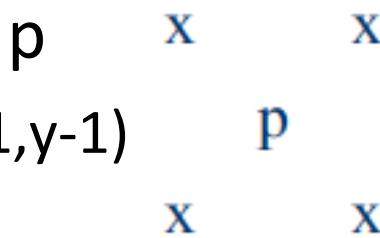
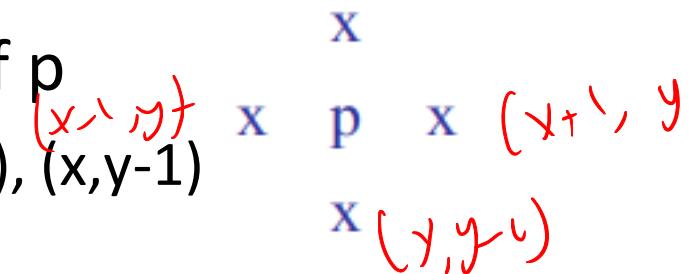


Basic Pixel Relationships

Neighbors of a Pixel

- A pixel p at coordinate (x,y) has (x_1, y_1)
 - $N_4(p)$: 4-neighbors of p
 - $(x+1, y), (x-1, y), (x, y+1), (x, y-1)$
 - $N_D(p)$: 4-diagonal neighbors of p
 - $(x+1, y+1), (x+1, y-1), (x-1, y+1), (x-1, y-1)$
 - $N_8(p)$: 8-neighbors of p :
 - a combination of $N_4(p)$ and $N_D(p)$



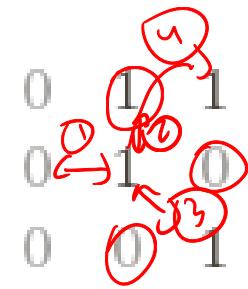
Definition

- Let V be the **set of intensity values** used to define adjacency (neighbors).
- In a binary image, $V = \{ 1 \}$ if we are referring to adjacency (neighbor) of pixels with value 1.
- In a gray-scale image, the idea is the same, but set V typically contains more elements.
 - For example, in the adjacency of pixels with a range of possible intensity values 0 to set V could be any subset of these 256 values.

Connectivity

- Let V be the set of gray-level values used to define connectivity
 - **4-connectivity** :
 - 2 pixels p and q with values from V are 4-connected if q is in the set $N_4(p)$
 - **8-connectivity** :
 - 2 pixels p and q with values from V are 8-connected if q is in the set $N_8(p)$
 - **m-connectivity** (mixed connectivity):
 - 2 pixels p and q with values from V are m-connected if
 - q is in the set $N_4(p)$ or
 - q is in the set $N_D(p)$ and the set $N_4(p) \cap N_4(q)$ is empty.

Example



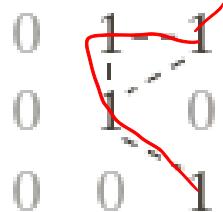
Arrangement
of pixels

① N_u, not connected

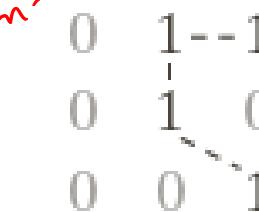
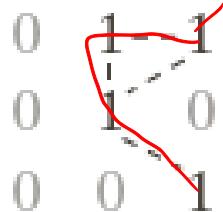
② N_u, connected (some value)

③ Diagonal and 8-connective, its intersections are empty (not!) so it's m-connective

(c) m-connectivity eliminates the multiple path connections that arise in (b) 8-connectivity.



8-neighbors of the
center pixel



m-neighbors of
the center pixel

④ diagonal and 8 connective joint if the intersections is not empty so it's not m connective

Adjacent

- A pixel p is adjacent to a pixel q if they are connected.
- Two image area subsets S_1 and S_2 are adjacent if some pixels in S_1 are adjacent to some pixels S_2 .

Regions

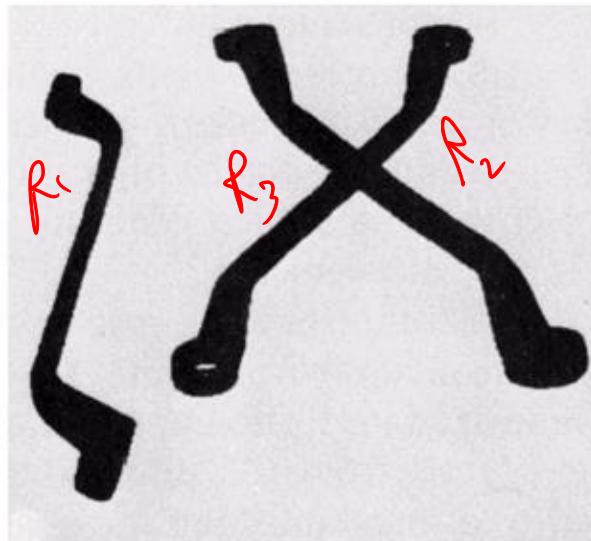
Let R be a subset of pixels in an image.

- We call R a ***region*** of the image if R is a connected set *of pixels*
- Two regions, R_i and R_j are said to be ***adjacent*** if their union forms a connected set.
- Regions that are not adjacent are said to be ***disjoint***.

Regions cont.

Let R_u denote the union of all the K regions, and let $(R_u)^c$ denote its complement

- We call all the points in R_u the **foreground**, and all the points in $(R_u)^c$ the **background** of the image.



R_1 and R_3
are connected

R_1 with R_2, R_3
are disjoint

Regions cont.

- The ***boundary*** (also called the ***border*** or ***contour***) of a region R is the set of points that are adjacent to points in the complement of R (***background***).
 - that have at least one background neighbor.
 - This set is called the **innerborder**.
 - The inner border may not form a **close path around the region**.
- Outer border; a border in the background.
 - **Always forms a close path around the region.**

Example

$\begin{matrix} 1 & 1 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{matrix} \Bigg\} R_i$	$\begin{matrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \end{matrix}$	$\begin{matrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \end{matrix}$	d
$\begin{matrix} 1 & 1 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{matrix} \Bigg\} R_j$	$\begin{matrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{matrix}$	$\begin{matrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{matrix}$	e
			f

g adjacency yes
n, no

- (d) Two regions that are **adjacent** if **8-adjacency** is used.
- (e) The circled point is part of the boundary of the 1-valued pixels only if 8-adjacency between the region and background is used.
border points of both 1-adjacency or 8-adjacency
- (f) The inner boundary of the 1-valued region does not form a closed path, but its outer boundary does.

Exercise

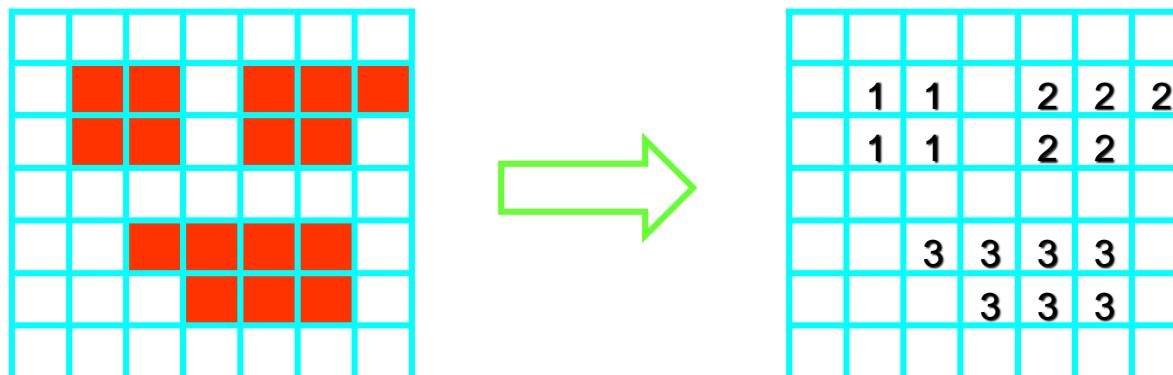
- Consider the two image subsets S_1 and S_2 :

	S_1					S_2					
0	0	0	0	0	0	0	0	1	1	0	
1	0	0	1	0	0	0	1	0	0	1	
1	0	0	1	0	1	1	1	0	0	0	
0	0	1	1	1	0	0	0	0	0	0	<i>or intersection = \emptyset</i>
0	0	1	1	1	0	0	0	1	1	1	

- For $V=\{1\}$, determine whether S_1 and S_2 are
 - 4-connected
 - 8-connected
 - m-connected
- because there's still 4-connectivity*
- 8-connected and m-connected but 4-connected*

Component Labeling

- **Component labeling** is one of the most fundamental operations on binary images. *Ø, I*
- Distinguishing different objects in an image
 - e.g. bacteria in microscopic images.
- We find all connected components in an image and assign a unique label to all pixels in the same component.



Images are from the Digital Image Processing textbook, see slide 7

labeling.xlsx - Excel

Anas Tomah

File Home Insert Page Layout Formulas Data Review View Help Tell me what you want to do

Cut Copy Format Painter Paste Clipboard Calibri 24 A B I U Merge & Center Conditional Format as Normal Bad Good Neutral Calculation Check Cell Explanatory... Input Linked Cell Note AutoSum Fill Clear Sort & Find & Filter Select

G8

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
1																	
2																	
3						1	1	1		2	2		1		✓		
4						1	1		2				2		✓		
5													3				
6							3	3	3	3			4		—		
7						3	3	3	3	3							
8																	
9																	

Sheet1

Ready

1 1 1 2 2 1
1 1 2 2 3
3 3 3 3 4
3 3 3 3

1 1 1 2 2 1
1 1 2 2 3
3 3 3 3 4
3 3 3 3

Component Labeling Algorithm

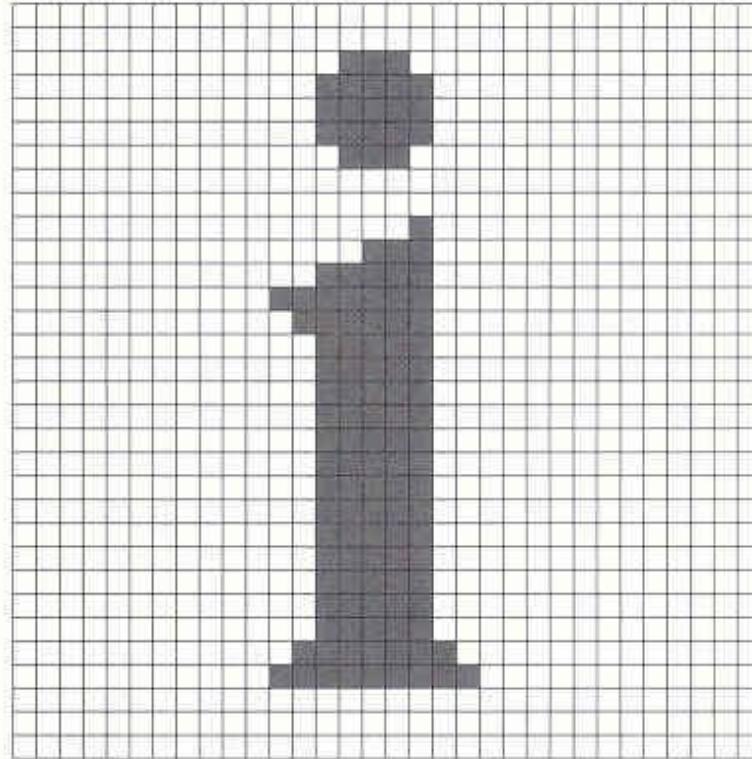
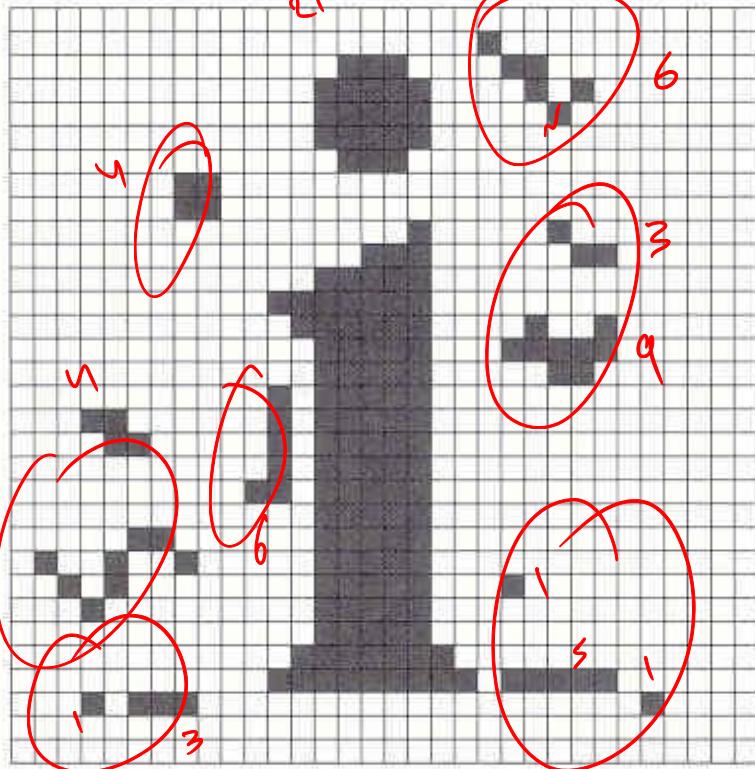
1. Scan the image left to right, top to bottom.
2. If the pixel is 1, then
 - If only one of its upper and left neighbors has a label, then copy the label.
 - If both have the same label, then copy the label.
 - If both have different labels, then copy the upper neighbor's label and enter both labels in the equivalence table as equivalent labels.
 - Otherwise assign a new label to this pixel and enter this label in the equivalence table.
3. If there are more pixels to consider, then go to Step 2.
4. Find the lowest label for each equivalence set in the equivalence table.
5. Scan the picture. Replace each label by the lowest label in its equivalence set.

Size Filter

- We can use component labeling to **remove noise** in binary images.
- For example, when we want to perform **optical character recognition (OCR)**, it often happens that there are small **groups of 1-pixels** outside the actual characters.
- Since these are usually very small, isolated blobs, we can remove them by applying a **size filter**, that is, labeling all components, computing their size, and for all components smaller than a **threshold T** , setting all of their pixels to 0.

Size Filter

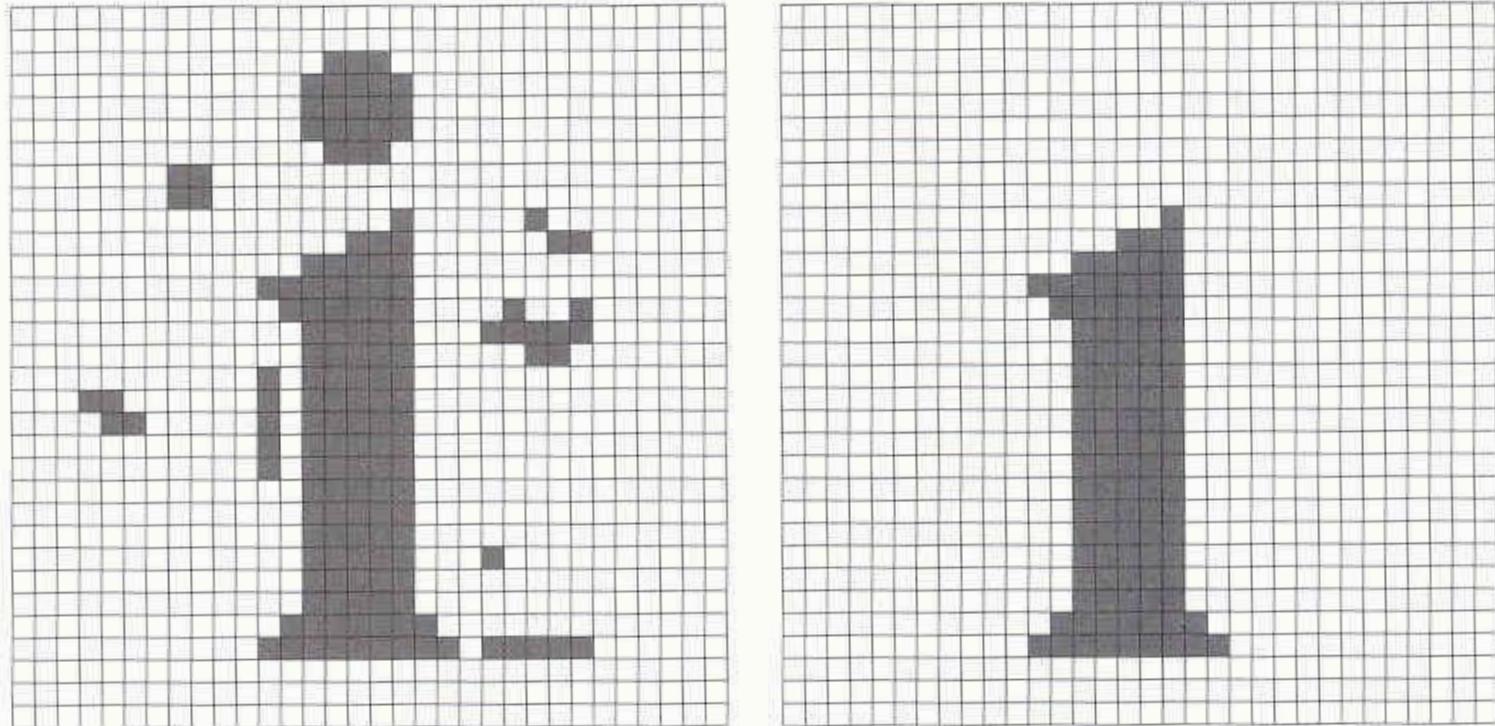
using component labeling we can find the sizes



we should decide on a threshold where if the component size exceeds the threshold
the component would be removed

- Here, for $T = 10$, the size filter perfectly removes all noise in the input image.

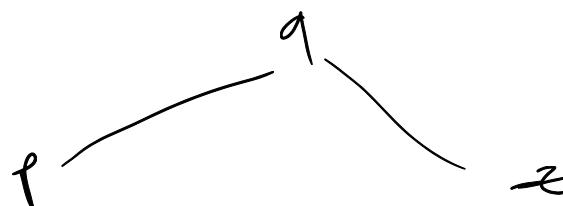
Size Filter



- However, if our threshold is too high, “accidents” may happen.

Distance Measures

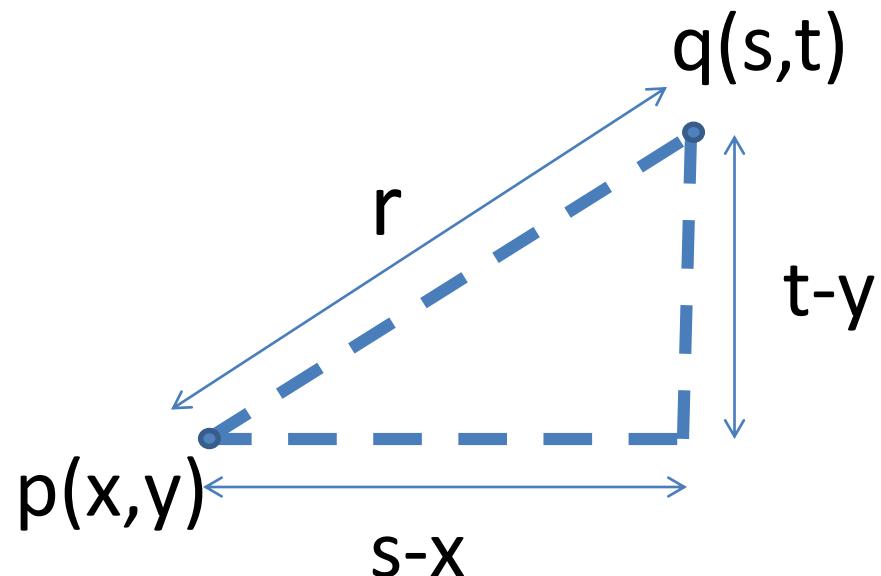
- For pixels p, q and z with coordinates(x,y), (s,t) and (u,v) respectively,
 - D is a distance function or metric if
 - (a) $D(p,q) \geq 0$; $D(p,q) = 0$ iff $p=q$ if $p \xrightarrow{D=0} q$ then $p=q$
 - (b) $D(p,q) = D(q,p)$ using the same route
 - (c) $D(p,z) \leq D(p,q) + D(q,z)$



Images are from the Digital Image Processing textbook, see slide 7

Euclidean Distance between p and q

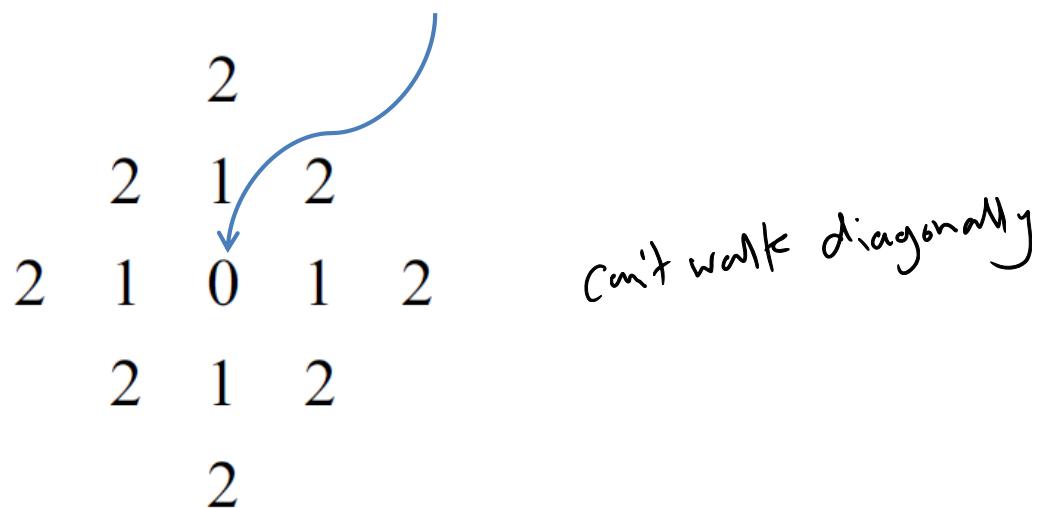
- $D_e(p, q) = \sqrt{(x - s)^2 + (y - t)^2}$



- Radius (r) centered at (x, y)

City-block Distance: D_4 distance

- $D_4(p, q) = |x - s| + |y - t|$
- Diamond centered at (x,y)
- **$D_4=1$ are 4-neighbors of (x,y)**



Chessboard Distance: D_8 distance

- $D_8(p, q) = \max(|x - s|, |y - t|)$
 - Square centered at (x, y)

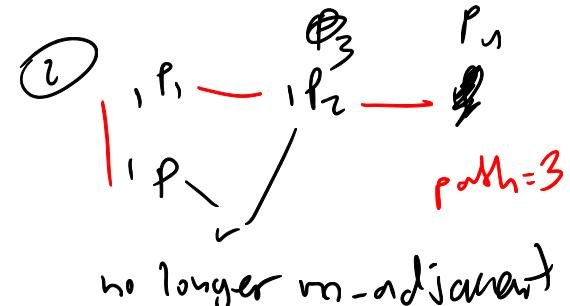
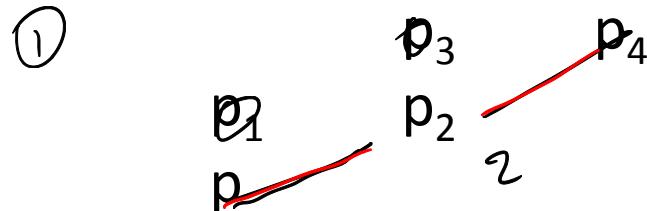
2	2	2	2	2
2	1	1	1	2
2	1	0	1	2
2	1	1	1	2
2	2	2	2	2

D_m distance

- The D_4 and D_8 distances involve only the coordinates of the points.
 - are independent of any paths that might exist between the points
- If the m-adjacency is used, the D distance between two points is defined as the shortest m-path between the m points.
 - the distance between two pixels will depend on the **values of the pixels along the path**, as well as the **values of their neighbors**.

D_m example

- Example, assume that p , p_2 , and p_4 have value 1 and that p_1 and p_3 can have a value of 0 or 1:



- 1 • If $V=\{1\}$, p_1 , p_3 are 0, the length of the shortest m -path between p and p_4 is 2.
- 2 • If p_1 is 1, then p_2 and p will no longer be m -adjacent. So the length of the shortest path between p and p_4 is now 3 ($pp_1p_2p_4$).
- 3 • If both p_1 and p_3 are one, the length of the shortest path is now 4 ($pp_1p_2p_3p_4$)

