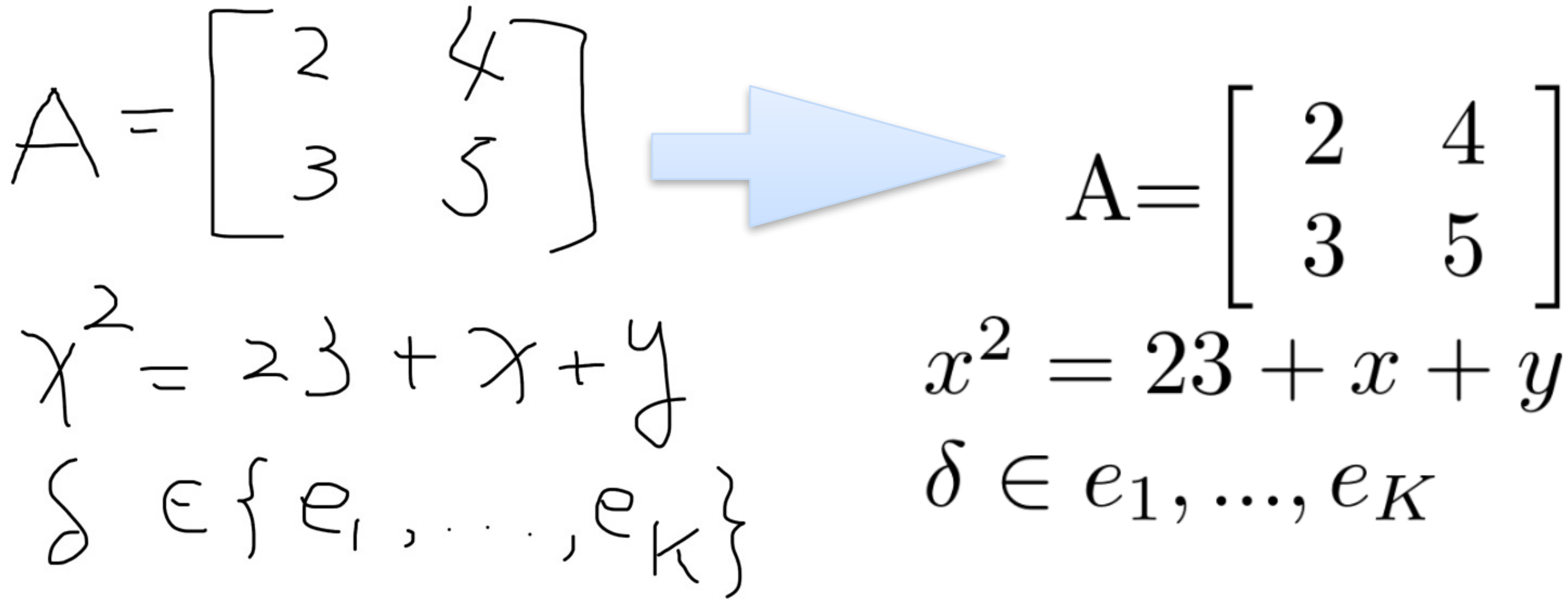
Discriminate analysis

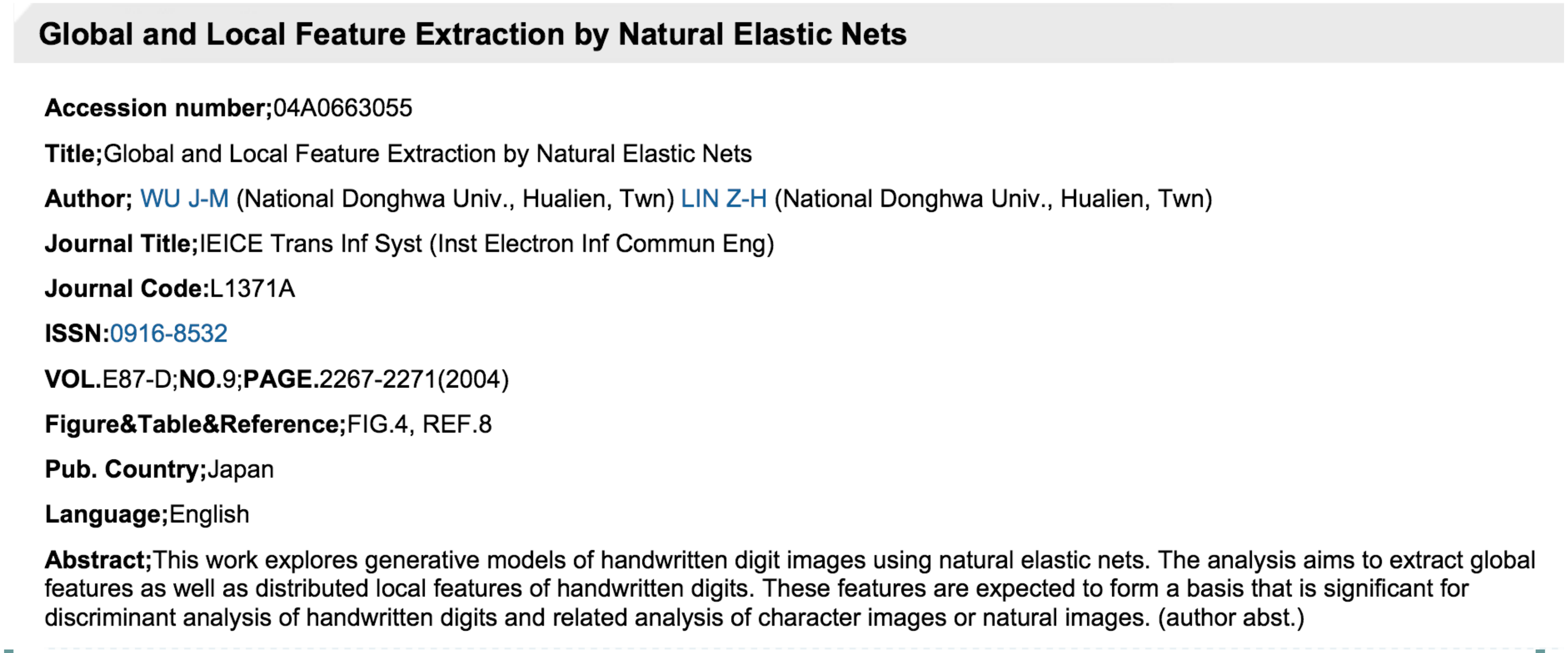
Pen written digit&character recognition

Outline

- Off-line pattern recognition
- Online pattern recognition

- One-line handwritten character recognition
- Off-line handwritten character recognition


$A = \begin{bmatrix} 2 & 4 \\ 3 & 5 \end{bmatrix}$
$x^2 = 23 + x + y$
$\delta \in \{e_1, \ldots, e_K\}$
$A = \begin{bmatrix} 2 & 4 \\ 3 & 5 \end{bmatrix}$
$x^2 = 23 + x + y$
$\delta \in e_1, \ldots, e_K$



Global and Local Feature Extraction by Natural Elastic Nets

Accession number;04A0663055

Title;Global and Local Feature Extraction by Natural Elastic Nets

Author; WU J-M (National Donghwa Univ., Hualien, Twn) LIN Z-H (National Donghwa Univ., Hualien, Twn)

Journal Title;IEICE Trans Inf Syst (Inst Electron Inf Commun Eng)

Journal Code:L1371A

ISSN:0916-8532

VOL.E87-D;**NO.**9;**PAGE.**2267-2271(2004)

Figure&Table&Reference;FIG.4, REF.8

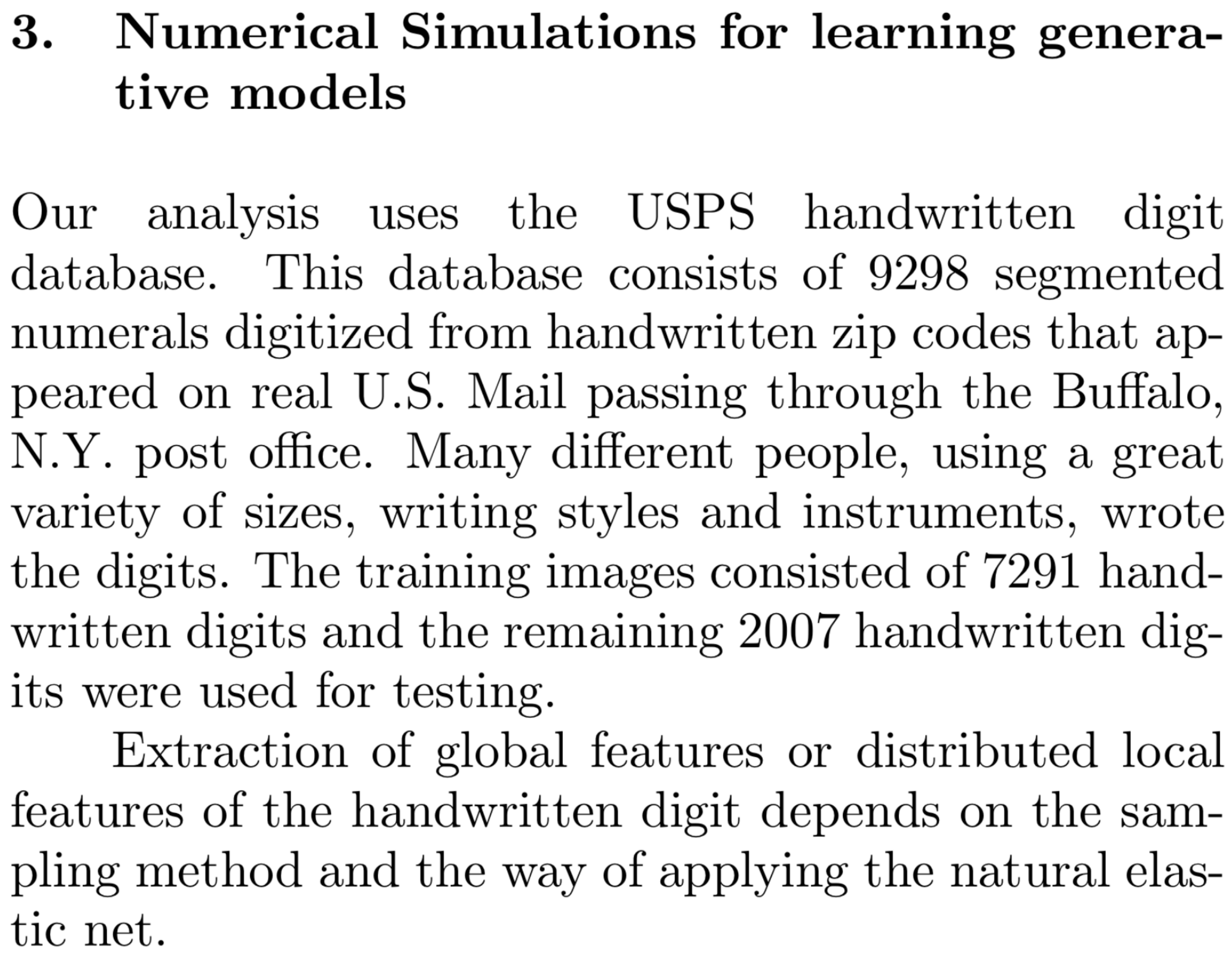
Pub. Country;Japan

Language;English

Abstract;This work explores generative models of handwritten digit images using natural elastic nets. The analysis aims to extract global features as well as distributed local features of handwritten digits. These features are expected to form a basis that is significant for discriminant analysis of handwritten digits and related analysis of character images or natural images. (author abst.)

Link

Methodologies



- **3. Numerical Simulations for learning generative models**

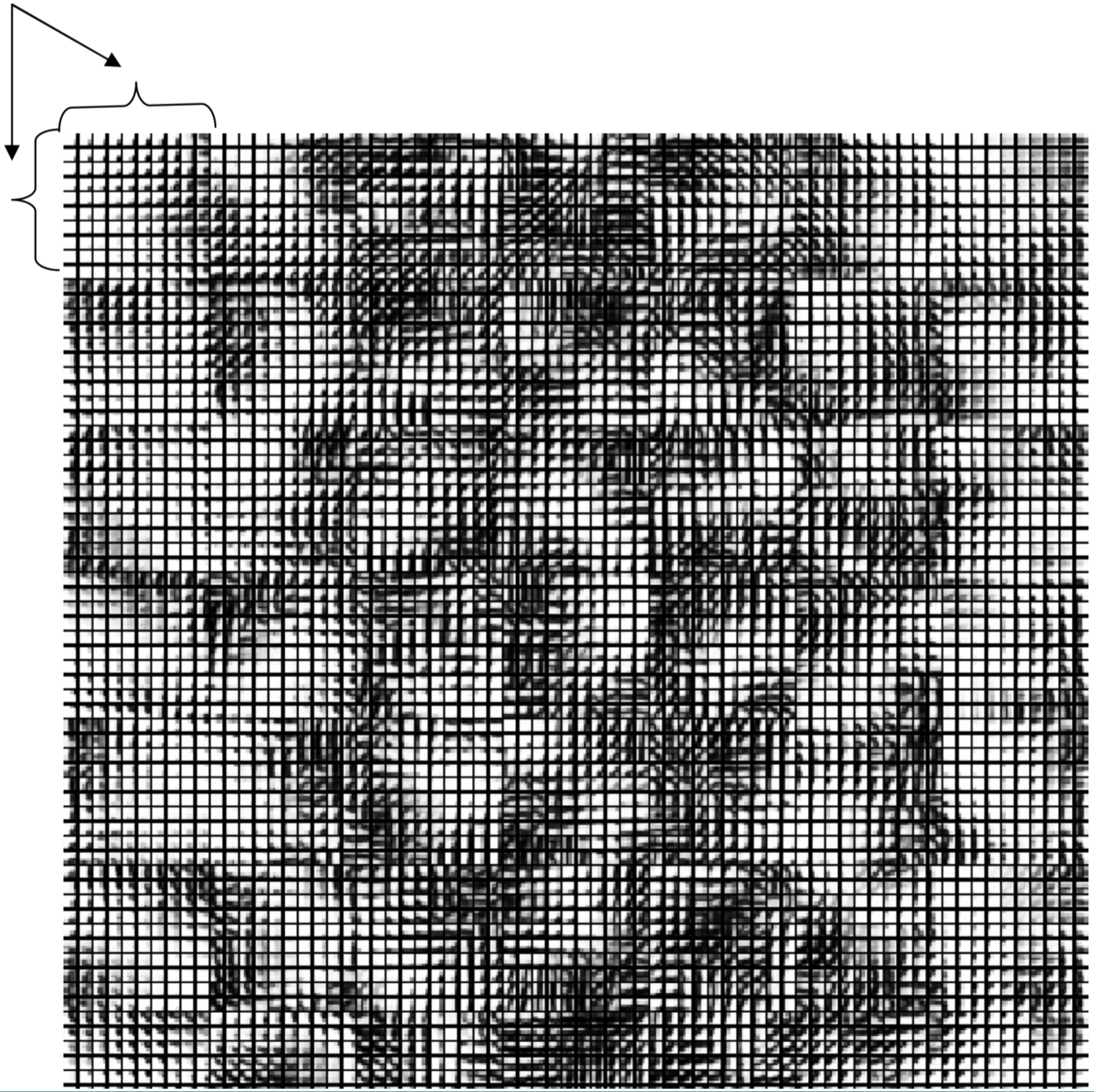
Our analysis uses the USPS handwritten digit database. This database consists of 9298 segmented numerals digitized from handwritten zip codes that appeared on real U.S. Mail passing through the Buffalo, N.Y. post office. Many different people, using a great variety of sizes, writing styles and instruments, wrote the digits. The training images consisted of 7291 handwritten digits and the remaining 2007 handwritten digits were used for testing.

Extraction of global features or distributed local features of the handwritten digit depends on the sampling method and the way of applying the natural elastic net.

As the preprocess applies the principle component analysis to training images, we have a set of eigenvectors or eigen-digits, each corresponding to an eigenvalue. We select forty significant eigenvectors to form a basis for dimensionality reduction. The projection of 7291 training images on the forty eigen-digits leads to a new set of training samples, each composed of 40 elements. We then apply a 20×20 natural elastic net to learn the generative model of these training samples. As a result, we have a set of cortical points ordered on the lattice and a covariance matrix for the generative model. Since each training sample is oriented from one and only one training image, it inherits the label of the training image. Let each training image have its own label denoting the digit type from zero to nine. By assigning each cortical point the label of the nearest training sample, we have a classifier based on the natural elastic net. After projecting the 2007 testing images on the space spanned by the forty eigen-digits, we have 2007 testing samples, each with its own label. Numerical simulations show that the classifier has a correct rate, 96.42%, for recognizing 2007 testing samples, which is comparable to the best known one, 97.5%[3].

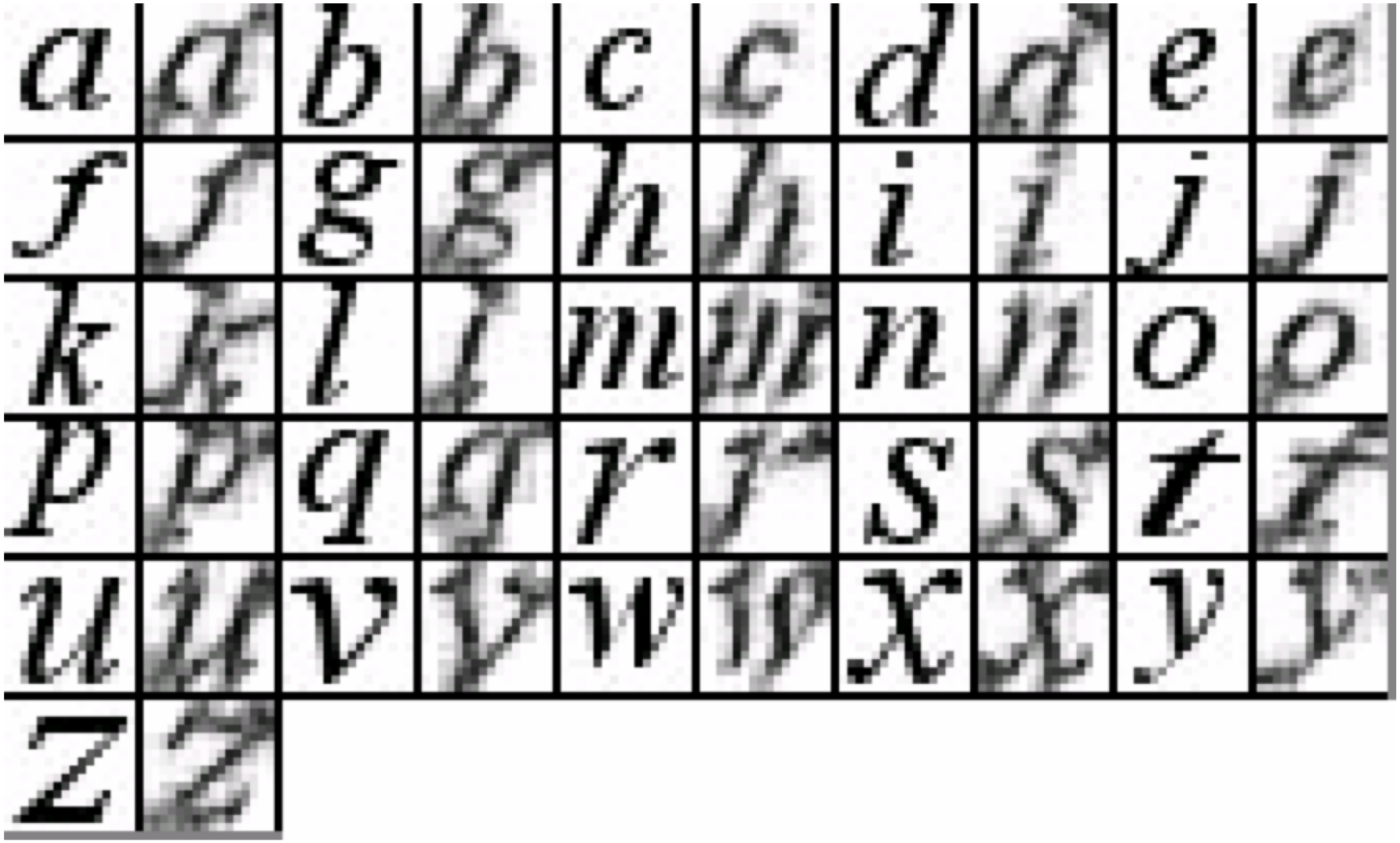
We combine 49 natural elastic nets to emulate a visual system for the process of handwritten digits. The whole digit image, 16×16 pixels, is uniformly partitioned into 49 overlapping receptive fields, each containing 4×4 pixels. Then sampling from the 7291 training images through these receptive fields produces 49 sets of training patterns, where each pattern contains 4×4 pixels. Each set of training patterns is analyzed by a 10×10 natural elastic net that emulates a population of visual neurons with respect to the same receptive field. As a result, each natural elastic net has 100 cortical points ordered on the 10×10 lattice and a matrix A as parameters of the generative model for 7291 patches obtained through a corresponding receptive field. Figure 2 show all cortical points of 49 natural elastic nets arranged by 7×7 blocks whose order is identical to the distribution of 49 receptive fields, and each block contains 10×10 cortical points of a natural elastic net. It reads that the most local features of handwritten digit images are line features, including vertical lines, horizontal lines and oblique lines, which are similar to the local feature extracted from natural images[1]. These distributed local features are primitive to visual analysis and can serve as internal representations essential for artificial visual functions..

A block for 10x10 cortical points of a natural elastic net



To accomplish our simulation, we apply the cortical points of 49 natural elastic nets to reconstruct the image beyond the training images. We select some from 2007 testing images, split each image into 49 overlapping patches through 49 receptive fields and feed each patch to a corresponding natural elastic net. Each natural elastic net generates a cortical point that is nearest to the patch. Combining 49 cortical points achieves a restored digit. The results are shown in figure 3, where 128 testing images and its reconstructed images are shown in odd columns and even columns, respectively. Compare the testing image and the reconstructed image, most of them are similar but some still have blur points. This may be improved by enlarging the number of emulating elements. The cortical points of 49 natural elastic nets are further applied to reconstruction of the characters from $'a'$ to $'z'$.

The results are shown in figure 4, where the characters and its reconstructed images are shown in odd columns and even columns, respectively. Although the training set contains only the digit images, the local features extracted by the natural elastic net is applicable to reconstruction of the characters that have similar features to digit images. The result implies the possibility of developing an artificial visual system incrementally from simple to complex.

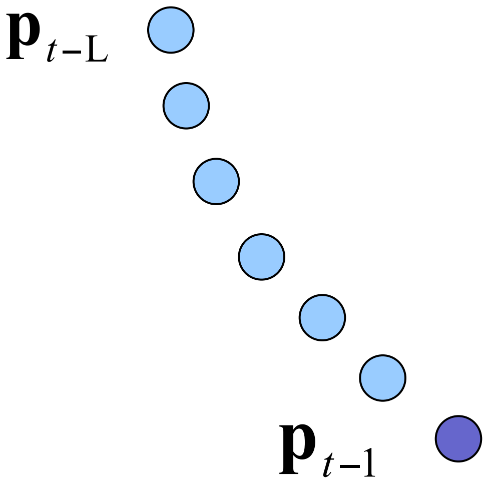


Machine Learning

- UCI Machine Learning Repository
- Pen-written database

Trajectory tracking

The position $\mathbf{p}_t$ is predictable by positions of previous L points



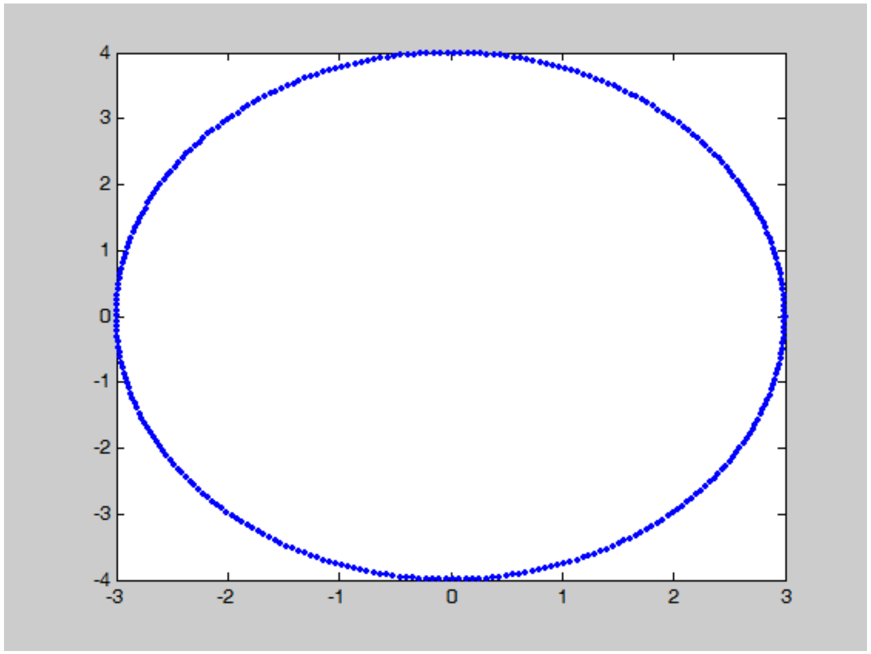
$$\mathbf{p}_t = \begin{bmatrix} p_{t1} \\ p_{t2} \end{bmatrix} \in R^2$$

$$\mathbf{p}_t = \begin{bmatrix} f_1(\mathbf{p}_{t-1},\ldots,\mathbf{p}_{t-L}) \\ f_2(\mathbf{p}_{t-1},\ldots,\mathbf{p}_{t-L}) \end{bmatrix}$$

Strategy

- Trajectory tracking
- Trajectory modeling by a recurrent multilayer neural network
- Integration of multiple models
- Fitting to multiple models
- Winner models

Tracking an ellipse



```
t=0:0.02:2*pi;
traject=[3*cos(t);4*sin(t)];
plot(traject(1,:),traject(2,:),'.');
```

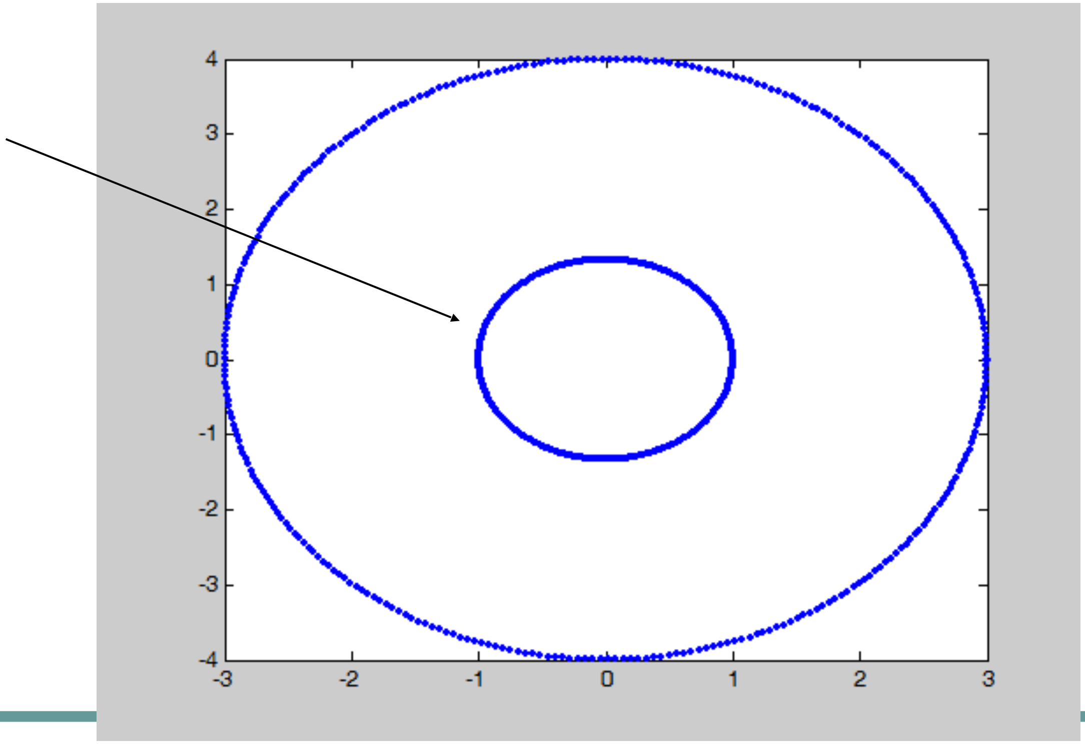
Function approximation

TRAINING PHASE

```
for i=1:2
    x=traject(:,1:1:length(t)-L)';
    y=traject(i,L+1:1:length(t))';
    max_x=max(max(x));max_y(i)=max(y);
    train_x=x/max_x;train_y=y/max_y(i);
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    % Training
    if i==1
        [E1,NetDef1,W11,W12]=noisecan(train_x,train_y,20);
    else
        [E1,NetDef2,W21,W22]=noisecan(train_x,train_y,20);
    end
end
```

Generate another ellipse

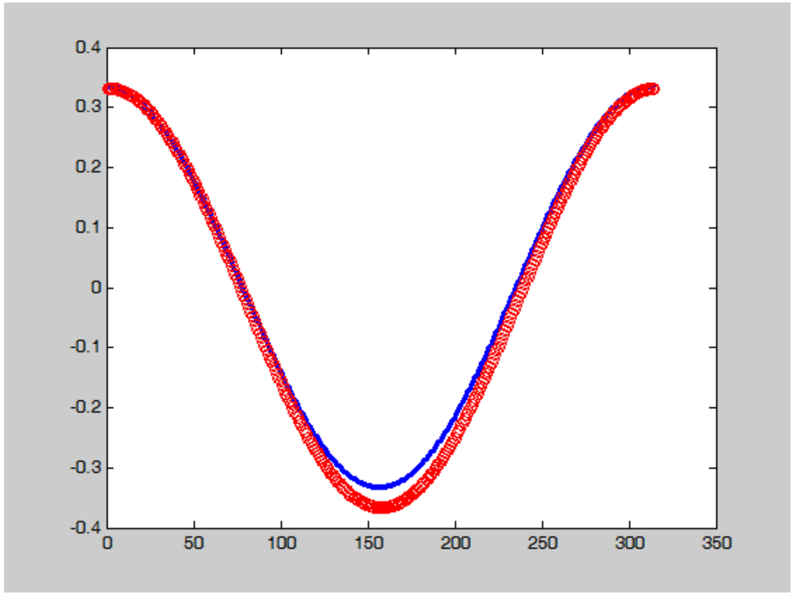
```
traject=[3/3*cos(t);4/3*sin(t)];
plot(traject(1,:),traject(2,:),'.');
```

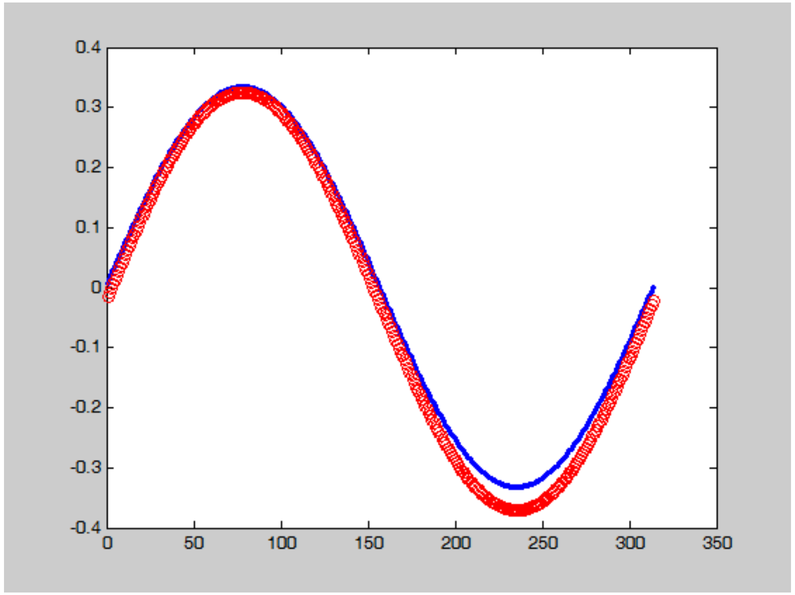


Prediction Phase

```
for i=1:2
    x=traject(:,1:1:length(t)-L)';
    y=traject(i,L+1:1:length(t))';
    test_x=x'/max_x;test_y=y'/max_y(i);
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    if i==1
        % Testing
        [Y_sim,E,PI] = nneval(NetDef1,W11,W12,test_x,test_y,1);
    else
        % Testing
        [Y_sim,E,PI] = nneval(NetDef2,W21,W22,test_x,test_y,1);
    end
end
```

Prediction error



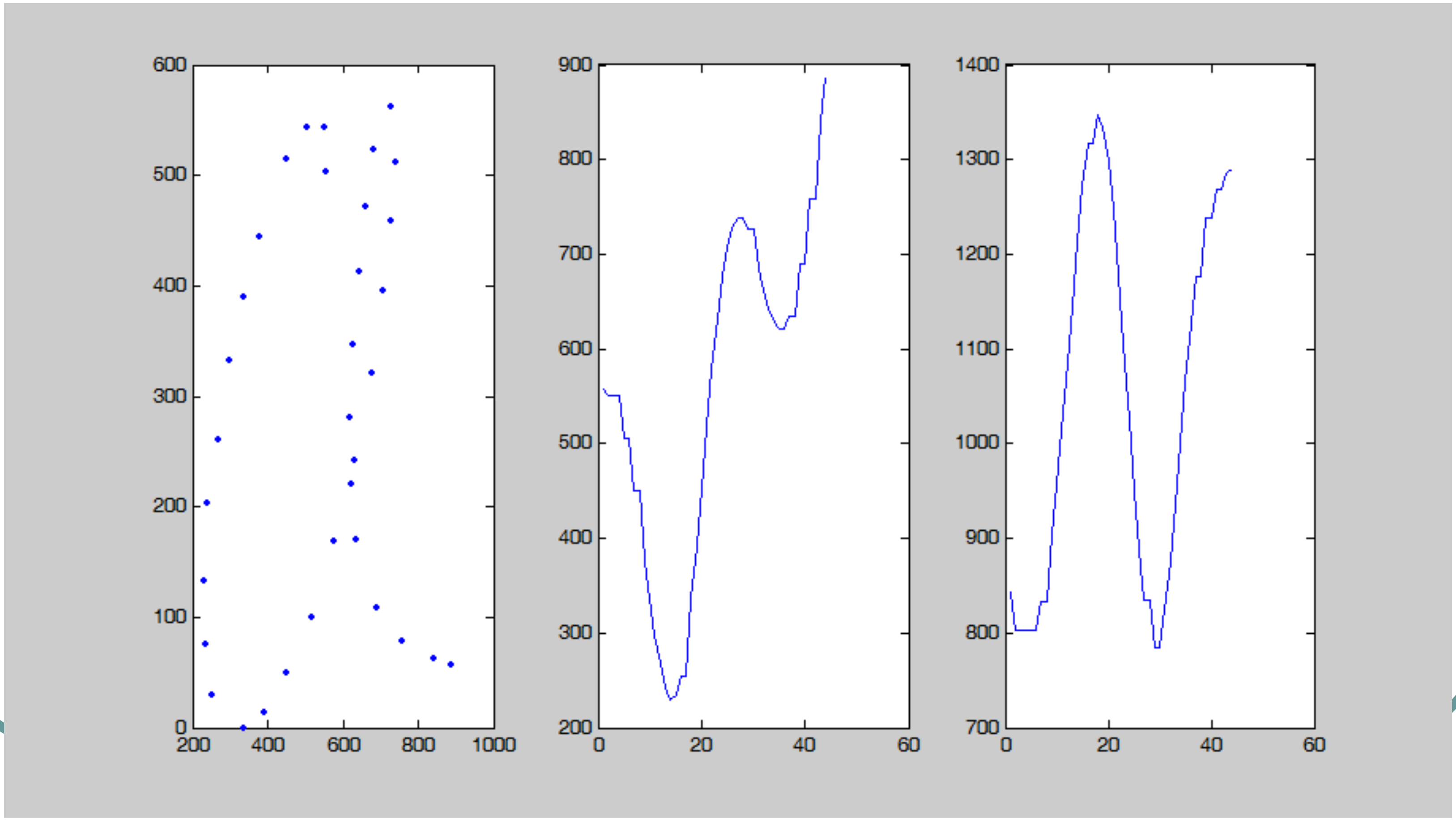


mean square testing error 0.000457
mean absolute testing error 0.017639
mean square testing error 0.000663
mean absolute testing error 0.023554

Pen Written Digit

```
>> load data\a1.txt
>> plot_curve(a1)
```

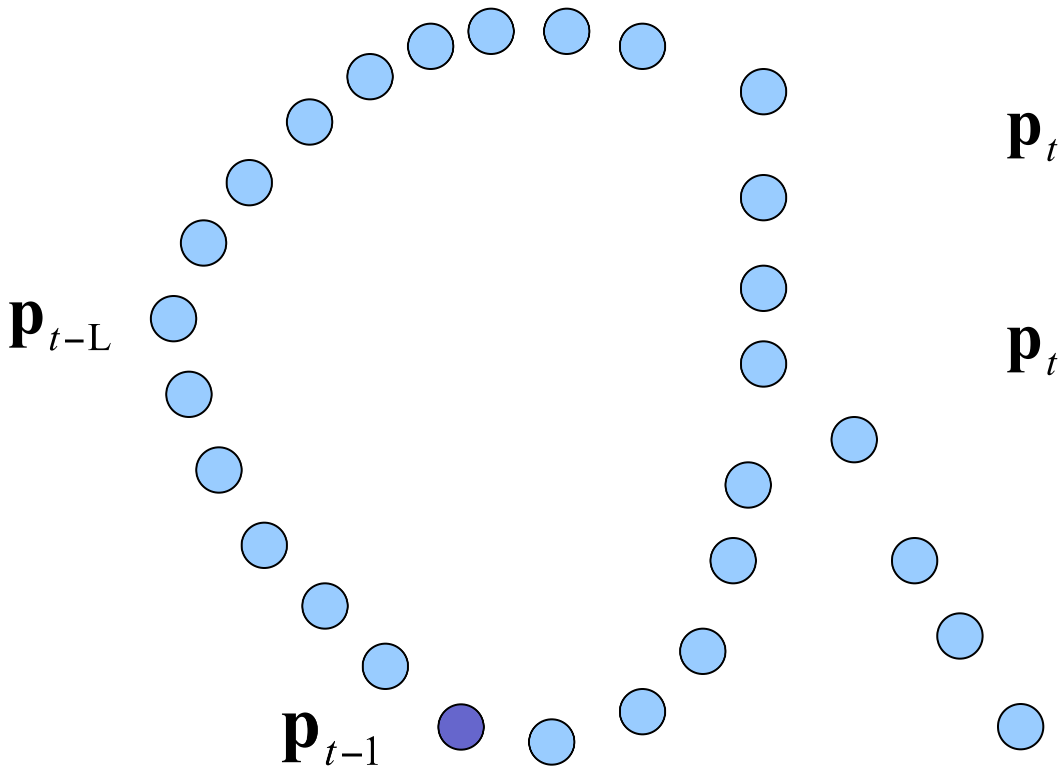
plot_curve.m



Preprocess

- Demean
- Rescale within [-1 1]x[-1 1]
- Form paired data

Trajectory tracking



$$\mathbf{p}_t = \begin{bmatrix} p_{t1} \\ p_{t2} \end{bmatrix} \in R^2$$

$$\mathbf{p}_t = \begin{bmatrix} f_1(\mathbf{p}_{t-1},\ldots,\mathbf{p}_{t-L}) \\ f_2(\mathbf{p}_{t-1},\ldots,\mathbf{p}_{t-L}) \end{bmatrix}$$

Paired data

$$\mathbf{p}_t = \begin{bmatrix} p_{t1} \\ p_{t2} \end{bmatrix} \in R^2$$

$$\mathbf{p}_t = \begin{bmatrix} f_1(\mathbf{p}_{t-1}, \ldots, \mathbf{p}_{t-L}) \\ f_2(\mathbf{p}_{t-1}, \ldots, \mathbf{p}_{t-L}) \end{bmatrix}$$

$$\mathrm{S1} = \{x[t], y_1[t]\}_{t=L+1}^{N}$$

$$S2 = \{x[t], y_2[t]\}_{t=L+1}^{N}$$

$$x[t] = [\mathbf{p}_{t-1}^T, \ldots, \mathbf{p}_{t-L}^T]$$

$$y_1[t] = p_{t1}, y_2[t] = p_{t1}$$

Function approximation

- MLP Learning subject to $S1 = \{x[t], y_1[t]\}_{t=L+1}^{N}$ leads to an approximating network

$$y = f(x;\theta_1)$$

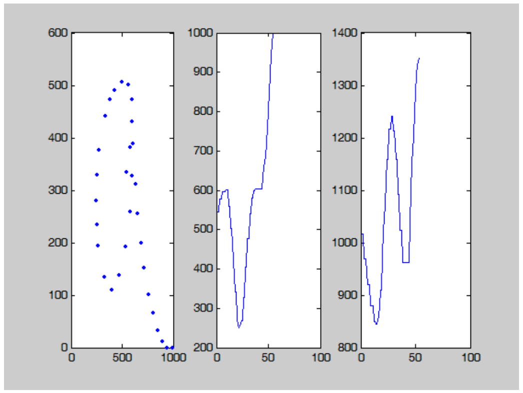
Function approximation

- MLP Learning subject to $S2 = \{x[t], y_2[t]\}_{t=L+1}^{N}$ leads to an approximating network

$$y = f(x;\theta_2)$$

Parsing phase

- Input a digit



$$e_1 + e_2$$

Two mean square errors

Form two sets of paired data

$$S1 = \{x[t], y_1[t]\}_{t=L+1}^{N}$$

$$S2 = \{x[t], y_2[t]\}_{t=L+1}^{N}$$

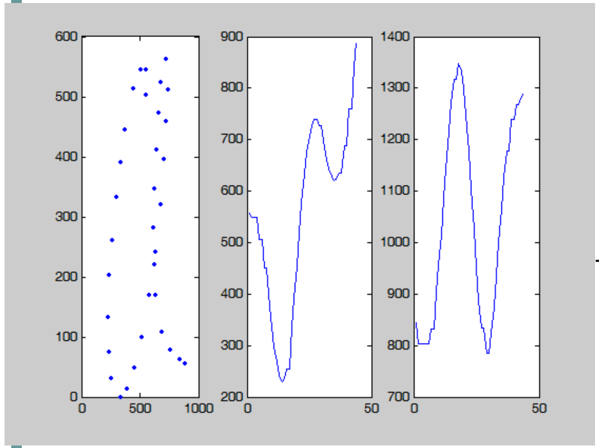
$$y = f(x;\theta_1)$$

$$y = f(x;\theta_2)$$

Inference

- Mean square errors of parsing score similarity between the input digit and the template.

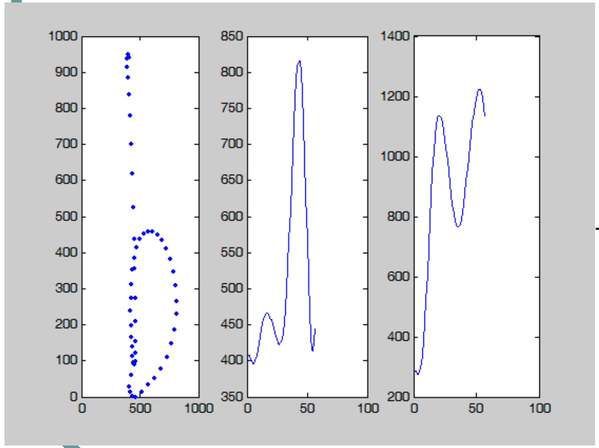
Numerical experiments : learning phase



MLPotts learning

$$y = f(x;\theta_{a1})$$

$$y = f(x;\theta_{a2})$$

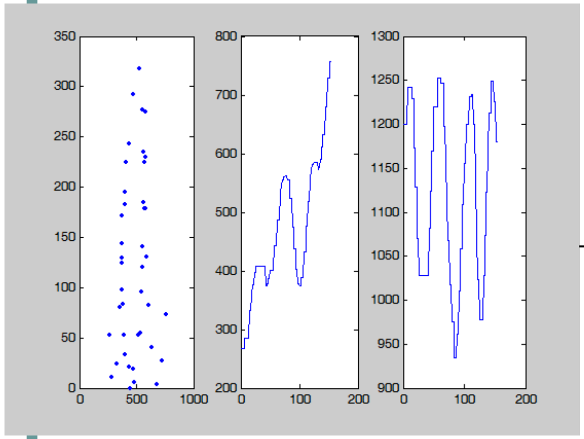


MLPotts learning

$$y = f(x;\theta_{b1})$$

$$y = f(x;\theta_{b2})$$

Parsing phase



$$\boxed{\begin{array}{l} y = f(x;\theta_{a1}) \\ y = f(x;\theta_{a2}) \end{array}} \longrightarrow e_a = e_1 + e_2$$

$$\boxed{\begin{array}{l} y = f(x;\theta_{b1}) \\ y = f(x;\theta_{b2}) \end{array}} \longrightarrow e_b = e_1 + e_2$$

Decision

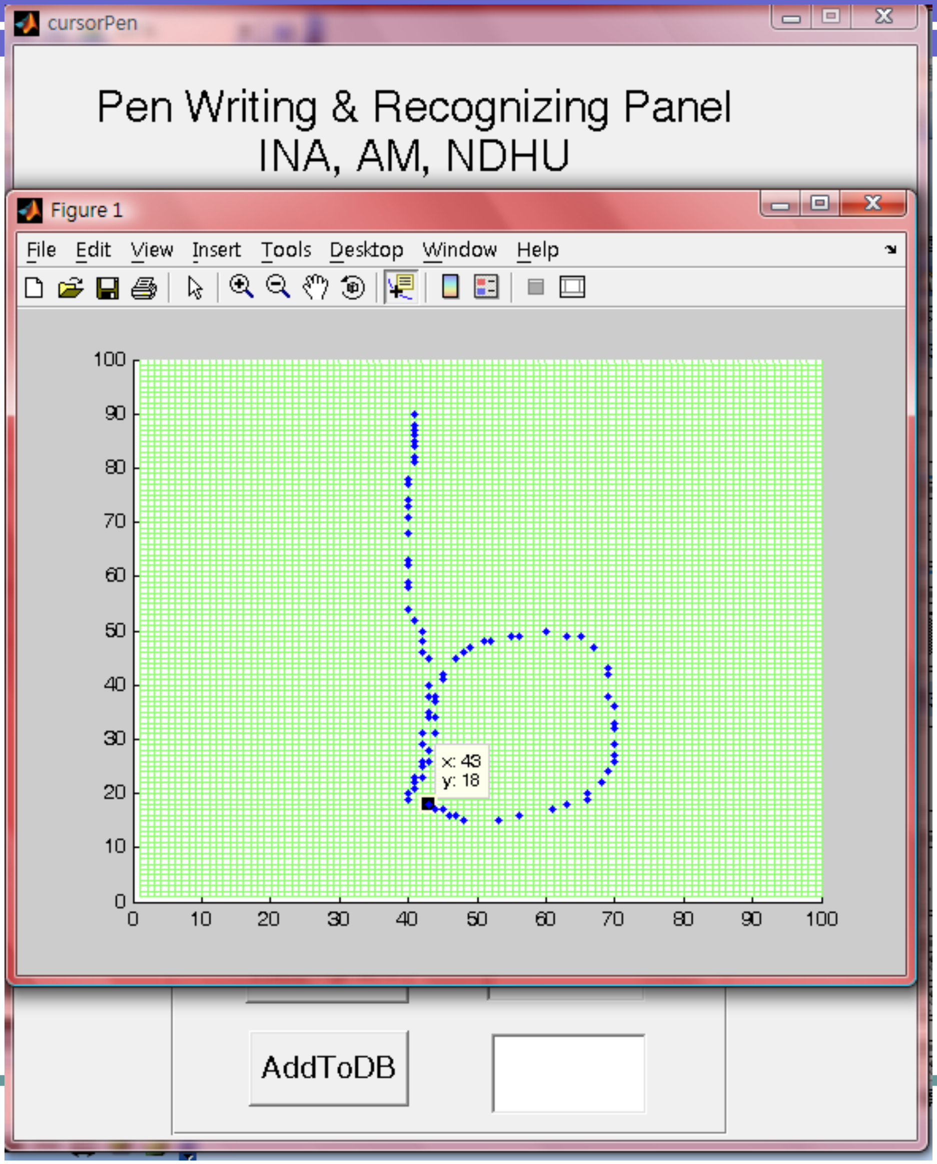
If $e_a = e_1 + e_2 < e_b = e_1 + e_2$

output 'a'

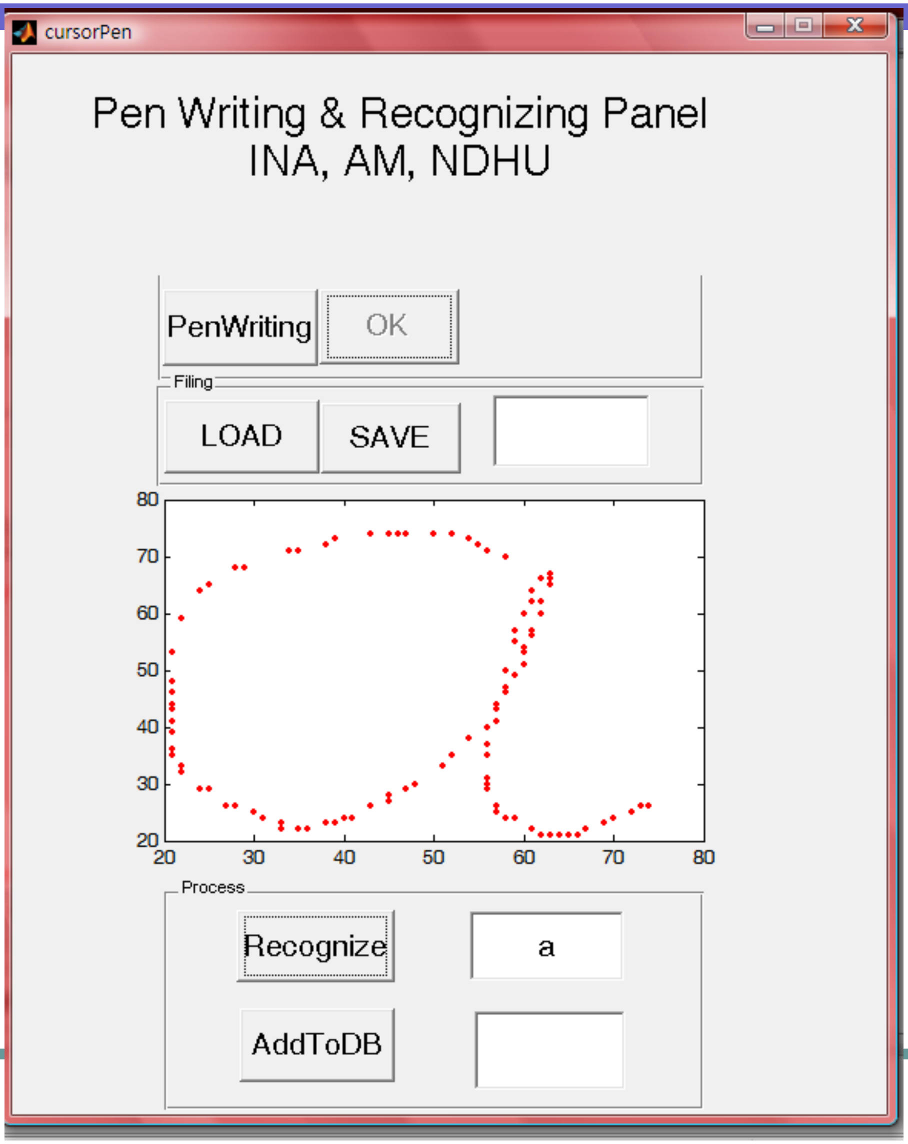
otherwise

output 'b'

Write a character


cursorPen
Pen Writing & Recognizing Panel
INA, AM, NDHU
Figure 1
File Edit View Insert Tools Desktop Window Help
x: 43
y: 18
AddToDB

Character Recognition


cursorPen
Pen Writing & Recognizing Panel
INA, AM, NDHU
PenWriting
OK
Filing
LOAD
SAVE
80
70
60
50
40
30
20
20
30
40
50
60
70
80
Process
Recognize
a
AddToDB