Appendix C

Source Code

C.1 Input Frame Design

```
* MainFrame.java
* Created on August 26, 2005, 10:20 AM

package Test1;

import java.awt.*;
import java.awt.geom.*;
import java.util.*;
import java.io.*;
import javax.swing.JFileChooser;
import javax.swing.JOptionPane;
* @author Cholwich and Santosh


public class MainFrame extends javax.swing.JFrame
{
/** Creates new form MainFrame */
        public MainFrame()
```

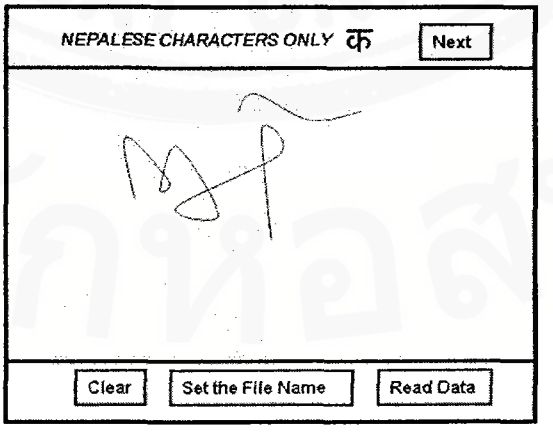

Figure C.1 Input Window

```
        {
        initComponents();
        this.setSize(600,500);
        }
/** This method is called from within the constructor to initialize the form.
* WARNING: Do NOT modify this code.
The content of this method is always regenerated by the Form Editor.*/

// editor-fold defaultstate="collapsed" desc="Generated Code "
private void initComponents()
{
jPanel1 = new javax.swing.JPanel();
clearButton = new javax.swing.JButton();
jButton1 = new javax.swing.JButton();
jButton3 = new javax.swing.JButton();
mainPanel = new javax.swing.JPanel();
jPanel2 = new javax.swing.JPanel();
jLabel1 = new javax.swing.JLabel();
jLabel2 = new javax.swing.JLabel();
jButton2 = new javax.swing.JButton();
setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
addWindowListener(new java.awt.event.WindowAdapter()
{
        public void windowClosing(java.awt.event.WindowEvent evt)
        {
        formWindowClosing(evt);
        }
});
clearButton.setBackground(new java.awt.Color(255, 0, 51));
clearButton.setText("Clear");
clearButton.addActionListener(new java.awt.event.ActionListener()
{
        public void actionPerformed(java.awt.event.ActionEvent evt)
        {
        clearButtonActionPerformed(evt);
        }
});
jPanel1.add(clearButton);
jButton1.setBackground(new java.awt.Color(255, 153, 0));
jButton1.setText("Set File Name");
jButton1.addActionListener(new java.awt.event.ActionListener()
{
        public void actionPerformed(java.awt.event.ActionEvent evt)
        {
        jButton1ActionPerformed(evt);
        }
});
jPanel1.add(jButton1);
```

```
jButton3.setBackground(new java.awt.Color(153, 153, 0));
jButton3.setText("read data");
jButton3.addActionListener(new java.awt.event.ActionListener()
{
        public void actionPerformed(java.awt.event.ActionEvent evt)
        {
        jButton3ActionPerformed(evt);
        }
});
jButton3.addMouseListener(new java.awt.event.MouseAdapter()
{
        public void mouseClicked(java.awt.event.MouseEvent evt)
        {
        jButton3MouseClicked(evt);
        }
});
jPanel1.add(jButton3);
getContentPane().add(jPanel1, java.awt.BorderLayout.SOUTH);
mainPanel.setBackground(new java.awt.Color(204, 204, 255));
mainPanel.setBorder(new javax.swing.border.MatteBorder(new java.awt.Insets(1, 1, 1, 1),
new java.awt.Color(28, 189, 189)));
mainPanel.addMouseMotionListener(new java.awt.event.MouseMotionAdapter()
{
        public void mouseDragged(java.awt.event.MouseEvent evt)
        {
        mainPanelMouseDragged(evt);
        }
});
mainPanel.addMouseListener(new java.awt.event.MouseAdapter()
{
        public void mouseReleased(java.awt.event.MouseEvent evt)
        {
        mainPanelMouseReleased(evt);
        }
});
getContentPane().add(mainPanel, java.awt.BorderLayout.CENTER);
jLabel1.setBackground(new java.awt.Color(0, 153, 153));
jLabel1.setFont(new java.awt.Font("Arabic Transparent", 3, 11));
jLabel1.setForeground(new java.awt.Color(0, 102, 102));
jLabel1.setText("NEPALESE CHARACTERS ONLY");
jLabel1.setMaximumSize(new java.awt.Dimension(155, 20));
jLabel1.setMinimumSize(new java.awt.Dimension(155, 20));
jLabel1.setPreferredSize(new java.awt.Dimension(155, 20));
jPanel2.add(jLabel1);
jLabel2.setFont(new java.awt.Font("Mangal", 1, 36));
jLabel2.setText("ŏ915");
jPanel2.add(jLabel2);
jButton2.setBackground(new java.awt.Color(204, 0, 204));
```

```
jButton2.setText("Next");
jButton2.addActionListener(new java.awt.event.ActionListener()
{
        public void actionPerformed(java.awt.event.ActionEvent evt)
        {
        jButton2ActionPerformed(evt);
        }
});
jPanel2.add(jButton2);
getContentPane().add(jPanel2, java.awt.BorderLayout.NORTH);
pack();
}
//editor_fold

private void jButton3MouseClicked(java.awt.event.MouseEvent evt)
{
// TODO add your handling code here:
JFileChooser fileChooser = new JFileChooser();
fileChooser.setFileSelectionMode(JFileChooser.FILES_ONLY);
int w = fileChooser.showOpenDialog(this);
if (w ==JFileChooser.CANCEL_OPTION)
return;
File fileName = fileChooser.getSelectedFile();
if (fileName ==null —— fileName.getName().equals(""))
JOptionPane.showMessageDialog(this,"Invalid File Name","Invalid File Name",
JOptionPane.ERROR_MESSAGE);
else
        {
          try
          {
          String line;
          BufferedReader input = new BufferedReader(new FileReader(fileName));
          while ((line= input.readLine())!=null);
          }
          catch (IOException ioException)
          {
          JOptionPane.showMessageDialog(this,"Error Opening File","Error",
          JOptionPane.ERROR_MESSAGE);
          }
        }
}

private void formWindowClosing(java.awt.event.WindowEvent evt)
{
// TODO add your handling code here:
if (output!=null)
        {
        output.flush();
```

```
        output.close();
        }
}

private PrintWriter output=null;
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt)
{
// TODO add your handling code here:
JFileChooser fileChooser = new JFileChooser();
fileChooser.setFileSelectionMode(JFileChooser.FILES_ONLY);
int w =fileChooser.showSaveDialog(this);
if(w == JFileChooser.CANCEL_OPTION)
return ;
File fileName = fileChooser.getSelectedFile();
if (fileName == null —— fileName.getName().equals(""))
JOptionPane.showMessageDialog(this,"InvalidFile Name","Invalid File Name",
JOptionPane.ERROR_MESSAGE);
else
        {
         try
         {
         output = new PrintWriter (new FileWriter(fileName));
         }
         catch (IOException ioException)
         {
         JOptionPane.showMessageDialog(this,"Error opening file","Error",
         JOptionPane.ERROR_MESSAGE);
         }
        }
}

private void jButton2ActionPerformed(java.awt.event.ActionEvent evt)
{
// TODO add your handling code here:
String error;
error="";
if (output!=null)
        {
        clearButtonActionPerformed(null);
        for(int i=0; i¡points.size(); i++)
           {
           Point2D.Double p = (Point2D.Double)points.get(i);
           output.println(p.x+","+p.y);
           }
        output.println(" ");
        output.flush();
        points.clear();
        }
```

```
else
        {
        System.err.println("Set File Name First");
        JOptionPane.showMessageDialog(null, error,"ERROR ..!! Please, Set filename first",
        JOptionPane.INFORMATION_MESSAGE);
        }
        mainPanel.setBackground(Color.GREEN);
}

private void mainPanelMouseReleased(java.awt.event.MouseEvent evt)
{
// TODO add your handling code here:
first = true;
points.add(new Point2D.Double(-1,-1));
}

private void clearButtonActionPerformed(java.awt.event.ActionEvent evt)
{
// TODO add your handling code here:
Graphics g = mainPanel.getGraphics();
g.clearRect(0,0, mainPanel.getWidth(), mainPanel.getHeight());
mainPanel.repaint();
}
private boolean first = true;
private Point2D.Double curP = new Point2D.Double();
private Point2D.Double newP = new Point2D.Double();
private Vector points = new Vector();

private void mainPanelMouseDragged(java.awt.event.MouseEvent evt)
{
// TODO add your handling code here:
Graphics2D g2 = (Graphics2D)mainPanel.getGraphics();
g2.setStroke(new BasicStroke(1));
if (first)
        {
        curP.setLocation(evt.getX(), evt.getY());
        first = false;
        points.add(new Point2D.Double(evt.getX(), evt.getY()));
        }
else
        {
        newP.setLocation(evt.getX(), evt.getY());
        g2.setColor(Color.RED);
        Line2D.Double l = new Line2D.Double(curP,newP);
        g2.draw(l);
        curP.setLocation(newP);
        points.add(new Point2D.Double(evt.getX(), evt.getY()));
        }
```

```
System.out.println(evt.getX()+","+evt.getY());
}

public static void main(String args[])
{
        java.awt.EventQueue.invokeLater(new Runnable()
        {
        public void run()
                {
                new MainFrame().setVisible(true);
                }
        });
}

// Variables declaration - do not modify
private javax.swing.JButton clearButton;
private javax.swing.JButton jButton1;
private javax.swing.JButton jButton2;
private javax.swing.JButton jButton3;
private javax.swing.JLabel jLabel1;
private javax.swing.JLabel jLabel2;
private javax.swing.JPanel jPanel1;
private javax.swing.JPanel jPanel2;
private javax.swing.JPanel mainPanel;
// End of variables declaration
}
```

C.2 Alignment of Two Non-linear Sequences and Averaging

Alignment of two sequences of different lengths (a string of coordinates)is possible with the use of Dynamic Time Warping (DTW). How it was done in the experiment is demonstrated below with the full source code in MATLAB.
Dist is un-normalized distance between T and R
D is the accumulated distance matrix
T is the vector you are testing against
R is the vector you are testing
k is the normalizing factor
w is the optimal path

function $[D, Dist, w, average] = dtw(T, R)$
$[rows, N] = size(T);$
$[rows, M] = size(R);$
for $n = 1 : N$
    for $m = 1 : M$
    $d(n,m)$ =sqrt$((T(n) - R(m))^2)$;
    end

end
$D = zeros(size(d))$;
$D(1,1) = d(1,1)$;
for $n = 2 : N$
$D(n,1) = d(n,1) + D(n-1,1)$;
end
for $m = 2 : M$
$D(1,m) = d(1,m) + D(1,m-1)$;
end
for $n = 2 : N$
    for $m = 2 : M$
    $D(n,m) = d(n,m) + min([D(n-1,m), D(n-1,m-1), D(n,m-1)])$;
    end
end
// Distance between T and R,
$Dist = D(N,M)$;

$n = N$;
$m = M$;
$k = 1$;
$w = [\,]$;
$w(1,:) = [N,M]$;
while $((n+m) = 2)$
if $(n-1) == 0$
$m = m-1$;
    elseif $(m-1) == 0$
    $n = n-1$;
    else
    $[values, number] = min([D(n-1,m), D(n,m-1), D(n-1,m-1)])$;
    $cord = [values]$;
    switch *number*
    case 1
    $n = n-1$;
    case 2
    $m = m-1$;
    case 3
    $n = n-1$;
    $m = m-1$;
    end
end
$k = k+1$;
$w = cat(1, w, [n,m])$;
end
$w1 = rot90(w)$;
$w2 = rot90(w1)$;
// Averaging two sequences T and R
for $i = 1 : length(w)$
$average(i) = (R(w2(i,1)) + T(w2(i,2)))/2$

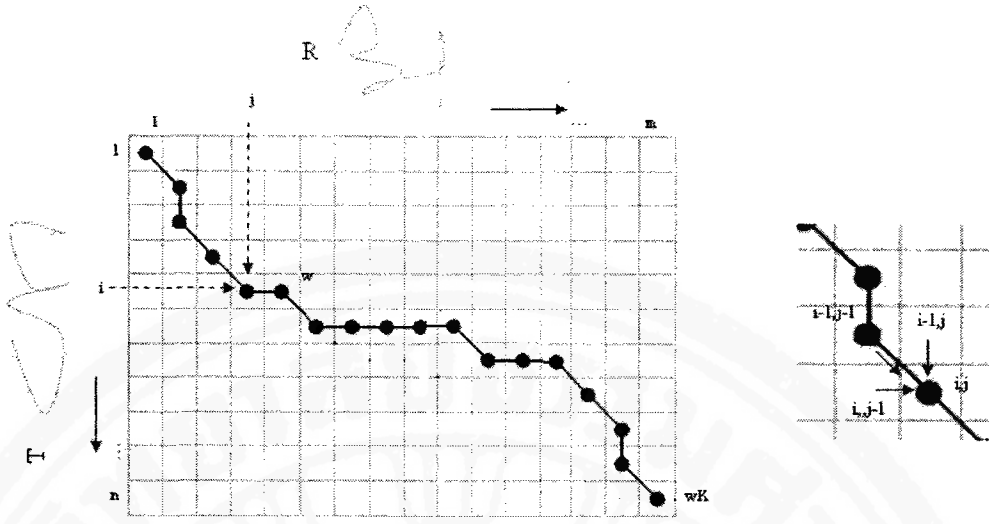

Figure C.2 Discrete Warping Path

	C1	C2	C3	C4	C5	C6
	A1	A2	A3	A4	A5	A6
A1	0	662	877	255	412	996
A2	662	0	295	468	268	400
A3	877	295	0	754	564	138
A4	255	468	754	0	219	869
A5	412	268	564	219	0	669
A6	996	400	138	869	669	0

Index: C1 = cluster 1, C2 = cluster 2,..., C6 = cluster 6

Figure C.3 Distance Matrix

end
average;

C.3 Agglomerative Clustering

The following source code provides an idea about the single-linkage, complete- linkage and centroid agglomerative hierarchical clustering.
Cluster = agg(Distance, Method, k)
Distance is square dissimilarity matrix, with Inf on leading diagonal
Method is one of 'single', 'complete' or 'centroid', and
k is the intended number of clusters
Cluster is a cell array showing which entities belong to which cluster

```
cluster = [1]  [2]  [1x2 double]  [4]  [5]  [6]        // c3 and c6 are going to be merged.

cluster = [1]  [2]  [1x2 double]  [4]  [5]             // c3 and c6 are merged.

cluster = [1]  [2]  [1x2 double]  [1x2 double]  [5]    // c4 and c5 are going to be merged

cluster = [1]  [2]  [1x2 double]  [1x2 double]         // c4 and c5 are merged.
```

Figure C.4 Clustering Results - Method: 'Single' and k =4

```
function cluster = agg(Distance,Method,k)

Distance = matrix;
Method = 'single';
k;

if nargin < 3
error('agg requires three arguments');
help(filename)
return
end

[R,C] = size(Distance);

//Put every point is in its own cluster

cluster = num2cell(1 : R);
for i = 2 : (R - k + 1)

//Find closest clusters

[MinRow,IdxDow] = min(Distance);
[temp,MinJ] = min(MinRow);
MinI = IdxDow(MinJ);

if MinI > MinJ
t = MinI;
MinI = MinJ;
MinJ = t;
end
```

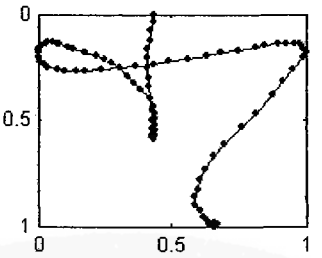

Figure C.5 A sample of a stroke

//Merge cluster j into cluster i, then delete j

$clusterMinI = [clusterMinIclusterMinJ]$
$cluster(MinJ) = [\,]$

//Calculate new Distance matrix

switch $Method$
case $'single'$
$Distance(:,MinI) = min(Distance(:,MinI), Distance(:,MinJ));$
$Distance(MinI,:) = min(Distance(MinI,:), Distance(MinJ,:));$
case $'complete'$
$Distance(:,MinI) = max(Distance(:,MinI), Distance(:,MinJ));$
$Distance(MinI,:) = max(Distance(MinI,:), Distance(MinJ,:));$
case $'centroid'$
$Distance(:,MinI) = (Distance(:,MinI) + Distance(:,MinJ))/2;$
$Distance(MinI,:) = (Distance(MinI,:) + Distance(MinJ,:))/2;$
otherwise
error('Unsupported Method in agg!');
end

$Distance(MinJ,:) = [\,];$
$Distance(:,MinJ) = [\,];$
$Distance(MinI,MinI) = inf;$
end
$cluster;$

C.4 Feature Vector Sequence

A sample of feature vector sequence is demonstrated in the following. It contains both pen tip positions and pen directions at every position. Feature is designated in the format of (x, y, θ).
Feature vector sequence of the stroke shown in Fig. C.5 is,

f = [(0.4406, 0.0053, 1.5708), (0.4406, 0.0370, 1.5708), (0.4266, 0.0794, -1.2517), (0.4266, 0.1217, 1.5708), (0.4126, 0.1640, 1.2517), (0.4126, 0.2011, 1.5708), (0.4126, 0.2487, 1.5708), (0.4196, 0.2963, 1.4250), (0.4196, 0.3386, 1.5708), (0.4266, 0.3968, 1.4512), (0.4336, 0.4550, 1.4512), (0.4336, 0.5026, 1.5708), (0.4336, 0.5397, 1.5708), (0.4336, 0.5714, 1.5708), (0.4336, 0.5820, 1.5708), (0.4406, 0.5873, 0.6477), (0.4476, 0.5714, -1.1558), (0.4476, 0.5503, -1.5708), (0.4476, 0.5238, -1.5708), (0.4476, 0.4921, -1.5708), (0.4476, 0.4656, -1.5708), (0.4406, 0.4339, 1.3540), (0.4266, 0.3968, 1.2097), (0.3916, 0.3598, 0.8142), (0.3636, 0.3228, 0.9239), (0.3357, 0.2910, 0.8485), (0.3077, 0.2487, 0.9868), (0.2727, 0.2275, 0.5443), (0.2517, 0.2116, 0.6477), (0.2168, 0.1958, 0.4261), (0.1748, 0.1799, 0.3617), (0.1329, 0.1640, 0.3617), (0.1119, 0.1587, 0.2471), (0.0769, 0.1429, 0.4261), (0.0699, 0.1376, 0.6477), (0.0559, 0.1323, 0.3617), (0.0350, 0.1323, 0), (0.0280, 0.1429, -0.9868), (0.0140, 0.1481, -0.3617), (0, 0.1640, -0.8485), (0, 0.1799, 1.5708), (0, 0.2011, 1.5708), (0.0070, 0.2275, 1.3124), (0.0210, 0.2381, 0.6477), (0.0350, 0.2487, 0.6477), (0.0699, 0.2593, 0.2939), (0.0979, 0.2646, 0.1869), (0.1399, 0.2698, 0.1254), (0.1748, 0.2698, 0), (0.2378, 0.2593, -0.1666), (0.3077, 0.2540, -0.0755), (0.3706, 0.2434, -0.1666), (0.4406, 0.2328, -0.1502), (0.4965, 0.2222, -0.1869), (0.5664, 0.2116, -0.1502), (0.6364, 0.1958, -0.2232), (0.6853, 0.1799, -0.3136), (0.7483, 0.1693, -0.1666), (0.8112, 0.1534, -0.2471), (0.8462, 0.1481, -0.1502), (0.8951, 0.1429, -0.1077), (0.9301, 0.1376, -0.1502), (0.9580, 0.1376, 0), (0.9790, 0.1376, 0), (0.9860, 0.1481, 0.9868), (0.9930, 0.1640, 1.1558), (1.0000, 0.1799, 1.1558), (0.9930, 0.2063, -1.3124), (0.9510, 0.2593, -0.9003), (0.9161, 0.3122, -0.9868), (0.8741, 0.3810, -1.0231), (0.8112, 0.4656, -0.9315), (0.7622, 0.5344, -0.9523), (0.6993, 0.6085, -0.8665), (0.6643, 0.6720, -1.0674), (0.6294, 0.7302, -1.0298), (0.6084, 0.7831, -1.1933), (0.6014, 0.8254, -1.4071), (0.5944, 0.8624, -1.3842), (0.5944, 0.8995, 1.5708), (0.6084, 0.9259, 1.0845), (0.6224, 0.9577, 1.1558), (0.6364, 0.9735, 0.8485), (0.6434, 0.9894, 1.1558), (0.6573, 0.9947, 0.3617), (0.6643, 1.0000, 0.6477), (0.6713, 0.9894, -0.9868), (0.6643, 0.9841, 0.6477), (0.6643, 0.9788, -1.5708)].