EECS 332 Digital Image Analysis

Binary Image Analysis (I)

Ying Wu

Dept. Electrical Engineering & Computer Science
Northwestern University
Evanston, IL 60208

http://www.ece.northwestern.edu/~yingwu
yingwu@ece.northwestern.edu

Preface …

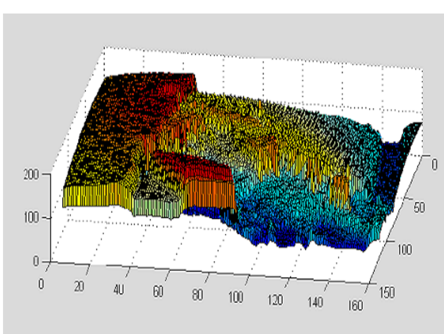
- "Life is complex, but the answer is simple."
- Sometimes, this is true.
- But sometimes, it is just the opposite.
  - For example, the operation of segmentation is natural and easy for human, but it is surprisingly difficult for computers.

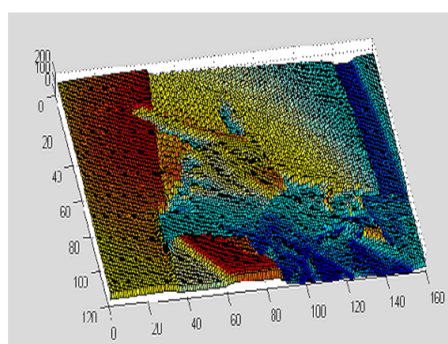
Look at our task!

139	144	147	151	152	154	157	111	117	121	83	93	88	102	83	84	86	80	82	73	64	57	31	59	76
139	146	148	151	154	157	159	102	104	114	112	84	111	82	103	98	95	79	82	76	65	59	31	53	77
140	146	150	152	154	159	159	126	126	125	93	83	111	79	104	100	95	82	82	75	68	57	32	48	79
142	149	151	153	157	159	161	126	124	124	116	96	101	108	83	79	87	80	87	75	71	61	32	43	79
144	150	152	153	158	161	161	126	115	111	125	122	80	101	98	104	88	95	87	76	72	61	32	39	79
144	151	153	154	158	160	162	126	132	132	117	102	87	114	91	107	88	93	90	80	73	66	32	37	79
145	151	154	158	158	162	162	130	135	132	131	125	94	86	112	81	97	87	86	76	74	68	33	34	77
144	150	155	158	159	123	163	91	108	125	109	104	68	104	116	88	96	91	91	83	79	66	33	36	77
144	152	155	157	160	74	97	90	116	132	125	125	61	80	98	94	95	81	76	76	79	66	33	34	77
146	151	154	157	159	110	61	109	117	138	135	133	55	105	81	114	88	100	90	81	76	68	32	36	75
146	150	156	158	160	161	55	89	120	95	100	110	61	83	87	115	82	102	88	76	81	69	32	34	75
147	152	155	160	160	163	161	53	95	89	107	126	51	87	93	97	114	86	90	91	75	69	32	36	73
146	152	156	159	161	162	162	53	71	108	108	136	47	98	122	86	91	95	88	79	71	73	31	34	72
147	152	156	159	161	162	159	147	57	68	116	125	45	93	129	116	94	102	89	89	82	77	32	37	69
147	154	157	160	160	162	154	103	83	54	82	129	33	87	126	97	82	110	89	87	79	75	34	37	68
149	152	157	159	160	163	146	107	145	59	71	108	123	94	135	126	115	83	100	88	88	80	37	36	62
145	153	156	159	160	161	147	130	137	98	83	114	109	111	123	126	122	110	82	88	73	81	43	38	57
146	152	157	158	161	161	114	153	90	111	86	111	95	104	115	131	126	104	82	91	75	83	46	37	53
146	152	155	159	160	160	122	147	72	80	86	105	100	95	101	131	117	121	112	89	91	73	48	38	50
147	151	155	157	159	160	149	131	66	71	82	91	114	91	102	111	94	98	97	103	83	79	50	38	48
144	151	154	158	160	158	152	152	76	87	79	94	116	101	111	137	116	115	107	107	86	77	59	38	47
145	149	154	155	158	158	154	160	77	79	87	72	109	112	110	128	95	91	88	81	91	90	65	39	46
143	150	152	155	157	157	153	150	58	79	94	83	109	114	107	142	123	108	90	86	93	91	60	39	43
145	149	151	153	158	158	152	80	61	62	83	86	110	118	123	102	116	95	101	107	88	76	72	38	41
143	149	151	153	154	158	133	51	88	76	75	90	89	115	118	74	93	119	125	110	103	94	75	38	39
142	145	151	153	152	155	68	100	62	77	74	77	77	122	115	121	65	104	130	93	84	80	68	39	39
143	146	149	152	153	157	91	52	53	62	75	67	86	90	115	73	71	52	79	93	87	81	66	38	39
142	146	149	152	153	152	54	29	55	54	64	67	59	75	77	72	65	58	101	110	111	97	79	38	38
139	144	149	150	150	66	74	75	67	46	60	65	60	71	59	66	71	77	87	81	79	44	46	43	39
138	143	146	150	144	64	73	73	73	55	54	64	60	54	59	65	79	79	94	58	79	43	44	44	40
136	143	146	149	145	61	73	74	74	67	50	50	51	38	40	66	64	68	128	75	76	43	57	44	37
133	139	146	147	146	61	72	73	74	69	40	37	44	39	39	51	73	72	79	72	72	45	57	54	55
133	139	144	145	146	61	71	73	73	71	45	38	41	30	40	41	65	68	61	27	40	58	55	51	54
132	139	144	145	145	30	33	39	53	59	54	38	38	32	36	54	47	64	58	58	40	52	58	45	46
130	135	140	143	145	71	142	80	46	30	16	13	32	34	38	36	53	57	11	17	44	47	51	50	62
129	133	139	144	144	77	159	159	155	100	81	48	29	39	30	41	47	30	34	59	44	45	55	48	61
129	133	137	143	145	84	114	166	165	162	152	145	25	34	43	32	36	53	62	57	55	89	45	47	74
125	131	133	140	144	107	107	173	168	167	154	153	20	26	47	25	31	60	55	37	68	146	36	47	77
125	131	132	138	143	109	104	170	174	172	154	151	24	55	31	27	32	47	53	30	50	143	29	54	76
123	126	131	134	139	105	101	123	176	175	173	153	27	39	34	37	26	37	43	38	28	68	57	47	73
121	124	130	133	138	104	101	103	178	178	176	150	30	32	31	19	26	40	34	33	26	48	79	53	68
118	123	125	131	135	100	100	100	160	178	178	174	102	30	29	22	23	24	32	36	31	40	64	47	72
117	121	123	128	133	96	98	96	102	176	175	174	173	36	26	17	33	20	44	32	29	27	30	58	58
114	119	123	126	132	96	97	96	97	171	180	177	175	31	33	17	41	19	17	41	31	31	32	45	61
115	118	122	126	130	95	95	96	93	115	150	149	150	145	44	22	38	15	39	57	33	25	27	23	69

- Can you tell something from it?
- image analysis is to "find" something out of it!

The clue …





- let's view a gray-scale image as a terrain
- analyzing an image is like exploring a terrain

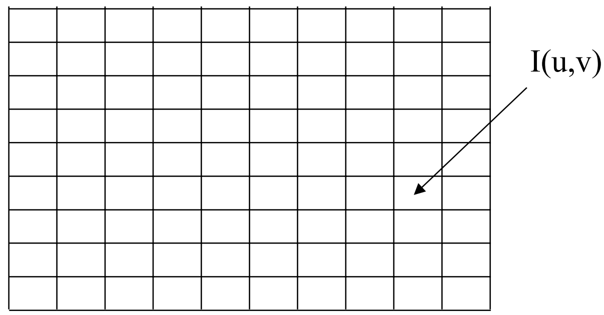
An analogy

	Terrain of Chicago	Depth image of Chicago
	Height of a location	Pixel intensity
P1	Find down Chicago?	Image segmentation
P2	How large is the downtown?	Region area
P3	Find I-90?	Edge detection
P4	Driving along Lakeshore?	Edge following
P5	Is Michigan Ave straight?	Line fitting

Questions for today

- Let's "find" *downtown Chicago*
  - **Fit an ellipse to an elongated region**
    - ✓ where
    - ✓ how large
    - ✓ what is the orientation

Where?

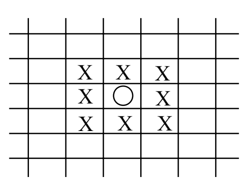


- The task: $\forall$I(u,v), make a decision 1/0
- Feature: the intensity $\Leftrightarrow$ "the height of the location"
- A simple solution: $\rightarrow$ thresholding

$$B(u,v) = \begin{cases} 1 \text{ (downtown)} & \textit{if } I(u,v) > t \\ 0 \quad \text{(suburban)} & \textit{otherwise} \end{cases}$$

A better idea

- Check the neighbor (or context)



- Then use the average of the neighbor

$$B(u,v) = \begin{cases} 1 & \textit{if } \bar{I}[N(u,v)] > t \\ 0 & \textit{otherwise} \end{cases}$$

Question

- How do you know the threshold?
  - Magic?
  - Ad hoc?
  - Heuristics?
  - Other ideas?
    - ✓ keep this question, we'll solve it in two weeks

How large?

- Size

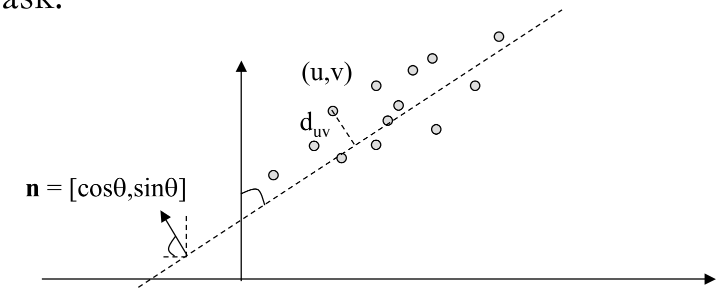
$$A = \sum_{u=1}^{n} \sum_{v=1}^{m} B(u,v)$$

- Center

$$\begin{cases} \bar{u} = \dfrac{\sum_{u=1}^{n} \sum_{v=1}^{m} uB(u,v)}{A} \\ \bar{v} = \dfrac{\sum_{u=1}^{n} \sum_{v=1}^{m} vB(u,v)}{A} \end{cases}$$

Orientation?

- Task:



– To find a direction (or a line), such that the sum of squared distance of all object points to the line is minimized, i.e.,

$$D = \sum_{u=1}^{n} \sum_{v=1}^{m} d_{uv}^2 B(u,v)$$

Problem Formulation

- To represent a line
  - $\rho = u \cos\theta + v \sin\theta$ (WHY?)
  - The normal of the line is $\mathbf{n} = [\cos\theta, \sin\theta]^T$
  - A more compact model
    - ✓ $\rho = \mathbf{n}^T \mathbf{p}$
    - ✓ i.e, ρ is the projection of $\mathbf{p}=[u,v]^T$ on **n**
- Then, $\forall \mathbf{p}$, its distance to the line is

$$d = (n^T p - \rho) = (u \cos\theta + v \sin\theta - \rho)^2$$

- So, the problem is formulated as:

$$(\rho^*, \theta^*) = \arg\min_{\rho,\theta} \sum_u \sum_v (u \cos\theta + v \sin\theta - \rho)^2 B(u,v)$$

Let's solve it

$$\partial D / \partial \rho = 2\sum_u \sum_v (u\cos\theta + v\sin\theta - \rho)B(u,v) = 0$$

$$\Rightarrow \quad \bar{u}\cos\theta + \bar{v}\sin\theta - \rho = 0$$

Let's prove it

i.e., the center is ON that line!

Cont.

$$\text{let} \begin{cases} \tilde{u} = u - \bar{u} \\ \tilde{v} = v - \bar{v} \end{cases}$$

$$d = (\tilde{u}\cos\theta + \tilde{v}\sin\theta)^2$$

$$D = \sum\sum \tilde{u}^2 B(u,v)\cos^2\theta + 2\sum\sum \tilde{u}\tilde{v}B(u,v)\cos\theta\sin\theta + \sum\sum \tilde{v}^2 B(u,v)\sin^2\theta$$

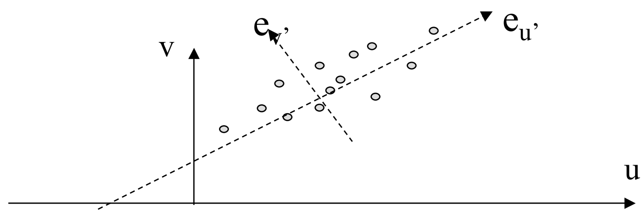
$$= a\cos^2\theta + b\sin\theta\cos\theta + c\cos^2\theta$$

$$= \frac{1}{2}[(a+c) + (a-c)\cos 2\theta + b\sin 2\theta]$$

$$\Rightarrow \frac{\partial D}{\partial \theta} = -\frac{1}{2}(a-c)\sin 2\theta + \frac{1}{2}b\cos 2\theta = 0$$

$$\Rightarrow \tan 2\theta = \frac{b}{a-c}$$

A better solution: PCA



$$\begin{bmatrix} u \\ v \end{bmatrix} \xrightarrow{R,t} \begin{bmatrix} u' \\ v' \end{bmatrix} = R\left(\begin{bmatrix} u \\ v \end{bmatrix} - t\right) = \begin{bmatrix} e_u^T \\ e_v^T \end{bmatrix}\left(\begin{bmatrix} u \\ v \end{bmatrix} - t\right)$$

The least square error is

$$d = [e_v^T(p-t)]^2 = e_v^T(p-t)(p-t)^T e_v$$

$$D = \sum_k e_v^T(p_i - t)(p_i - t)^T e_v$$

$$= e_v^T \sum_k (p_i - t)(p_i - t)^T e_v = e_v^T \psi e_v$$

ψ is the covariance matrix

Formulation and solution

$$e_v^* = \arg\min_{e_v} e_v^T \psi e_v \quad \text{s.t,} \quad e_v^T e_v = 1$$

Let's solve this constrained optimization problem

Construct the Largrangian $\quad L = e_v^T \psi e_v - \lambda(e_v^T e_v - 1)$

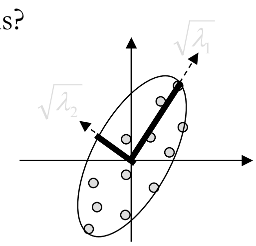
$$\frac{\partial L}{\partial e_v} = 0 \quad \Rightarrow \quad \psi e_v - \lambda e_v = (\psi - \lambda I)e_v = 0$$

What is this?

$\Rightarrow$ e_v is an eigenvector of ψ!

So let $\quad U = [e_1 \;\; e_2] \quad$ Then, $\quad \psi = U^T \Sigma U$

$$D = e_v^T U^T \begin{bmatrix} \lambda_u & \\ & \lambda_v \end{bmatrix} U e_v = \lambda_v$$



Principal Component Analysis

dataset $S = [s_1, s_2, ..., s_N]$

[1] mean $t = \frac{1}{N}\sum_{k=1}^{N} s_i$ and $\tilde{s}_k = s_k - t$

[2] covariance matrix $M = \frac{1}{N}\sum_{k=1}^{N} \tilde{s}\tilde{s}^T$

[3] eigenvalue decomposition $M = U^T \Sigma U$

[4] sort eigenvalue $M = \begin{bmatrix} e_1^T \\ e_2^T \end{bmatrix} \begin{bmatrix} \lambda_1 & \\ & \lambda_2 \end{bmatrix} [e_1 \ \ e_2],\ \lambda_1 > \lambda_2$

principal axis: e_1

transformation: $\hat{s} = U(s - t)$

line: $e_1^T(x - t) = 0$

principal variance: $\sqrt{\lambda_1}$

EECS332 Digital Image Analysis

Binary Image Analysis (II)

Ying Wu

Dept. Electrical Engineering and Computer Science
Northwestern University
Evanston, IL 60208

http://www.ece.northwestern.edu/~yingwu
yingwu@ece.northwestern.edu

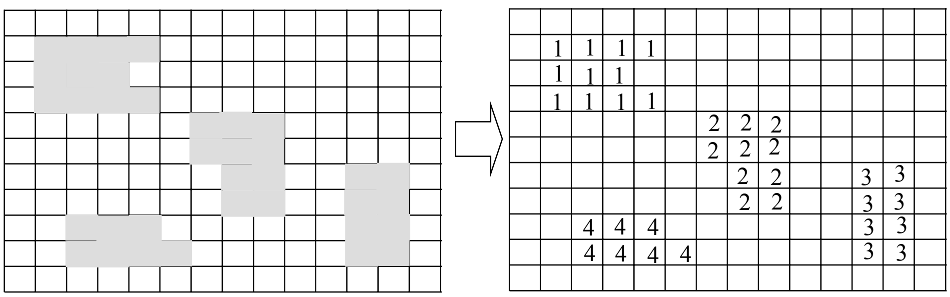
What we've learnt …

- Calculate region area and centroid
- Calculate region orientation

- We can do these because the region has been segmented or isolated.
  – In other words, if there are a number of isolated regions, we have to identify them and label them.

CCL: the task

- How many isolated regions are there in an image, and where are they?



I(u,v) L(u,v)

connected component labeling (CCL)

Definitions

- Neighbors

4-neighbor 8-neighbor

- Path: a sequence of pixel indices

  (u_0,v_0), (u_1,v_1), …, (u_n,v_n), s.t., (u_k,v_k) is neighbor of (u_{k+1},v_{k+1}), $\forall k$, $0 \le k < n$

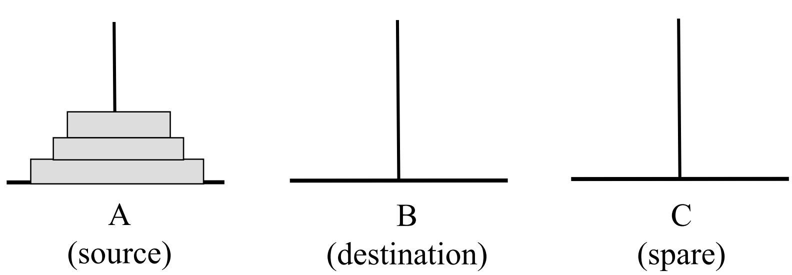
Definition

- Connectivity
  – p∈S, p is connected to q ∈S, if path(p,q) ∈S
- Boundary

$$S' = \{p \mid p \in S, \ \ 4N(p) \in \overline{S}\}$$

The Towers of Hanoi

- Given: N disks, three poles



- The puzzle: to move the disks one by one from A→B
- Constraints: a disk can only be placed on top of a larger disk

Recursion

- Statement:
  - Begin w/ N disks on A, and 0 disks on B and C
  - Solve Hanoi(N, A, B, C), or $A \xrightarrow[C]{N} B$
- Solution
  - Hanoi(N-1, A, C, B) $A \xrightarrow[B]{N-1} C$
  - Hanoi(1, A, B, C) $A \xrightarrow{1} B$
  - Hanoi(N-1, C, B, A) $C \xrightarrow[A]{N-1} B$
- Pseudo code

```
Hanoi(count, source, dest, spare)
  if count ==1
   move from source to dest;
  else {
   Hanoi(count-1, source, spare, dest);
   Hanoi(1, source, dest, spare);
   Hanoi(count-1, spare, dest, source);
  }
}
```

A recursive solution to CCL

- Find a "starting point", `I(u,v)=1 & L(u,v)=0`
- Recursion: `Labeling(I, L, u, v, label)`

```
if I(u,v)=0 | (I(u,v)=1 & L(u,v)≠0)
    Return;
else if L(u,v)=0 {
    L(u,v)=label;
    Labeling(I, L, u, v+1, label);
    Labeling(I, L, u+1, v, label);
    Labeling(I, L, u, v-1, label);
    Labeling(I, L, u-1, v, label);
}
```

- Iteration (find component one by one)

```
Label ++
```

- Discussion:

  is this algorithm good?

A sequential solution to CCL

- Scan image: left→right, top→ down
- cases

  L_u: label of the upper pixel

  L_l: label of the left pixel

	$L_u=0$	$L_u\neq 0$
$L_l=0$	$L(u,v)=L+1$	$L(u,v)=\max(L_u, L_l)$
$L_l\neq 0$	$L(u,v)=\max(L_u, L_l)$	$L_u=L_l$
		$L_u \neq L_l$ (E_table)

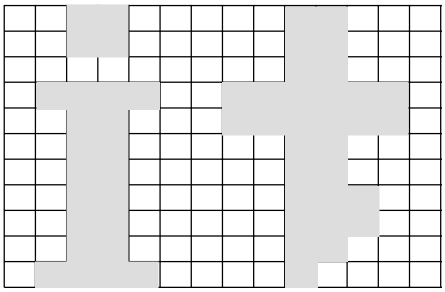
A sequential solution to CCL

- First scanning

```
if I(u,v)=1{
   Lu = L(u-1,v);   // upper label
   Ll = L(u,v-1);   // left label
   if Lu = Ll & Lu≠0 & Ll ≠0   // the same label
      L(u,v) = Lu;
   else if Lu≠Ll  & !(Lu&Ll) //either is 0
      L(u,v) = max(Lu, Ll)
   else if Lu ≠Ll & Lu>0 & Ll>0 // both
      L(u,v)=min(Lu, Ll);
      E_table(Lu, Ll);
   else L(u,v) = L+1; // none
}
```

- Second scanning
  – Renumbering the labels using the E_table

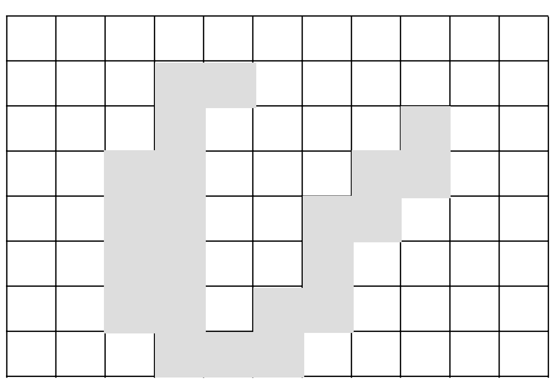
Example



		1	1						2	2			
		1	1						2	2			
									2	2			
	3	3	3	3			4	4	2	2	2	2	
		3	3				4	4	2	2	2	2	
		3	3						2	2			
		3	3						2	2			
		3	3						2	2	2		
		3	3						2	2	2		
		3	3						2	2			
	5	3	3	3					2				

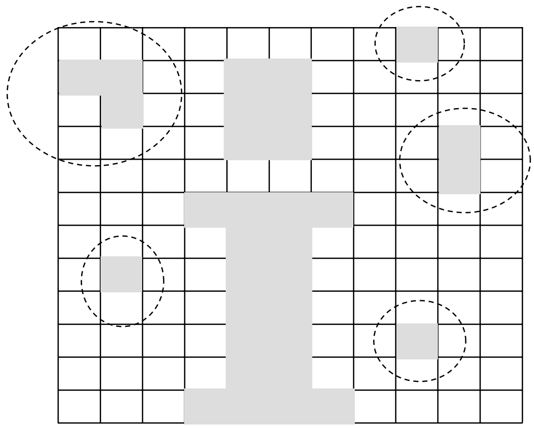
	1	2	3	4	5	6	7	8	9	10	11	12
2←→ 4	1	2	3	2	5	6	7	8	9	10	11	12
3←→ 5	1	2	3	2	3	6	7	8	9	10	11	12

Let’s work this out



Size Filter

- Set a threshold to get rid of those components whose areas are less than the threshold



Region Boundary

- Boundary following algo:
  1. Starting pixel: $p_0 \in S$
  2. Current pixel $c = p_0$, and $b = West(c)$, $b \in \bar{S}$
  3. $8N(c) = \{n_1, n_2, \ldots, n_8\}$ clockwise
  4. Find $k^* = \{k \mid \text{first } n_k \in S\}$
  5. Then, set

     $c \leftarrow n_{k^*}$

     $b \leftarrow n_{k^*-1}$
  6. Loop until $c = p_o$

Example