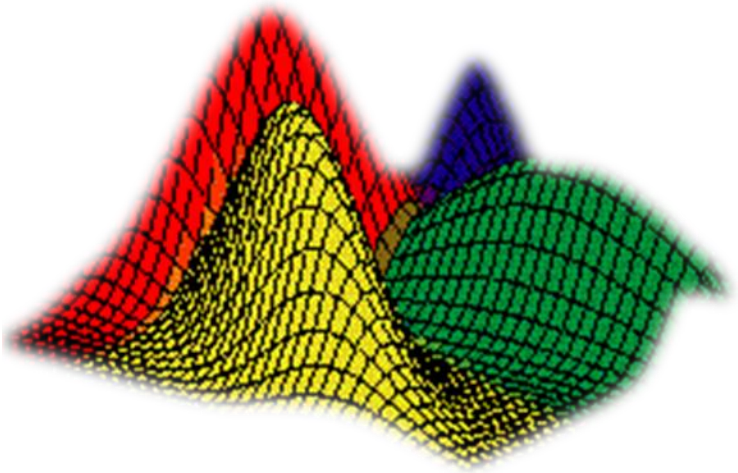
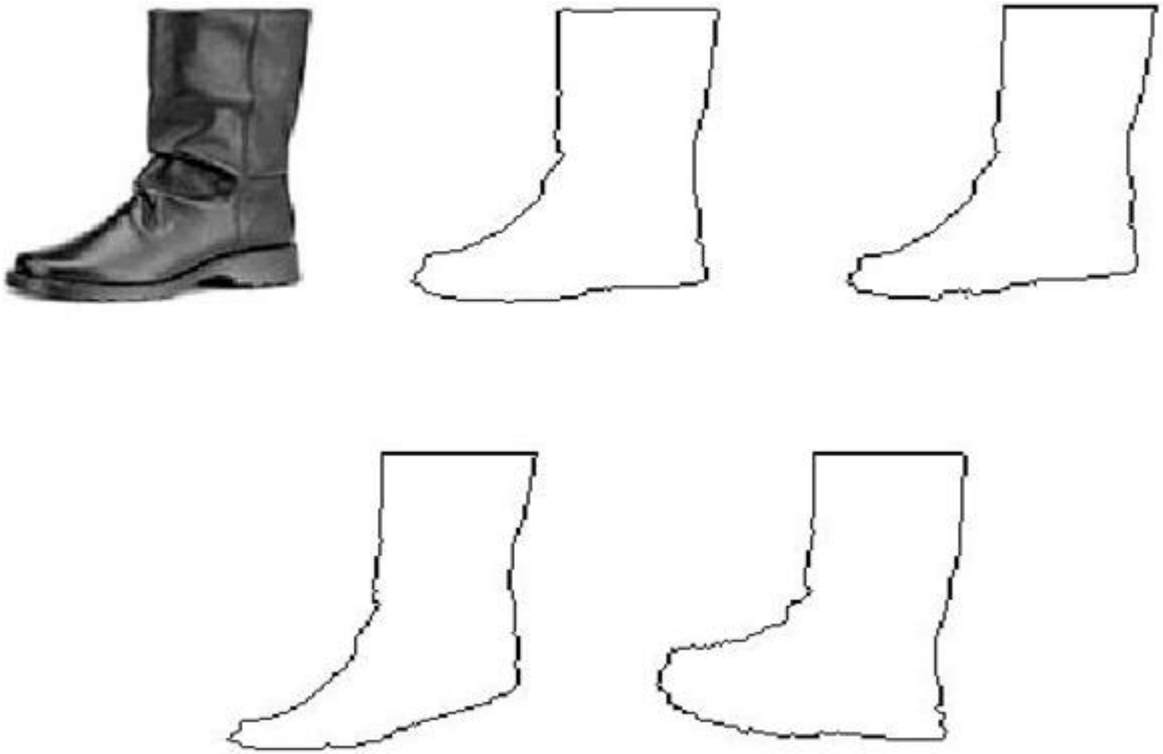


CSE 467
Pattern Recognition

Template Matching



Template Matching

- Typical Applications
  - Speech Recognition
  - Motion Estimation in Video Coding
  - Data Base Image Retrieval
  - Written Word Recognition
  - Bioinformatics

Template Matching

- The Goal:
  - Given a set of reference patterns known as TEMPLATES,
  - find the best match for unknown pattern
  - each class represented by a single typical pattern.
- requires an appropriate "measure" to quantify similarity or matching.

Template Matching

- The cost "measure":
  - deviations between the template and the test pattern.

Template Matching

- The cost "measure":
    - deviations between the template and the test pattern.
    - For example:
        - The word beauty may have been read as beeauty or beuty, etc., due to errors.
        - The same person may speak the same word differently.

Template Matching Methods

- Optimal path searching techniques
- Correlation
- Deformable models

TM using Optimal Path Searching

- Representation: Represent the template by a sequence of measurement vectors or string patterns

Template: $\underline{r}(1), \underline{r}(2), \ldots, \underline{r}(I)$

Test pattern: $\underline{t}(1), \underline{t}(2), \ldots, \underline{t}(J)$

TM using Optimal Path Searching

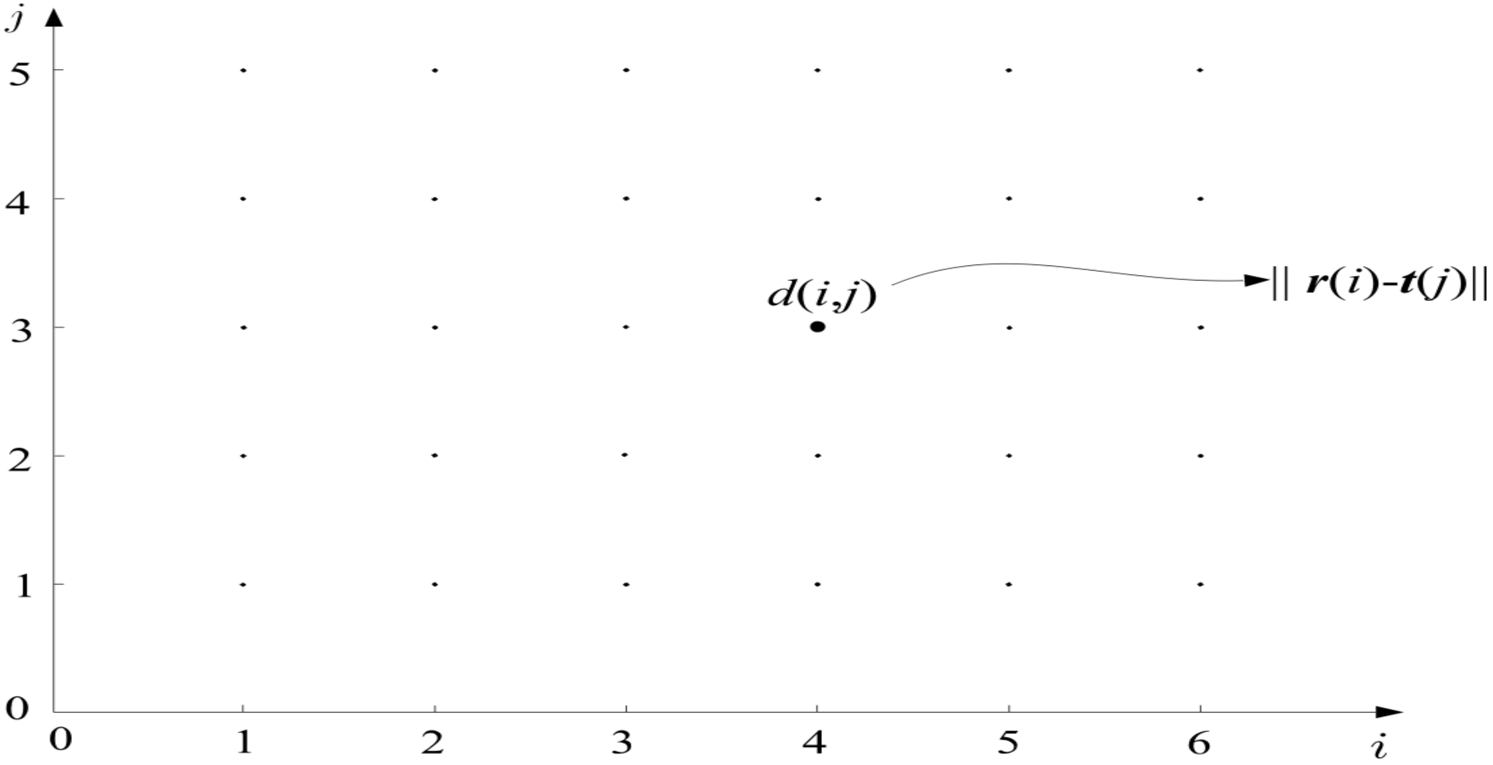
Template: $\underline{r}(1), \underline{r}(2), \ldots, \underline{r}(I)$

Test pattern: $\underline{t}(1), \underline{t}(2), \ldots, \underline{t}(J)$

- In general $I \neq J$
- We need to find an appropriate distance measure between test and reference patterns.

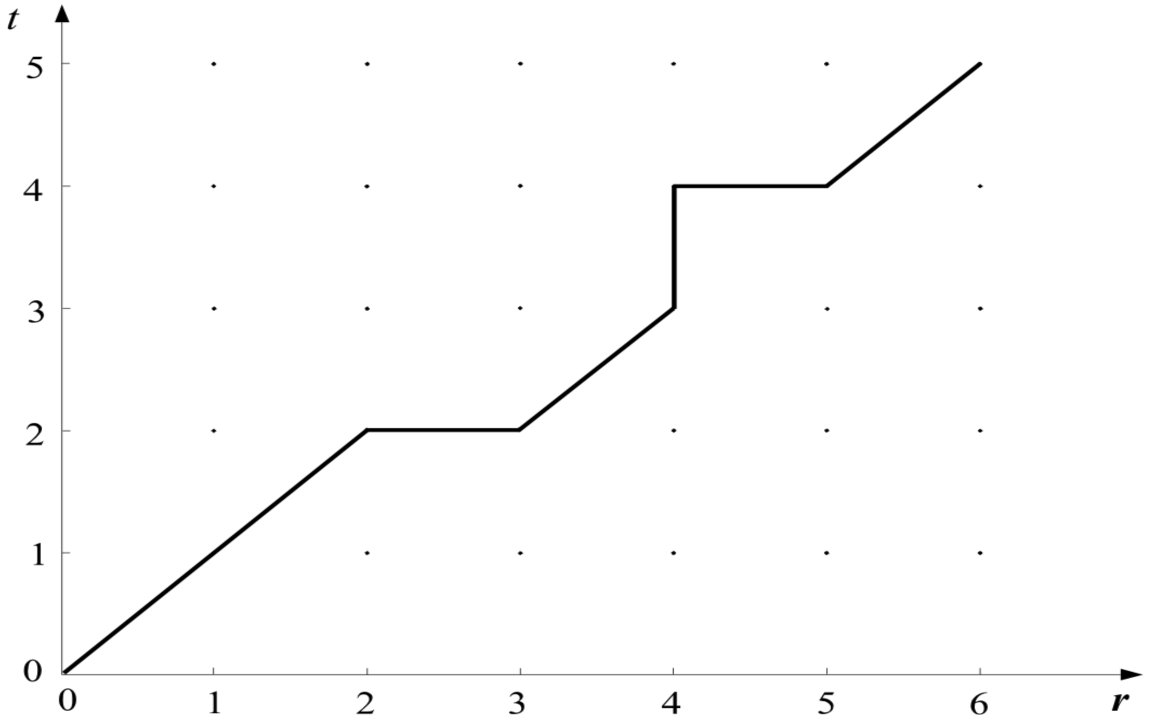
TM using Optimal Path Searching

- Form a grid with I points (template) in horizontal and J points (test) in vertical
- Each point (i,j) of the grid measures the distance between $\underline{r}(i)$ and $\underline{t}(j)$



TM using Optimal Path Searching

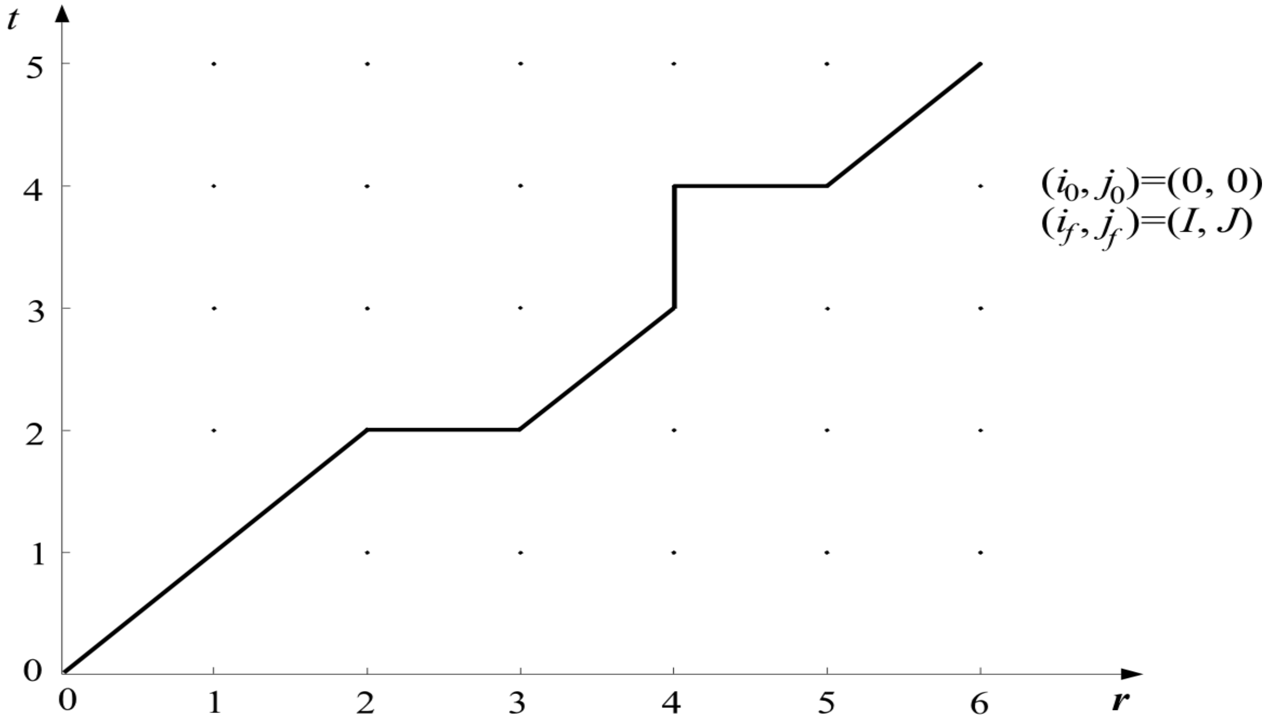
- Path: A path through the grid, from an initial node (i_0, j_0) to a final one (i_f, j_f), is an ordered set of nodes $(i_0, j_0), (i_1, j_1), (i_2, j_2) \ldots (i_k, j_k) \ldots (i_f, j_f)$



TM using Optimal Path Searching

- Path: A path is complete path if:

$$(i_0, j_0) = (0, 0), (i_1, j_1), (i_2, j_2), \ldots, (i_f, j_f) = (I, J)$$

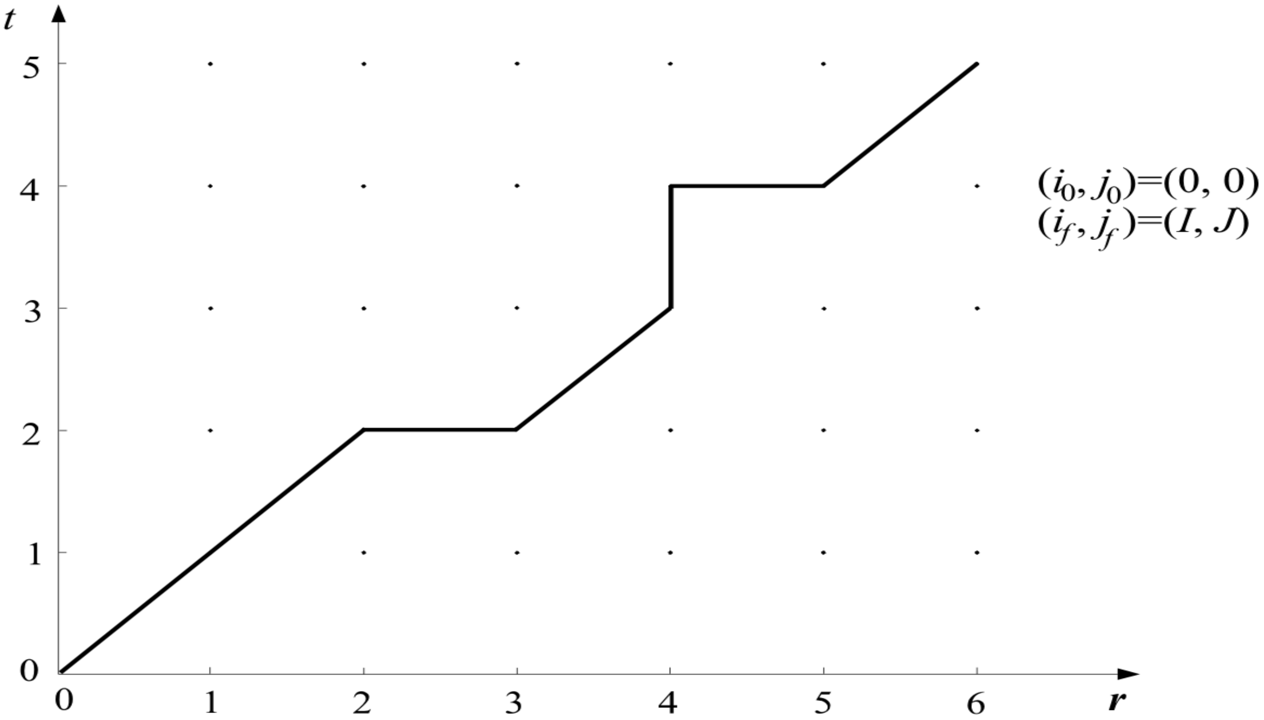


TM using Optimal Path Searching

- Each path is associated with a cost

$$D = \sum_{k=0}^{K-1} d(i_k, j_k)$$

where K is the number of nodes across the path



TM using Optimal Path Searching

- Let the cost up to node (i_k, j_k) be $D(i_k, j_k)$
- By convention
  - $D(0, 0)=0$
  - $d(0,0)=0$

TM using Optimal Path Searching

- The equation

$$D = \sum_{k=0}^{K-1} d(i_k, j_k)$$

assumes that each node has been associated with some cost

TM using Optimal Path Searching

- The equation

$$D = \sum_{k=0}^{K-1} d(i_k, j_k)$$

assumes that each node has been associated with some cost

- However, each transition $(i_{k\text{-}1}, j_{k\text{-}1})$ to (i_k, j_k) may also associate with a cost
- The new equation is:

$$D = \sum_{k} d(i_k, j_k | i_{k-1}, j_{k-1})$$

TM using Optimal Path Searching

$$D = \sum_{k} d(i_k, j_k | i_{k-1}, j_{k-1})$$

- Search for the path with the optimal cost $D_{opt.}$
- The matching cost between template $\underline{r}$ and test pattern $\underline{t}$ is $D_{opt.}$
- Costly operation
- Needs efficient computation

Bellman's Optimality Principle

- Optimal path:

$$(i_0, j_0) \xrightarrow{opt} (i_f, j_f)$$

Bellman's Optimality Principle

- Optimal path:

$$(i_0, j_0) \xrightarrow{opt} (i_f, j_f)$$

- Let (i,j) be an intermediate node, i.e.

$$(i_0, j_0) \rightarrow \ldots \rightarrow (i, j) \rightarrow \ldots \rightarrow (i_f, j_f)$$

Bellman’s Optimality Principle

- Optimal path:

$$(i_0, j_0) \xrightarrow{opt} (i_f, j_f)$$

- Let *(i,j)* be an intermediate node, i.e.

$$(i_0, j_0) \rightarrow \ldots \rightarrow (i, j) \rightarrow \ldots \rightarrow (i_f, j_f)$$

Then, write the optimal path through (i, j)

$$(i_0, j_0) \xrightarrow[(i,j)]{opt} (i_f, j_f)$$

Bellman's Optimality Principle

- Bellman's Principle:

$(i_0, j_0) \xrightarrow{opt} (i_f, j_f)$ can be obtained as

$$(i_0, j_0) \xrightarrow{opt} (i, j) \oplus (i, j) \xrightarrow{opt} (i_f, j_f)$$

- meaning: The overall optimal path from (i_0,j_0) to (i_f,j_f) through (i,j) is the concatenation of the optimal paths from (i_0,j_0) to (i,j) and from (i,j) to (i_f,j_f)

Bellman's Optimality Principle

- Bellman's Principle:

$$(i_0, j_0) \xrightarrow{opt} (i_f, j_f) \Leftrightarrow (i_0, j_0) \xrightarrow{opt} (i, j) \oplus (i, j) \xrightarrow{opt} (i_f, j_f)$$

- Let $D_{opt.}(i_{k\text{-}1}, j_{k\text{-}1})$ is the optimal path to reach $(i_{k\text{-}1}, j_{k\text{-}1})$ from (i_0, j_0), then Bellman's principle is stated as:

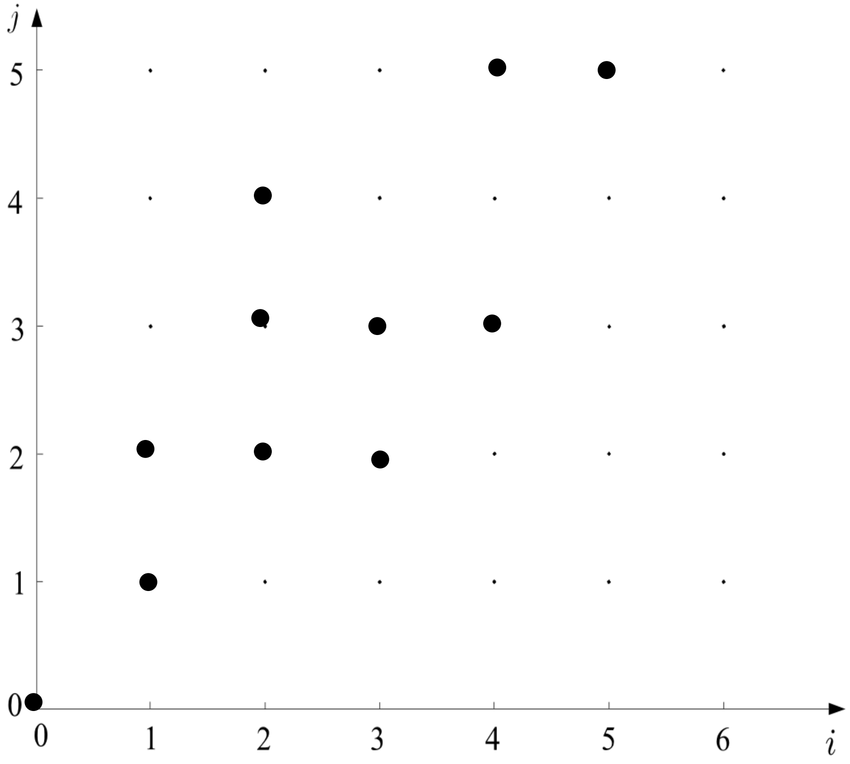
$$D_{opt}(i_k, j_k) = opt\{D_{opt}(i_{k-1}, j_{k-1}) + d(i_k, j_k \mid i_{k-1}, j_{k-1})\}$$

Bellman's Optimality Principle

$$D_{opt}(i_k, j_k) = opt\{D_{opt}(i_{k-1}, j_{k-1}) + d(i_k, j_k \mid i_{k-1}, j_{k-1})\}$$

- We don't need to search the whole space to find the optimal path

- Global and local constraints may be imposed to reduce the search space

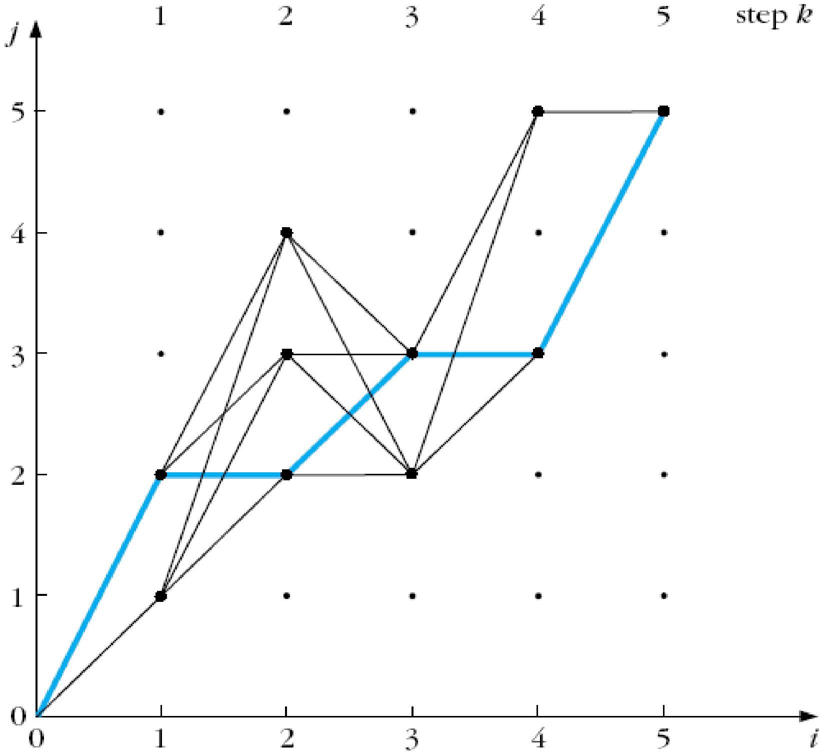


Bellman's Optimality Principle

$$D_{opt}(i_k, j_k) = opt\{D_{opt}(i_{k-1}, j_{k-1}) + d(i_k, j_k \mid i_{k-1}, j_{k-1})\}$$

- We don't need to search the whole space to find the optimal path

- Global and local constraints may be imposed to reduce the search space



Application of TM in Text Matching: The Edit Distance

- The Edit distance
  - It is used for matching written words. Applications:
    - Automatic Editing
    - Text Retrieval

Application of TM in Text Matching: The Edit Distance

- The Edit distance
  - It is used for matching written words. Applications:
    - Automatic Editing
    - Text Retrieval
  - The measure to be adopted for matching, must take into account:
    - Wrongly identified symbols
      e.g. "befuty" instead of "beauty"
    - Insertion errors, e.g. "bearuty"
    - Deletion errors, e.g. "beuty"

The Edit Distance

- Edit distance: **Minimal** total number of changes, C, insertions I and deletions R, required to change pattern A into pattern B,

$$D(A,B)=\min_{j}[C(j)+I(j)+R(j)]$$

where j runs over All possible variations of symbols, in order to convert $A \longrightarrow B$

The Edit Distance

- Edit distance: **Minimal** total number of changes, C, insertions I and deletions R, required to change pattern A into pattern B,

  $$D(A,B) = \min_j [C(j) + I(j) + R(j)]$$

  where j runs over All possible variations of symbols, in order to convert $A \longrightarrow B$

- *Example*: many ways to change **beuty** to **beauty**

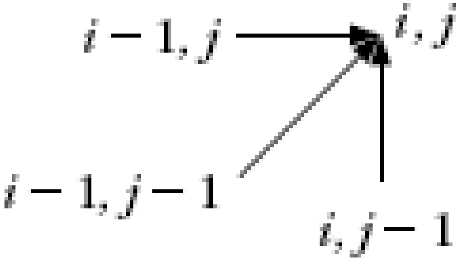
The Edit Distance

- The optimal path search algorithm can be used, provided we know
  - Initial conditions
  - Search space
  - Allowable transitions
  - Distance measure

The Edit Distance

- Cost $D(0,0) = 0$,
- Complete path is searched
- Allowable predecessors and costs:
  - $(i-1, j-1) \rightarrow (i, j)$

    $$d(i, j \mid i-1, j-1) = \begin{cases} 0, & \text{if } t(i) = r(j) \\ 1, & t(i) \neq r(j) \end{cases}$$
  - Horizontal $d(i, j \mid i-1, j) = 1$
  - Vertical $d(i, j \mid i, j-1) = 1$

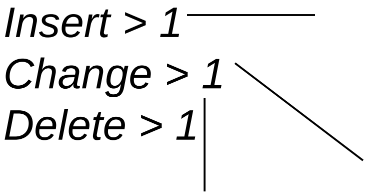


The Minimum Edit Distance

Ref. Word ***j***

Test Word ***i***

	Null	D	R	E	A	M
Null	0	1	2	3	4	5
D	1					
R	2					
R	3					
E	4					
U	5					
M	6					



Insert > 1

Change > 1

Delete > 1

If(r==c)

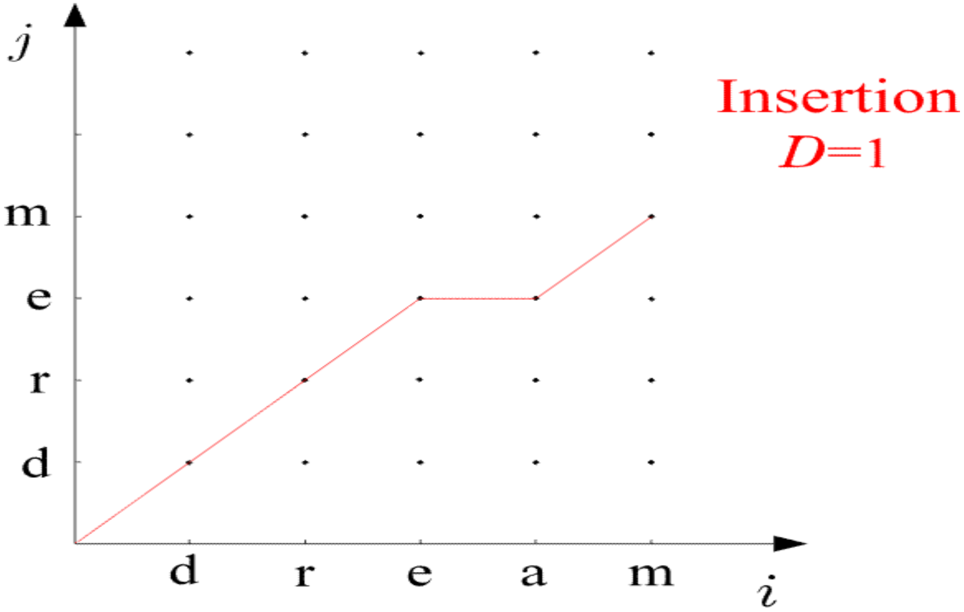
$d(i, j)= d((i-1), (j-1))$

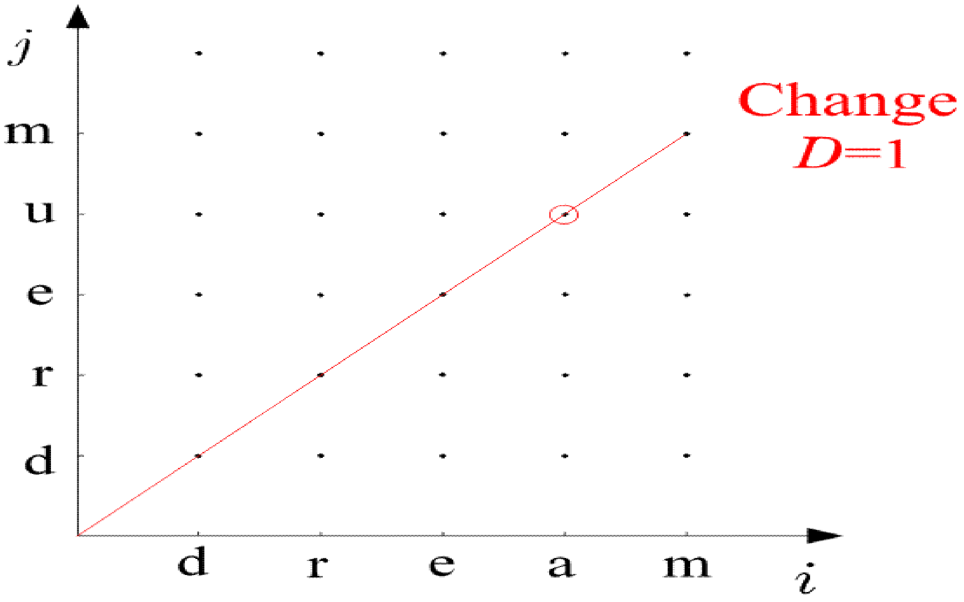
Else if(r!=c)

$d(i, j)= min\{d((i-1), (j)), d((i-1), (j-1)$

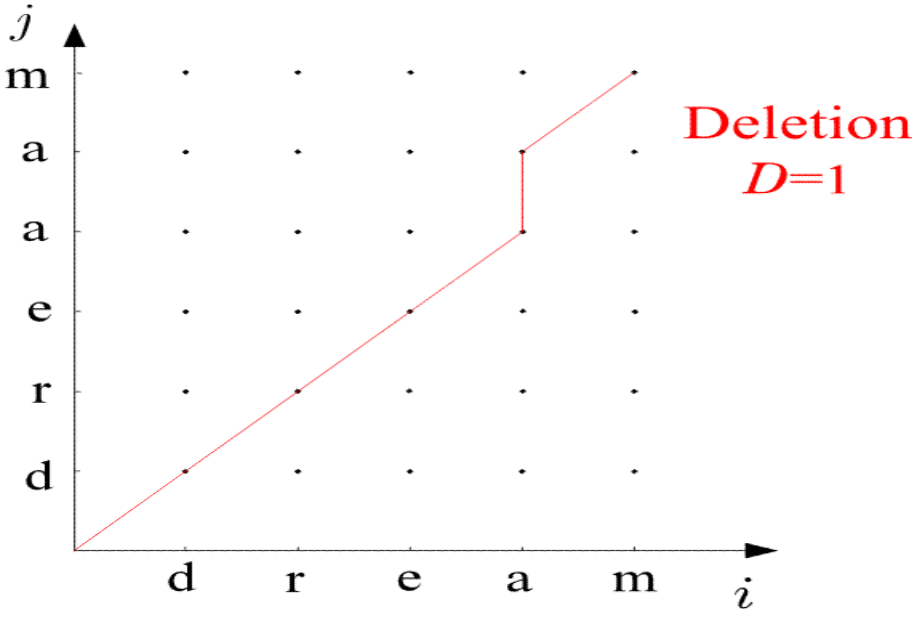
$d((i), (j-1))\}$

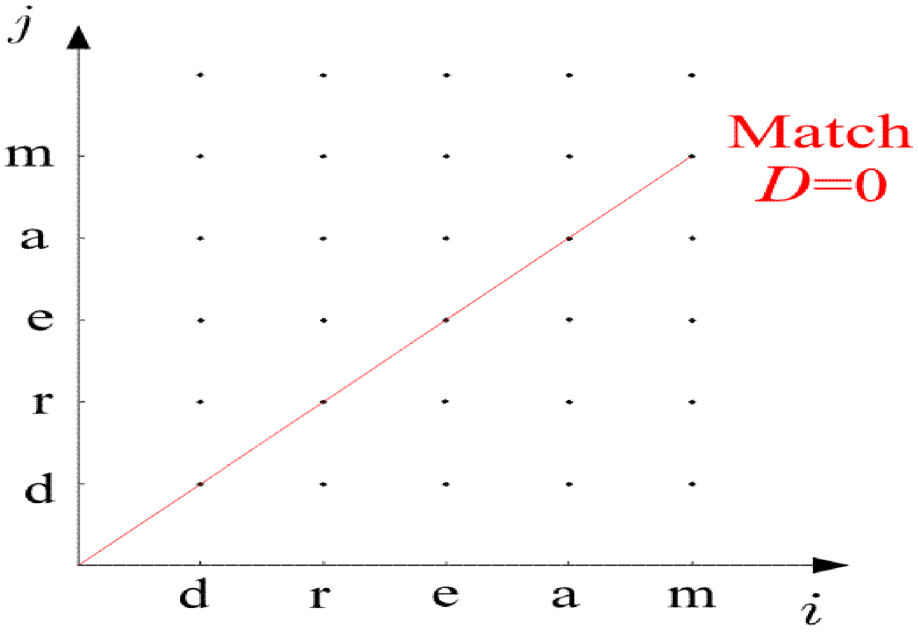
- Examples:





- Examples:

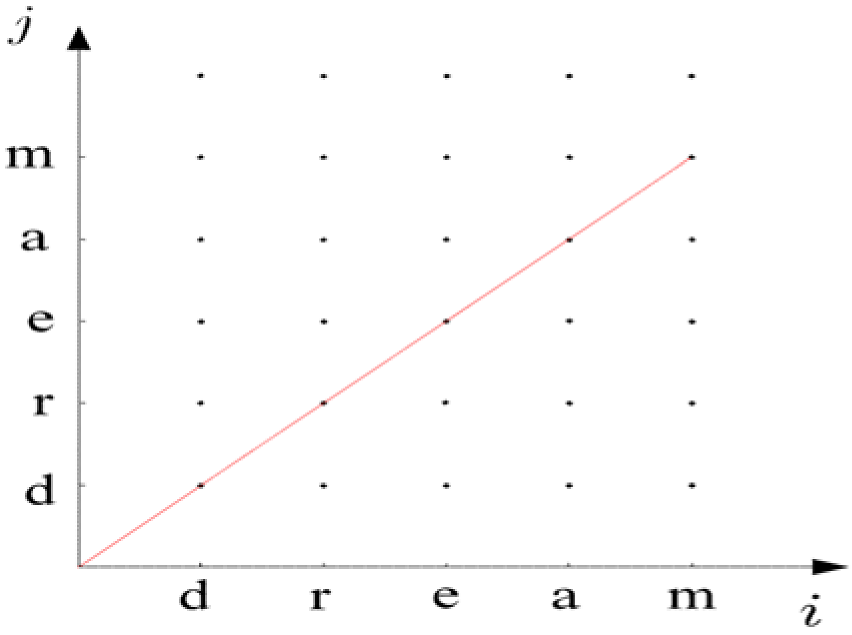




The Edit Distance

- The Algorithm

```
$D(0,0)=0$
For $i=1$, to $I$
    $D(i,0)=D(i-1,0)+1$
END {FOR}
For $j=1$ to $J$
    $D(0,j)=D(0,j-1)+1$
END{FOR}
For $i=1$ to $I$
    For $j=1$, to $J$
        $C_1=D(i-1,j-1)+d(i,j \mid i-1,j-1)$
        $C_2=D(i-1,j)+1$
        $C_3=D(i,j-1)+1$
        $D(i,j)=\min(C_1,C_2,C_3)$
    END {FOR}
END {FOR}
$D(A,B)=D(I,J)$
```



Application of TM in Speech Recognition

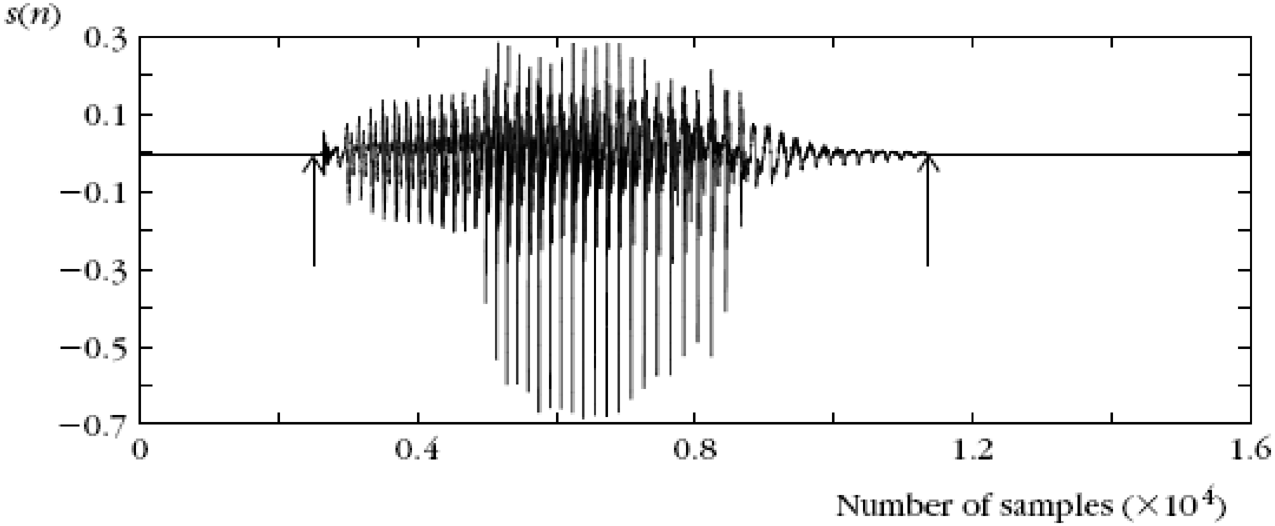
- A number of variations
  - Speaker Independent Speech Recognition
  - Speaker Dependent Speech Recognition
  - Continuous Speech Recognition
  - Isolated word recognition (IWR)

Application of TM in IWR

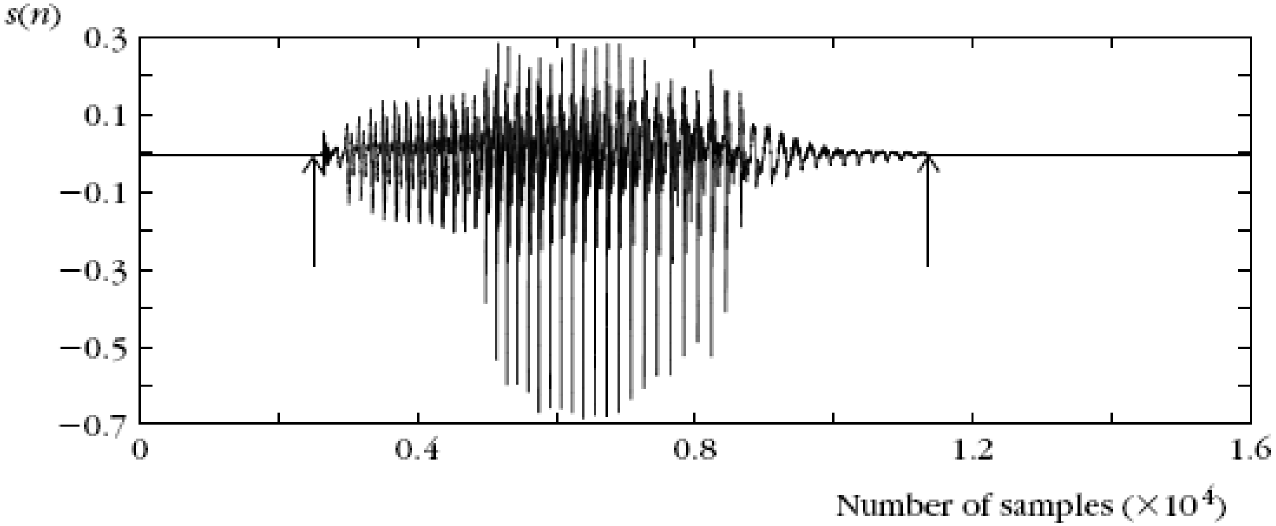
- The goal:
  - Given a number of known spoken words in a data base (reference patterns)
  - find the best match of an unknown spoken word (test pattern).

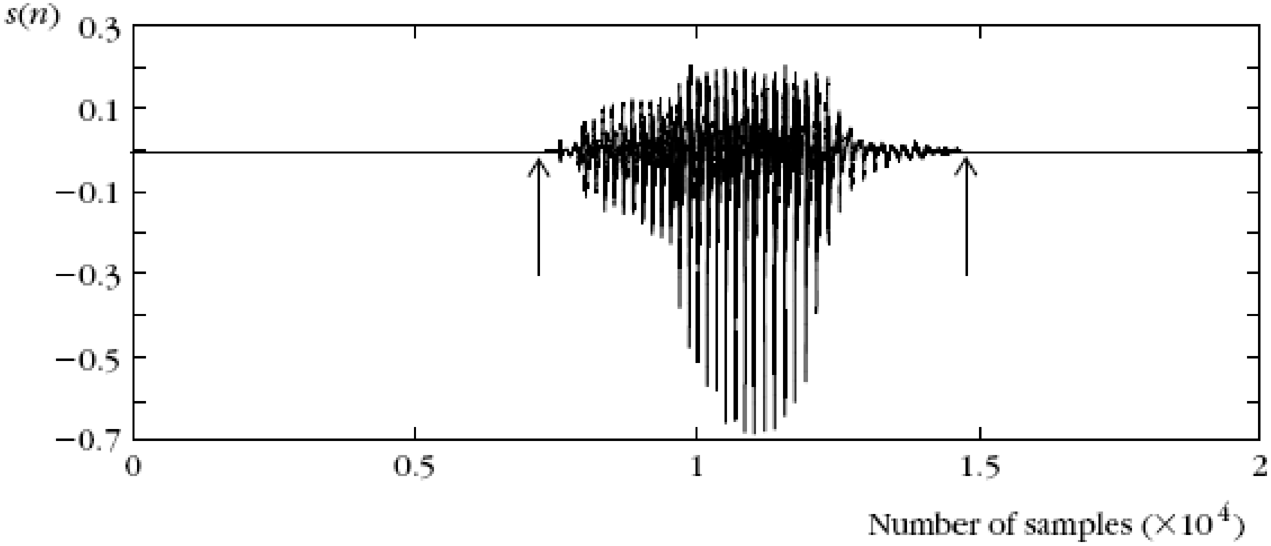
- Procedure:
  - compare the test word against reference words

Application of TM in IWR



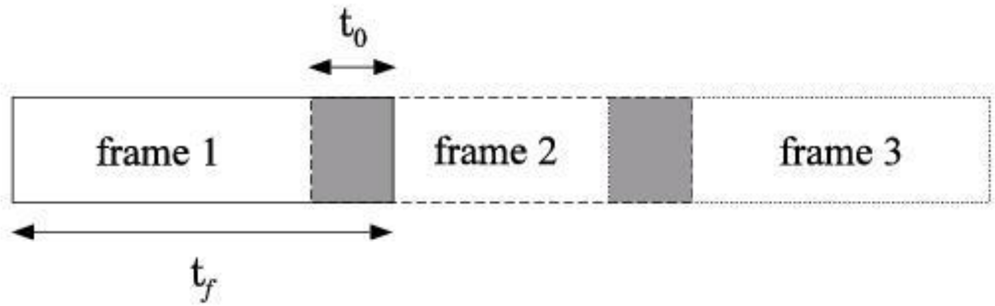
Application of TM in IWR





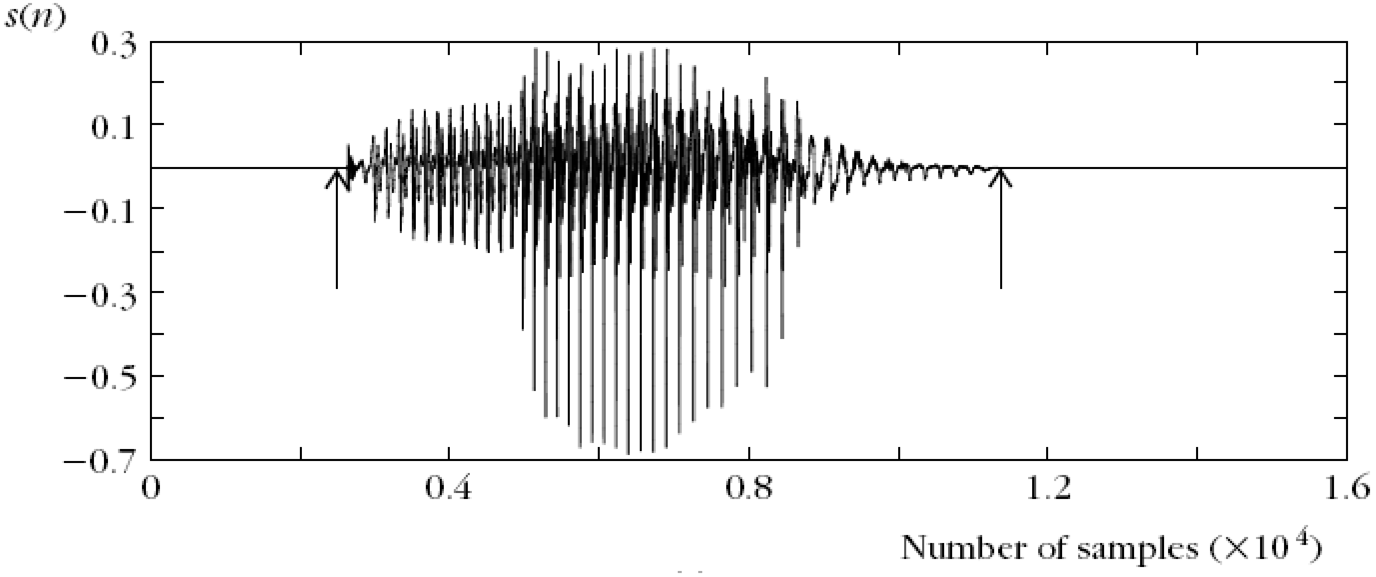
Application of TM in IWR

- The procedure:
  - Express the test and each of the reference patterns as sequences of feature vectors , $\underline{r}(i)$, $\underline{t}(j)$.
  - To this end, divide each of the speech segments in a number of successive frames.



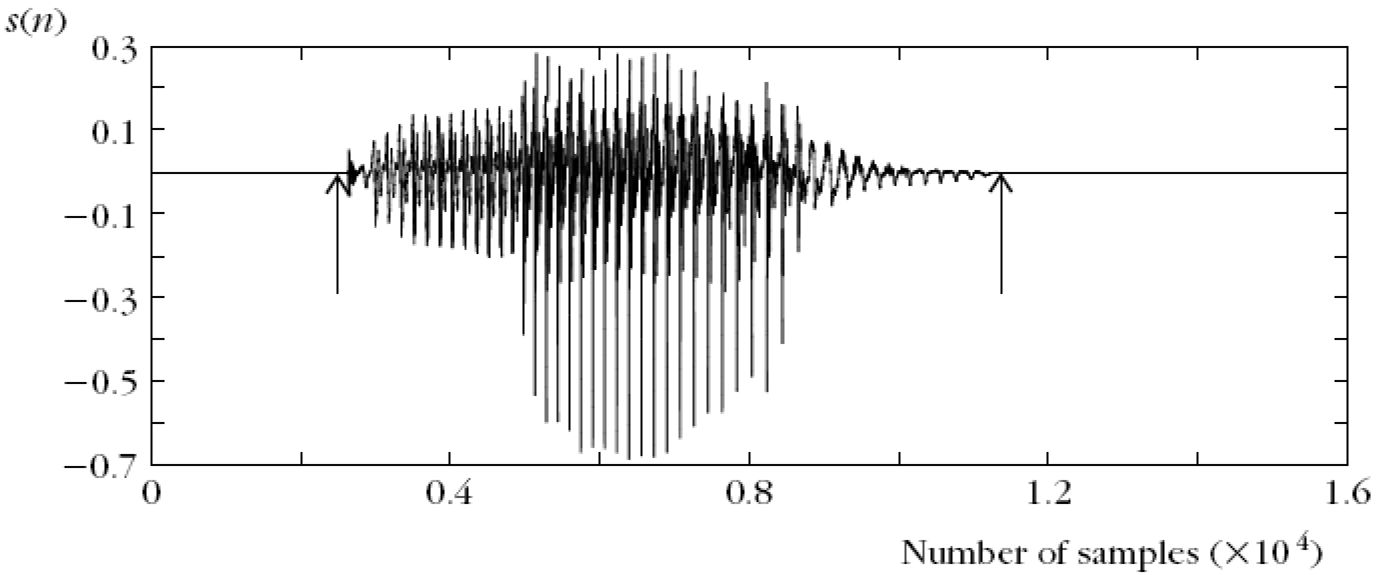
Application of TM in IWR

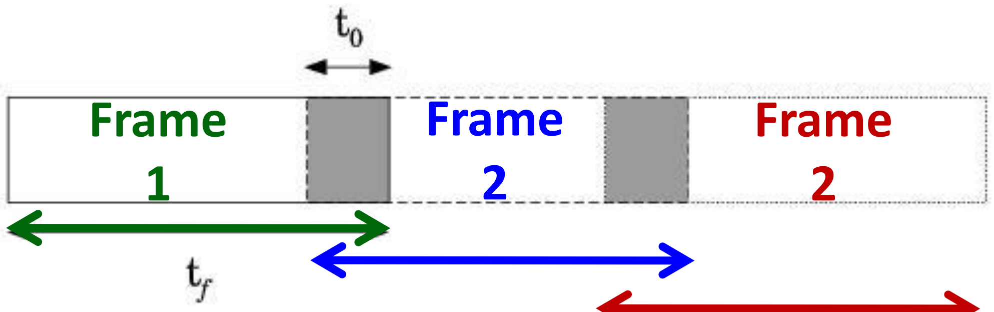
- The procedure:
  - Sample a speech segment from a microphone:



Application of TM in IWR

- The procedure:





Frame 1

Frame 2

Frame 2

$$t_f = 512$$

$$t_0 = 100$$

- each frame is represented by a vector of 512 samples

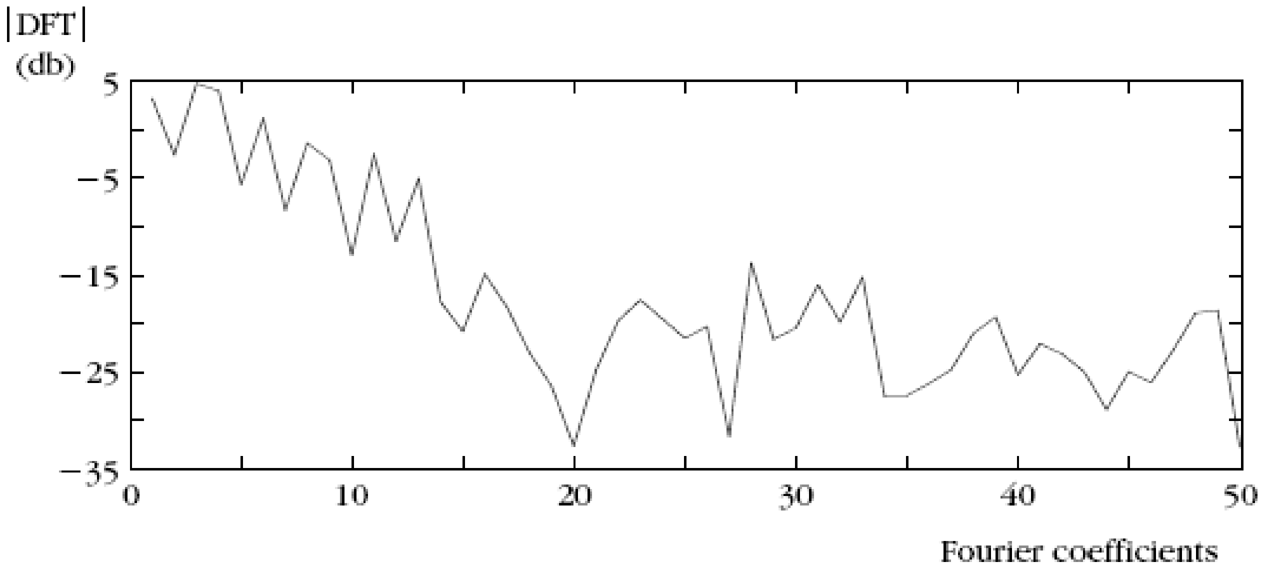
$$\underline{r}(i) = \begin{bmatrix} x_i(0) \\ x_i(1) \\ \ldots \\ \ldots \\ x_i(512) \end{bmatrix}, \; i = 1, \ldots, I \qquad \underline{t}(j) = \begin{bmatrix} x_j(0) \\ x_j(1) \\ \ldots \\ \ldots \\ x_j(512) \end{bmatrix}, \; j = 1, \ldots, J$$

- convert them to DFT

$$DFT(\underline{r}(i)) = DFT(\begin{bmatrix} x_i(0) \\ x_i(1) \\ \ldots \\ \ldots \\ x_i(512) \end{bmatrix}) = \begin{bmatrix} X_i(0) \\ X_i(1) \\ \ldots \\ \ldots \\ X_i(512) \end{bmatrix}$$

$$DFT(\underline{t}(j)) = DFT(\begin{bmatrix} x_i(0) \\ x_i(1) \\ \ldots \\ \ldots \\ x_i(512) \end{bmatrix}) = \begin{bmatrix} X_i(0) \\ X_i(1) \\ \ldots \\ \ldots \\ X_i(512) \end{bmatrix}$$

- convert them to DFT



- For each frame compute a feature vector. For example, the DFT coefficients and use, say, ℓ of those:

$$\underline{r}(i)=\begin{bmatrix} x_i(0) \\ x_i(1) \\ \ldots \\ \ldots \\ x_i(\ell-1) \end{bmatrix},\ i=1,\ \ldots,\ I \qquad \underline{t}(j)=\begin{bmatrix} x_j(0) \\ x_j(1) \\ \ldots \\ \ldots \\ x_j(\ell-1) \end{bmatrix},\ \ j=1,\ \ldots,\ J$$

- For each frame compute a feature vector. For example, the DFT coefficients and use, say, ℓ of those:

$$\underline{r}(i) = \begin{bmatrix} x_i(0) \\ x_i(1) \\ \dots \\ \dots \\ x_i(\ell - 1) \end{bmatrix}, \; i = 1, \dots, I \qquad \underline{t}(j) = \begin{bmatrix} x_j(0) \\ x_j(1) \\ \dots \\ \dots \\ x_j(\ell - 1) \end{bmatrix}, \; j = 1, \dots, J$$

- Choose a cost function associated with each node across a path, e.g., the Euclidean distance

$$\left\| \underline{r}(i_k) - \underline{t}(j_k) \right\| = d(i_k, j_k)$$

- For each frame compute a feature vector. For example, the DFT coefficients and use, say, ℓ of those:

$$\underline{r}(i)=\begin{bmatrix} x_i(0) \\ x_i(1) \\ \dots \\ \dots \\ x_i(\ell-1) \end{bmatrix}, \ i=1,\ \dots,\ I \qquad \underline{t}(j)=\begin{bmatrix} x_j(0) \\ x_j(1) \\ \dots \\ \dots \\ x_j(\ell-1) \end{bmatrix}, \ j=1,\ \dots,\ J$$

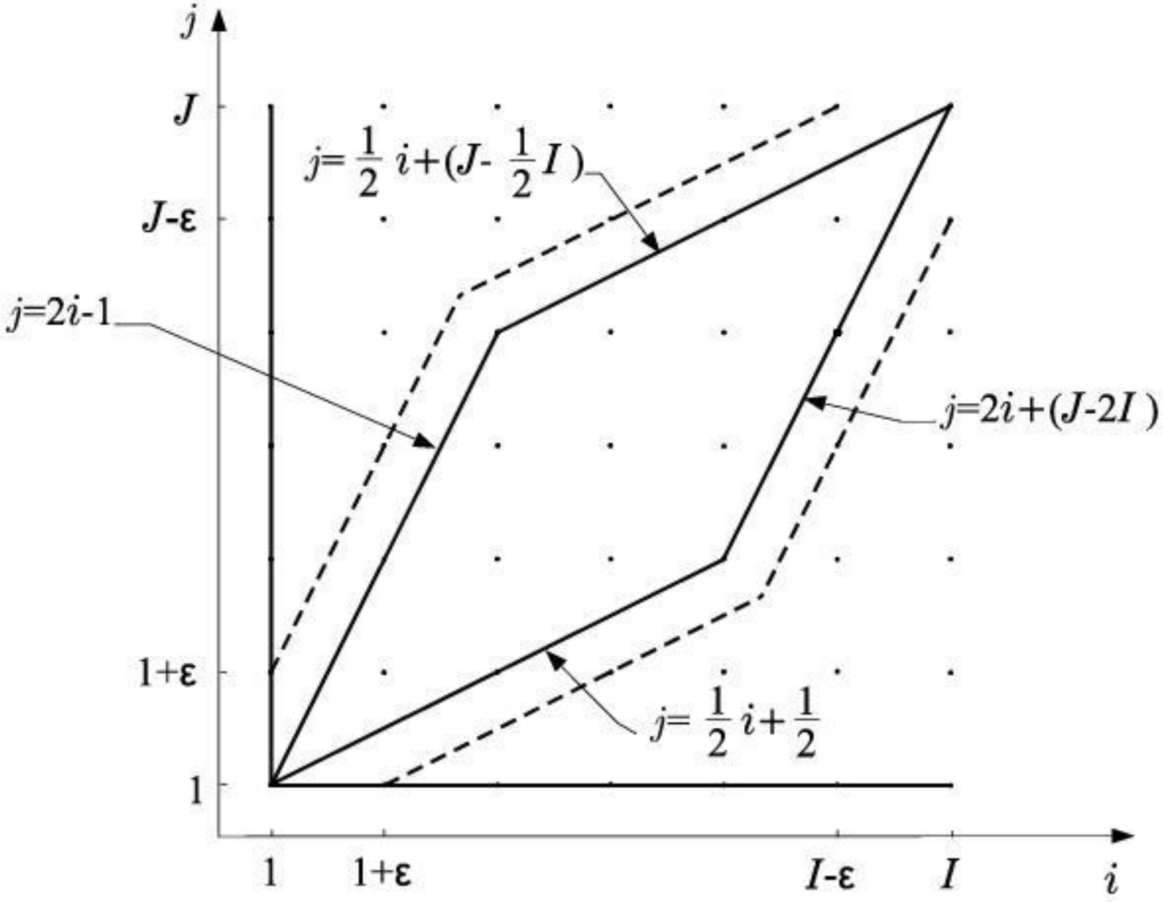
- Choose a cost function associated with each node across a path, e.g., the Euclidean distance

$$\left\| \underline{r}(i_k)-\underline{t}(j_k) \right\| = d(i_k,\ j_k)$$

- find the optimal path in the grid
- Match the test pattern to the reference pattern associated with the optimal path

- Prior to performing the math one has to choose:
    - End point constraints
    - global constraints
    - local constraints
    - distance

- Prior to performing the math one has to choose:
  - The global constraints: Defining the region of space within which the search for the optimal path will be performed.



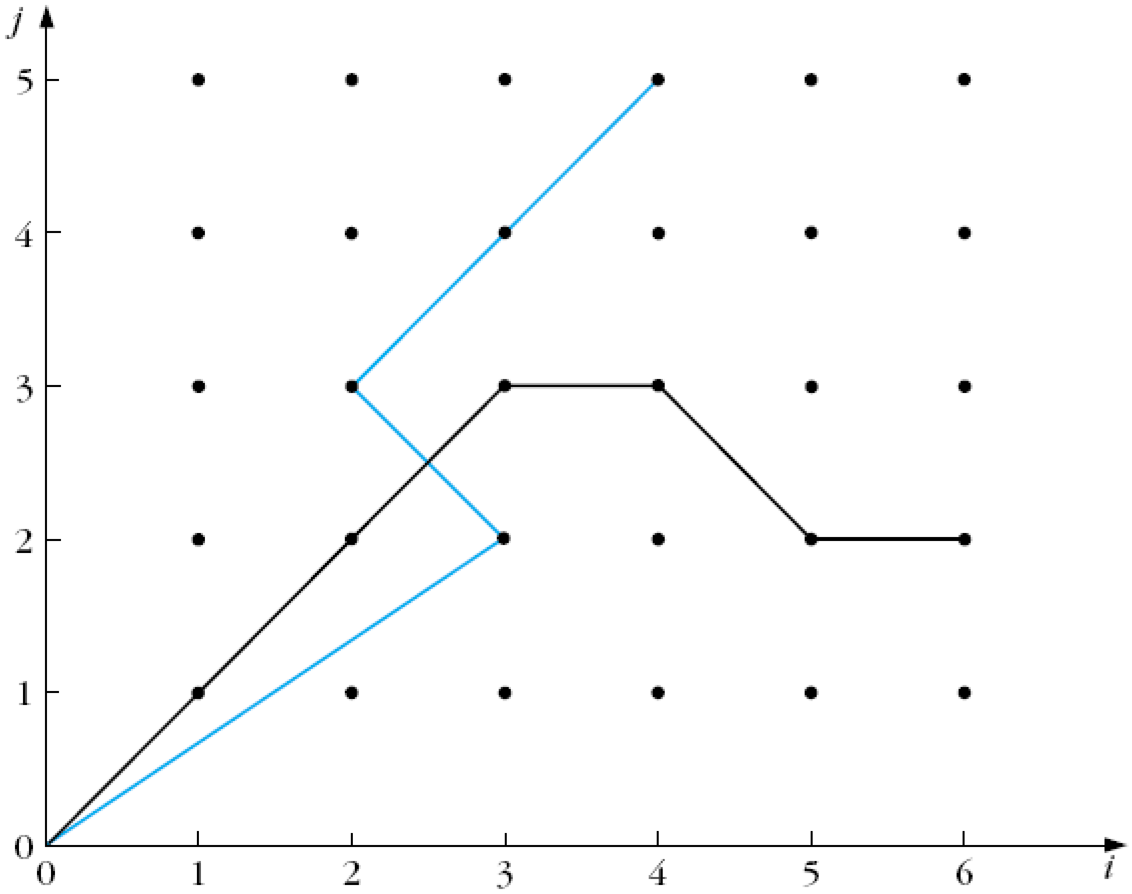
- The local constraints: monotonic path

$$i_{k-1} \leq i_k \quad \text{and} \quad j_{k-1} \leq j_k$$

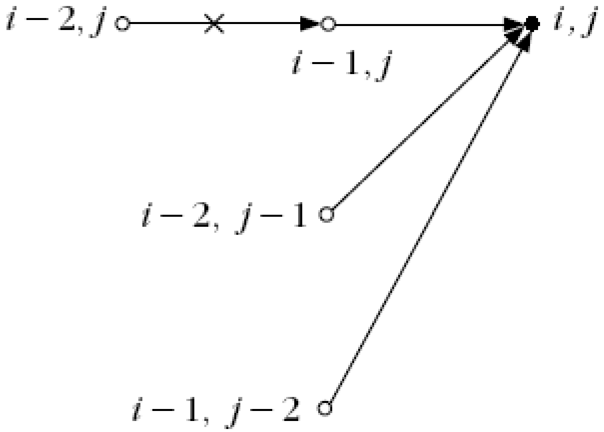
- The local constraints: monotonic path

$$i_{k-1} \le i_k \quad \text{and} \quad j_{k-1} \le j_k$$

- Non-monotonic path

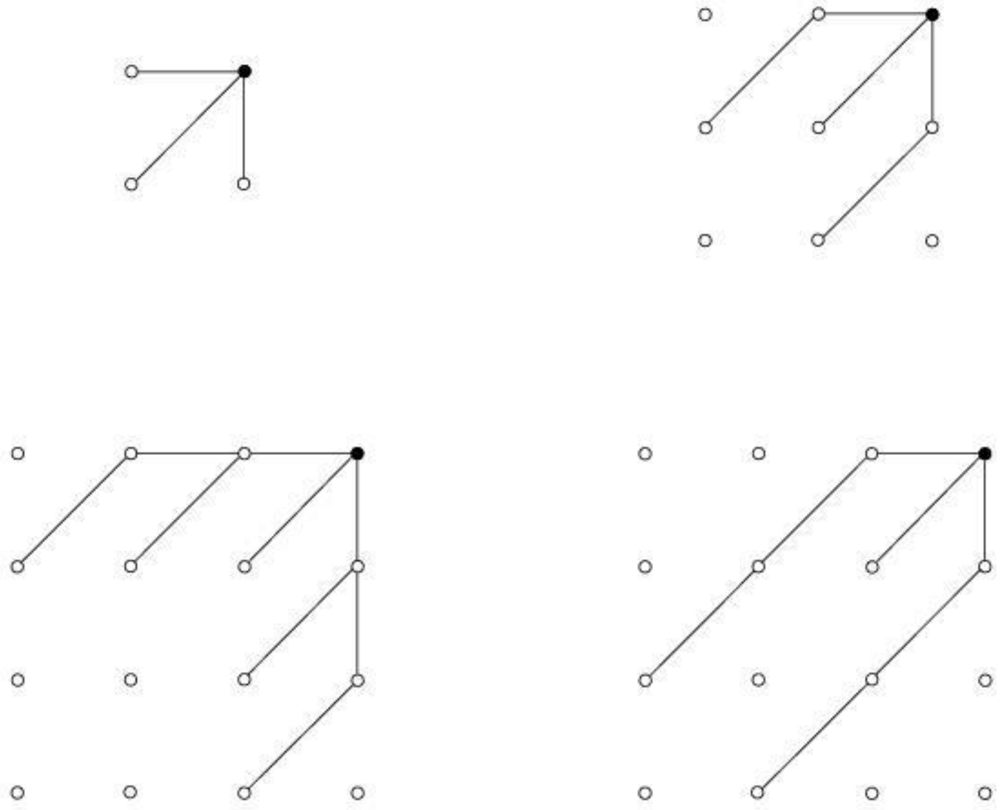


- The local constraints: Defining the type of transitions allowed between the nodes of the grid.



Itakura local constraints

- The local constraints: Defining the type of transitions allowed between the nodes of the grid.



Sakoe and Chiba local constraints

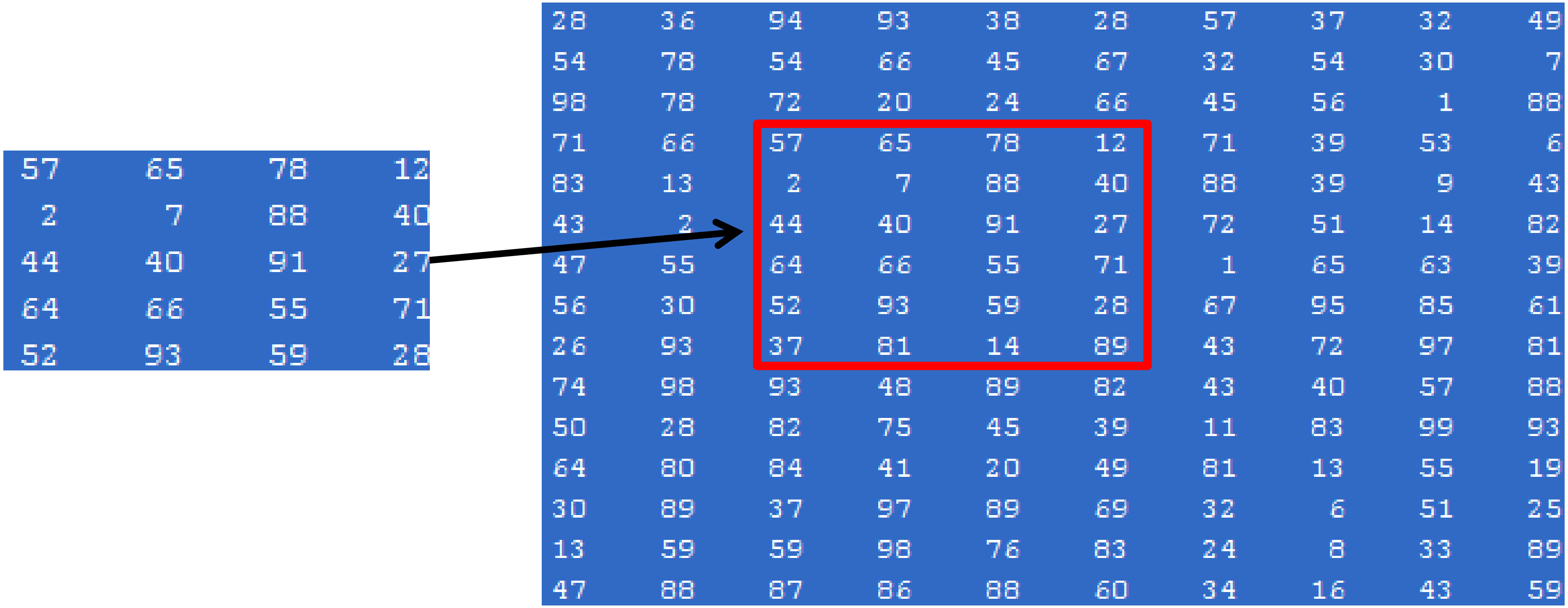
- cost function:
  - Euclidean distance
  - only node distance

$$d(i_k, j_k \mid i_{k-1}, j_{k-1}) = d(i_k, j_k)$$

$$= \left\| \underline{r}(i_k) - \underline{t}(j_k) \right\|$$

Correlation based TM

- Goal: to find whether a specific known reference pattern resides within a given block of data.



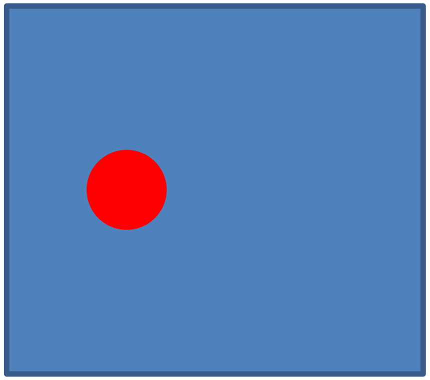
57	65	78	12
2	7	88	40
44	40	91	27
64	66	55	71
52	93	59	28

28	36	94	93	38	28	57	37	32	49
54	78	54	66	45	67	32	54	30	7
98	78	72	20	24	66	45	56	1	88
71	66	57	65	78	12	71	39	53	6
83	13	2	7	88	40	88	39	9	43
43	2	44	40	91	27	72	51	14	82
47	55	64	66	55	71	1	65	63	39
56	30	52	93	59	28	67	95	85	61
26	93	37	81	14	89	43	72	97	81
74	98	93	48	89	82	43	40	57	88
50	28	82	75	45	39	11	83	99	93
64	80	84	41	20	49	81	13	55	19
30	89	37	97	89	69	32	6	51	25
13	59	59	98	76	83	24	8	33	89
47	88	87	86	88	60	34	16	43	59

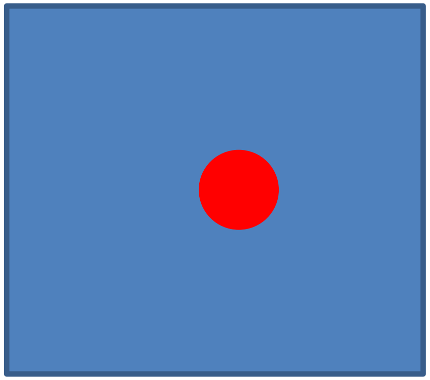
Correlation based TM

- Application: target detection, robot vision, video coding.

Frame at time *t*

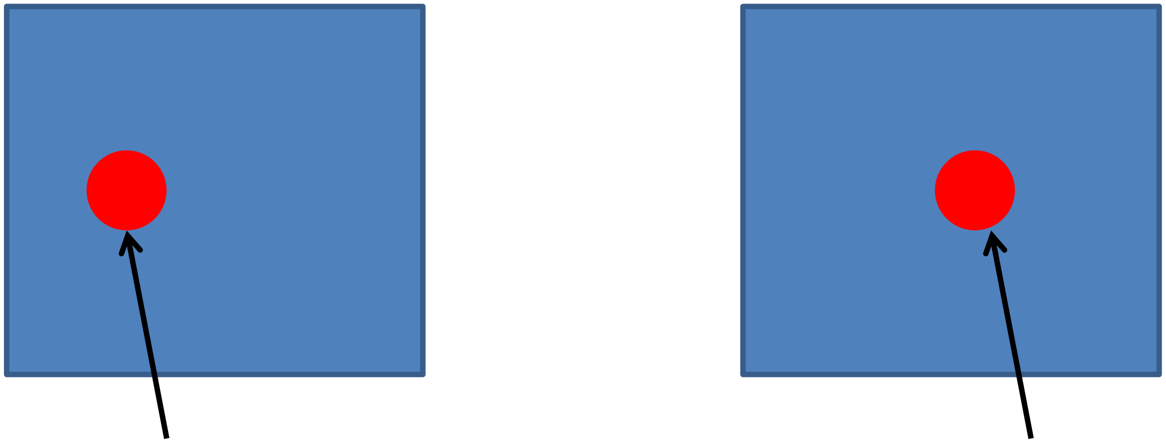


Frame at time *t*-1



Correlation based TM

- Application: target detection, robot vision, video coding.



Frame at time *t*

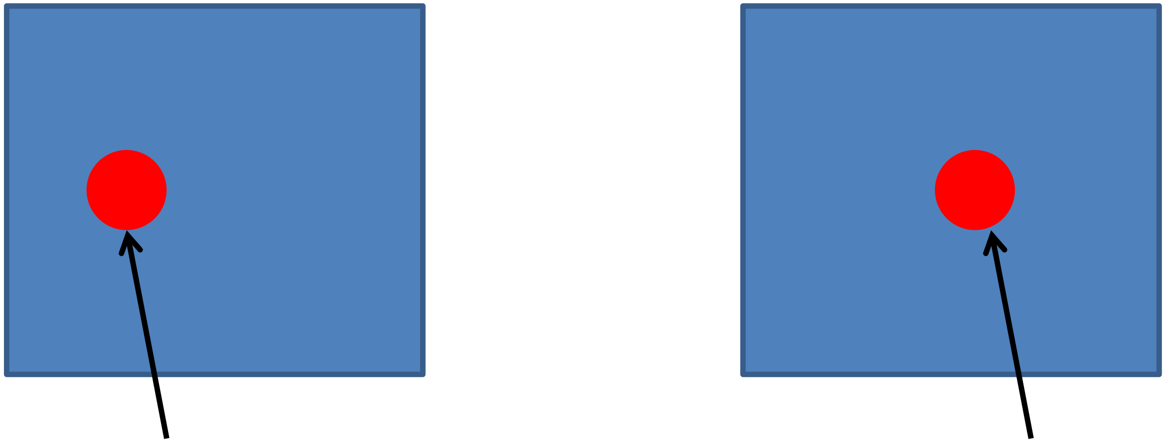
Frame at time *t*-1

Pixel value: $r(i, j, t)$

Pixel value $r(i-m, j-n, t-1)$

Correlation based TM

- Application: target detection, robot vision, video coding.



Frame at time *t*

Frame at time *t*-1

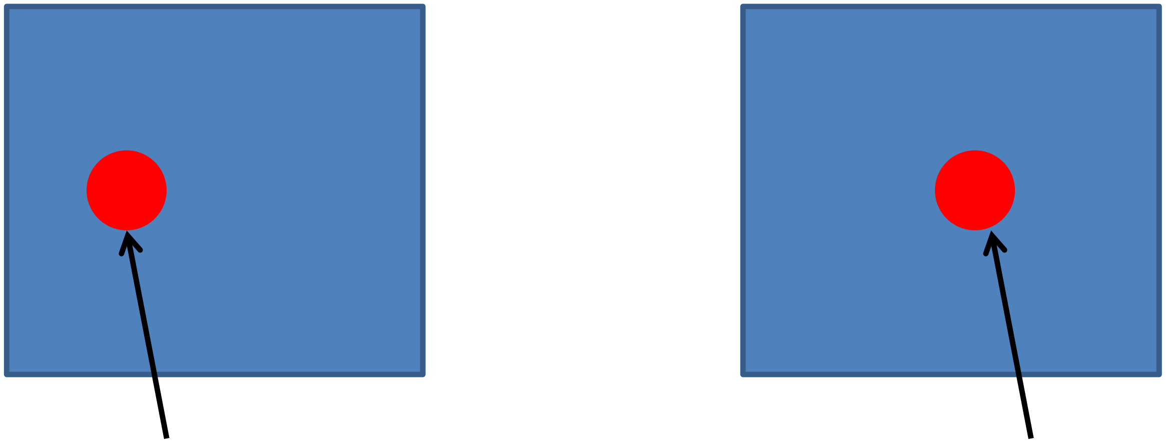
Pixel value: $r(i, j, t)$

Pixel value $r(i-m, j-n, t-1)$

Difference $e(i, j, t) = r(i, j, t) - r(i-m, j-n, t-1)$

Correlation based TM

- Application: target detection, robot vision, video coding.



Frame at time *t*

Frame at time *t*-1

Pixel value: $r(i, j, t)$

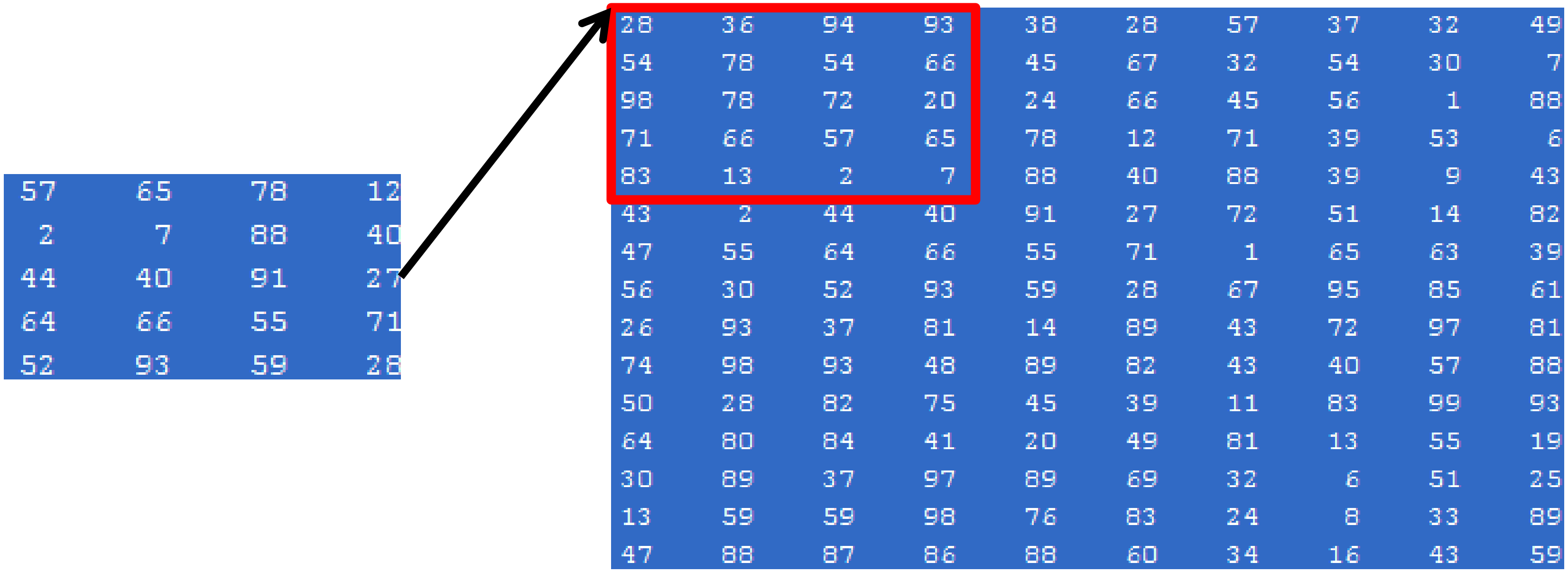
Pixel value $r(i-m, j-n, t-1)$

Difference $e(i, j, t) = r(i, j, t) - r(i-m, j-n, t-1)$

- We need to encode only the difference

Correlation based TM

- There are two basic steps in such a procedure:
  - Step 1: Move the reference pattern to all possible positions within the block of data. For each position, compute the "similarity" between the reference pattern and the respective part of the block of data.
  - Step 2: Compute the best matching value.

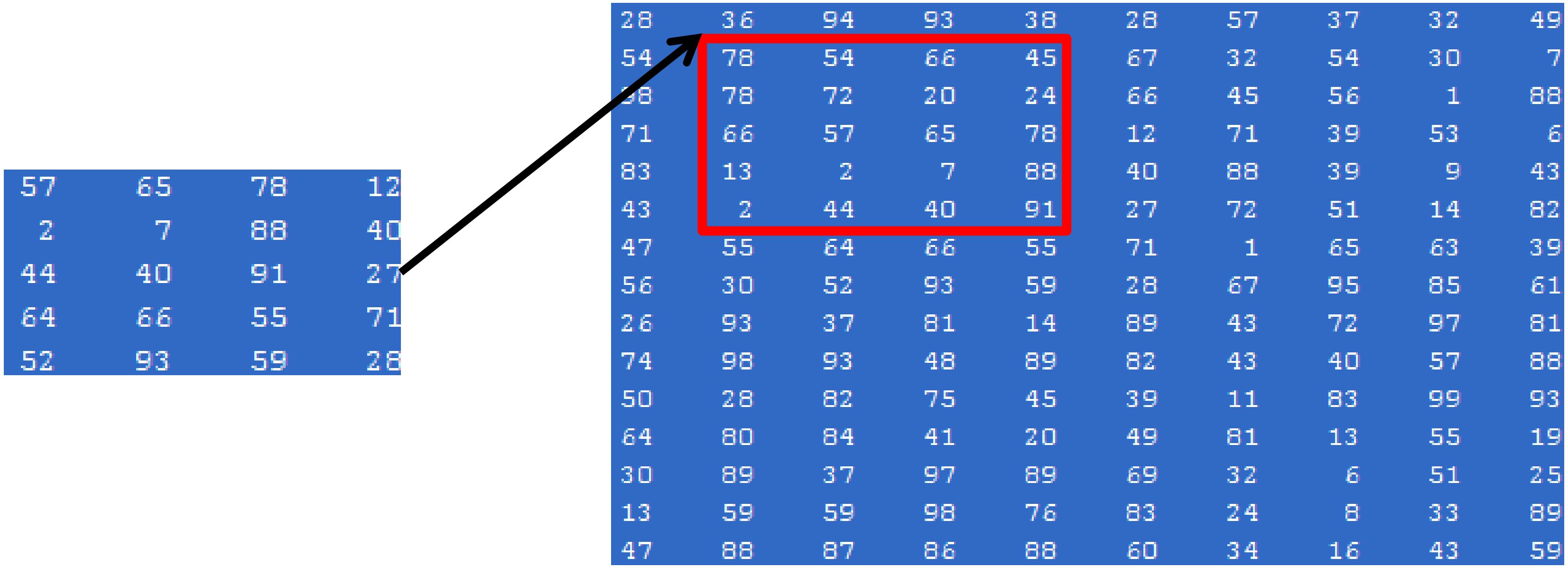


57	65	78	12
2	7	88	40
44	40	91	27
64	66	55	71
52	93	59	28

28	36	94	93	38	28	57	37	32	49
54	78	54	66	45	67	32	54	30	7
98	78	72	20	24	66	45	56	1	88
71	66	57	65	78	12	71	39	53	6
83	13	2	7	88	40	88	39	9	43
43	2	44	40	91	27	72	51	14	82
47	55	64	66	55	71	1	65	63	39
56	30	52	93	59	28	67	95	85	61
26	93	37	81	14	89	43	72	97	81
74	98	93	48	89	82	43	40	57	88
50	28	82	75	45	39	11	83	99	93
64	80	84	41	20	49	81	13	55	19
30	89	37	97	89	69	32	6	51	25
13	59	59	98	76	83	24	8	33	89
47	88	87	86	88	60	34	16	43	59

Correlation based TM

- There are two basic steps in such a procedure:
  - Step 1: Move the reference pattern to all possible positions within the block of data. For each position, compute the “similarity” between the reference pattern and the respective part of the block of data.
  - Step 2: Compute the best matching value.

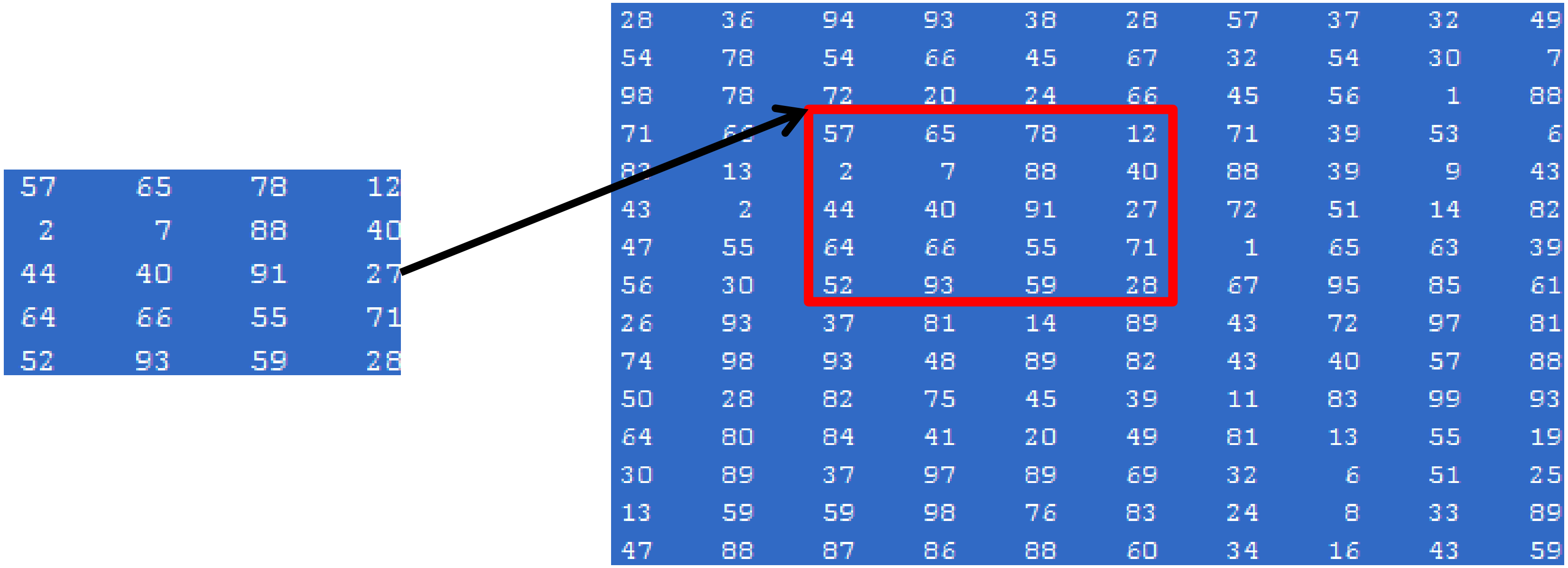


57	65	78	12
2	7	88	40
44	40	91	27
64	66	55	71
52	93	59	28

28	36	94	93	38	28	57	37	32	49
54	78	54	66	45	67	32	54	30	7
98	78	72	20	24	66	45	56	1	88
71	66	57	65	78	12	71	39	53	6
83	13	2	7	88	40	88	39	9	43
43	2	44	40	91	27	72	51	14	82
47	55	64	66	55	71	1	65	63	39
56	30	52	93	59	28	67	95	85	61
26	93	37	81	14	89	43	72	97	81
74	98	93	48	89	82	43	40	57	88
50	28	82	75	45	39	11	83	99	93
64	80	84	41	20	49	81	13	55	19
30	89	37	97	89	69	32	6	51	25
13	59	59	98	76	83	24	8	33	89
47	88	87	86	88	60	34	16	43	59

Correlation based TM

- There are two basic steps in such a procedure:
  - Step 1: Move the reference pattern to all possible positions within the block of data. For each position, compute the "similarity" between the reference pattern and the respective part of the block of data.
  - Step 2: Compute the best matching value.

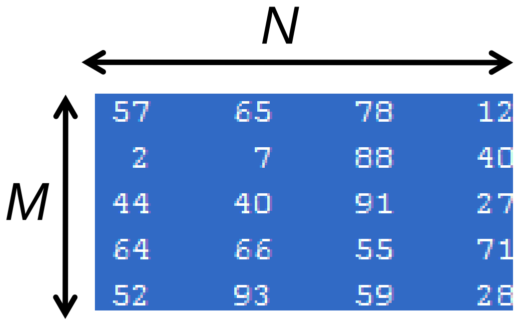


57	65	78	12
2	7	88	40
44	40	91	27
64	66	55	71
52	93	59	28

28	36	94	93	38	28	57	37	32	49
54	78	54	66	45	67	32	54	30	7
98	78	72	20	24	66	45	56	1	88
71	66	57	65	78	12	71	39	53	6
82	13	2	7	88	40	88	39	9	43
43	2	44	40	91	27	72	51	14	82
47	55	64	66	55	71	1	65	63	39
56	30	52	93	59	28	67	95	85	61
26	93	37	81	14	89	43	72	97	81
74	98	93	48	89	82	43	40	57	88
50	28	82	75	45	39	11	83	99	93
64	80	84	41	20	49	81	13	55	19
30	89	37	97	89	69	32	6	51	25
13	59	59	98	76	83	24	8	33	89
47	88	87	86	88	60	34	16	43	59

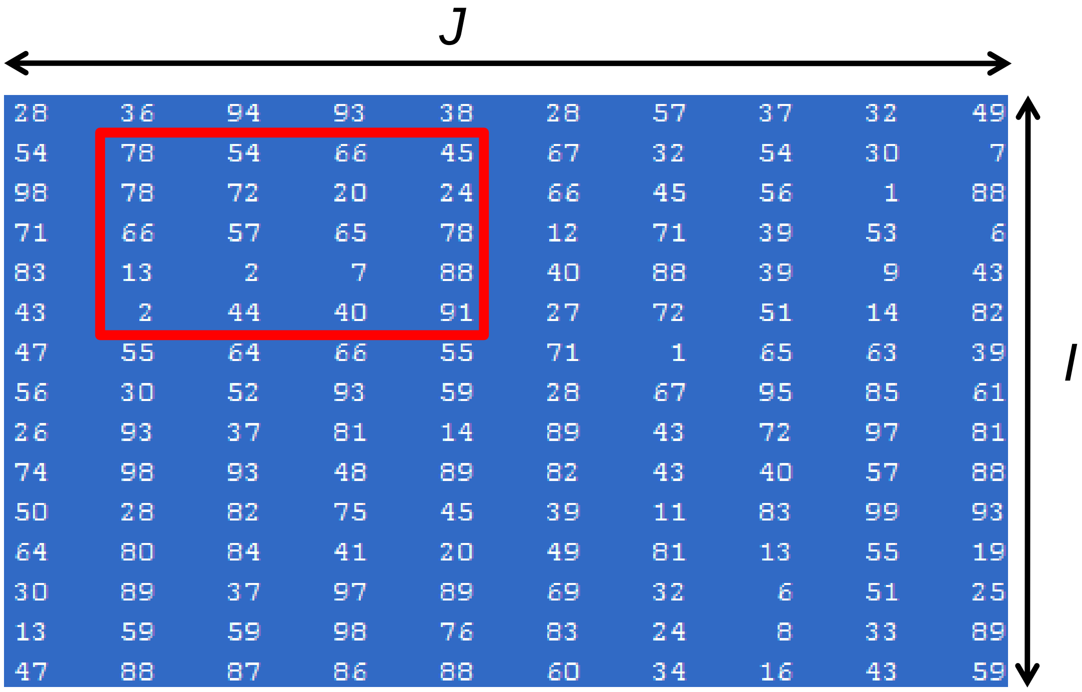
Correlation In Image Matching

- – Application to images: Given a reference image, $r(i,j)$ of MxN size, and an IxJ image array $t(i,j)$. Move $r(i,j)$ to all possible positions (m,n) within $t(i,j)$.



57	65	78	12
2	7	88	40
44	40	91	27
64	66	55	71
52	93	59	28

Reference, *r*



28	36	94	93	38	28	57	37	32	49
54	78	54	66	45	67	32	54	30	7
98	78	72	20	24	66	45	56	1	88
71	66	57	65	78	12	71	39	53	6
83	13	2	7	88	40	88	39	9	43
43	2	44	40	91	27	72	51	14	82
47	55	64	66	55	71	1	65	63	39
56	30	52	93	59	28	67	95	85	61
26	93	37	81	14	89	43	72	97	81
74	98	93	48	89	82	43	40	57	88
50	28	82	75	45	39	11	83	99	93
64	80	84	41	20	49	81	13	55	19
30	89	37	97	89	69	32	6	51	25
13	59	59	98	76	83	24	8	33	89
47	88	87	86	88	60	34	16	43	59

Test, *t*

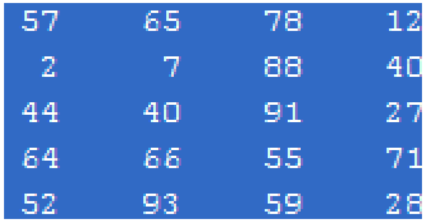
Correlation In Image Matching

- Compute the distance:

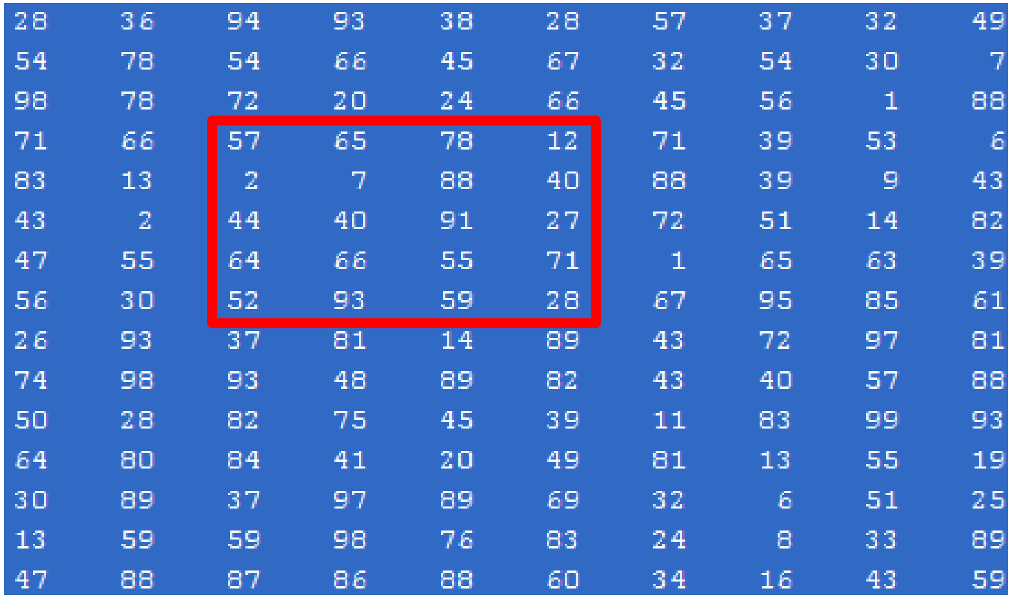
$$D(m,n) = \sum_{i=m}^{m+M-1} \sum_{j=n}^{n+N-1} |t(i,j) - r(i-m, j-n)|^2$$

for every (m,n).

- For all (m,n), compute the minimum.



57	65	78	12
2	7	88	40
44	40	91	27
64	66	55	71
52	93	59	28



28	36	94	93	38	28	57	37	32	49
54	78	54	66	45	67	32	54	30	7
98	78	72	20	24	66	45	56	1	88
71	66	57	65	78	12	71	39	53	6
83	13	2	7	88	40	88	39	9	43
43	2	44	40	91	27	72	51	14	82
47	55	64	66	55	71	1	65	63	39
56	30	52	93	59	28	67	95	85	61
26	93	37	81	14	89	43	72	97	81
74	98	93	48	89	82	43	40	57	88
50	28	82	75	45	39	11	83	99	93
64	80	84	41	20	49	81	13	55	19
30	89	37	97	89	69	32	6	51	25
13	59	59	98	76	83	24	8	33	89
47	88	87	86	88	60	34	16	43	59

Correlation In Image Matching

- The equation

$$D(m,n) = \sum_{i=m}^{m+M-1} \sum_{j=n}^{n+N-1} |t(i,j) - r(i-m, j-n)|^2$$

can be written as

$$D(m,n) = \sum_{i} \sum_{j} |t(i,j)|^2 + \sum_{i} \sum_{j} |r(i,j)|^2$$

$$- 2 \sum_{i} \sum_{j} t(i,j) r(i-m, j-n)$$

Correlation In Image Matching

- In the equation

$$D(m,n) = \sum_i \sum_j |t(i,j)|^2 + \sum_i \sum_j |r(i,j)|^2$$

$$- 2 \sum_i \sum_j t(i,j) r(i-m, j-n)$$

shaded terms are constant

provided pixel levels do not change much across the test image

Correlation In Image Matching

$$D(m,n) = \boxed{\sum_i \sum_j |t(i,j)|^2} + \boxed{\sum_i \sum_j |r(i,j)|^2}$$

$$- 2 \sum_i \sum_j t(i, j) r(i - m, j - n)$$

- Canceling out the shaded terms, find point (m, n) that maximize:

$$c(m,n) = \sum_i \sum_j t(i,j) r(i - m, j - n)$$

Correlation In Image Matching

$$c(m,n) = \sum_{i}\sum_{j} t(i,j)r(i-m,j-n)$$

- c (m, n) is no longer a difference term
- This is called cross correlation

Correlation In Image Matching

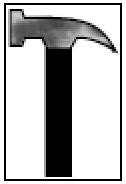
$$c(m,n) = \sum_i \sum_j t(i,j) r(i-m, j-n)$$

- In case gray level variation is valid, normalize:

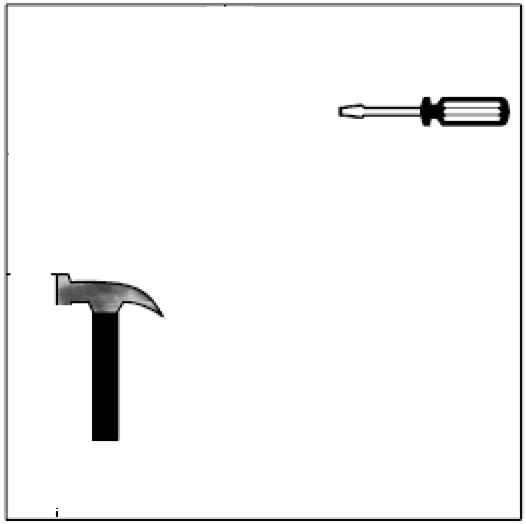
$$c_N(m,n) = \frac{c(m,n)}{\sqrt{\sum_i \sum_j |t(i,j)|^2 \sum_i \sum_j |r(i,j)|^2}}$$

Correlation In Image Matching

- Example:



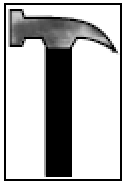
Reference, *r*



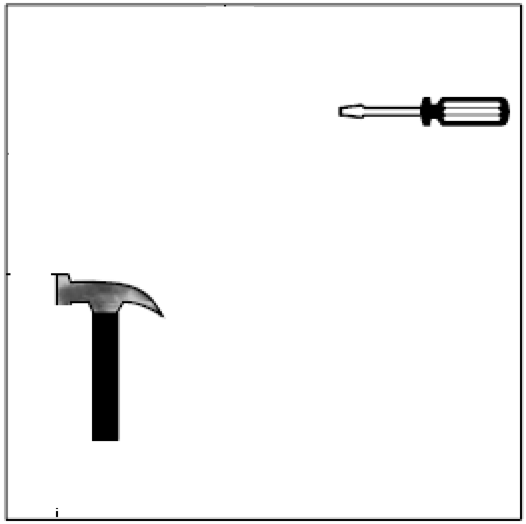
Test Image, *t*

Correlation In Image Matching

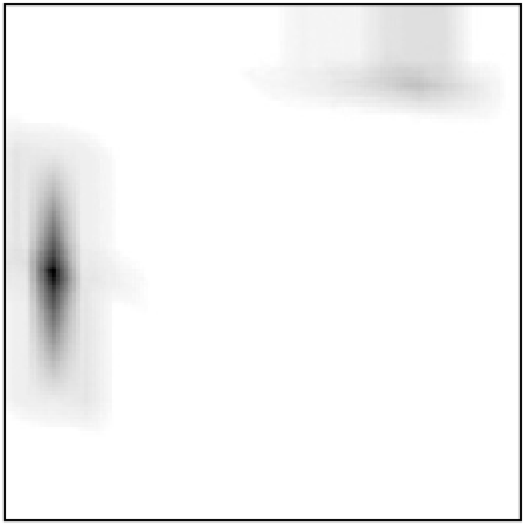
- Example:



Reference, *r*



Test Image, *t*



Correlation Image

Computation Considerations in Correlation Based TM (1)

- Find *c* (*m*,*n*) at every pixel

$$c(m,n) = \sum_{i} \sum_{j} t(i,j) r(i-m, j-n)$$

- This equation looks like convolution operation
- Alternate is to calculate in the frequency domain

Computation Considerations in Correlation Based TM (1)

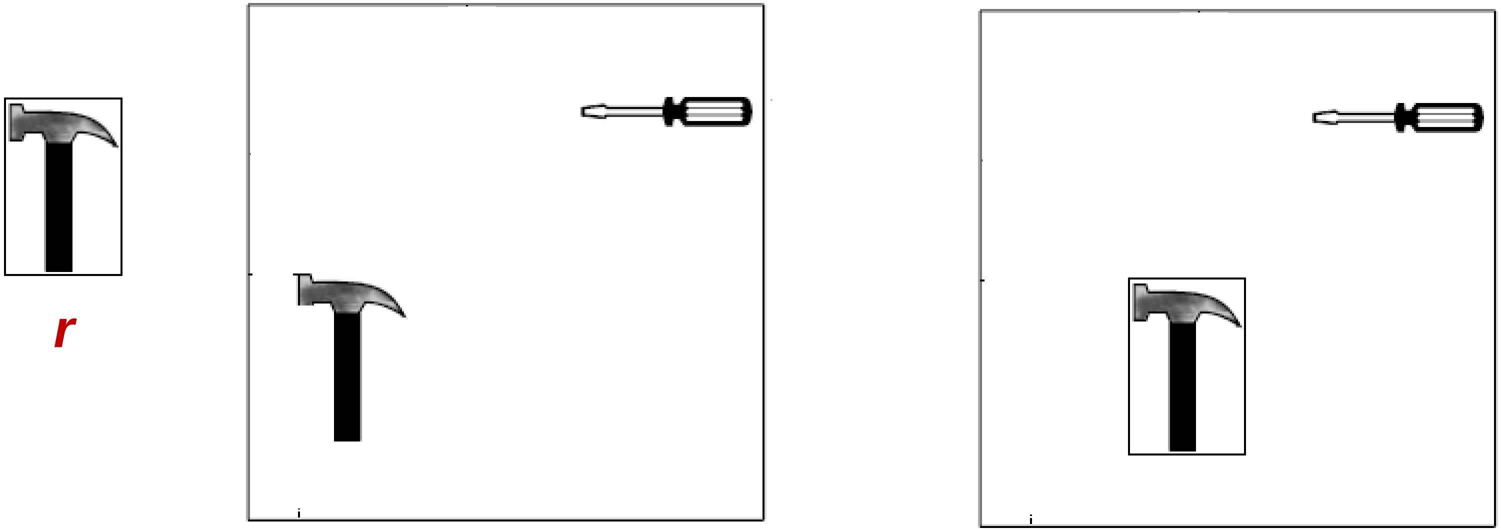
- The frequency domain representation of

$$c(m,n) = \sum_i \sum_j t(i,j) r(i-m, j-n)$$

is $c = IDFT(DFT(t)DFT(r))$

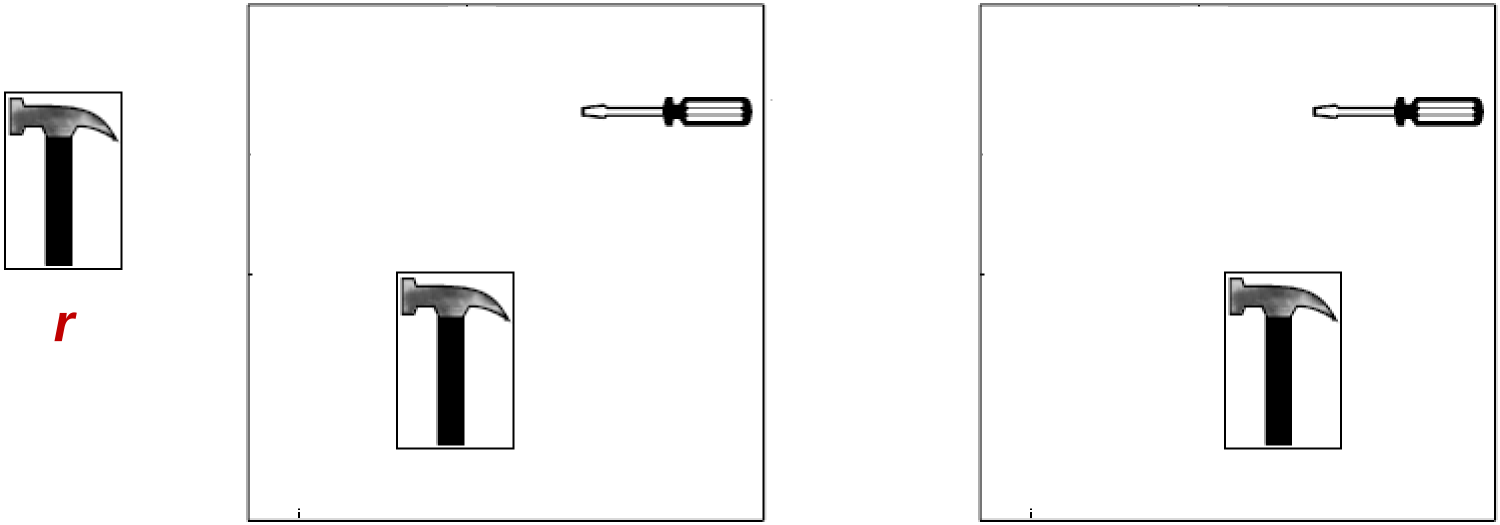
Computation Considerations in Correlation Based TM (2)

- Limit the search space
  - Search only in the area of [-p, p] X [-p, p] centered at (x, y)



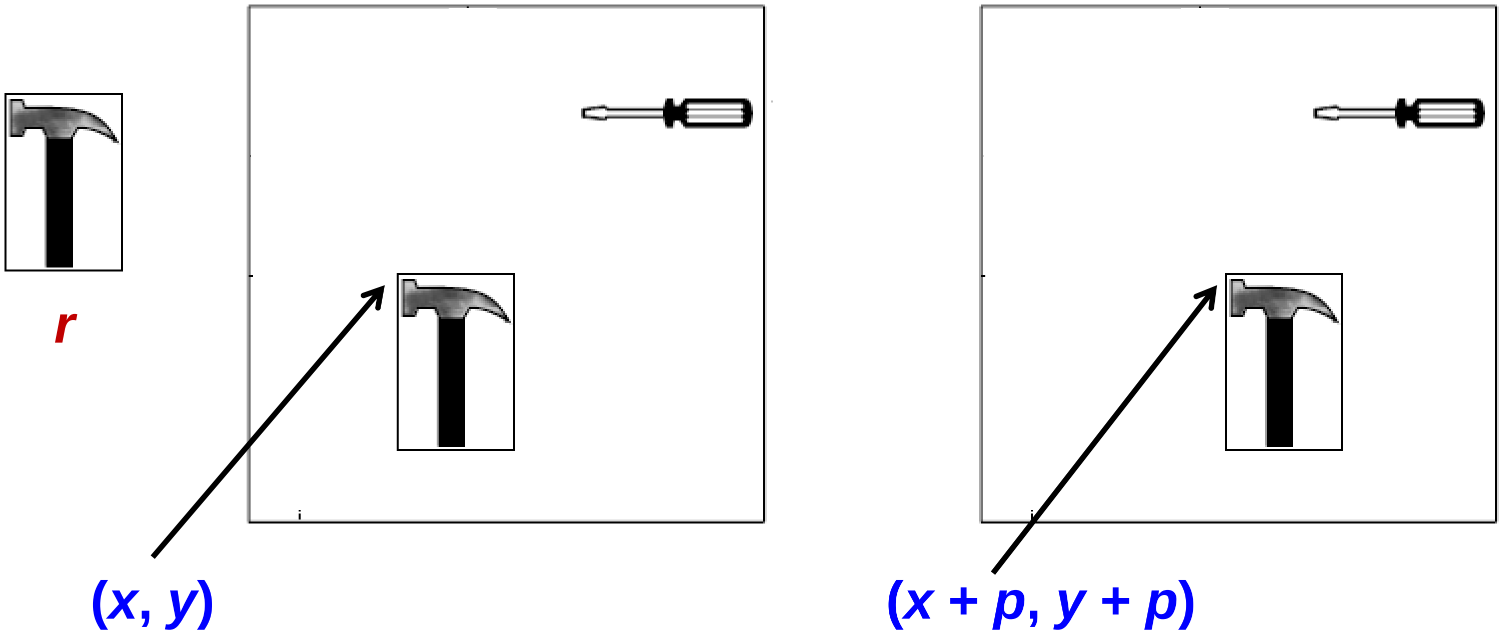
Computation Considerations in Correlation Based TM (2)

- Limit the search space
  - Search only in the area of [-p, p] X [-p, p] centered at (x, y)



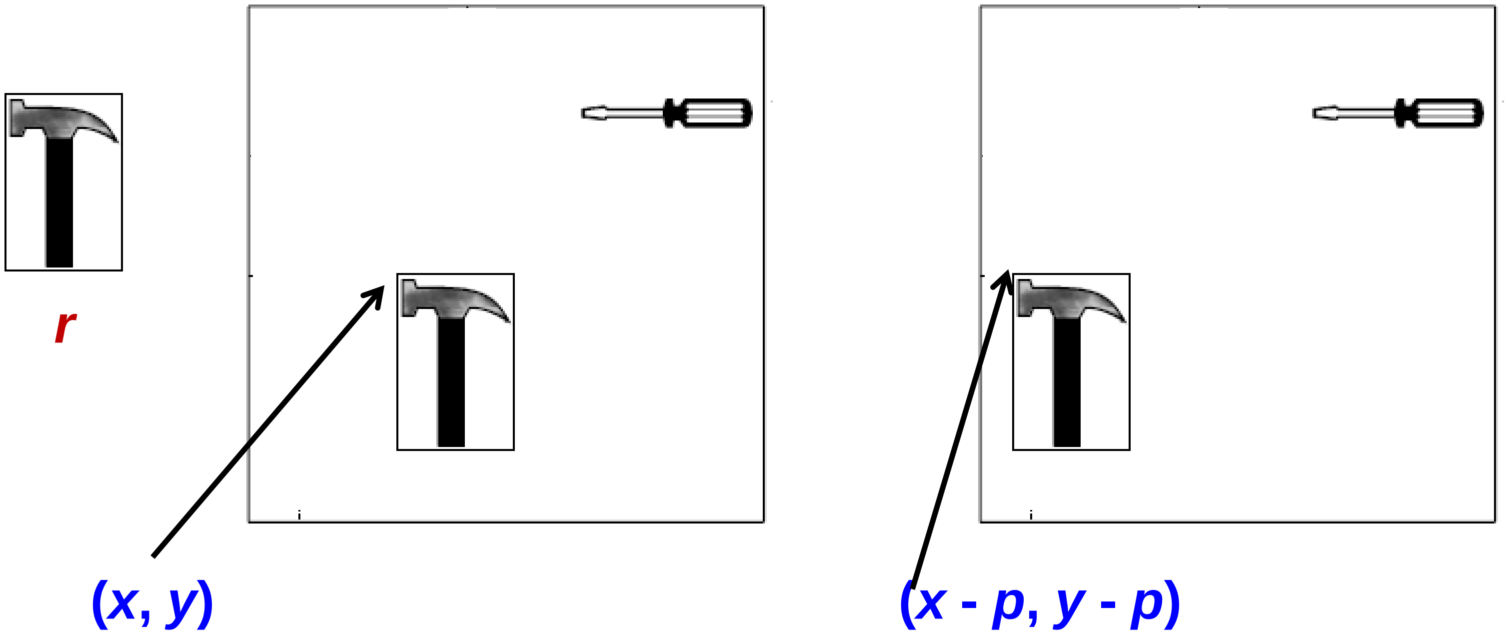
Computation Considerations in Correlation Based TM (2)

- Limit the search space
  - Search only in the area of [-p, p] X [-p, p] centered at (x, y)



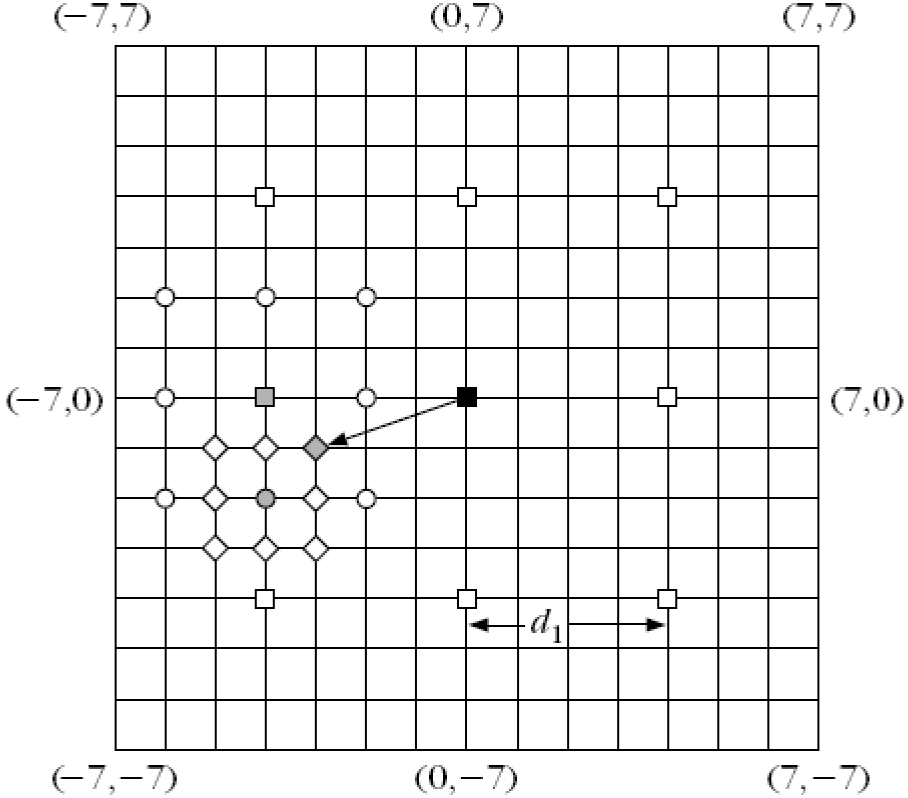
Computation Considerations in Correlation Based TM (2)

- Limit the search space
  - Search only in the area of [-*p*, *p*] X [-*p*, *p*] centered at (*x*, *y*)



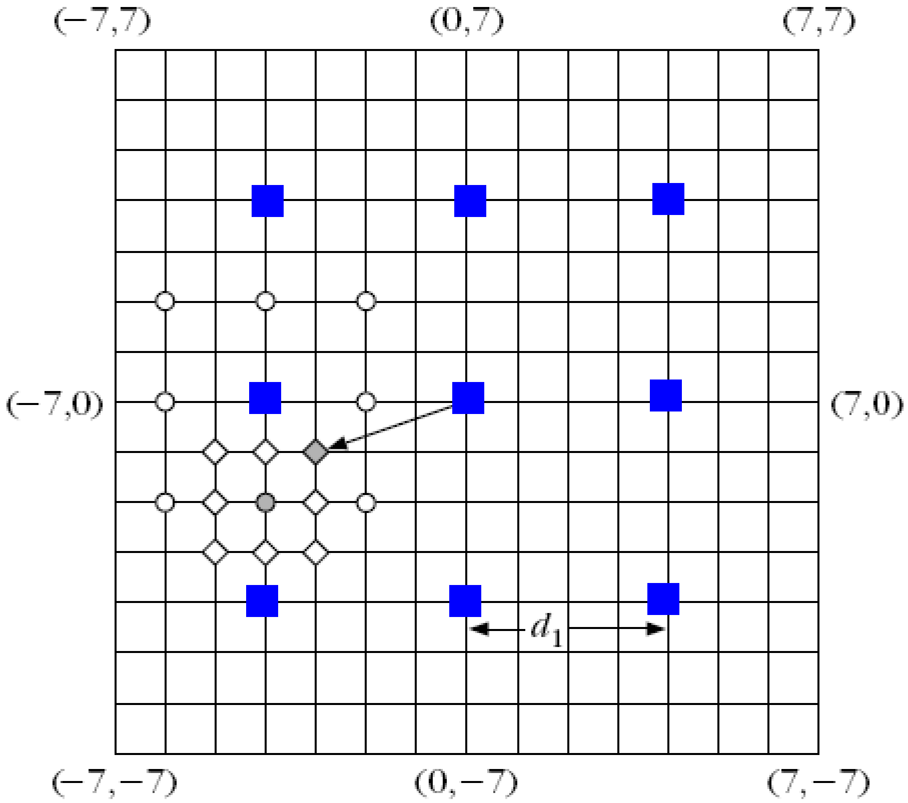
Computation Considerations in Correlation Based TM (3)

- 2D Logarithmic search
  - Start with a rectangle of size [-*p*, *p*] X [-*p*, *p*]



Computation Considerations in Correlation Based TM (3)

- 2D Logarithmic search
  - Search only at 9 points separated by d_1

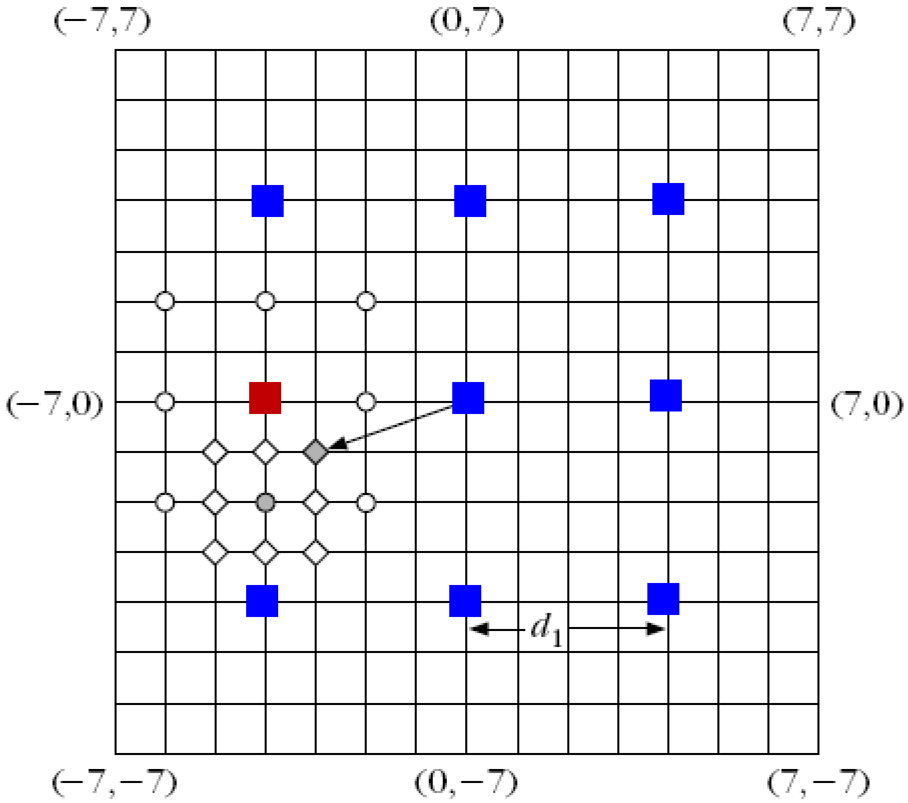


$$d_1 = 2^{k-1}$$

$$k = \lceil \log_2 p \rceil$$

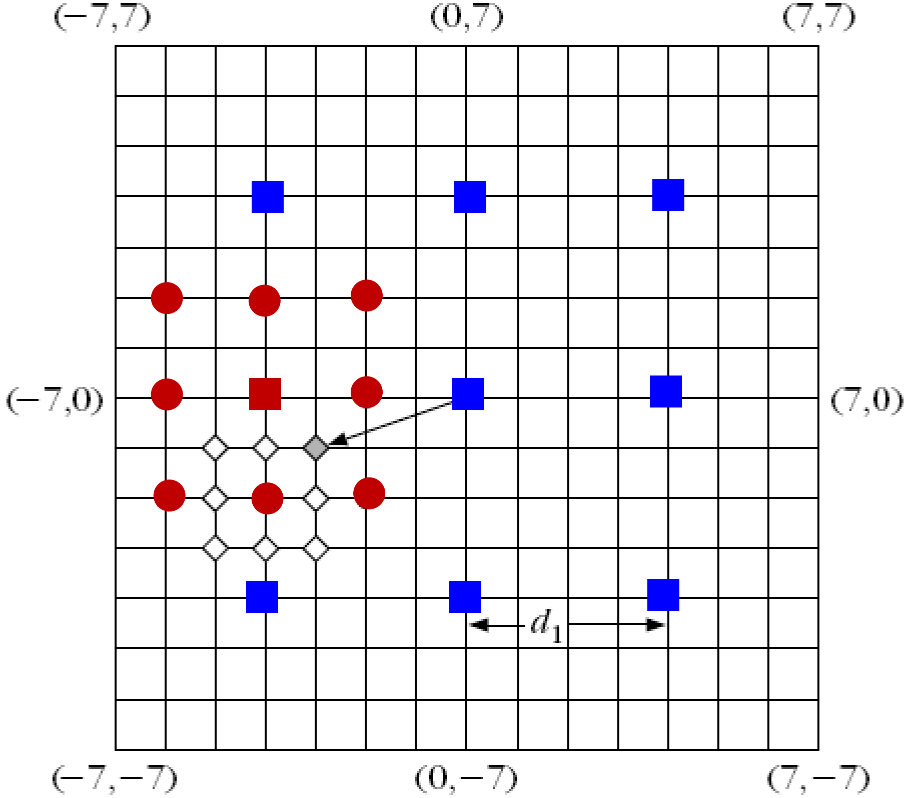
Computation Considerations in Correlation Based TM (3)

- 2D Logarithmic search
  - Maximum found at dark red



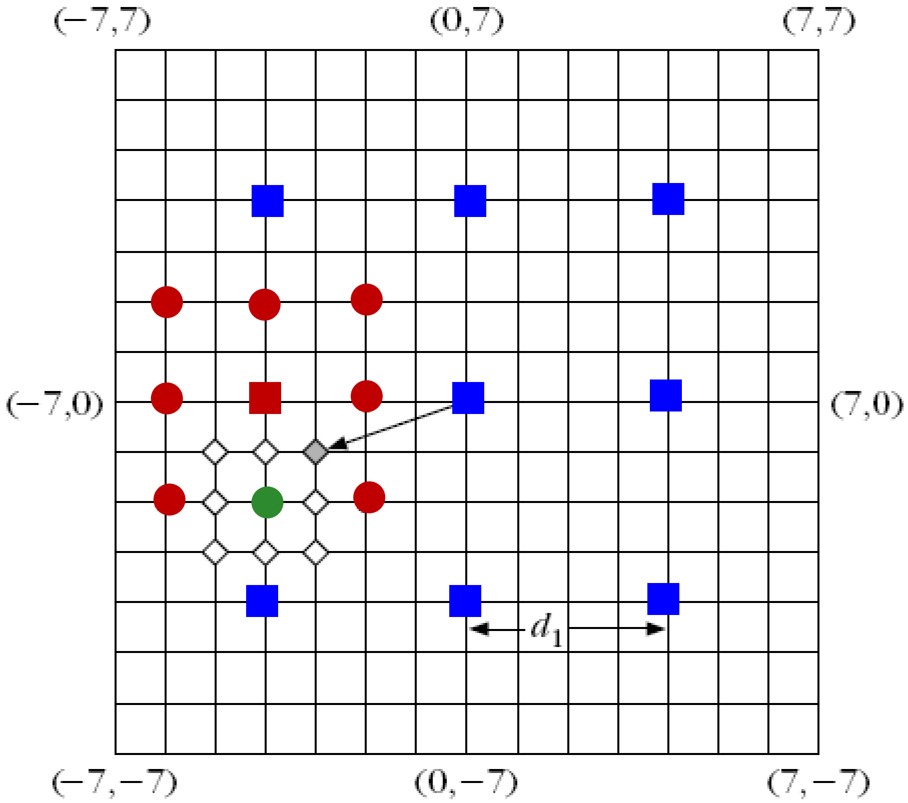
Computation Considerations in Correlation Based TM (3)

- 2D Logarithmic search
  - Search in the rectangle of size [-*p/4*, *p/4*] X [-*p/4*, *p/4*] centered at dark red



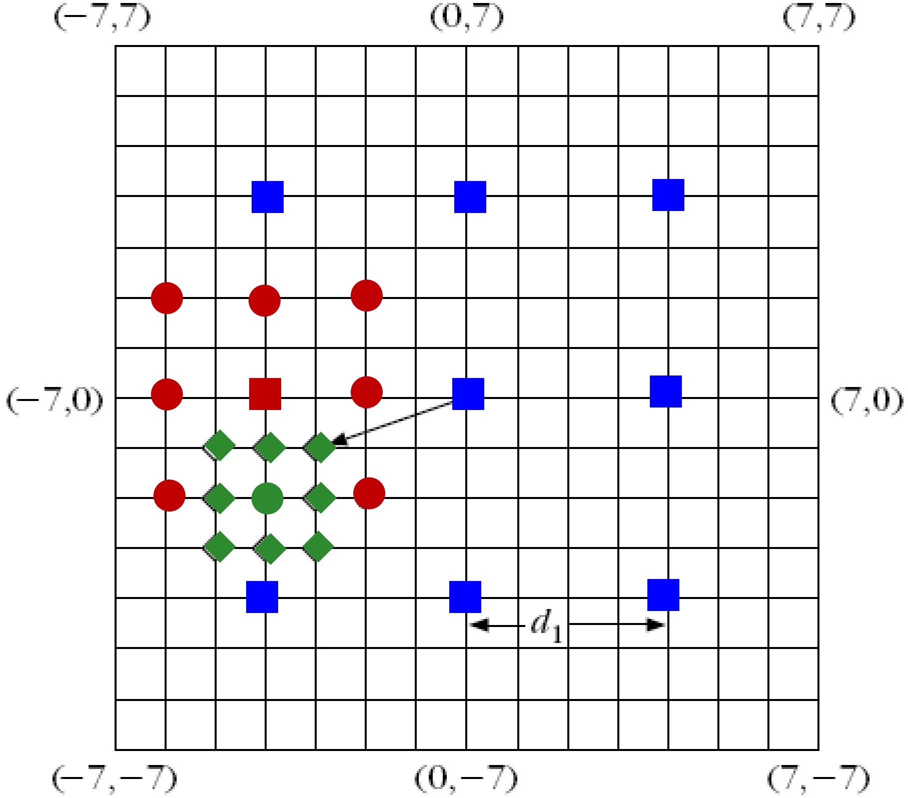
Computation Considerations in Correlation Based TM (3)

- 2D Logarithmic search
  - Maximum found at green



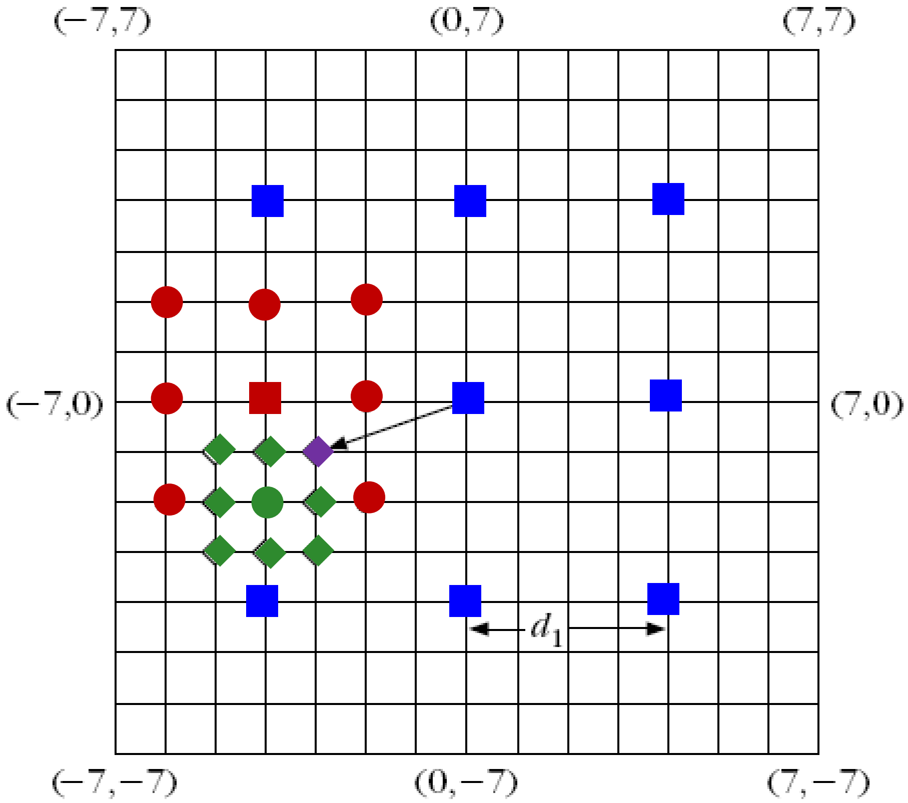
Computation Considerations in Correlation Based TM (3)

- 2D Logarithmic search
  - Search in the rectangle of size [-*p/8*, *p/8*] X [-*p/8*, *p/8*] centered at green



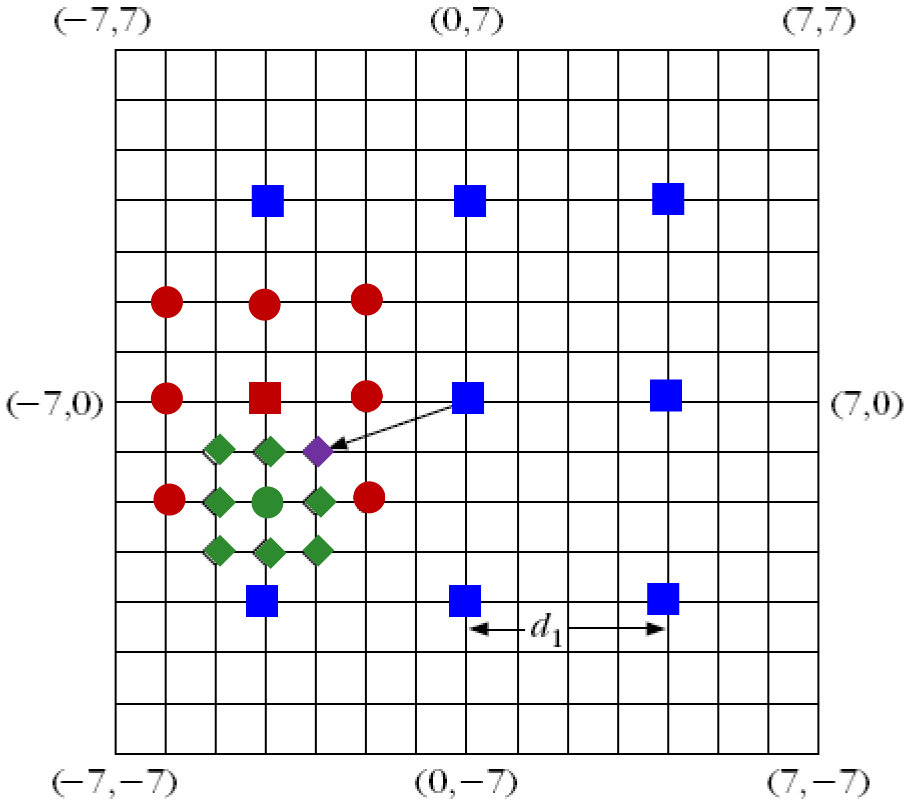
Computation Considerations in Correlation Based TM (3)

- 2D Logarithmic search
  - Maximum found at purple



Computation Considerations in Correlation Based TM (3)

- Complexity $MN(8k+1)$



$$k = \lceil \log_2 p \rceil$$

Computation Considerations in Correlation Based TM (4)

- Hierarchical Search
  - Search the reference in the area of size [-*p*, *p*] X [-*p*, *p*] centered at (*x*, *y*)
  - Let, reference be of size 16X16

reference

test

Computation Considerations in Correlation Based TM (4)

- Hierarchical Search

Level 0

Original reference and test image

Low pass Filter of Level 0

Computation Considerations in Correlation Based TM (4)

- Hierarchical Search

Level 0

Low pass Filter of Level 0

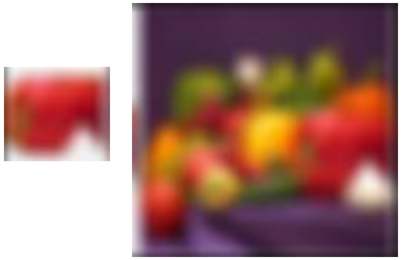
Sub-sampled by 2

Level 1

Computation Considerations in Correlation Based TM (4)

- Hierarchical Search

Level 1



Low pass Filter of Level 1

Sub-sampled by 2

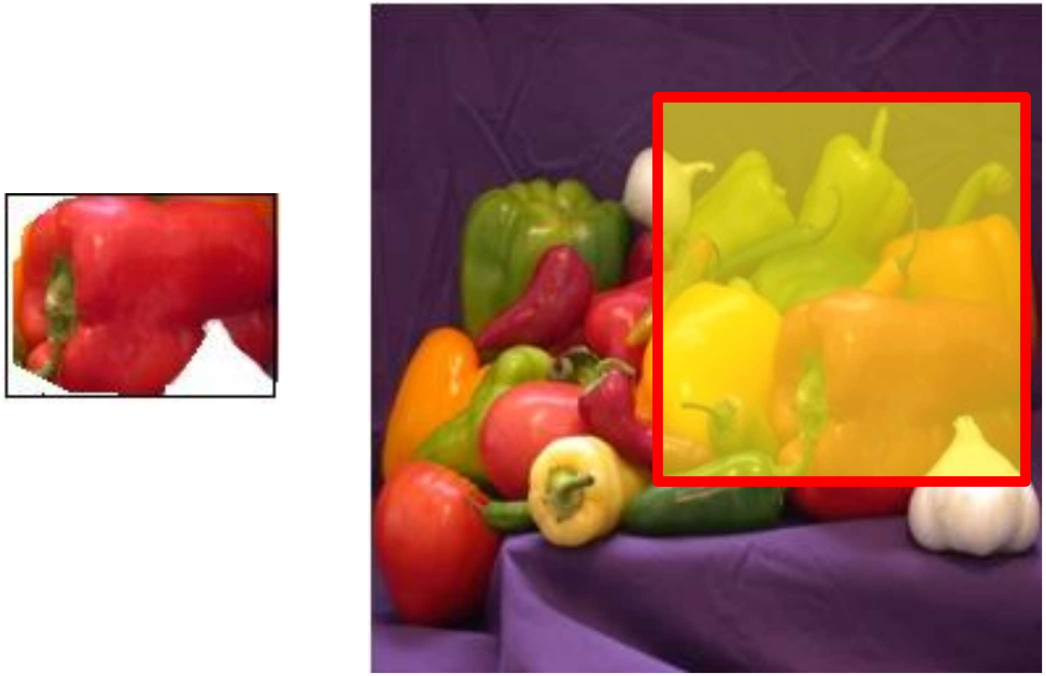
Level 2

Computation Considerations in Correlation Based TM (4)

- Hierarchical Search

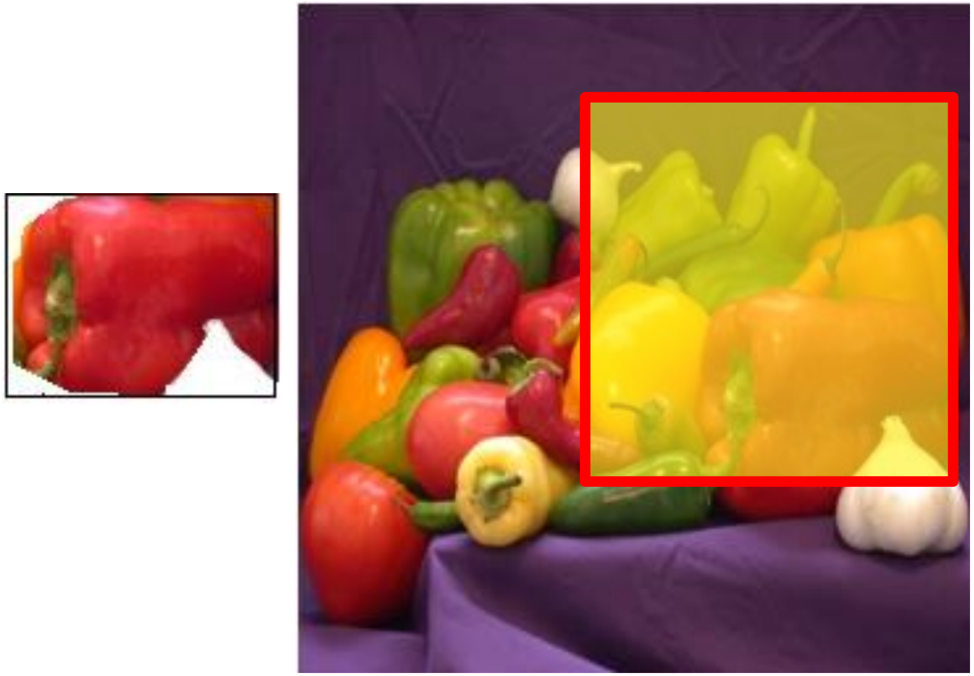
Computation Considerations in Correlation Based TM (4)

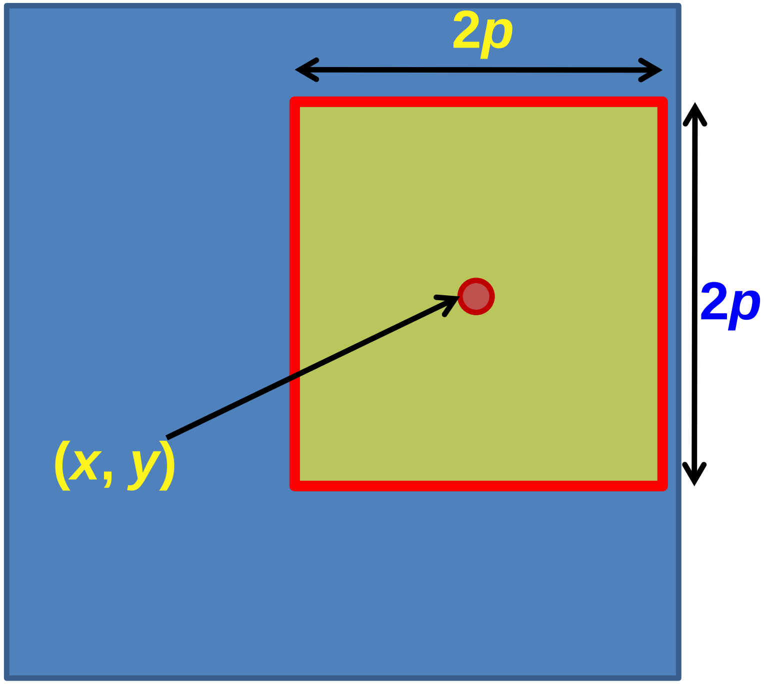
- Hierarchical Search



Computation Considerations in Correlation Based TM (4)

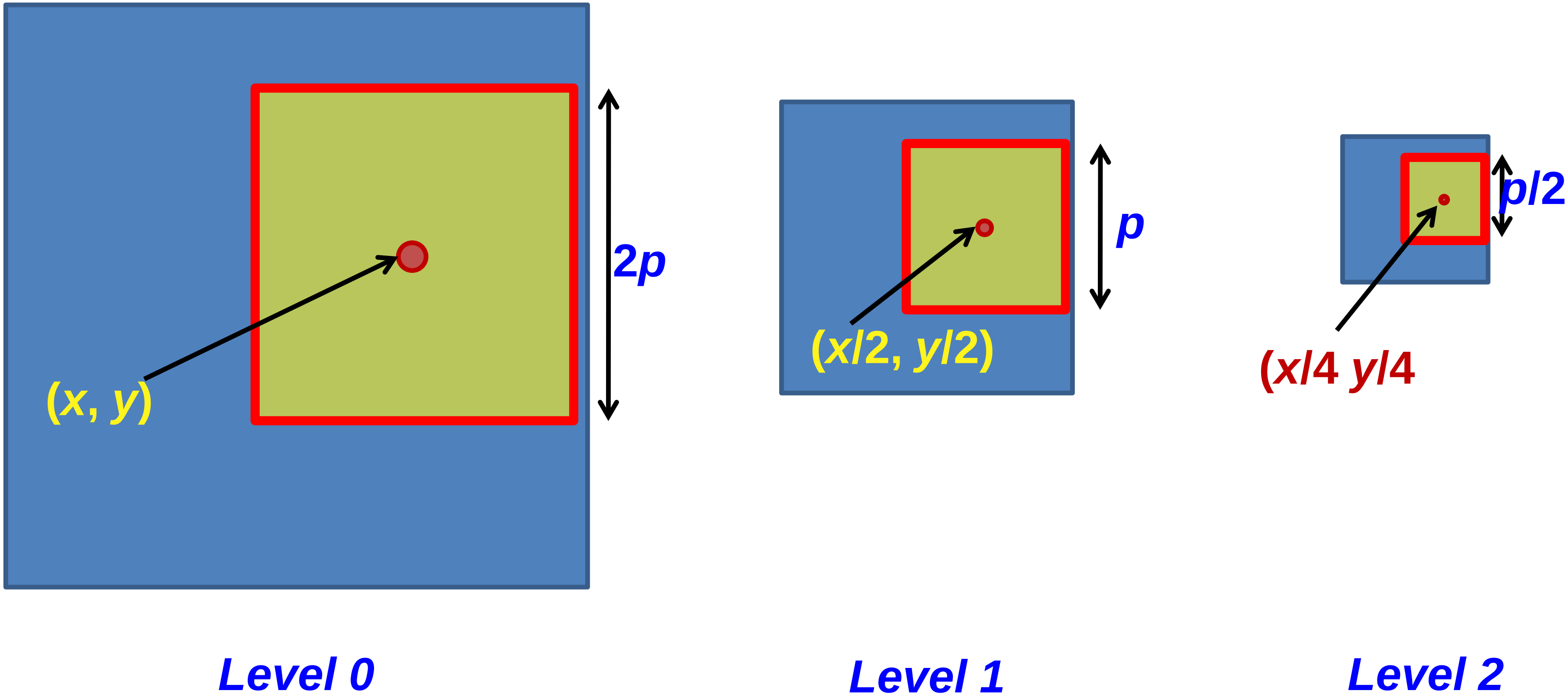
- Hierarchical Search





Computation Considerations in Correlation Based TM (4)

- Hierarchical Search



Computation Considerations in Correlation Based TM (4)

- Hierarchical Search
  - Start at Level 2 with the reference of size 4X4
  - Search in the rectangle $[-p/4, p/4]$ $[-p/4, p/4]$ centered at $(x/4, y/4)$

Computation Considerations in Correlation Based TM (4)

- Hierarchical Search
  - Start at Level 2 with the reference of size 4X4
  - Search in the rectangle $[-p/4, p/4]$ $[-p/4, p/4]$ centered at $(x/4, y/4)$

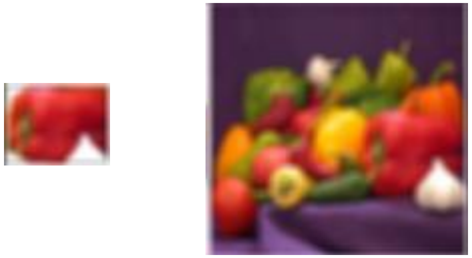
  - Let optimal found at (x_1, y_1) *with respect to* $(x/4, y/4)$.

Computation Considerations in Correlation Based TM (4)

- Hierarchical Search
  - At Level 1, with the reference of size 8X8
  - Search in the rectangle [-1, 1] X [-1, 1] centered at $(x/2 + 2x_1, y/2 + 2y_1)$

Computation Considerations in Correlation Based TM (4)

- Hierarchical Search
  - At Level 1, with the reference of size 8X8
  - Search in the rectangle [-1, 1] X [-1, 1] centered at $(x/2 + 2x_1, y/2 + 2y_1)$



  - Let optimal found at (x_2, y_2) *with respect to* $(x/2, y/2)$.

Computation Considerations in Correlation Based TM (4)

- Hierarchical Search
  - At Level 0, with the reference of size 16X16
  - Search in the rectangle [-1, 1] X [-1, 1] centered at $(x+2x_2, y+2y_2)$

Computation Considerations in Correlation Based TM (4)

- Hierarchical Search
  - At Level 0, with the reference of size 16X16
  - Search in the rectangle [-1, 1] X [-1, 1]
    centered at $(x+2x_2, y+2y_2)$

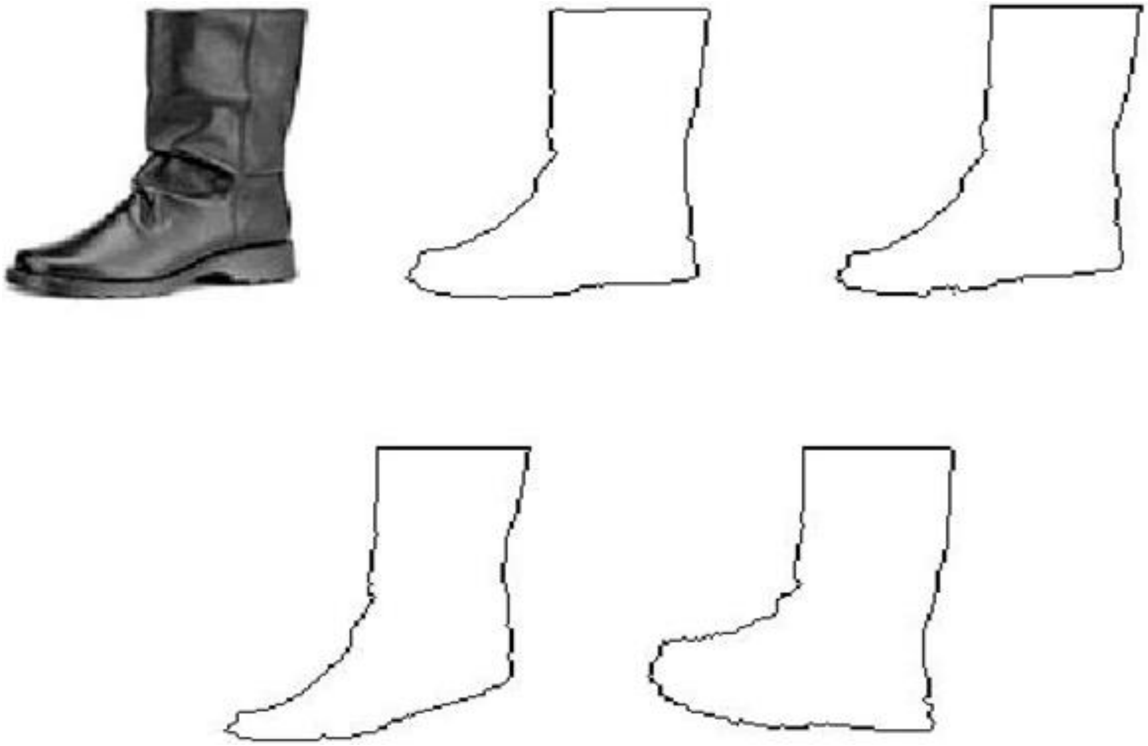
  - Location at this time is the final one

Computation Considerations in Correlation Based TM (4)

- Complexity of Hierarchical Search
    - 9 × No. of Decompositions +
    - Complexity at highest level

Deformable Template Matching

- Why is This?
  - Test and reference patterns are seldom exact
  - Rather, they are ‘similar’
  - In CBIR, query sketch significantly differs from the shapes in image DB



Deformable Template Matching

- The philosophy: Given a reference pattern $r(i,j)$ known as prototype:
  - Deform the prototype to produce different variants. Deformation is described by the application of a parametric transform on $r(i,j)$:

$$T_{\xi}\left[r(i, j)\right]$$

  - Match the test pattern with each of the deformed patterns

Deformable Template Matching

- For different values of the parameter $\underline{\xi}$, the goodness of fit with the test pattern is given by the matching energy:

$$E_m\left(\underline{\xi}\right)$$

- The goal is to chose $\underline{\xi}$ so that $E_m\left(\underline{\xi}\right)$ is minimum

Deformable Template Matching

- However, the higher the deformation, $\underline{\xi}$ the higher the deviation from the prototype. This is quantified by a cost known as deformation energy:

$$E_d\left(\underline{\xi}\right)$$

- In deformable template matching,

$$\text{compute } \underline{\xi} \text{ so that } \underline{\xi} : \min_{\underline{\xi}} \left[E_m(\underline{\xi}) + E_d(\underline{\xi}) \right]$$

- Thus target: small deformation and small matching energy

Deformable Template Matching

- The essential elements
  - A prototype of the reference
  - Transformation function
  - Matching Energy cost
  - Deformation Energy cost

Deformable Template Matching

- The prototype of the reference
  - Should be representable
  - Capture the *mean shape* characteristics of an object

Deformable Template Matching

- Transformation function
  - Any appropriate parametric operation
  - A suitable transformation is:

$$(x,y) \longrightarrow (x,y) + (D^{x}(x,y), D^{y}(x,y))$$

Deformable Template Matching

- Transformation function
  - Any appropriate parametric operation
  - A suitable transformation is:

$$(x, y) \longrightarrow (x, y) + (D^x(x, y), D^y(x, y))$$

where,

$$D^x(x, y) = \sum_{m=1}^{M} \sum_{n=1}^{N} \xi^x_{mn} e^x_{mn}(x, y)$$

$$D^y(x, y) = \sum_{m=1}^{M} \sum_{n=1}^{N} \xi^y_{mn} e^y_{mn}(x, y)$$

$$e^x_{mn}(x, y) = \alpha_{mn} \sin \pi nx \cos \pi my$$

$$e^y_{mn}(x, y) = \alpha_{mn} \cos \pi mx \sin \pi ny$$

$$\alpha_{mn} = \frac{1}{\pi^2(n^2 + m^2)}$$

Deformable Template Matching

- Deformation Energy cost
  - This should be minimum for no deformation, that is, for $\boldsymbol{\xi} = \mathbf{0}$.
  - Alternately,

$$E_d(\boldsymbol{\xi}) = \sum_m \sum_n ((\xi^x_{mn})^2 + (\xi^y_{mn})^2)$$

Deformable Template Matching

- Matching Energy cost
  - Captured as a function of point-to-point distance between reference and test pattern:

$$E_m(\boldsymbol{\xi}, \boldsymbol{\theta}, I) = \frac{1}{N_d} \sum_{i,j} (1 + \Phi(i, j))$$

Deformable Template Matching

- Matching Energy cost
  - Captured as a function of point-to-point distance between reference and test pattern:

$$E_m(\boldsymbol{\xi}, \boldsymbol{\theta}, I) = \frac{1}{N_d} \sum_{i,j} (1 + \Phi(i, j))$$

where,

$$\Phi(i,j) = -\exp\left(-\rho(\delta_i^2 + \delta_j^2)^{1/2}\right)$$

    - (δ_i, δ_j) is the displacement of the (*i*, *j*) pixel of the deformed template from the nearest pixel of the test template
    - ρ is a constant