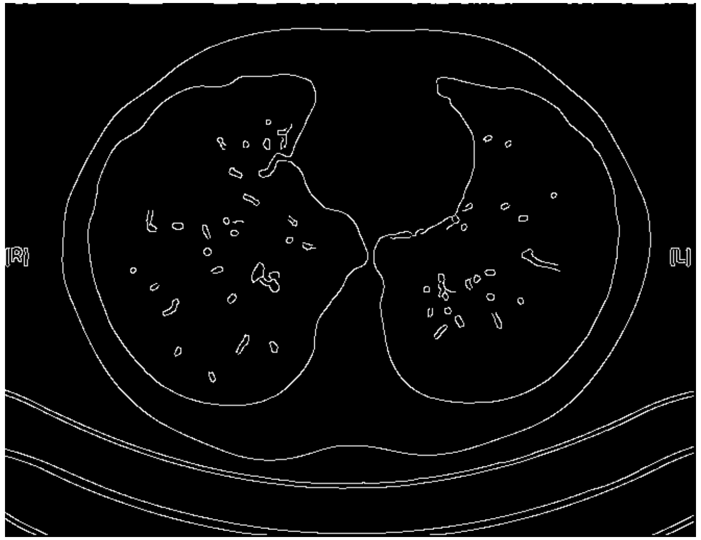
Boundary following

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

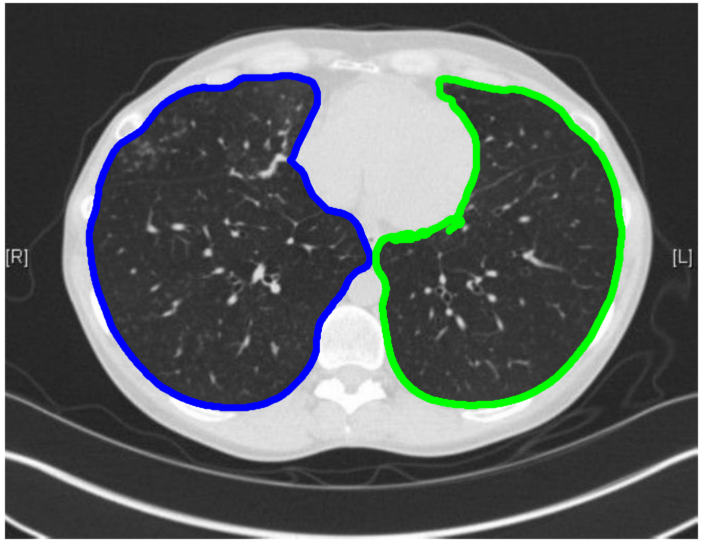
Input image



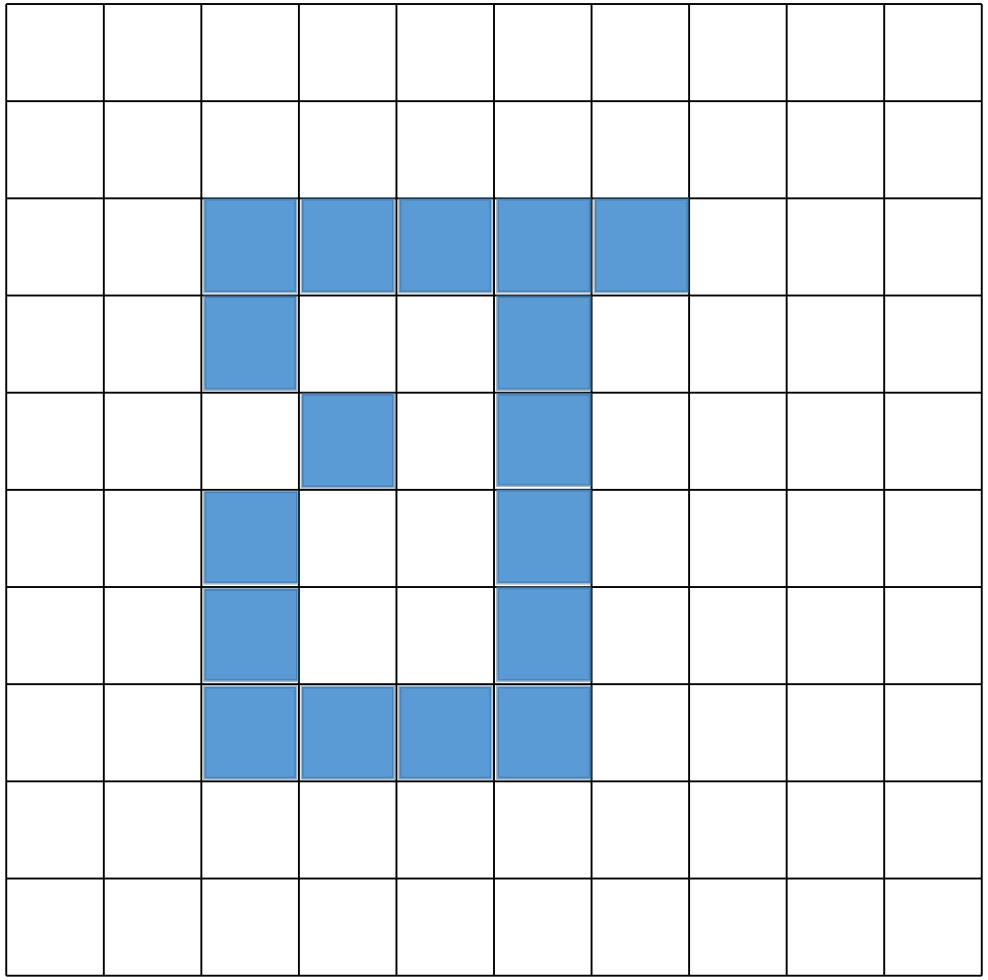
Edge detection



Boundary following



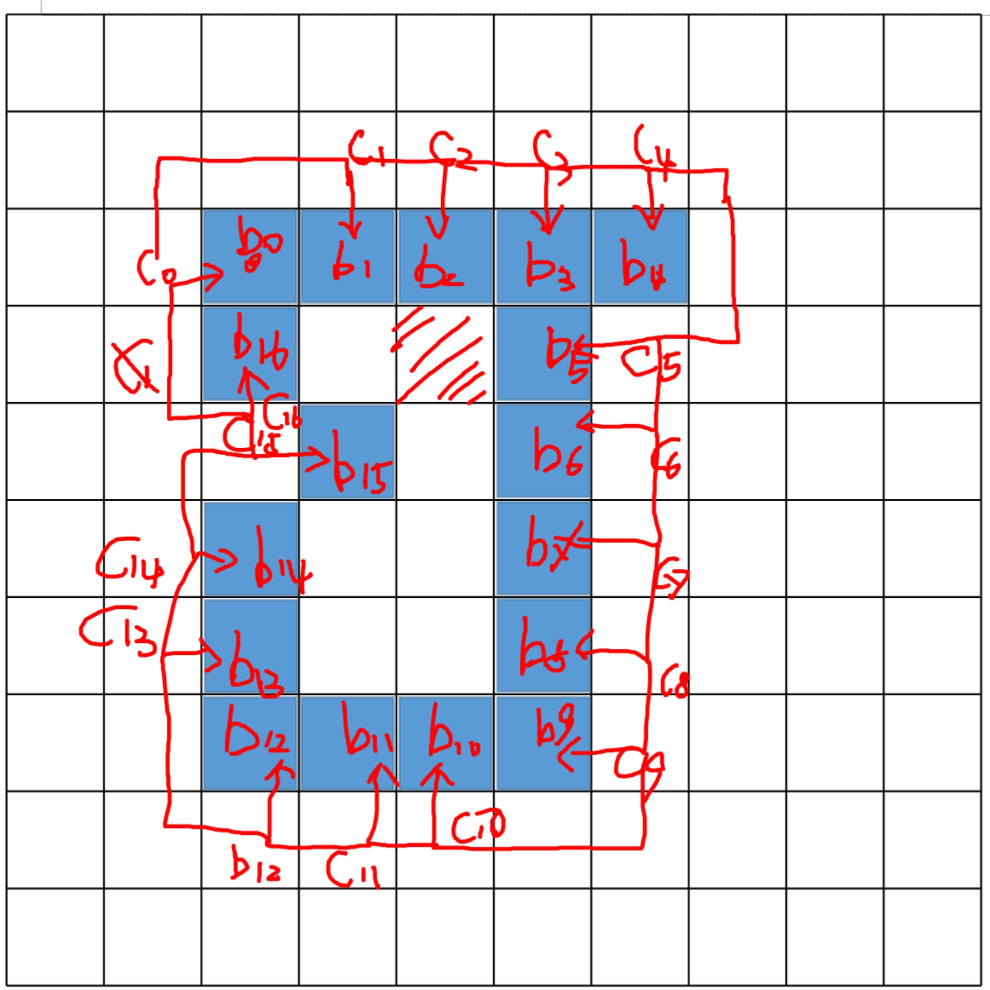
Boundary following

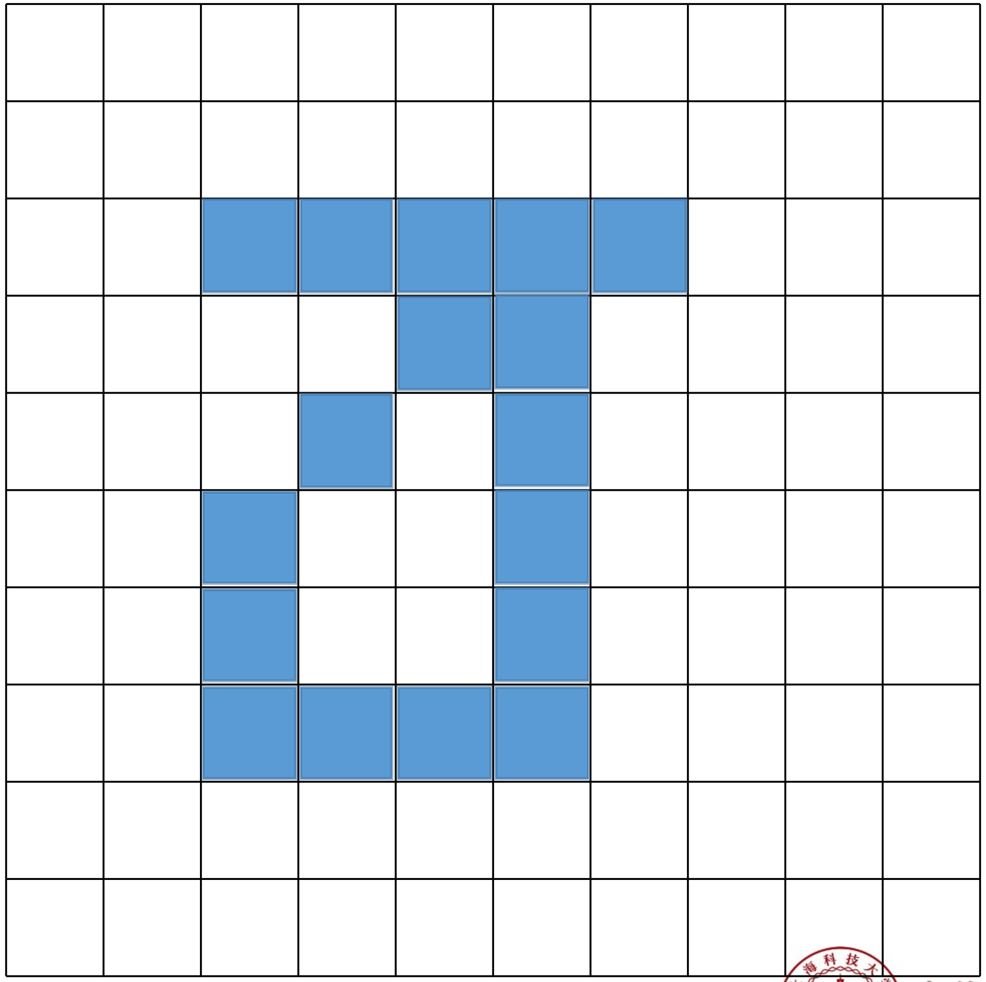


- **Moore’s boundary following algorithms:**

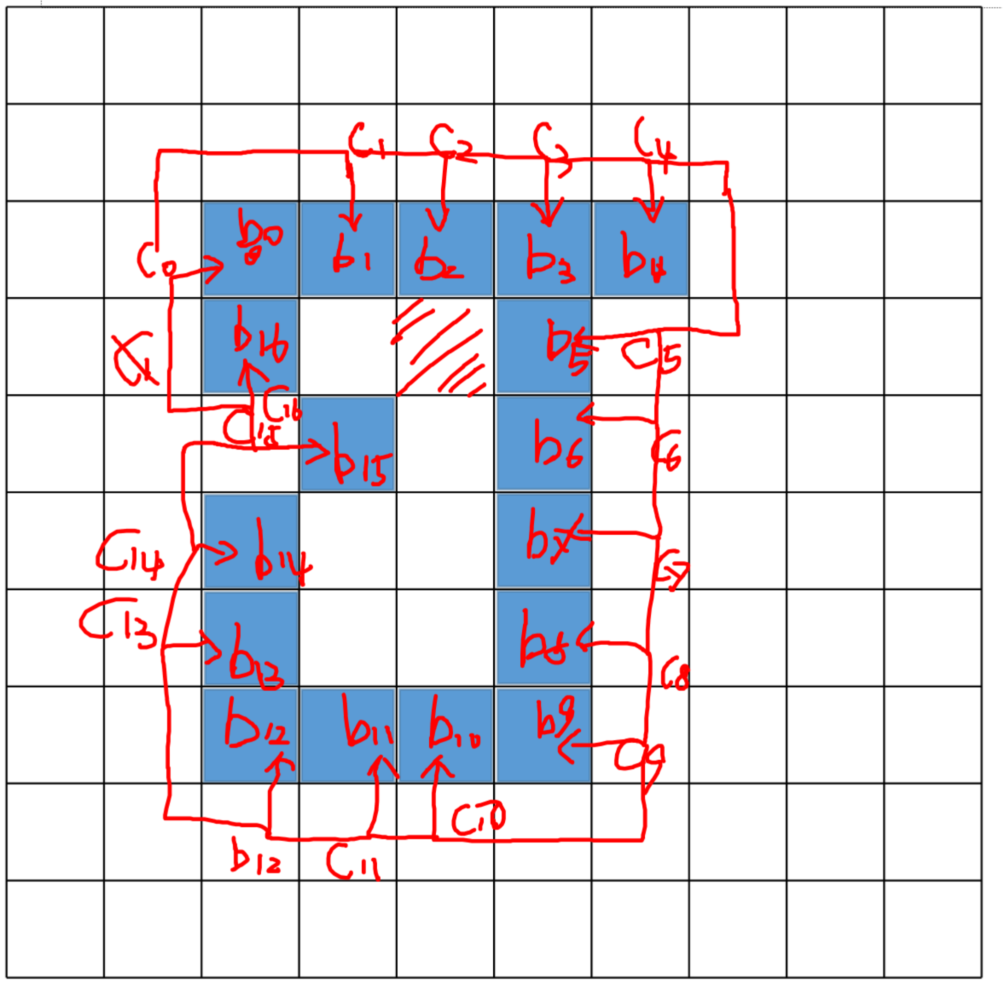
1. Start with edge maps (binary).
2. Let starting point b_0 be the uppermost, leftmost point labelled “1”. Let c_0 be the left neighbor of b_0.
3. Examine 8-neighbors of b_0, starting at c_0, and going clock-wise. Let b_1 be the first 1 pixel and c_1 be the preceding 0 pixel.
4. Let $b = b_1, c = c_1$.
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.

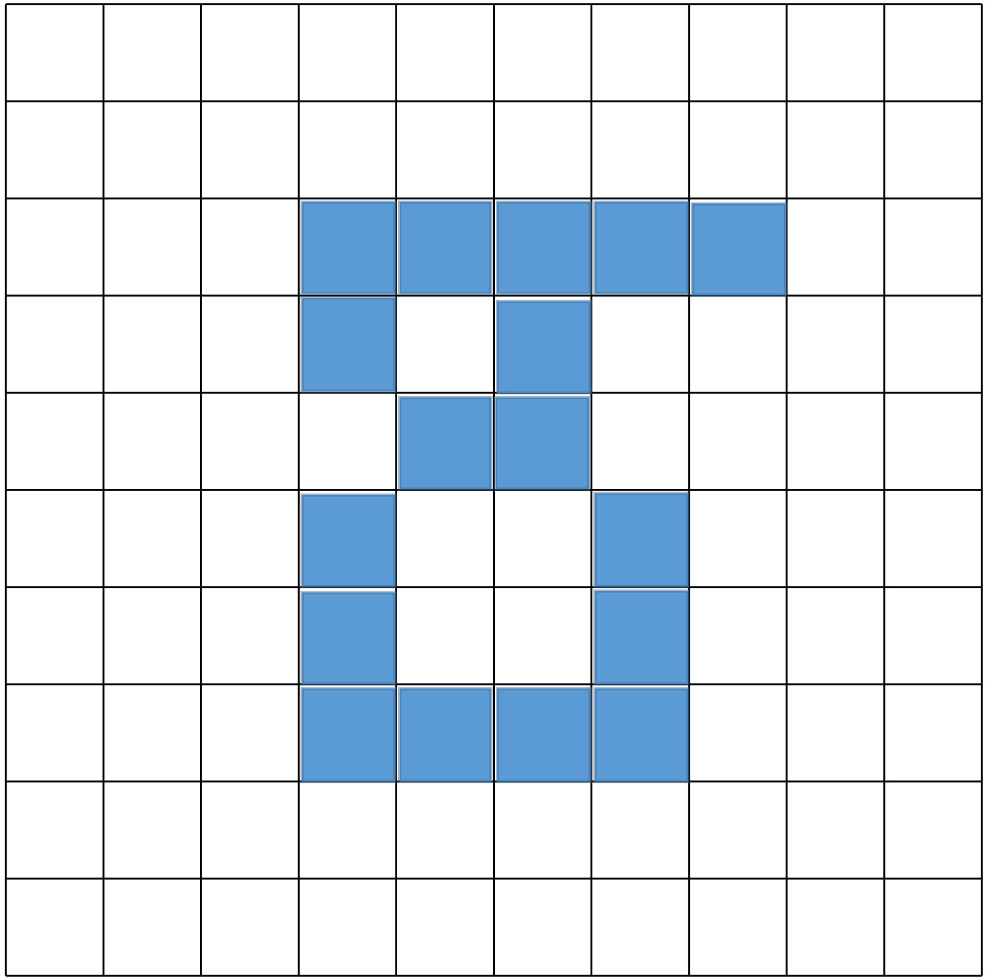
Boundary following



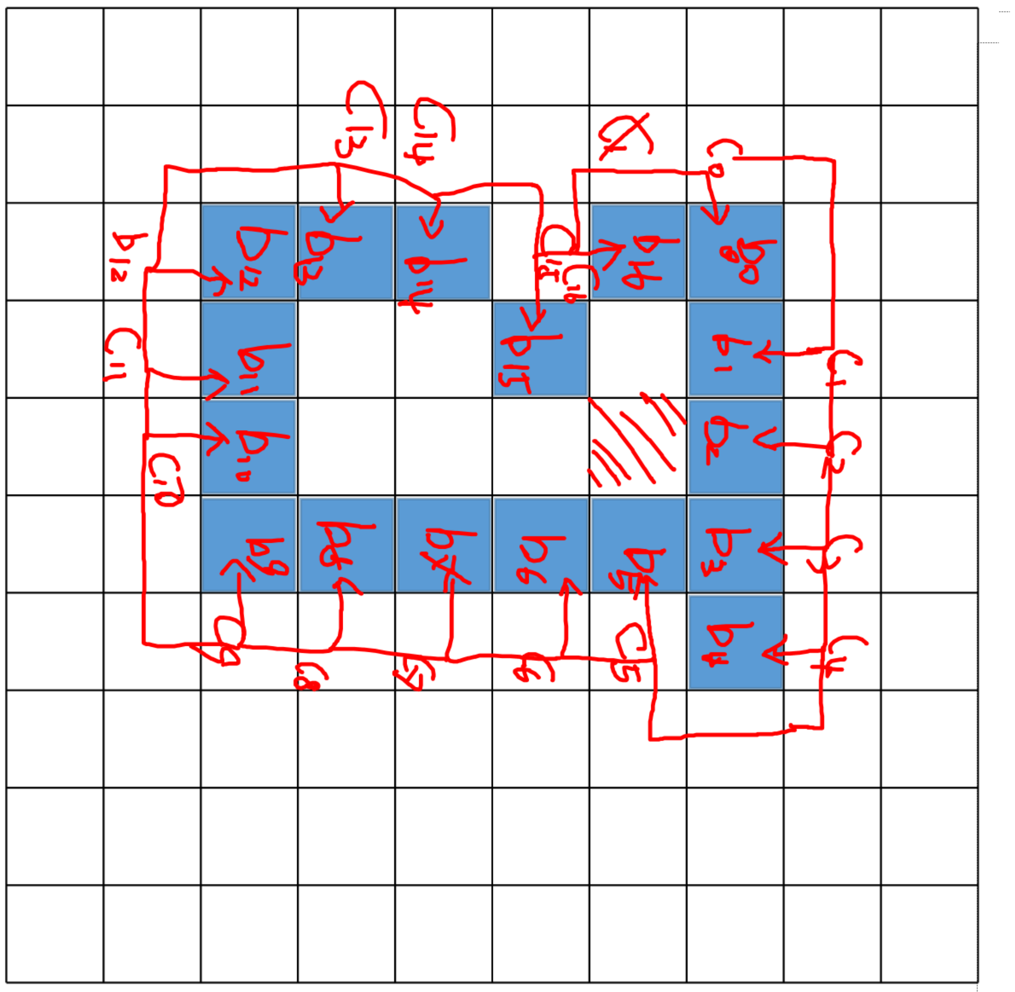


Boundary following



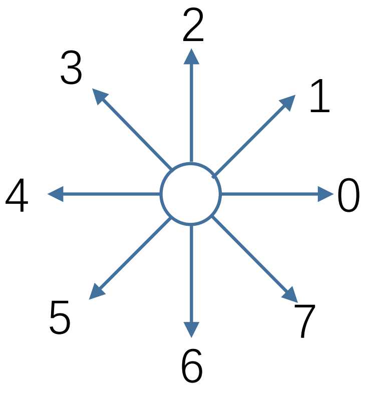


Boundary following



➢ Describe the boundary with a chain code:

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

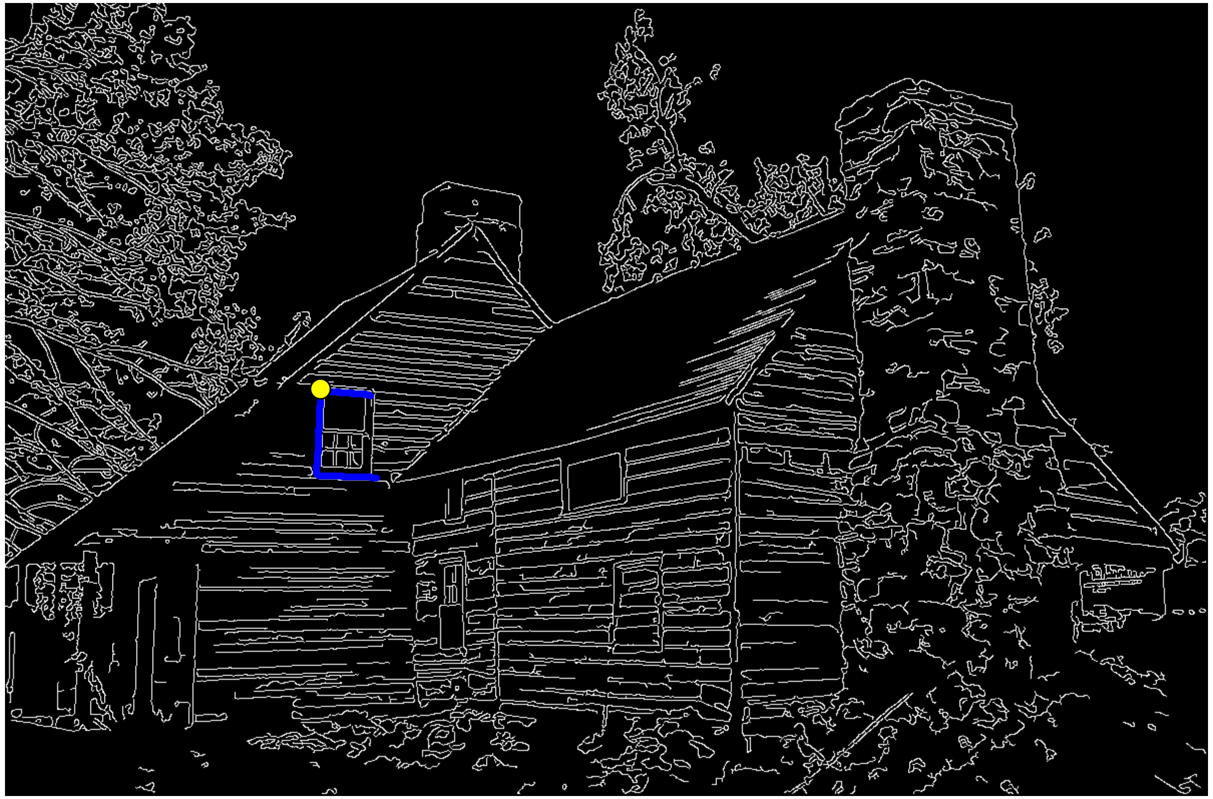


b0	b1	b2	b3	b4	b5	b6	b7	b8	b9
Direction for next P	0	0	0	0	5	6	6	6	6
ΔDirection	0	0	0	0	5	1	0	0	0

b10	b11	b12	b13	b14	b15	b16			
4	4	4	2	2	1	3			
−2	0	0	-2	0	-1	2			

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.

Polygonal fitting

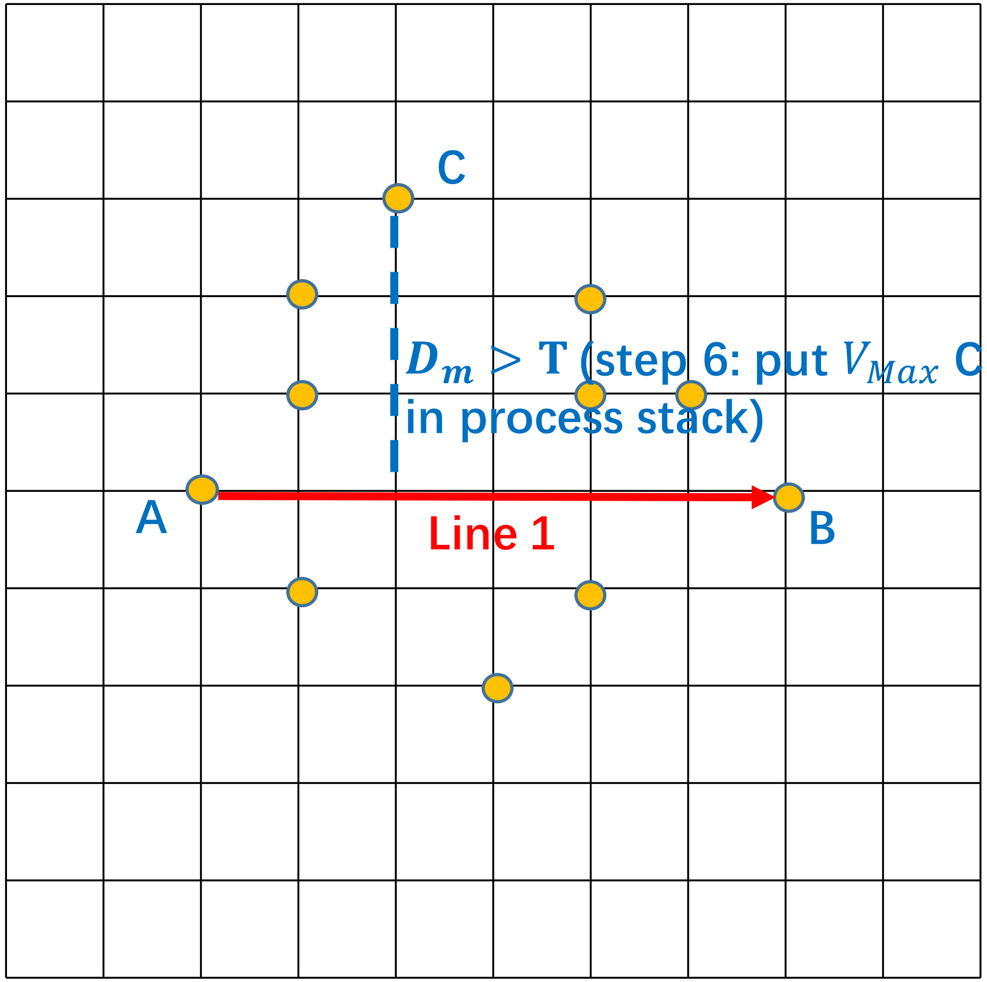
- **Purpose of Polygonal fitting**

Fitting a set of points or an edge map with convex boundary.

Polygonal fitting

Initialization & iter1

final	In process
A	AB
A	ABC



$D_m > \mathrm{T}$ (step 6: put V_{Max} C on top of in process stack)

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

2.Specify two starting points A,B with largest distance among all points.

3.Specify a threshold T (pixel distance).

4.Creating the stacks using the two starting points: for example [final] as A and [in process] A,B. Then connect the vertices on top of each stack with a **directed line**.

5. Compute the distance from this line to all the points in the **clock-wise or anti-clock-wise** side of the directed line 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].

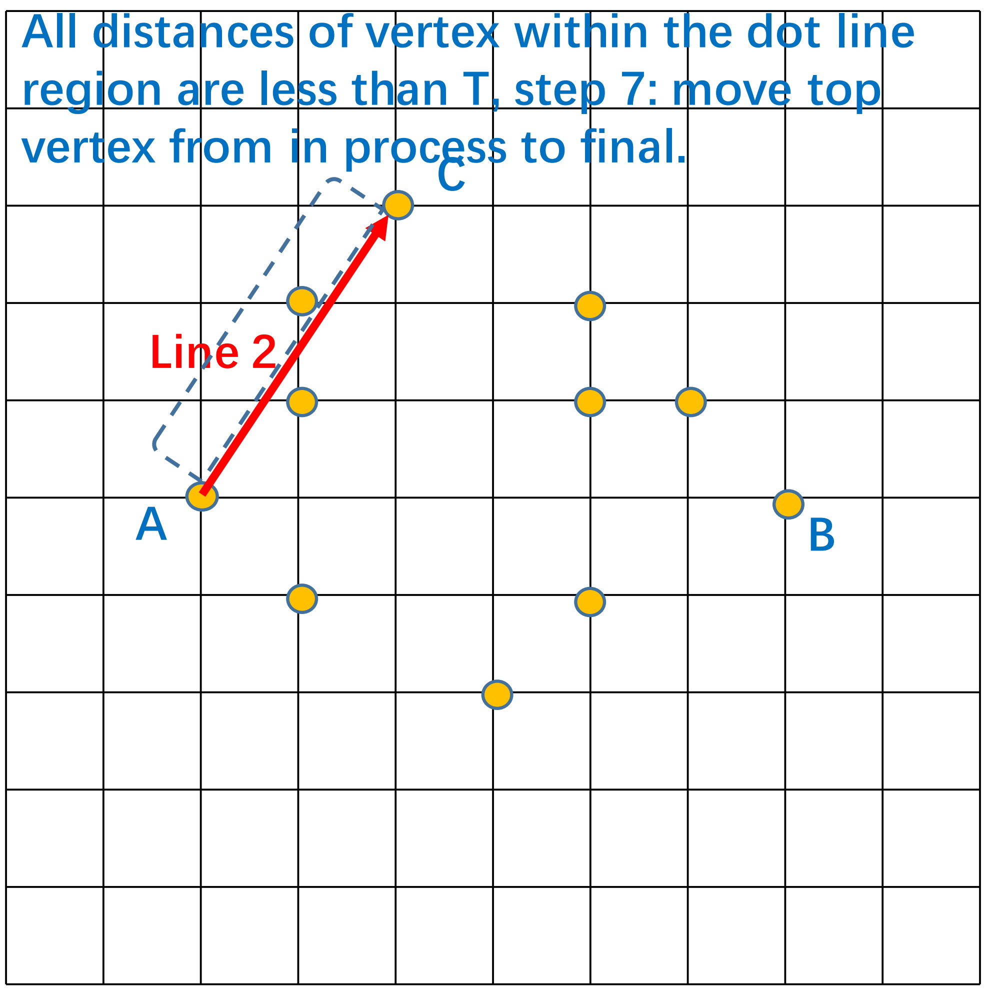
8. If [in process] is not empty, go to step 4.

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

Polygonal fitting

iter2

final	In process
A	AB
A	ABC
AC	AB



All distances of vertex within the dot line region are less than T, step 7: move top vertex from in process to final.

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

2.Specify two starting points A,B with largest distance among all points.

3.Specify a threshold T (pixel distance).

4.Creating the stacks using the two starting points: for example [final] as A and [in process] A,B. Then connect the vertices on top of each stack with a **directed line**.

5. Compute the distance from this line to all the points in the **clock-wise or anti-clock-wise** side of the directed line 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

iter3

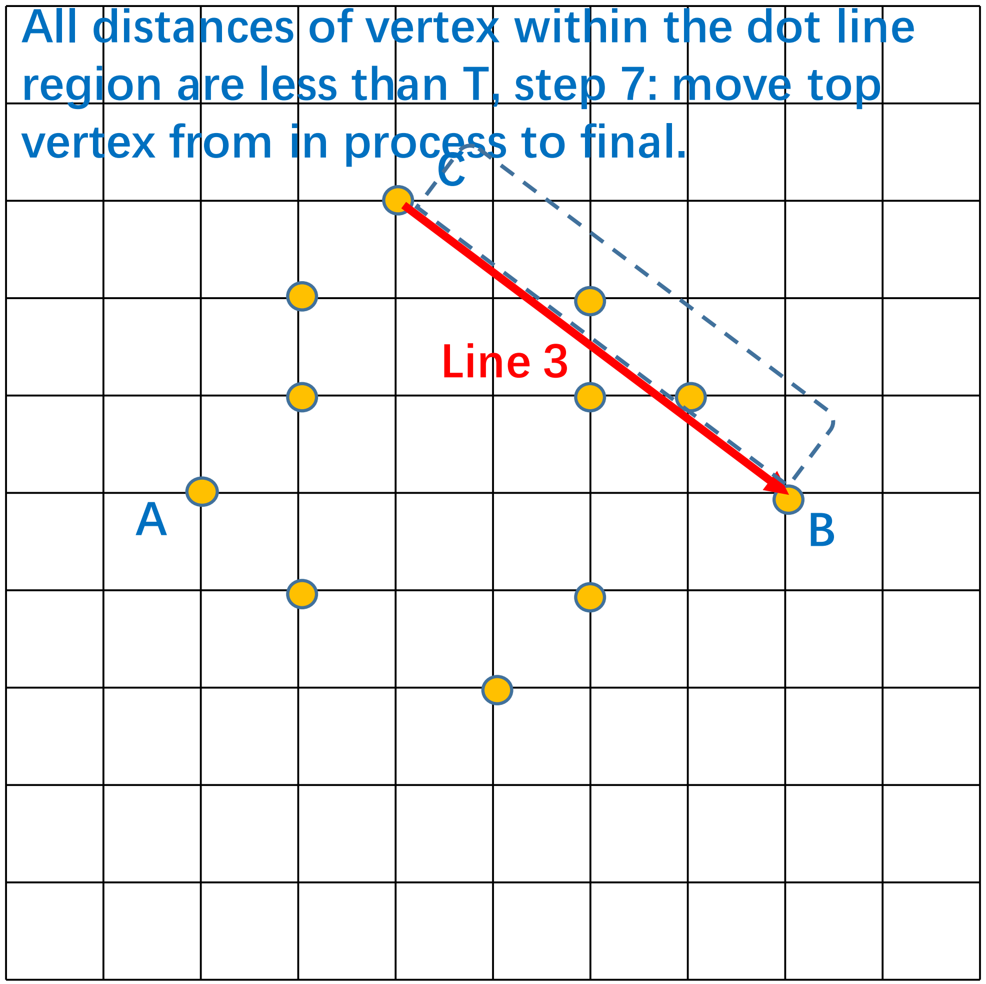
final

A

A

AC

ACB


All distances of vertex within the dot line region are less than T, step 7: move top vertex from in process to final.
C
Line 3
A
B

In process

AB

ABC

AB

A

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

2.Specify two starting points A,B with largest distance among all points.

3.Specify a threshold T (pixel distance).

4.Creating the stacks using the two starting points: for example [final] as A and [in process] A,B. Then connect the vertices on top of each stack with a **directed line**.

5. Compute the distance from this line to all the points in the **clock-wise or anti-clock-wise** side of the directed line 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.

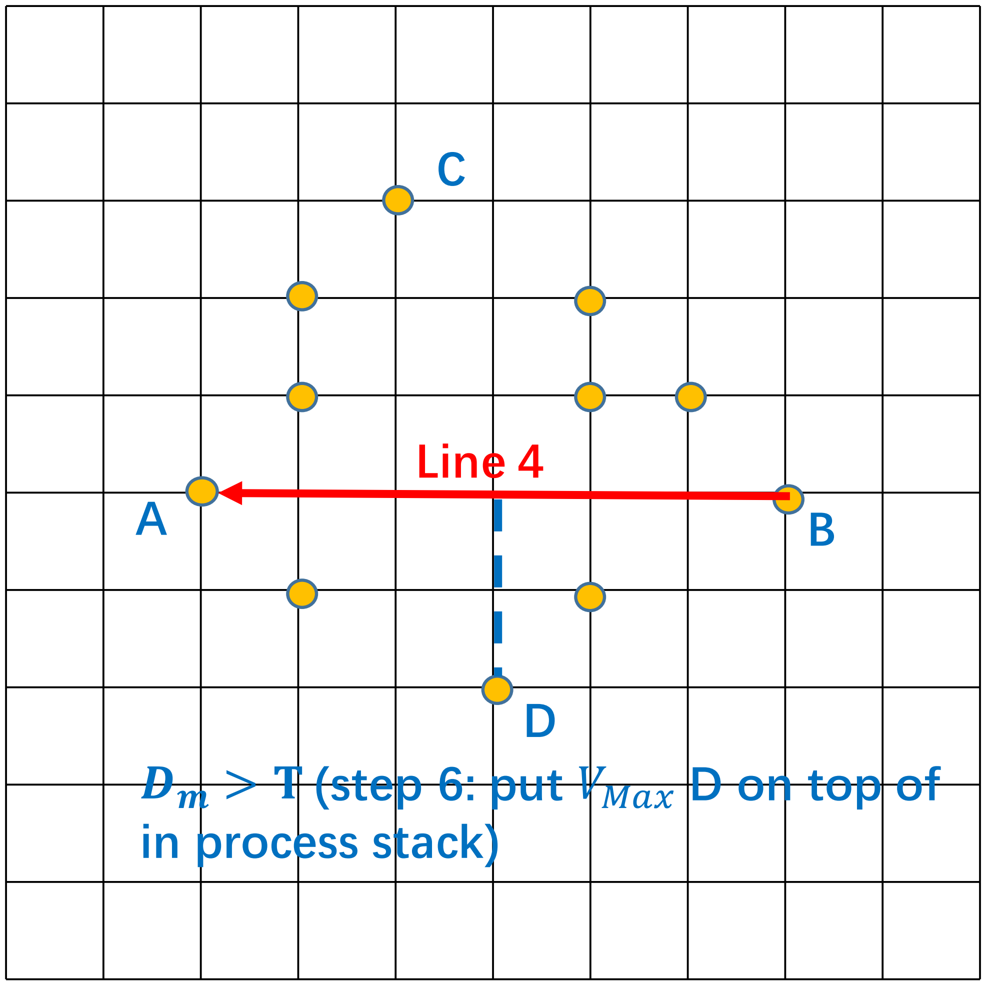
7. Otherwise, remove the last vertex from [in process] and make it the last vertex in [final].

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

iter4



$D_m > \mathrm{T}$ (step 6: put V_{Max} D on top of in process stack)

final	In process
A	AB
A	ABC
AC	AB
ACB	A
ACB	AD

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

2.Specify two starting points A,B with largest distance among all points.

3.Specify a threshold T (pixel distance).

4.Creating the stacks using the two starting points: for example [final] as A and [in process] A,B. Then connect the vertices on top of each stack with a **directed line**.

5. Compute the distance from this line to all the points in the **clock-wise or anti-clock-wise** side of the directed line 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.

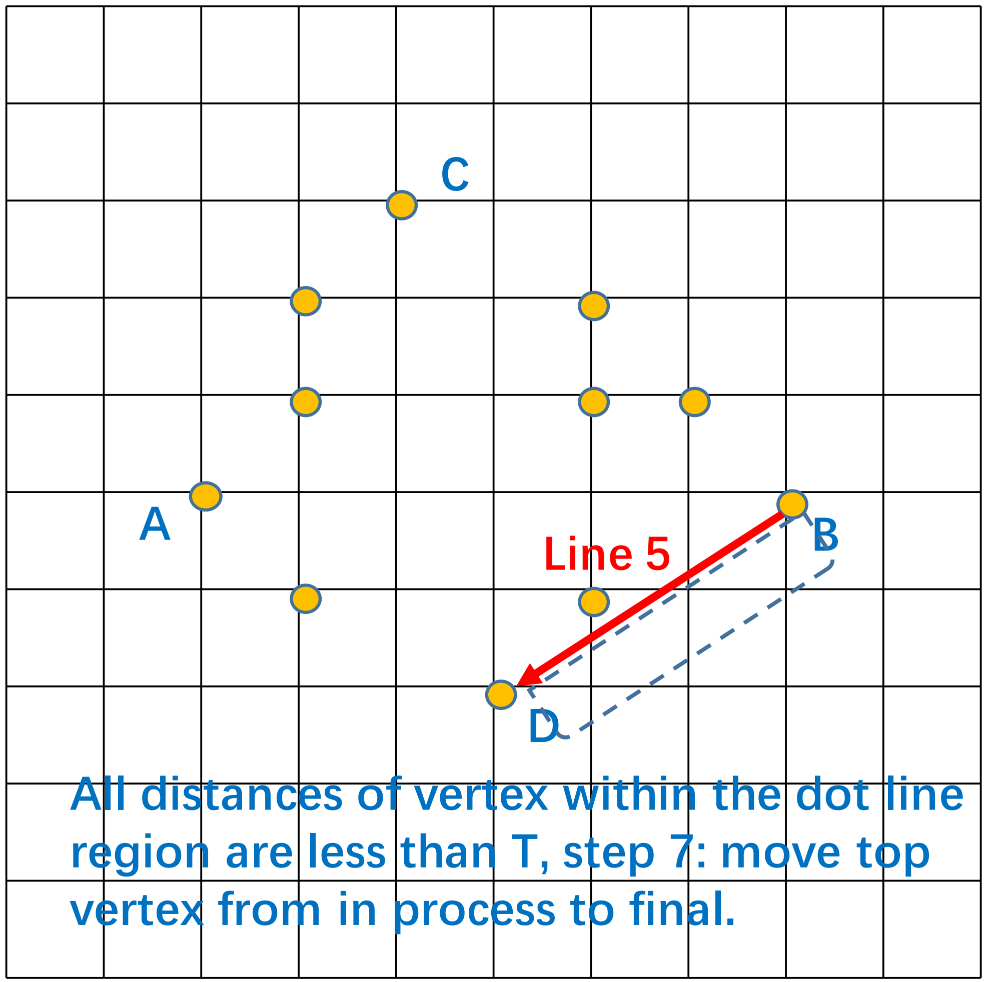
7. Otherwise, remove the last vertex from [in process] and make it the last vertex in [final].

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

iter5



All distances of vertex within the dot line region are less than T, step 7: move top vertex from in process to final.

final	In process
A	AB
A	ABC
AC	AB
ACB	A
ACB	AD
ACBD	A

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

2.Specify two starting points A,B with largest distance among all points.

3.Specify a threshold T (pixel distance).

4.Creating the stacks using the two starting points: for example [final] as A and [in process] A,B. Then connect the vertices on top of each stack with a **directed line**.

5. Compute the distance from this line to all the points in the **clock-wise or anti-clock-wise** side of the directed line 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.

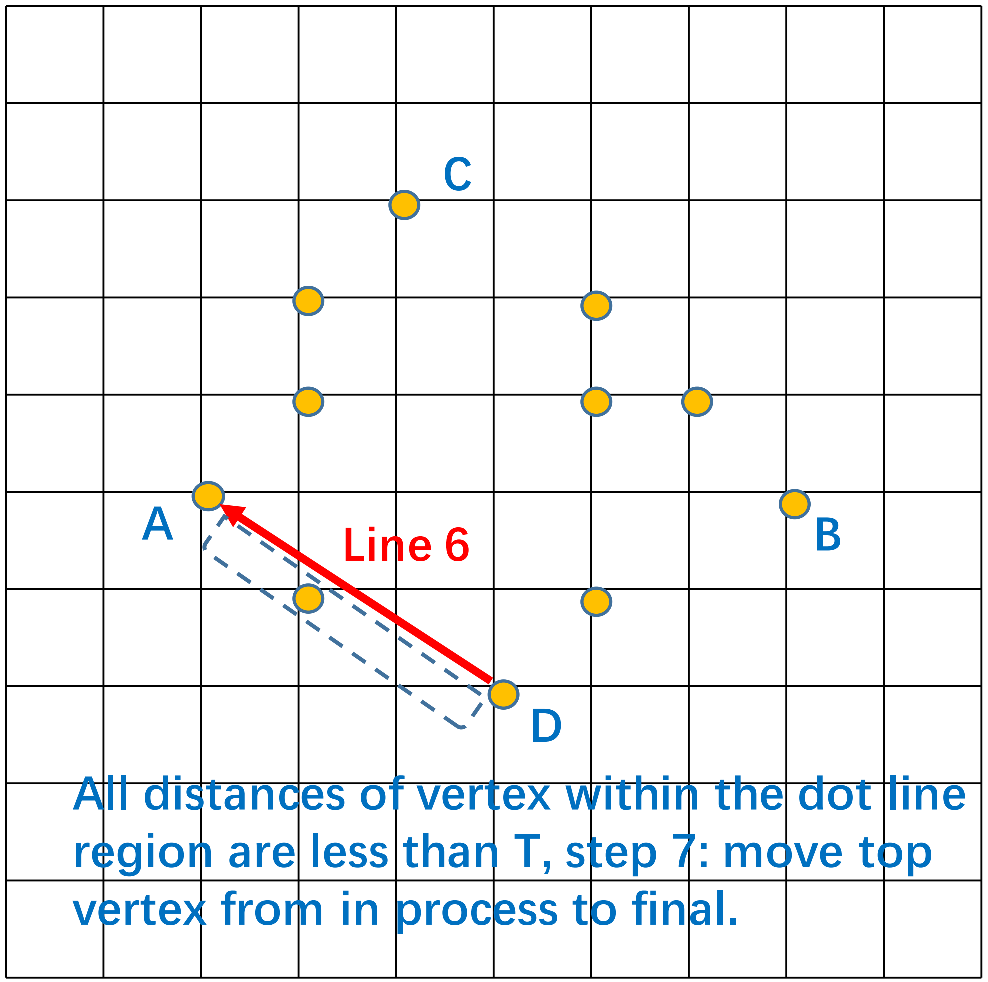
7. Otherwise, remove the last vertex from [in process] and make it the last vertex in [final].

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

iter6



All distances of vertex within the dot line region are less than T, step 7: move top vertex from in process to final.

final	In process
A	AB
A	ABC
AC	AB
ACB	A
ACB	AD
ACBD	A
ACBDA	null

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

2.Specify two starting points A,B with largest distance among all points.

3.Specify a threshold T (pixel distance).

4.Creating the stacks using the two starting points: for example [final] as A and [in process] A,B. Then connect the vertices on top of each stack with a **directed line**.

5. Compute the distance from this line to all the points in the **clock-wise or anti-clock-wise** side of the directed line 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.

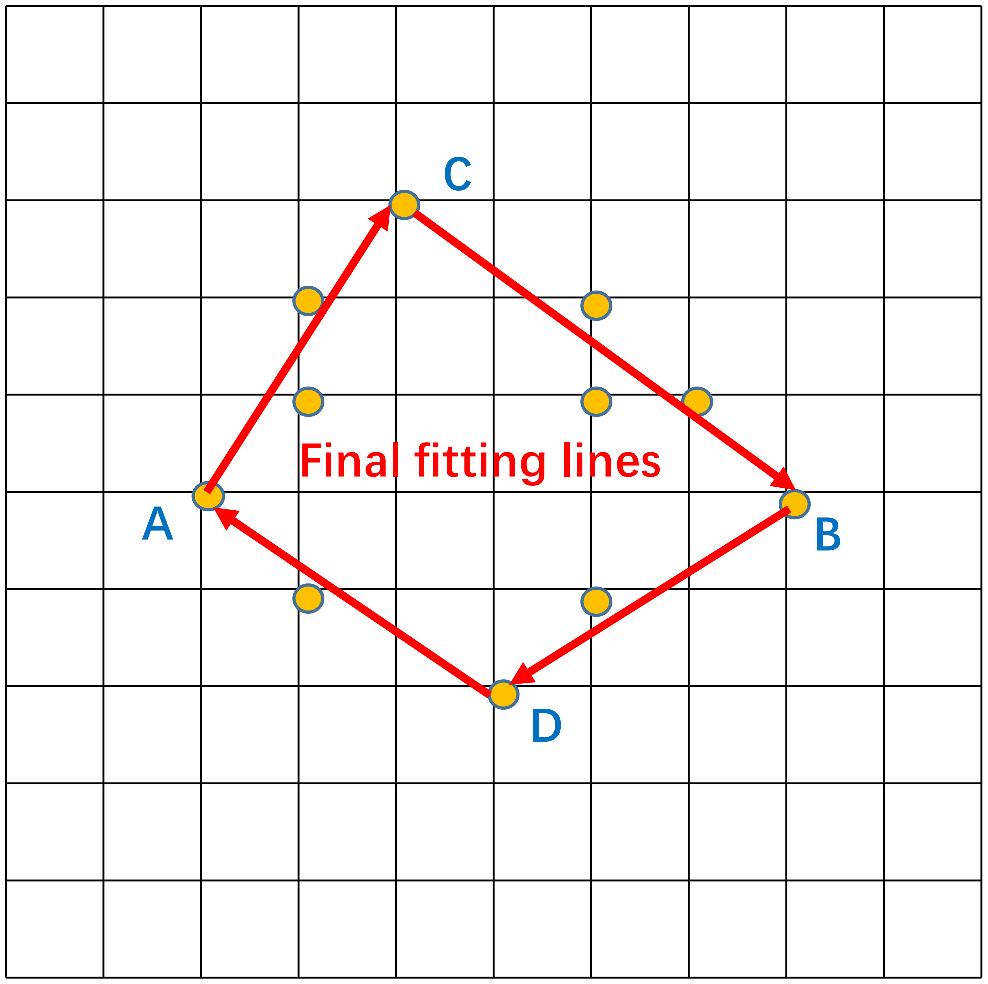
7. Otherwise, remove the last vertex from [in process] and make it the last vertex in [final].

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

iter6



final	In process
A	AB
A	ABC
AC	AB
ACB	A
ACB	AD
ACBD	A
ACBDA	null

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

2.Specify two starting points A,B with largest distance among all points.

3.Specify a threshold T (pixel distance).

4.Creating the stacks using the two starting points: for example [final] as A and [in process] A,B. Then connect the vertices on top of each stack with a **directed line**.

5. Compute the distance from this line to all the points in the **clock-wise or anti-clock-wise** side of the directed line 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].

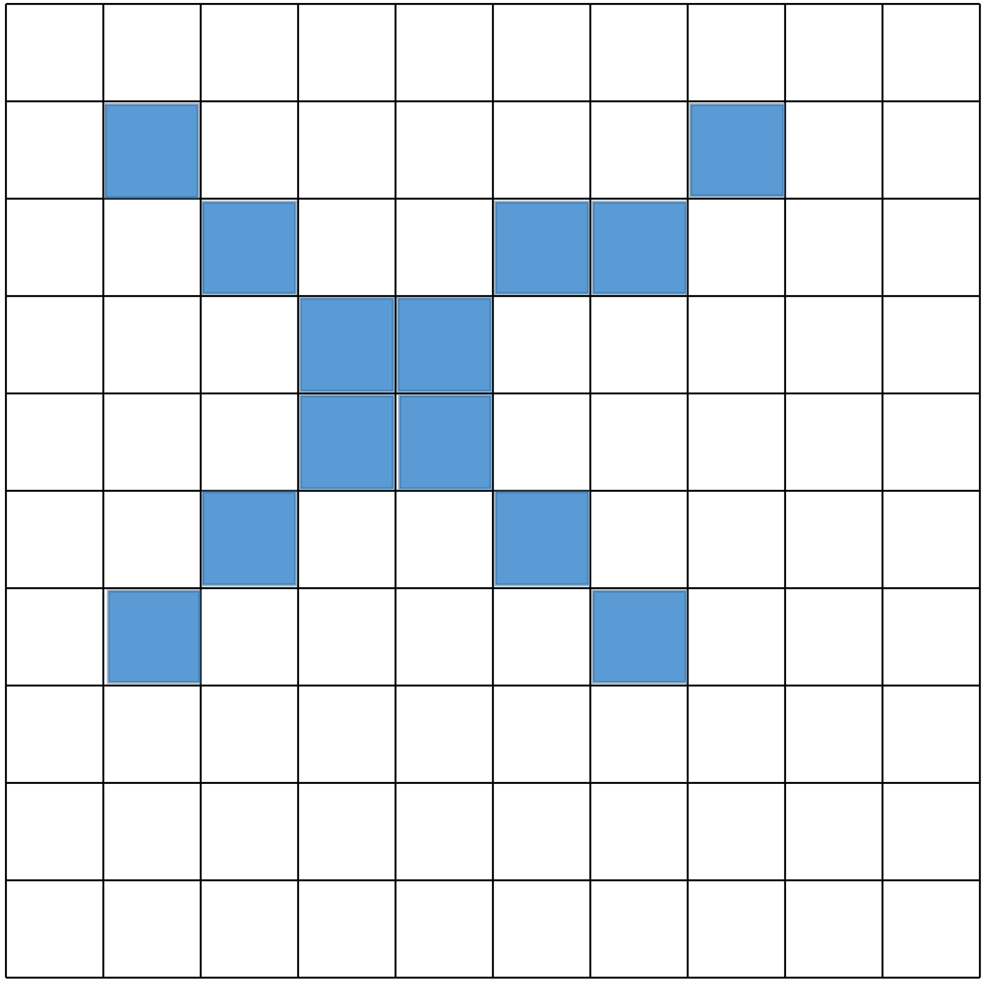
8. If [in process] is not empty, go to step 4.

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

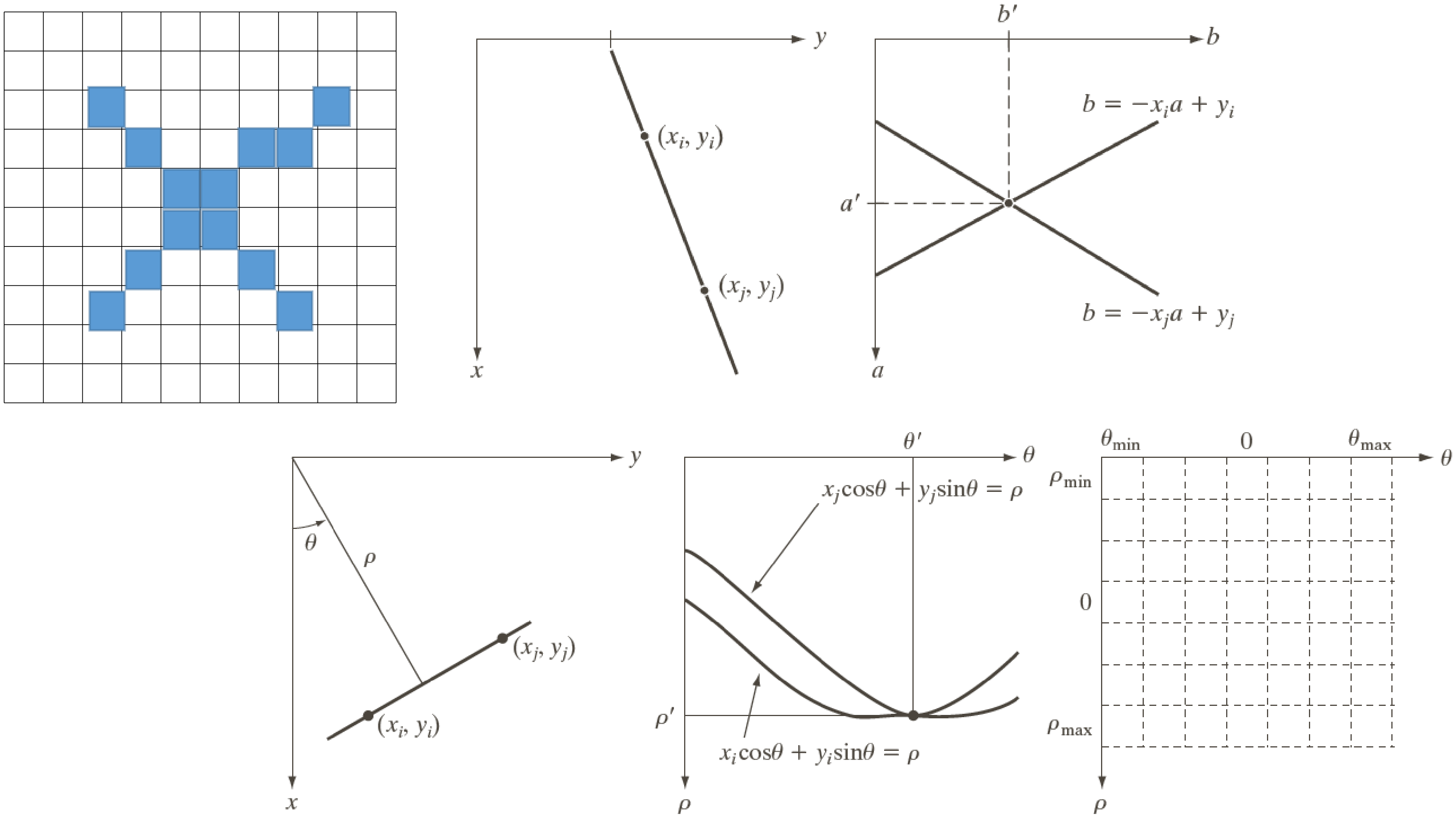
Polygonal fitting

- 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



Hough Transform (霍夫变换)



Hough Transform (霍夫变换)

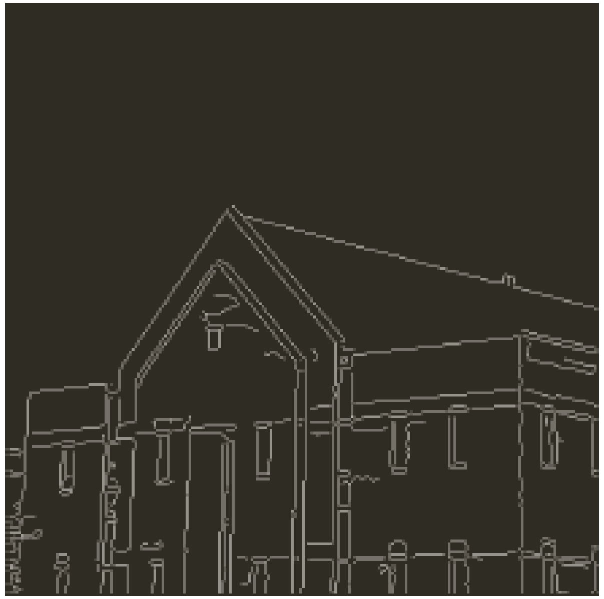
➢ An approach based on Hough Transform

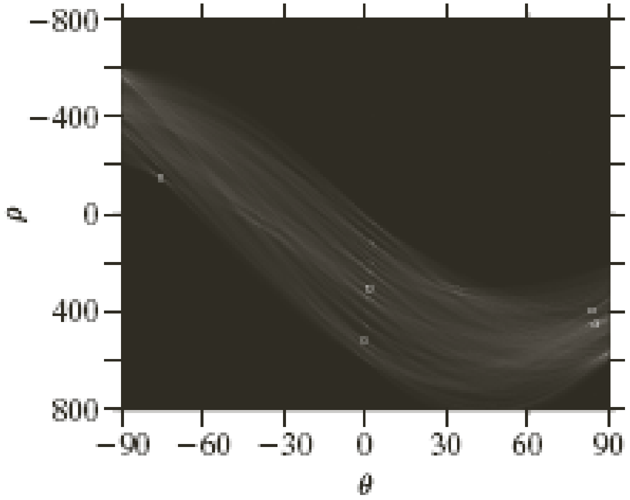
1. Obtain a binary edge image using any edge detector;
2. Specify subdivisions in the $\rho\theta$-plane;
3. Examine the counts of the accumulator cells (累加器单元) for high pixel concentrations;
4. Examine the relationship between pixels in a chosen cell.

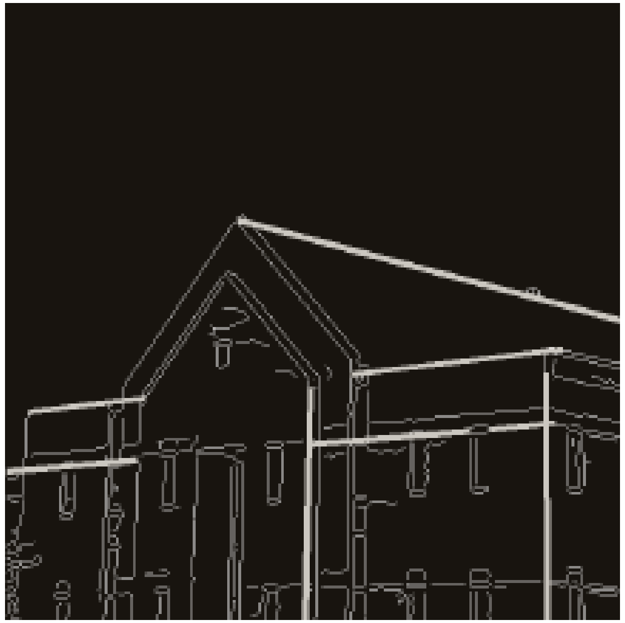
➢ Matlab function:

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

Hough Transform (霍夫变换)

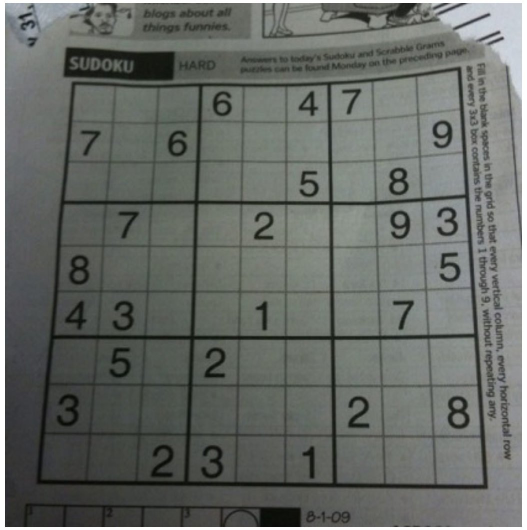






Take home message & Discussion

- Boundary detection & Global structure detection:



blogs about all things funnies.

SUDOKU HARD

Answers to today's Sudoku and Scrabble Grams puzzles can be found Monday on the preceding page.

			6		4	7		
7		6						9
					5		8	
	7			2			9	3
8								5
4	3			1			7	
	5		2					
3						2		8
		2	3		1			

Fill in the blank spaces in the grid so that every vertical column, every horizontal row and every 3x3 box contains the numbers 1 through 9, without repeating any.

8-1-09