APPENDIX G

Program Listing

```
    function       varargout       =
OBR(varargin)
% OBR M-file for OBR.fig
%      OBR, by itself, creates a new
OBR or raises the existing
%      singleton*.
%
%      H = OBR returns the handle to
a new OBR or the handle to
%      the existing singleton*.
%
%
OBR('CALLBACK',hObject,eventData,han
dles,...) calls the local
%        function named CALLBACK in
OBR.M    with    the    given    input
arguments.
%
%        OBR('Property','Value',...)
creates a new OBR or raises the
%              existing  singleton*.
Starting  from  the  left,  property
value pairs
%      are
%         applied to the GUI before
OBR_OpeningFunction gets called.  An
%      unrecognized property name or
invalid    value    makes    property
application
%       stop.  All inputs are passed
to OBR_OpeningFcn via varargin.
%
%       *See GUI Options on GUIDE's
Tools menu.  Choose "GUI allows only
one
%      instance to run (singleton)".
%
%    See    also:    GUIDE,    GUIDATA,
GUIHANDLES

% Copyright 2002-2003 The MathWorks,
Inc.

% Edit the above text to modify the
response to help OBR

% Last Modified by GUIDE v2.5 23-
Oct-2009 14:20:06

% Begin initialization code - DO NOT
EDIT
gui_Singleton = 1;
gui_State    =    struct('gui_Name',
mfilename, ...
                   'gui_Singleton',
gui_Singleton, ...
                   'gui_OpeningFcn',
@OBR_OpeningFcn, ...
                   'gui_OutputFcn',
@OBR_OutputFcn, ...
                   'gui_LayoutFcn',
[] , ...
                   'gui_Callback',
[]);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback         =
str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}]         =
gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State,
varargin{:});
end
% End initialization code - DO NOT
EDIT


% --- Executes just before OBR is
made visible.
function     OBR_OpeningFcn(hObject,
eventdata, handles, varargin)
% This function has no output args,
see OutputFcn.
% hObject    handle to figure
%  eventdata    reserved  -  to  be
defined  in  a  future  version  of
MATLAB
% handles     structure with handles
and user data (see GUIDATA)
% varargin    command line arguments
to OBR (see VARARGIN)
set(handles.BrowseImage,'Enable','on
');
set(handles.CleanImage,'Enable','off
');
set(handles.DeskewImage,'Enable','of
f');
set(handles.SegmentImage,  'Enable',
'off');
set(handles.RecognizeImage,'Enable',
'off');
set(handles.ImageDisplay,'xcolor','w
','ycolor','w','xtick',[],'ytick',[]
);
set(handles.Deskew,'xcolor','w','yco
lor','w','xtick',[],'ytick',[]);
set(handles.Backside,'xcolor','w','y
color','w','xtick',[],'ytick',[]);
set(handles.Oneclick,'Enable','off')
;
```

```
% Choose default command line output
for OBR
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes OBR wait for user
response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are
returned to the command line.
function varargout =
OBR_OutputFcn(hObject, eventdata,
handles)
% varargout cell array for
returning output args (see
VARARGOUT);
% hObject handle to figure
% eventdata reserved - to be
defined in a future version of
MATLAB
% handles structure with handles
and user data (see GUIDATA)

% Get default command line output
from handles structure
varargout{1} = handles.output;


% --- Executes on button press in
BrailleContractions.
function
BrailleContractions_Callback(hObject
, eventdata, handles)
% hObject handle to
BrailleContractions (see GCBO)
% eventdata reserved - to be
defined in a future version of
MATLAB
% handles structure with handles
and user data (see GUIDATA)
%set(handles.contractionsmenu,'visib
le','on');
backgroundImage =
importdata('C:\MATLAB701\work\image\
Braille Contractions (new).jpg');
figure('Name','Braille
Alphabet','NumberTitle','off'),imsho
w(backgroundImage);

% --- Executes on button press in
ResetAll.
function ResetAll_Callback(hObject,
eventdata, handles)
% hObject handle to ResetAll (see
GCBO)
% eventdata reserved - to be
defined in a future version of
MATLAB
% handles structure with handles
and user data (see GUIDATA)
cla(handles.ImageDisplay,'reset');
cla(handles.Deskew,'reset');
cla(handles.Backside,'reset');
cla;
set(handles.text1,'String','Browsed
Image');
set(handles.BrowseImage,'Enable','on
');
set(handles.DeskewImage,'Enable','of
f');
set(handles.CleanImage,'Enable','off
');
set(handles.SegmentImage,'Enable','o
ff');
set(handles.RecognizeImage,'Enable',
'off');
set(handles.RecognizedImage,'String'
,' ');
set(handles.ImageDisplay,'xcolor','w
','ycolor','w','xtick',[],'ytick',[]
);
set(handles.Deskew,'xcolor','w','yco
lor','w','xtick',[],'ytick',[]);
set(handles.Backside,'xcolor','w','y
color','w','xtick',[],'ytick',[]);
set(handles.text1,'visible','on');
set(handles.Oneclick,'Enable','off')
;
set(handles.Angle,'String',' ');
clear

% --- Executes on button press in
BrailleAlphabet.
function
BrailleAlphabet_Callback(hObject,
eventdata, handles)
% hObject handle to
BrailleAlphabet (see GCBO)
% eventdata reserved - to be
defined in a future version of
MATLAB
% handles structure with handles
and user data (see GUIDATA)
backgroundImage =
importdata('C:\MATLAB701\work\image\
braille.jpg');
figure('Name','Braille
Alphabet','NumberTitle','off'),imsho
w(backgroundImage);

% --- Executes on button press in
BrowseImage.
function
BrowseImage_Callback(hObject,
eventdata, handles)
% hObject handle to BrowseImage
(see GCBO)
% eventdata reserved - to be
defined in a future version of
MATLAB
```

```
% handles     structure with handles
and user data (see GUIDATA)
set(handles.RecognizedImage,'String'
,' ');
[filename,        pathname]        =
uigetfile({'*.jpg';'*.bmp'},'Pick an
Image(.jpg,.bmp) File');
%[filename,        pathname]        =
uigetfile({'*.jpg';'*.jpeg';'*.bmp'}
,'File
%Selector');
if  ~ischar([pathname  filename])  ||
isempty([pathname filename])
else
info= imfinfo([pathname filename]);
I   =   imread([pathname,filename]);
%Place   the   image   file   to   the
variable I;
save I;
set(handles.text1,'String',info.File
name);
set(handles.Oneclick,'Enable','on');
set(handles.BrowseImage,'Enable','of
f');
set(handles.CleanImage,'Enable','off
');
set(handles.DeskewImage,'Enable','on
');
set(handles.SegmentImage,   'Enable',
'off');
set(handles.RecognizeImage,'Enable',
'off');
axes(handles.ImageDisplay)
image(I);
axis off
save info
end


% --- Executes on button press in
DeSkewImage.
function
DeSkewImage_Callback(hObject,
eventdata, handles)
% hObject     handle to DeSkewImage
(see GCBO)
% eventdata     reserved - to be
defined in a future version of
MATLAB
% handles     structure with handles
and user data (see GUIDATA)

load I;
cropedimage   =   I;   %copies   I   to
cropedimage
save cropedimage;
gray=imadjust(rgb2gray(cropedimage))
; %Grayscales cropedimage and then
increase its contrast
imwrite(gray,'C:\MATLAB701\work\dump
\gray.jpg'); %saves the image bw
imwrite(gray,'C:\MATLAB701\work\dump
\gray.jpg'); %saves the image bw
level=graythresh(gray);
bw=im2bw(gray,level);
imwrite(bw,'C:\MATLAB701\work\dump\b
w.jpg');
img=imread('C:\MATLAB701\work\dump\b
w.jpg');
img=double(img);
inv(:,:,:)=255-img(:,:,:);
inv=uint8(inv);
inv=im2bw(inv);
inv=bwareaopen(inv,30);
imwrite(inv,'C:\MATLAB701\work\dump\
inverse.jpg');

%rotated initalization
rotated = inv;
mediangray=gray;
save mediangray;
totalangle=0;

%Showing the skewed image with lines
L= bwlabel(rotated);
s = regionprops(L,'centroid');
figure('Name','Slopes:   Yellow   are
the
Centers','NumberTitle','off'),imshow
(rotated);
hold on
numObj = numel(s);

if numObj<3

ee='Insufficient Braille Dots'
cla(handles.ImageDisplay,'reset');
cla(handles.Deskew,'reset');
cla(handles.Backside,'reset');
cla;
set(handles.text1,'String','Browsed
Image');
set(handles.BrowseImage,'Enable','on
');
set(handles.DeskewImage,'Enable','of
f');
set(handles.CleanImage,'Enable','off
');
set(handles.SegmentImage,'Enable','o
ff');
set(handles.RecognizeImage,'Enable',
'off');
set(handles.RecognizedImage,'String'
,ee);
set(handles.ImageDisplay,'xcolor','w
','ycolor','w','xtick',[],'ytick',[]
);
set(handles.Deskew,'xcolor','w','yco
lor','w','xtick',[],'ytick',[]);
set(handles.Backside,'xcolor','w','y
color','w','xtick',[],'ytick',[]);
set(handles.text1,'visible','on');
set(handles.Oneclick,'Enable','off')
;
set(LOADING,'visible','off');
clear
```

```
else

for k = 1 :  numObj
    plot(s(k).Centroid(1),
s(k).Centroid(2), 'y*');
    plot(s(k).Centroid(1),
s(k).Centroid(2), 'yo');
    for i=1 : numObj
        if         (s(i).Centroid(1)-
s(k).Centroid(1)) >0
    slope=double((s(i).Centroid(2)-
s(k).Centroid(2))/(s(i).Centroid(1)-
s(k).Centroid(1)));
    distance                          =
sqrt((s(i).Centroid(1)-
s(k).Centroid(1))^2+(s(i).Centroid(2
)-s(k).Centroid(2))^2);
    format long
    slope;
             if  ((slope  <  0.1)  &&
(slope > -0.1)) && distance <150

plot([s(k).Centroid(1)
s(i).Centroid(1)],  [s(k).Centroid(2)
s(i).Centroid(2)],'r');
             end
        end
    end
end

hold off;


%Loop start and angle extraction
loop=1;
angle=1;
z=0;
while z<=5

L= bwlabel(rotated);
s = regionprops(L,'centroid');

%get total number of accepted slopes
ctrr=0;
numObj = numel(s);
for ctr = 1 :  numObj
    for ctr2=1 : numObj
        if      (s(ctr2).Centroid(1)-
s(ctr).Centroid(1))>0

slope=double((s(ctr2).Centroid(2)-
s(ctr).Centroid(2))/(s(ctr2).Centroi
d(1)-s(ctr).Centroid(1)));
    distance                          =
sqrt((s(ctr2).Centroid(1)-
s(ctr).Centroid(1))^2+(s(ctr2).Centr
oid(2)-s(ctr).Centroid(2))^2);
    format long
    slope;
           if loop==1
             if  ((slope  <  0.1)  &&
(slope > -0.1)) && distance <150
                ctrr=ctrr+1;
             end
           else
             if  ((slope  <  0.05)  &&
(slope > -0.05)) && distance <150
                ctrr=ctrr+1;
             end
           end
        end
    end
end

%Declaration of array and pointer
array =1:ctrr;
point=0;

%Extraction of the useful slopes
hold on
numObj = numel(s);
for k = 1 :  numObj
    for i=1 : numObj
        if         (s(i).Centroid(1)-
s(k).Centroid(1)) >0
    slope=double((s(i).Centroid(2)-
s(k).Centroid(2))/(s(i).Centroid(1)-
s(k).Centroid(1)));
    distance                          =
sqrt((s(i).Centroid(1)-
s(k).Centroid(1))^2+(s(i).Centroid(2
)-s(k).Centroid(2))^2);
    format long
    slope;
          if loop==1
             if  ((slope  <  0.1)  &&
(slope > -0.1)) && distance <150
                point=point+1;

array(:,point)=slope;

%plot([s(k).Centroid(1)
s(i).Centroid(1)],  [s(k).Centroid(2)
s(i).Centroid(2)],'r');
             end
           else
             if  ((slope  <  0.05)  &&
(slope > -0.05)) && distance <150
                point=point+1;

array(:,point)=slope;
             end
           end
        end
    end
end
hold off;

array;
medianave=median(array);

%Deskewing
angle=atan(medianave);
angle=(angle*180)/pi;
rotated= imrotate(rotated, angle);
```

```
mediangray=imrotate(mediangray,
angle);
save mediangray;
imwrite(mediangray,'C:\MATLAB701\wor
k\dump\mediangray.jpg');
imwrite(rotated,'C:\MATLAB701\work\d
ump\Rotated.jpg');

clear array;
loop=loop+1;
totalangle=totalangle+angle;
z=z+1;
end

%final display label
figure('Name','Deskew             nth
run','NumberTitle','off'),imshow(rot
ated);
L = bwlabel(rotated);
s = regionprops(L,'centroid');
hold on
numObj = numel(s);
for k = 1 :  numObj
    plot(s(k).Centroid(1),
s(k).Centroid(2), 'y*');
    plot(s(k).Centroid(1),
s(k).Centroid(2), 'yo');

    %backside filtering
    x=s(k).Centroid(1);
    y=s(k).Centroid(2)-7.5;
    plot(x,y, 'b*');
    p=impixel(mediangray,x,y);

    for i=1 : numObj
        if         (s(i).Centroid(1)-
s(k).Centroid(1))>0
    slope=double((s(i).Centroid(2)-
s(k).Centroid(2))/(s(i).Centroid(1)-
s(k).Centroid(1)));
    distance                        =
sqrt((s(i).Centroid(1)-
s(k).Centroid(1))^2+(s(i).Centroid(2
)-s(k).Centroid(2))^2);
    format long

            if loop==1
                if ((slope < 0.1) &&
(slope > -0.1)) && distance <150

plot([s(k).Centroid(1)
s(i).Centroid(1)],  [s(k).Centroid(2)
s(i).Centroid(2)],'r');
                end
            else
                if  ((slope  <  0.05)
&& (slope > -0.05)) && distance <150

plot([s(k).Centroid(1)
s(i).Centroid(1)],  [s(k).Centroid(2)
s(i).Centroid(2)],'r');
                end
            end
        end
    end
end
hold off;


%dis1=mediangray;
%dis1=im2bw(dis1);

dis1=imread('C:\MATLAB701\work\dump\
mediangray.jpg');
dis1 =cat (3, dis1 ,dis1 ,dis1);
axes(handles.Deskew);
image(dis1);
axis off;

totalangle
set(handles.Angle,'String',totalangl
e);

set(handles.BrowseImage,'Enable','of
f');
set(handles.CleanImage,'Enable','on'
);
set(handles.DeskewImage,'Enable','of
f');
set(handles.SegmentImage,  'Enable',
'off');
set(handles.RecognizeImage,'Enable',
'off');
end

% --- Executes on button press in
RecognizeImage.
function
RecognizeImage_Callback(hObject,
eventdata, handles)
%   hObject             handle   to
RecognizeImage (see GCBO)
%  eventdata    reserved  -  to  be
defined  in  a  future  version  of
MATLAB
% handles     structure with handles
and user data (see GUIDATA)

set(LOADING,'Visible','on')


load charcount;
charcount

if charcount==0
ee='No Character Detected'
set(LOADING,'Visible','off')
set(handles.RecognizedImage,'String'
,ee);
set(handles.BrowseImage,'Enable','on
');
set(handles.CleanImage,'Enable','off
');
set(handles.DeskewImage,'Enable','of
f');
```

```
set(handles.SegmentImage,   'Enable',
'off');
set(handles.RecognizeImage,'Enable',
'off');


elseif charcount>=1
load binary;
binary

%for u = 1 : charcount;
u=1
while(u<=charcount)
    z=num2str(binary(1,u));
    b=[z];
    for o = 2 : 6
    a=binary(o,u);
    str1=num2str(a);
    b=[b str1];
    end
    db(u,:)=strvcat(b);
    u=u+1;
end

db

    db1='123456';
    conn  =  database('Braille',  '',
'');

setdbprefs('DataReturnFormat','cella
rray');
    curs   =   exec(conn,   ['select
Letter from Grade1 where ID= ' ''''
db1 '''']);
    curs = fetch(curs);
    bb = curs.data;

    db1='000000';
    conn  =  database('Braille',  '',
'');

setdbprefs('DataReturnFormat','cella
rray');
    curs   =   exec(conn,   ['select
Letter from Grade1 where ID= ' ''''
db1 '''']);
    curs = fetch(curs);
    space = curs.data;

end

if charcount>=1
set(handles.RecognizedImage,'String'
,'Loading...');
db1=[db(1,:)]
met=1;
epox=1;
while(epox<=charcount)
    epox
    lft=0;
    rgt=0;
    if epox<charcount && charcount>1
    db2=[db1 db(epox+1,:) '%']
    elseif charcount==1
    db2='12345'
    end
    conn  =  database('Braille',  '',
'');

setdbprefs('DataReturnFormat','cella
rray')
    curs = exec(conn, ['select Words
from Grade2 where ID like ' '''' db2
''''])
    curs = fetch(curs)
    aa = curs.data

    %Empty query checker
    n = numel(curs.data)
    b=0;
    if n==1
    b = strcmp(bb,aa)
    end
    %left & right checker
    if epox+1<charcount

rgt=strcmp(db(epox+1,:),'000000')
    end
    if  epox-met>=1%error  catcher  if
ctr-met will equal to 0
    lft=strcmp(db(epox-
met,:),'000000')
    end

    %query  for  standalone  grade  2
braille  characters(group)  for  the
first
    %including the first char
    if  (b  ==  1  &&  n  ==  1)  &&  rgt==1
&& met>1 && epox-met<1
        curs  =  exec(conn,  ['select
Words  from  Grade2  where  ID=  '  ''''
db1 '''']);
        curs = fetch(curs)
        aa = curs.data
        n = numel(curs.data)
        b=strcmp(bb,aa)
        arrayy(epox) = aa

        spaceb=met-1
        while spaceb>=1
        arrayy(epox-spaceb)=space;
        spaceb=spaceb-1;
        end

        db1=db(epox+1,:)
        met=1
        epox=epox+1;

     %Sure standalone
     elseif  (b  ==  1  &&  n  ==  1)  &&
lft==1 &&  rgt==1 && met>1
        curs  =  exec(conn,  ['select
Words  from  Grade2  where  ID=  '  ''''
db1 '''']);
```

```
        curs = fetch(curs)
        aa = curs.data
        n = numel(curs.data)
        b=strcmp(bb,aa)
        arrayy(epox) = aa

        spaceb=met-1
        while spaceb>=1
        arrayy(epox-spaceb)=space;
        spaceb=spaceb-1;
        end

        db1=db(epox+1,:)
        met=1
        epox=epox+1;


      %Prefix
      elseif (b == 1 && n == 1) &&
lft==1 &&  rgt==0  && met>1
         curs = exec(conn, ['select
Words from Pre_fix where ID= ' ''''
db1 ''''])
        curs = fetch(curs)
        aa = curs.data
        n = numel(curs.data)
        b=strcmp(bb,aa)

        if  (b == 1 && n == 1)
        ppp=met-1
        while ppp>=1
        epox=epox-ppp
        db1=db(epox,:)
        curs = exec(conn, ['select
Letter from Grade1 where ID= ' ''''
db1 ''''])
        curs = fetch(curs)
        aa = curs.data
        n = numel(curs.data)
        b=strcmp(bb,aa)
        arrayy(epox) = aa
        db1=db(epox+1,:)
        met=1
        epox=epox+ppp;
        ppp=ppp-1;
        end

        else
        curs = exec(conn, ['select
Words from Pre_fix where ID= ' ''''
db1 ''''])
        curs = fetch(curs)
        aa = curs.data
        n = numel(curs.data)
        b=strcmp(bb,aa)
        arrayy(epox) = aa

        spaceb=met-1
        while spaceb>=1
        arrayy(epox-spaceb)=space;
        spaceb=spaceb-1;
        end

        db1=db(epox+1,:)
        met=1
        epox=epox+1;
        end

      %Suffix
      elseif (b == 1 && n == 1) &&
lft==0 &&  rgt==1  && met>1

        curs = exec(conn, ['select
Words from Suffix where ID= ' ''''
db1 ''''])
        curs = fetch(curs)
        aa = curs.data
        n = numel(curs.data)
        b=strcmp(bb,aa)

        if  (b == 1 && n == 1)
        ppp=met-1
        while ppp>=1
        epox=epox-ppp
        db1=db(epox,:)
        curs = exec(conn, ['select
Letter from Grade1 where ID= ' ''''
db1 ''''])
        curs = fetch(curs)
        aa = curs.data
        n = numel(curs.data)
        b=strcmp(bb,aa)
        arrayy(epox) = aa
        db1=db(epox+1,:)
        met=1
        epox=epox+ppp;
        ppp=ppp-1;
        end

        else
        curs = exec(conn, ['select
Words from Suffix where ID= ' ''''
db1 ''''])
        curs = fetch(curs)
        aa = curs.data
        n = numel(curs.data)
        b=strcmp(bb,aa)
        arrayy(epox) = aa

        spaceb=met-1
        while spaceb>=1
        arrayy(epox-spaceb)=space;
        spaceb=spaceb-1;
        end

        db1=db(epox+1,:)
        met=1
        epox=epox+1;
        end


       %Midfix
       elseif (b == 1 && n == 1) &&
lft==0 &&  rgt==0  && met>1
```

```
        curs  =  exec(conn,  ['select
Words from Mid_fix where ID= ' ''''
db1 ''''])
        curs = fetch(curs)
        aa = curs.data
        n = numel(curs.data)
        b=strcmp(bb,aa)

        if  (b == 1 && n == 1)
        ppp=met-1
        while ppp>=1
        epox=epox-ppp
        db1=db(epox,:)
        curs  =  exec(conn,  ['select
Letter from Grade1 where ID= ' ''''
db1 ''''])
        curs = fetch(curs)
        aa = curs.data
        n = numel(curs.data)
        b=strcmp(bb,aa)
        arrayy(epox) = aa
        db1=db(epox+1,:)
        met=1
        epox=epox+ppp;
        ppp=ppp-1;
        end

        else
        curs  =  exec(conn,  ['select
Words from Mid_fix where ID= ' ''''
db1 ''''])
        curs = fetch(curs)
        aa = curs.data
        n = numel(curs.data)
        b=strcmp(bb,aa)
        arrayy(epox) = aa

        spaceb=met-1
        while spaceb>=1
        arrayy(epox-spaceb)=space;
        spaceb=spaceb-1;
        end

        db1=db(epox+1,:)
        met=1
        epox=epox+1;
        end


        %try
    elseif  (b==1  &&  n==1)&&  met==1
&& lft==1 && rgt==1
        curs  =  exec(conn,  ['select
Words  from  Stand_Alone  where  ID=  '
'''' db1 ''''])
        curs = fetch(curs)
        aa = curs.data
        n = numel(curs.data)
        b=strcmp(bb,aa)
        arrayy(epox) = aa
        if epox<charcount
        db1=db(epox+1,:)
        end
        met=1
        epox=epox+1;

    elseif  (b==1  &&  n==1)&&  met==1
&& lft==1 && epox+1>charcount
        curs  =  exec(conn,  ['select
Words  from  Stand_Alone  where  ID=  '
'''' db1 ''''])
        curs = fetch(curs)
        aa = curs.data
        n = numel(curs.data)
        b=strcmp(bb,aa)
        arrayy(epox) = aa
        if epox<charcount
        db1=db(epox+1,:)
        end
        met=1
        epox=epox+1;

     elseif  (b==1  &&  n==1)&&  met==1
&& rgt==1 && epox-met<1
        curs  =  exec(conn,  ['select
Words  from  Stand_Alone  where  ID=  '
'''' db1 ''''])
        curs = fetch(curs)
        aa = curs.data
        n = numel(curs.data)
        b=strcmp(bb,aa)
        arrayy(epox) = aa
        if epox<charcount
        db1=db(epox+1,:)
        end
        met=1
        epox=epox+1;

        elseif   (b==1   &&   n==1)&&
met==1  &&  epox+1>charcount  &&  epox-
met<1
        curs  =  exec(conn,  ['select
Words  from  Stand_Alone  where  ID=  '
'''' db1 ''''])
        curs = fetch(curs)
        aa = curs.data
        n = numel(curs.data)
        b=strcmp(bb,aa)
        arrayy(epox) = aa
        if epox<charcount
        db1=db(epox+1,:)
        end
        met=1
        epox=epox+1;
%not sure



    %query for grade 1 braille(since
grade  2  1  braille  char  standalone
are unique)
    elseif    (b==1  &&  n  ==  1)  &&
met==1
        curs  =  exec(conn,  ['select
Letter from Grade1 where ID= ' ''''
db1 ''''])
```

```
        curs = fetch(curs)
        aa = curs.data
        n = numel(curs.data)
        b=strcmp(bb,aa)

        %query for grade 2 single
braille char standalone(condition is
if
        %there's no match in the
grade 1 DB)
        if (b == 1 && n == 1)
        curs = exec(conn, ['select
Words from Grade2 where ID= ' '''' 
db1 ''''])
        curs = fetch(curs)
        aa = curs.data
        n = numel(curs.data)
        b=strcmp(bb,aa)

            if n>1 && epox-met<1 &&
rgt==0
                curs = exec(conn,
['select Words from Pre_Fix where
ID= ' '''' db1 ''''])
                curs = fetch(curs)
                aa = curs.data
                n = numel(curs.data)
                b=strcmp(bb,aa)

            elseif n>1 && lft==0 &&
rgt==1 && epox==1
                curs = exec(conn,
['select Words from Stand_Alone
where ID= ' '''' db1 ''''])
                curs = fetch(curs)
                aa = curs.data
                n = numel(curs.data)
                b=strcmp(bb,aa)

            elseif n>1 && lft==0 &&
rgt==0
                curs = exec(conn,
['select Words from Mid_Fix where
ID= ' '''' db1 ''''])
                curs = fetch(curs)
                aa = curs.data
                n = numel(curs.data)
                b=strcmp(bb,aa)

            elseif n>1 && lft==0 &&
rgt==1
                curs = exec(conn,
['select Words from Suffix where ID=
' '''' db1 ''''])
                curs = fetch(curs)
                aa = curs.data
                n = numel(curs.data)
                b=strcmp(bb,aa)

            elseif n>1 && lft==1 &&
rgt==0
                curs = exec(conn,
['select Words from Pre_Fix where
ID= ' '''' db1 ''''])
                curs = fetch(curs)
                aa = curs.data
                n = numel(curs.data)
                b=strcmp(bb,aa)

            elseif n>1 && lft==1 &&
rgt==1
                curs = exec(conn,
['select Words from Stand_Alone
where ID= ' '''' db1 ''''])
                curs = fetch(curs)
                aa = curs.data
                n = numel(curs.data)
                b=strcmp(bb,aa)

            end
        end

        %proceed here if match is
already found
        arrayy(epox) = aa
        if epox<charcount
        db1=db(epox+1,:)
        end
        met=1
        epox=epox+1;

    else
    %db1 incrementation if there's
no match in any of the database
    if epox+1<=charcount
    db1=[db1 db(epox+1,:)]
    met=met+1
    epox=epox+1;

    %trial
    else
        curs = exec(conn, ['select
Words from Grade2 where ID= ' '''' 
db1 ''''])
        curs = fetch(curs)
        aa = curs.data
        n = numel(curs.data)
        b=strcmp(bb,aa)
        arrayy(epox) = aa

        spaceb=met-1
        while spaceb>=1
        arrayy(epox-spaceb)=space;
        spaceb=spaceb-1;
        end

        epox=epox+1
    end
    %trial
end
    close(conn)
    close(curs)
```

```
end

result = [arrayy(:)]

ee=strvcat(arrayy(1));
for cc = 2 : charcount
ff=strvcat(arrayy(cc));
ee=[ee ff];
end
ee

set(LOADING,'Visible','off')
set(handles.RecognizedImage,'String'
,ee);


set(handles.BrowseImage,'Enable','on
');
set(handles.CleanImage,'Enable','off
');
set(handles.DeskewImage,'Enable','of
f');
set(handles.SegmentImage,  'Enable',
'off');
set(handles.RecognizeImage,'Enable',
'off');
end

% --- Executes on button press in
AnalyzeImage.
function
AnalyzeImage_Callback(hObject,
eventdata, handles)
% hObject    handle to AnalyzeImage
(see GCBO)
% eventdata    reserved - to be
defined in a future version of
MATLAB
% handles    structure with handles
and user data (see GUIDATA)



% --- Executes on button press in
Exit.
function      Exit_Callback(hObject,
eventdata, handles)
% hObject      handle to Exit (see
GCBO)
% eventdata    reserved - to be
defined in a future version of
MATLAB
% handles    structure with handles
and user data (see GUIDATA)
close

%handles.output = hObject;
%if             ispc             &&
isequal(get(hObject,'BackgroundColor
'),
get(0,'defaultUicontrolBackgroundCol
or'))
%
set(hObject,'BackgroundColor','white
');
%end

% --- Executes when figure1 is
resized.
function  figure1_ResizeFcn(hObject,
eventdata, handles)
% hObject    handle to figure1 (see
GCBO)
% eventdata    reserved - to be
defined in a future version of
MATLAB
% handles    structure with handles
and user data (see GUIDATA)


% --- Executes on button press in
pushbutton16.
function
pushbutton16_Callback(hObject,
eventdata, handles)
% hObject    handle to pushbutton16
(see GCBO)
% eventdata    reserved - to be
defined in a future version of
MATLAB
% handles    structure with handles
and user data (see GUIDATA)

%erasing
load mediangray;
for po =1 : 3
rotated=imread('C:\MATLAB701\work\du
mp\rotated.jpg');
rotated=uint8(rotated);
rotated=im2bw(rotated);

%figure('Name','Erasing','NumberTitl
e','off'),imshow(mediangray);
hold on
L = bwlabel(rotated);
s = regionprops(L,'centroid');
numObj = numel(s);
for k = 1 :  numObj

    %backside filtering
    x=s(k).Centroid(1);
    y=s(k).Centroid(2);
    x2=s(k).Centroid(1)-4;
    y2=s(k).Centroid(2)-7;
    plot(x2,y2, 'b.');
    p=impixel(mediangray,x2,y2);
    if p(1,1)<=210
       plot(s(k).Centroid(1),
s(k).Centroid(2),'g*');
       plot(s(k).Centroid(1),
s(k).Centroid(2),'go');
        I2                          =
imread('C:\MATLAB701\work\dump\media
ngray.jpg');
```

```
        K                           =
imread('C:\MATLAB701\work\dump\rotat
ed.jpg');
        col  =  [(s(k).Centroid(1)-7)
(s(k).Centroid(1)+7)
(s(k).Centroid(1)+7)
(s(k).Centroid(1)-7)];
        row  =  [(s(k).Centroid(2)-7)
(s(k).Centroid(2)-7)
(s(k).Centroid(2)+7)
(s(k).Centroid(2)+7)];
        J = roifill(I2,col,row);
        M = roifill(K,col,row);

imwrite(J,'C:\MATLAB701\work\dump\me
diangray.jpg');

imwrite(M,'C:\MATLAB701\work\dump\ro
tated.jpg');
        rotated=M;
        mediangray=J;
        save mediangray;
    end

end
hold off;
end

dis2=imread('C:\MATLAB701\work\dump\
mediangray.jpg');
dis2 =cat (3, dis2 ,dis2 ,dis2);
axes(handles.Backside);
image(dis2);
axis off;



set(handles.BrowseImage,'Enable','of
f');
set(handles.CleanImage,'Enable','off
');
set(handles.DeskewImage,'Enable','of
f');
set(handles.SegmentImage,   'Enable',
'on');
set(handles.RecognizeImage,'Enable',
'off');



% --- Executes on button press in
SegmentImage.
function
SegmentImage_Callback(hObject,
eventdata, handles)
% hObject     handle to SegmentImage
(see GCBO)
%  eventdata     reserved  -  to  be
defined  in  a  future  version  of
MATLAB
% handles     structure with handles
and user data (see GUIDATA)
%segmentation

rotated=imread('C:\MATLAB701\work\du
mp\rotated.jpg');
rotated=uint8(rotated);
rotated=im2bw(rotated);
figure('Name','Segments','NumberTitl
e','off'),imshow(rotated);

L = bwlabel(rotated);
s = regionprops(L,'centroid');
hold on
numObj = numel(s);

if numObj>0
%xholder
xholder=1:numObj;
for iz = 1 : numObj
xholder(:,iz)=s(iz).Centroid(1);
format short;
end
xholder

%yholder
yholder=1:numObj;
for iz = 1 : numObj
yholder(:,iz)=s(iz).Centroid(2);
format short;
end
yholder

%xyholder
xyholder=[1:numObj;1:numObj];
for iz = 1 : numObj
xyholder(1,iz)=s(iz).Centroid(1);
xyholder(2,iz)=s(iz).Centroid(2);
format short;
end
xyholder



y1=-999;
y2=-999;
y3=-999;

for pt=1 : numObj
    y2=(yholder(:,pt));

    for pt2=1 : numObj
        if  yholder(:,pt2)>=y2-22  &&
yholder(:,pt2)<y2-9
        y1=yholder(:,pt2);
        break;
        end
    end

    for pt2=1 : numObj
        if  yholder(:,pt2)>=y2+9  &&
yholder(:,pt2)<y2+22
        y3=yholder(:,pt2);
        break;
        end
    end
```

```
if y3>=y2+9 && y3<y2+22 && y1>=y2-22
&& y1<y2-9
break;
end

end


%charcount
charcount=1;
pt=0;
firstx=xholder(:,1);
secondx=0;
for pt=1 : numObj
    save charcount;
        if  (xholder(:,pt)>=firstx+16
&&    xholder(:,pt)<firstx+21)     &&
(yholder(:,pt)>=y1-4               &&
yholder(:,pt)<=y3+4)
            secondx=xholder(:,pt);
            break;
        elseif
(xholder(:,pt)>=firstx+21          &&
xholder(:,pt)<=firstx+32)          &&
(yholder(:,pt)>=y1-4               &&
yholder(:,pt)<=y3+14)
            secondx=firstx;
            firstx=secondx-20;
            break;
         elseif
(xholder(:,pt)>=firstx+47          &&
xholder(:,pt)<=firstx+51)          &&
(yholder(:,pt)>=y1-4               &&
yholder(:,pt)<=y3+4)
            secondx=firstx+20;
            break;
         %subject to change
         elseif
(xholder(:,pt)>=firstx+74          &&
xholder(:,pt)<=firstx+83)          &&
(yholder(:,pt)>=y1-4               &&
yholder(:,pt)<=y3+4)
            secondx=firstx;
            firstx=secondx-20;
            break;
         elseif
(xholder(:,pt)>=firstx+96          &&
xholder(:,pt)<=firstx+102)         &&
(yholder(:,pt)>=y1-4               &&
yholder(:,pt)<=y3+4)
            secondx=firstx+20;
            break;
          elseif
(xholder(:,pt)>=firstx+113         &&
xholder(:,pt)<=firstx+124)         &&
(yholder(:,pt)>=y1-4               &&
yholder(:,pt)<=y3+4)
            secondx=firstx+20;
            break;
         %end subject to change
        else
            firstx=xholder(:,pt);
        end
end

plot(firstx,80,'g*');
plot(secondx,80,'g*');
plot([firstx secondx], [80 80],'b');

 for i = 1 : 50

    for pt=1:numObj
        if
(xholder(:,pt)>=secondx+21         &&
xholder(:,pt)<secondx+34)          &&
(yholder(:,pt)>=y1-4               &&
yholder(:,pt)<=y3+4)
            firstx=xholder(:,pt);
            for pt2=1:numObj
                if
(xholder(:,pt2)>=firstx+16         &&
xholder(:,pt2)<firstx+21)          &&
(yholder(:,pt)>=y1-4               &&
yholder(:,pt)<=y3+4)

secondx=xholder(:,pt2);
                    break;
                else

secondx=firstx+20;
                    break;
                end
            end
            charcount=charcount+1;
            break;
        elseif
(xholder(:,pt)>=secondx+44         &&
xholder(:,pt)<secondx+53)          &&
(yholder(:,pt)>=y1-4               &&
yholder(:,pt)<=y3+4)
            secondx=xholder(:,pt);
            firstx=secondx-20;
            charcount=charcount+1;
            break;
        elseif
(xholder(:,pt)>=secondx+74         &&
xholder(:,pt)<secondx+83)          &&
(yholder(:,pt)>=y1-4               &&
yholder(:,pt)<=y3+4)
            firstx=xholder(:,pt);
            for pt2=1:numObj
                if
(xholder(:,pt2)>=firstx+16         &&
xholder(:,pt2)<firstx+21)          &&
(yholder(:,pt)>=y1-4               &&
yholder(:,pt)<=y3+4)

secondx=xholder(:,pt2);
                    break;
                else

secondx=firstx+20;
                    break;
                end
            end
            charcount=charcount+2;
```

```
            break;
        elseif
(xholder(:,pt)>=secondx+96          &&
xholder(:,pt)<secondx+102)          &&
(yholder(:,pt)>=y1-4                &&
yholder(:,pt)<=y3+4)

            secondx=xholder(:,pt);
            firstx=secondx-20;
            charcount=charcount+2;
            break;

        end
    end
plot(firstx,80,'g*');
plot(secondx,80,'g*');
plot([firstx secondx], [80 80],'b');
end

charcount
save charcount;



y1
y2
y3

%test

binary=[1:charcount  ;  1:charcount  ;
1:charcount    ;    1:charcount    ;
1:charcount ; 1:charcount];
for cl=1:6
    for cl2=1:charcount
        binary(cl,cl2)=0;
    end
end

marker=1;
firstx=xholder(:,1);
secondx=0;
for pt=1 : numObj
        if  (xholder(:,pt)>=firstx+16
&&    xholder(:,pt)<firstx+21)     &&
(yholder(:,pt)>=y1-4                &&
yholder(:,pt)<=y3+4)
           secondx=xholder(:,pt);
           break;

        elseif
(xholder(:,pt)>=firstx+21           &&
xholder(:,pt)<=firstx+32)           &&
(yholder(:,pt)>=y1-4                &&
yholder(:,pt)<=y3+4)
            secondx=firstx;
            firstx=secondx-20;
            break;

        elseif
(xholder(:,pt)>=firstx+47           &&
xholder(:,pt)<=firstx+51)           &&
(yholder(:,pt)>=y1-4                &&
yholder(:,pt)<=y3+4)
            secondx=firstx+20;
            break;

        else
            firstx=xholder(:,pt);
        end
end

                    for  pt2  =  1  :
numObj
                    if
xyholder(1,pt2)>=firstx-10          &&
xyholder(1,pt2)<=firstx+10
                    if
xyholder(2,pt2)>=y1-8               &&
xyholder(2,pt2)<=y1+8

binary(1,marker)=1;
                    elseif
xyholder(2,pt2)>=y2-8               &&
xyholder(2,pt2)<=y2+8

binary(2,marker)=1;
                    elseif
xyholder(2,pt2)>=y3-8               &&
xyholder(2,pt2)<=y3+8

binary(3,marker)=1;
                    end
                    end
                    end

                    for  pt2  =  1  :
numObj
                    if
xyholder(1,pt2)>=secondx-10         &&
xyholder(1,pt2)<=secondx+10
                    if
xyholder(2,pt2)>=y1-8               &&
xyholder(2,pt2)<=y1+4

binary(4,marker)=1;
                    elseif
xyholder(2,pt2)>=y2-8               &&
xyholder(2,pt2)<=y2+8

binary(5,marker)=1;
                    elseif
xyholder(2,pt2)>=y3-8               &&
xyholder(2,pt2)<=y3+8

binary(6,marker)=1;
                    end
                    end
                    end



for i = 1 : 50
    for pt=1:numObj
```

```
        if
(xholder(:,pt)>=secondx+21         &&
xholder(:,pt)<secondx+32)          &&
(yholder(:,pt)>=y1-4               &&
yholder(:,pt)<=y3+4)
            firstx=xholder(:,pt);

            for pt2=1:numObj
                if
(xholder(:,pt2)>=firstx+16         &&
xholder(:,pt2)<firstx+21)          &&
(yholder(:,pt)>=y1-4               &&
yholder(:,pt)<=y3+4)

secondx=xholder(:,pt2);
                    break;
                else

secondx=firstx+20;
                    break;
                end
            end
            marker=marker+1;
            break;

        elseif
(xholder(:,pt)>=secondx+47         &&
xholder(:,pt)<secondx+51)          &&
(yholder(:,pt)>=y1-4               &&
yholder(:,pt)<=y3+4)
            secondx=xholder(:,pt);
            firstx=secondx-20;
            marker=marker+1;
            break;

        elseif
(xholder(:,pt)>=secondx+74         &&
xholder(:,pt)<secondx+83)          &&
(yholder(:,pt)>=y1-4               &&
yholder(:,pt)<=y3+4)
            firstx=xholder(:,pt);

            for pt2=1:numObj
                if
(xholder(:,pt2)>=firstx+16         &&
xholder(:,pt2)<firstx+21)          &&
(yholder(:,pt)>=y1-4               &&
yholder(:,pt)<=y3+4)

secondx=xholder(:,pt2);
                break;

                else
                secondx=firstx+20;
                break;
                end
            end
            marker=marker+2;
            break;

        elseif
(xholder(:,pt)>=secondx+96         &&
xholder(:,pt)<secondx+102)         &&
(yholder(:,pt)>=y1-4               &&
yholder(:,pt)<=y3+4)

secondx=xholder(:,pt);
                firstx=secondx-20;
                marker=marker+2;
            break;

                    end

    end


                    for  pt2  =  1  :
numObj
                    if
xyholder(1,pt2)>=firstx-10         &&
xyholder(1,pt2)<=firstx+10
                    if
xyholder(2,pt2)>=y1-8              &&
xyholder(2,pt2)<=y1+8

binary(1,marker)=1;
                    elseif
xyholder(2,pt2)>=y2-8              &&
xyholder(2,pt2)<=y2+8

binary(2,marker)=1;
                    elseif
xyholder(2,pt2)>=y3-8              &&
xyholder(2,pt2)<=y3+8

binary(3,marker)=1;
                    end
                    end
                    end

                    for  pt2  =  1  :
numObj
                    if
xyholder(1,pt2)>=secondx-10        &&
xyholder(1,pt2)<=secondx+10
                    if
xyholder(2,pt2)>=y1-8              &&
xyholder(2,pt2)<=y1+8

binary(4,marker)=1;
                    elseif
xyholder(2,pt2)>=y2-8              &&
xyholder(2,pt2)<=y2+8

binary(5,marker)=1;
                    elseif
xyholder(2,pt2)>=y3-8              &&
xyholder(2,pt2)<=y3+8

binary(6,marker)=1;
                    end
                    end
                    end

save binary;
end
```

```
elseif numObj==0
charcount=0
save charcount;
end

set(handles.BrowseImage,'Enable','of
f');
set(handles.CleanImage,'Enable','off
');
set(handles.DeskewImage,'Enable','of
f');
set(handles.SegmentImage,   'Enable',
'off');
set(handles.RecognizeImage,'Enable',
'on');



% --- Executes on button press in
Oneclick.
function   Oneclick_Callback(hObject,
eventdata, handles)
% hObject    handle to Oneclick (see
GCBO)
%  eventdata     reserved  -  to  be
defined  in  a  future  version  of
MATLAB
% handles     structure with handles
and user data (see GUIDATA)


set(LOADING,'Visible','on')


load I;
cropedimage   =   I;   %copies   I   to
cropedimage
save cropedimage;
gray=imadjust(rgb2gray(cropedimage))
; %Grayscales cropedimage and then
increase its contrast
imwrite(gray,'C:\MATLAB701\work\dump
\gray.jpg'); %saves the image bw
imwrite(gray,'C:\MATLAB701\work\dump
\gray.jpg'); %saves the image bw
level=graythresh(gray);
bw=im2bw(gray,level);
imwrite(bw,'C:\MATLAB701\work\dump\b
w.jpg');
img=imread('C:\MATLAB701\work\dump\b
w.jpg');
img=double(img);
inv(:,:,:)=255-img(:,:,:);
inv=uint8(inv);
inv=im2bw(inv);
inv=bwareaopen(inv,30);
imwrite(inv,'C:\MATLAB701\work\dump\
inverse.jpg');

%rotated initalization
rotated = inv;
mediangray=gray;
save mediangray;
totalangle=0;

%Showing the skewed image with lines
L= bwlabel(rotated);
s = regionprops(L,'centroid');
%figure('Name','Slopes','NumberTitle
','off'),imshow(rotated);
%title('Yellow are the centers');
hold on
numObj = numel(s);

if numObj<3

ee='Insufficient Braille Dots'
cla(handles.ImageDisplay,'reset');
cla(handles.Deskew,'reset');
cla(handles.Backside,'reset');
cla;
set(handles.text1,'String','Browsed
Image');
set(handles.BrowseImage,'Enable','on
');
set(handles.DeskewImage,'Enable','of
f');
set(handles.CleanImage,'Enable','off
');
set(handles.SegmentImage,'Enable','o
ff');
set(handles.RecognizeImage,'Enable',
'off');
set(handles.RecognizedImage,'String'
,ee);
set(handles.ImageDisplay,'xcolor','w
','ycolor','w','xtick',[],'ytick',[]
);
set(handles.Deskew,'xcolor','w','yco
lor','w','xtick',[],'ytick',[]);
set(handles.Backside,'xcolor','w','y
color','w','xtick',[],'ytick',[]);
set(handles.text1,'visible','on');
set(handles.Oneclick,'Enable','off')
;
set(LOADING,'visible','off');
clear



else
for k = 1 :  numObj
    for i=1 : numObj
        if         (s(i).Centroid(1)-
s(k).Centroid(1)) >0
    slope=double((s(i).Centroid(2)-
s(k).Centroid(2))/(s(i).Centroid(1)-
s(k).Centroid(1)));
    distance                        =
sqrt((s(i).Centroid(1)-
s(k).Centroid(1))^2+(s(i).Centroid(2
)-s(k).Centroid(2))^2);
    format long
    slope;
             if  ((slope  <  0.1)  &&
(slope > -0.1)) && distance <150
             end
```

```
        end
    end
end

hold off;


%Loop start and angle extraction
loop=1;
angle=1;
z=0;
while z<=5

L= bwlabel(rotated);
s = regionprops(L,'centroid');

%get total number of accepted slopes
ctrr=0;
numObj = numel(s);
for ctr = 1 :  numObj
    for ctr2=1 : numObj
        if      (s(ctr2).Centroid(1)-
s(ctr).Centroid(1))>0

slope=double((s(ctr2).Centroid(2)-
s(ctr).Centroid(2))/(s(ctr2).Centroi
d(1)-s(ctr).Centroid(1)));
    distance                        =
sqrt((s(ctr2).Centroid(1)-
s(ctr).Centroid(1))^2+(s(ctr2).Centr
oid(2)-s(ctr).Centroid(2))^2);
    format long
    slope;
           if loop==1
             if  ((slope  <  0.1)  &&
(slope > -0.1)) && distance <150
                 ctrr=ctrr+1;
             end
           else
             if  ((slope  <  0.05)  &&
(slope > -0.05)) && distance <150
                 ctrr=ctrr+1;
             end
           end
        end
    end
end

%Declaration of array and pointer
array =1:ctrr;
point=0;

%Extraction of the useful slopes
hold on
numObj = numel(s);
for k = 1 :  numObj
    for i=1 : numObj
        if         (s(i).Centroid(1)-
s(k).Centroid(1)) >0
    slope=double((s(i).Centroid(2)-
s(k).Centroid(2))/(s(i).Centroid(1)-
s(k).Centroid(1)));
    distance                        =
sqrt((s(i).Centroid(1)-
s(k).Centroid(1))^2+(s(i).Centroid(2
)-s(k).Centroid(2))^2);
    format long
    slope;
          if loop==1
             if  ((slope  <  0.1)  &&
(slope > -0.1)) && distance <150
                point=point+1;

array(:,point)=slope;

%plot([s(k).Centroid(1)
s(i).Centroid(1)],  [s(k).Centroid(2)
s(i).Centroid(2)],'r');
            end
           else
             if  ((slope  <  0.05)  &&
(slope > -0.05)) && distance <150
                point=point+1;

array(:,point)=slope;
            end
           end
        end
    end
end
hold off;

array;
medianave=median(array);

%Deskewing
angle=atan(medianave);
angle=(angle*180)/pi;
rotated= imrotate(rotated, angle);
mediangray=imrotate(mediangray,
angle);
save mediangray;
imwrite(mediangray,'C:\MATLAB701\wor
k\dump\mediangray.jpg');
imwrite(rotated,'C:\MATLAB701\work\d
ump\Rotated.jpg');

clear array;
loop=loop+1;
totalangle=totalangle+angle;
z=z+1;
end

%final display label
L = bwlabel(rotated);
s = regionprops(L,'centroid');
hold on
numObj = numel(s);
for k = 1 :  numObj
    %backside filtering
    x=s(k).Centroid(1);
    y=s(k).Centroid(2)-7.5;
    p=impixel(mediangray,x,y);

    for i=1 : numObj
```

```
        if         (s(i).Centroid(1)-
s(k).Centroid(1))>0
    slope=double((s(i).Centroid(2)-
s(k).Centroid(2))/(s(i).Centroid(1)-
s(k).Centroid(1)));
    distance                        =
sqrt((s(i).Centroid(1)-
s(k).Centroid(1))^2+(s(i).Centroid(2
)-s(k).Centroid(2))^2);
    format long

            if loop==1
                if ((slope < 0.1) &&
(slope > -0.1)) && distance <150
                end
            else
                if  ((slope  <  0.05)
&& (slope > -0.05)) && distance <150
                end
            end
        end
    end
end
hold off;


dis1=imread('C:\MATLAB701\work\dump\
mediangray.jpg');
dis1 =cat (3, dis1 ,dis1 ,dis1);
axes(handles.Deskew);
image(dis1);
axis off;

totalangle

%erasing
load mediangray;
for po =1 : 3
rotated=imread('C:\MATLAB701\work\du
mp\rotated.jpg');
rotated=uint8(rotated);
rotated=im2bw(rotated);

%figure('Name','Erasing','NumberTitl
e','off'),imshow(mediangray);
hold on
L = bwlabel(rotated);
s = regionprops(L,'centroid');
numObj = numel(s);
for k = 1 :  numObj

    %backside filtering
    x=s(k).Centroid(1);
    y=s(k).Centroid(2);
    x2=s(k).Centroid(1)-4;
    y2=s(k).Centroid(2)-7;
    p=impixel(mediangray,x2,y2);
    if p(1,1)<=210
        I2                          =
imread('C:\MATLAB701\work\dump\media
ngray.jpg');
        K                           =
imread('C:\MATLAB701\work\dump\rotat
ed.jpg');
        col  =  [(s(k).Centroid(1)-7)
(s(k).Centroid(1)+7)
(s(k).Centroid(1)+7)
(s(k).Centroid(1)-7)];
        row  =  [(s(k).Centroid(2)-7)
(s(k).Centroid(2)-7)
(s(k).Centroid(2)+7)
(s(k).Centroid(2)+7)];
        J = roifill(I2,col,row);
        M = roifill(K,col,row);

imwrite(J,'C:\MATLAB701\work\dump\me
diangray.jpg');

imwrite(M,'C:\MATLAB701\work\dump\ro
tated.jpg');
        rotated=M;
        mediangray=J;
        save mediangray;
    end

end
hold off;
end



dis2=imread('C:\MATLAB701\work\dump\
mediangray.jpg');
dis2 =cat (3, dis2 ,dis2 ,dis2);
axes(handles.Backside);
image(dis2);
axis off;


%segmentation
rotated=imread('C:\MATLAB701\work\du
mp\rotated.jpg');
rotated=uint8(rotated);
rotated=im2bw(rotated);

L = bwlabel(rotated);
s = regionprops(L,'centroid');
hold on
numObj = numel(s);

if numObj>0
%xholder
xholder=1:numObj;
for iz = 1 : numObj
xholder(:,iz)=s(iz).Centroid(1);
format short;
end
xholder

%yholder
yholder=1:numObj;
for iz = 1 : numObj
yholder(:,iz)=s(iz).Centroid(2);
format short;
```

```
end
yholder

%xyholder
xyholder=[1:numObj;1:numObj];
for iz = 1 : numObj
xyholder(1,iz)=s(iz).Centroid(1);
xyholder(2,iz)=s(iz).Centroid(2);
format short;
end
xyholder


y1=-999;
y2=-999;
y3=-999;

for pt=1 : numObj
    y2=(yholder(:,pt));

    for pt2=1 : numObj
        if  yholder(:,pt2)>=y2-22  &&
yholder(:,pt2)<y2-9
        y1=yholder(:,pt2);
        break;
        end
    end

    for pt2=1 : numObj
        if  yholder(:,pt2)>=y2+9  &&
yholder(:,pt2)<y2+22
        y3=yholder(:,pt2);
        break;
        end
    end

if y3>=y2+9 && y3<y2+22 && y1>=y2-22
&& y1<y2-9
break;
end

end

%charcount
charcount=1;
pt=0;
firstx=xholder(:,1);
secondx=0;
for pt=1 : numObj
    save charcount;
        if  (xholder(:,pt)>=firstx+16
&&    xholder(:,pt)<firstx+21)    &&
(yholder(:,pt)>=y1-4               &&
yholder(:,pt)<=y3+4);
            secondx=xholder(:,pt);
            break;
        elseif
(xholder(:,pt)>=firstx+21          &&
xholder(:,pt)<=firstx+32)          &&
(yholder(:,pt)>=y1-4               &&
yholder(:,pt)<=y3+14);
            secondx=firstx;
            firstx=secondx-20;
            break;
        elseif
(xholder(:,pt)>=firstx+47          &&
xholder(:,pt)<=firstx+51)          &&
(yholder(:,pt)>=y1-4               &&
yholder(:,pt)<=y3+4);
            secondx=firstx+20;
            break;
        %subject to change
        elseif
(xholder(:,pt)>=firstx+74          &&
xholder(:,pt)<=firstx+83)          &&
(yholder(:,pt)>=y1-4               &&
yholder(:,pt)<=y3+4);
            secondx=firstx;
            firstx=secondx-20;
            break;
        elseif
(xholder(:,pt)>=firstx+96          &&
xholder(:,pt)<=firstx+102)         &&
(yholder(:,pt)>=y1-4               &&
yholder(:,pt)<=y3+4);
            secondx=firstx+20;
            break;
         elseif
(xholder(:,pt)>=firstx+113         &&
xholder(:,pt)<=firstx+124)         &&
(yholder(:,pt)>=y1-4               &&
yholder(:,pt)<=y3+4);
            secondx=firstx+20;
            break;
         %end subject to change
        else
            firstx=xholder(:,pt);
        end
end

plot(firstx,80,'g*');
plot(secondx,80,'g*');
plot([firstx secondx], [80 80],'b');

 for i = 1 : 50

    for pt=1:numObj
        if
(xholder(:,pt)>=secondx+21         &&
xholder(:,pt)<secondx+34)          &&
(yholder(:,pt)>=y1-4               &&
yholder(:,pt)<=y3+4);
            firstx=xholder(:,pt);
            for pt2=1:numObj
                if
(xholder(:,pt2)>=firstx+16         &&
xholder(:,pt2)<firstx+21)          &&
(yholder(:,pt)>=y1-4               &&
yholder(:,pt)<=y3+4);

secondx=xholder(:,pt2);
                    break;
                else

secondx=firstx+20;
                    break;
```

```
                end
            end
            charcount=charcount+1;
            break;
        elseif
(xholder(:,pt)>=secondx+44          &&
xholder(:,pt)<secondx+53)           &&
(yholder(:,pt)>=y1-4                &&
yholder(:,pt)<=y3+4);
            secondx=xholder(:,pt);
            firstx=secondx-20;
            charcount=charcount+1;
            break;
        elseif
(xholder(:,pt)>=secondx+74          &&
xholder(:,pt)<secondx+83)           &&
(yholder(:,pt)>=y1-4                &&
yholder(:,pt)<=y3+4);;
            firstx=xholder(:,pt);
            for pt2=1:numObj
                if
(xholder(:,pt2)>=firstx+16          &&
xholder(:,pt2)<firstx+21)           &&
(yholder(:,pt)>=y1-4                &&
yholder(:,pt)<=y3+4);

secondx=xholder(:,pt2);
                    break;
                else

secondx=firstx+20;
                    break;
                end
            end
            charcount=charcount+2;
            break;
        elseif
(xholder(:,pt)>=secondx+96          &&
xholder(:,pt)<secondx+102)          &&
(yholder(:,pt)>=y1-4                &&
yholder(:,pt)<=y3+4);

            secondx=xholder(:,pt);
            firstx=secondx-20;
            charcount=charcount+2;
            break;

        end
    end
plot(firstx,80,'g*');
plot(secondx,80,'g*');
plot([firstx secondx], [80 80],'b');
end

charcount
save charcount;




y1
y2
y3
```

```
%test

binary=[1:charcount  ;  1:charcount  ;
1:charcount    ;    1:charcount    ;
1:charcount ; 1:charcount];
for cl=1:6
    for cl2=1:charcount
        binary(cl,cl2)=0;
    end
end

marker=1;
firstx=xholder(:,1);
secondx=0;
for pt=1 : numObj
        if  (xholder(:,pt)>=firstx+16
&&    xholder(:,pt)<firstx+21)     &&
(yholder(:,pt)>=y1-4                &&
yholder(:,pt)<=y3+4);
           secondx=xholder(:,pt);
           break;

        elseif
(xholder(:,pt)>=firstx+21           &&
xholder(:,pt)<=firstx+32)           &&
(yholder(:,pt)>=y1-4                &&
yholder(:,pt)<=y3+4);
            secondx=firstx;
            firstx=secondx-20;
            break;

        elseif
(xholder(:,pt)>=firstx+47           &&
xholder(:,pt)<=firstx+51)           &&
(yholder(:,pt)>=y1-4                &&
yholder(:,pt)<=y3+4);
            secondx=firstx+20;
            break;

        else
            firstx=xholder(:,pt);
        end
end

                    for  pt2  =  1  :
numObj
                    if
xyholder(1,pt2)>=firstx-10          &&
xyholder(1,pt2)<=firstx+10;
                    if
xyholder(2,pt2)>=y1-8               &&
xyholder(2,pt2)<=y1+8;

binary(1,marker)=1;
                    elseif
xyholder(2,pt2)>=y2-8               &&
xyholder(2,pt2)<=y2+8;

binary(2,marker)=1;
                    elseif
xyholder(2,pt2)>=y3-8               &&
xyholder(2,pt2)<=y3+8;
```

```
binary(3,marker)=1;
                    end
                    end
                    end

                    for  pt2  =  1  :
numObj
                    if
xyholder(1,pt2)>=secondx-10        &&
xyholder(1,pt2)<=secondx+10;
                    if
xyholder(2,pt2)>=y1-8              &&
xyholder(2,pt2)<=y1+4

binary(4,marker)=1;
                    elseif
xyholder(2,pt2)>=y2-8              &&
xyholder(2,pt2)<=y2+8;

binary(5,marker)=1;
                    elseif
xyholder(2,pt2)>=y3-8              &&
xyholder(2,pt2)<=y3+8;

binary(6,marker)=1;
                    end
                    end
                    end



for i = 1 : 50;
    for pt=1:numObj;
        if
(xholder(:,pt)>=secondx+21         &&
xholder(:,pt)<secondx+32)          &&
(yholder(:,pt)>=y1-4               &&
yholder(:,pt)<=y3+4);
            firstx=xholder(:,pt);

            for pt2=1:numObj;
                if
(xholder(:,pt2)>=firstx+16         &&
xholder(:,pt2)<firstx+21)          &&
(yholder(:,pt)>=y1-4               &&
yholder(:,pt)<=y3+4);

secondx=xholder(:,pt2);
                    break;
                else

secondx=firstx+20;
                    break;
                end
            end
            marker=marker+1;
            break;

        elseif
(xholder(:,pt)>=secondx+47         &&
xholder(:,pt)<secondx+51)          &&
(yholder(:,pt)>=y1-4               &&
yholder(:,pt)<=y3+4);
            secondx=xholder(:,pt);
            firstx=secondx-20;
            marker=marker+1;
            break;

        elseif
(xholder(:,pt)>=secondx+74         &&
xholder(:,pt)<secondx+83)          &&
(yholder(:,pt)>=y1-4               &&
yholder(:,pt)<=y3+4);
            firstx=xholder(:,pt);

            for pt2=1:numObj;
                if
(xholder(:,pt2)>=firstx+16         &&
xholder(:,pt2)<firstx+21)          &&
(yholder(:,pt)>=y1-4               &&
yholder(:,pt)<=y3+4);

secondx=xholder(:,pt2);
                break;

                else
                secondx=firstx+20;
                break;
                end
            end
            marker=marker+2;
            break;

        elseif
(xholder(:,pt)>=secondx+96         &&
xholder(:,pt)<secondx+102)         &&
(yholder(:,pt)>=y1-4               &&
yholder(:,pt)<=y3+4);

secondx=xholder(:,pt);
                firstx=secondx-20;
                marker=marker+2;
            break;

                    end

    end


                    for  pt2  =  1  :
numObj;
                    if
xyholder(1,pt2)>=firstx-10         &&
xyholder(1,pt2)<=firstx+10;
                    if
xyholder(2,pt2)>=y1-8              &&
xyholder(2,pt2)<=y1+8;

binary(1,marker)=1;
                    elseif
xyholder(2,pt2)>=y2-8              &&
xyholder(2,pt2)<=y2+8;

binary(2,marker)=1;
```

```
                    elseif
xyholder(2,pt2)>=y3-8              &&
xyholder(2,pt2)<=y3+8;

binary(3,marker)=1;
                    end
                    end
                    end

                    for  pt2  =  1  :
numObj;
                    if
xyholder(1,pt2)>=secondx-10        &&
xyholder(1,pt2)<=secondx+10;
                    if
xyholder(2,pt2)>=y1-8              &&
xyholder(2,pt2)<=y1+8;

binary(4,marker)=1;
                    elseif
xyholder(2,pt2)>=y2-8              &&
xyholder(2,pt2)<=y2+8;

binary(5,marker)=1;
                    elseif
xyholder(2,pt2)>=y3-8              &&
xyholder(2,pt2)<=y3+8;

binary(6,marker)=1;
                    end
                    end
                    end
save binary;
end
elseif numObj==0
charcount=0
save charcount;
end


load charcount;
charcount

if charcount==0
ee='No Character Detected'
set(LOADING,'Visible','off')
set(handles.RecognizedImage,'String'
,ee);

set(handles.BrowseImage,'Enable','on
');
set(handles.CleanImage,'Enable','off
');
set(handles.DeskewImage,'Enable','of
f');
set(handles.SegmentImage,   'Enable',
'off');
set(handles.RecognizeImage,'Enable',
'off');
set(handles.Oneclick,'Enable','off')
;

elseif charcount>=1

load binary;
binary

u=1
while(u<=charcount)
    z=num2str(binary(1,u));
    b=[z];
    for o = 2 : 6
    a=binary(o,u);
    str1=num2str(a);
    b=[b str1];
    end
    db(u,:)=strvcat(b);
    u=u+1;
end

db

    db1='123456';
    conn  =  database('Braille',  '',
'');

setdbprefs('DataReturnFormat','cella
rray');
    curs   =   exec(conn,   ['select
Letter from Grade1 where ID= ' ''''
db1 '''']);
    curs = fetch(curs);
    bb = curs.data;

    db1='000000';
    conn  =  database('Braille',  '',
'');

setdbprefs('DataReturnFormat','cella
rray');
    curs   =   exec(conn,   ['select
Letter from Grade1 where ID= ' ''''
db1 '''']);
    curs = fetch(curs);
    space = curs.data;


end

if charcount>=1

db1=[db(1,:)]
met=1;
epox=1;
while(epox<=charcount)
    epox
    lft=0;
    rgt=0;
    if epox<charcount && charcount>1
    db2=[db1 db(epox+1,:) '%']
    elseif charcount==1
    db2='12345'
    end
    conn  =  database('Braille',  '',
'');
```

```
setdbprefs('DataReturnFormat','cella
rray')
    curs = exec(conn, ['select Words
from Grade2 where ID like ' '''' db2
''''])
    curs = fetch(curs)
    aa = curs.data

    %Empty query checker
    n = numel(curs.data)
    b=0;
    if n==1
    b = strcmp(bb,aa)
    end
    %left & right checker
    if epox+1<charcount

rgt=strcmp(db(epox+1,:),'000000')
    end
    if epox-met>=1%error catcher if
ctr-met will equal to 0
    lft=strcmp(db(epox-
met,:),'000000')
    end

    %query for standalone grade 2
braille characters(group) for the
first
    %including the first char
    if (b == 1 && n == 1) && rgt==1
&& met>1 && epox-met<1
        curs = exec(conn, ['select
Words from Grade2 where ID= ' ''''
db1 ''''])
        curs = fetch(curs)
        aa = curs.data
        n = numel(curs.data)
        b=strcmp(bb,aa)
        arrayy(epox) = aa

        spaceb=met-1
        while spaceb>=1
        arrayy(epox-spaceb)=space;
        spaceb=spaceb-1;
        end

        db1=db(epox+1,:)
        met=1
        epox=epox+1;


     %Sure standalone
     elseif (b == 1 && n == 1) &&
lft==1 &&  rgt==1 && met>1
        curs = exec(conn, ['select
Words from Grade2 where ID= ' ''''
db1 ''''])
        curs = fetch(curs)
        aa = curs.data
        n = numel(curs.data)
        b=strcmp(bb,aa)
        arrayy(epox) = aa

        spaceb=met-1
        while spaceb>=1
        arrayy(epox-spaceb)=space;
        spaceb=spaceb-1;
        end

        db1=db(epox+1,:)
        met=1
        epox=epox+1;

      %Prefix
      elseif (b == 1 && n == 1) &&
lft==1 &&  rgt==0  && met>1
         curs = exec(conn, ['select
Words from Pre_fix where ID= ' ''''
db1 ''''])
        curs = fetch(curs)
        aa = curs.data
        n = numel(curs.data)
        b=strcmp(bb,aa)

        if  (b == 1 && n == 1)
        ppp=met-1
        while ppp>=1
        epox=epox-ppp
        db1=db(epox,:)
        curs = exec(conn, ['select
Letter from Grade1 where ID= ' ''''
db1 ''''])
        curs = fetch(curs)
        aa = curs.data
        n = numel(curs.data)
        b=strcmp(bb,aa)
        arrayy(epox) = aa
        db1=db(epox+1,:)
        met=1
        epox=epox+ppp;
        ppp=ppp-1;
        end

        else
        curs = exec(conn, ['select
Words from Pre_fix where ID= ' ''''
db1 ''''])
        curs = fetch(curs)
        aa = curs.data
        n = numel(curs.data)
        b=strcmp(bb,aa)
        arrayy(epox) = aa

        spaceb=met-1
        while spaceb>=1
        arrayy(epox-spaceb)=space;
        spaceb=spaceb-1;
        end

        db1=db(epox+1,:)
        met=1
        epox=epox+1;
        end

      %Suffix
```

```
      elseif (b == 1 && n == 1) &&
lft==0 &&  rgt==1  && met>1

        curs  =  exec(conn,  ['select
Words from Suffix where ID= ' ''''
db1 ''''])
        curs = fetch(curs)
        aa = curs.data
        n = numel(curs.data)
        b=strcmp(bb,aa)

        if  (b == 1 && n == 1)
        ppp=met-1
        while ppp>=1
        epox=epox-ppp
        db1=db(epox,:)
        curs  =  exec(conn,  ['select
Letter from Grade1 where ID= ' ''''
db1 ''''])
        curs = fetch(curs)
        aa = curs.data
        n = numel(curs.data)
        b=strcmp(bb,aa)
        arrayy(epox) = aa
        db1=db(epox+1,:)
        met=1
        epox=epox+ppp;
        ppp=ppp-1;
        end

        else
        curs  =  exec(conn,  ['select
Words from Suffix where ID= ' ''''
db1 ''''])
        curs = fetch(curs)
        aa = curs.data
        n = numel(curs.data)
        b=strcmp(bb,aa)
        arrayy(epox) = aa

        spaceb=met-1
        while spaceb>=1
        arrayy(epox-spaceb)=space;
        spaceb=spaceb-1;
        end

        db1=db(epox+1,:)
        met=1
        epox=epox+1;
        end


       %Midfix
       elseif (b == 1 && n == 1) &&
lft==0 &&  rgt==0  && met>1

        curs  =  exec(conn,  ['select
Words from Mid_fix where ID= ' ''''
db1 ''''])
        curs = fetch(curs)
        aa = curs.data
        n = numel(curs.data)
        b=strcmp(bb,aa)
        if  (b == 1 && n == 1)
        ppp=met-1
        while ppp>=1
        epox=epox-ppp
        db1=db(epox,:)
        curs  =  exec(conn,  ['select
Letter from Grade1 where ID= ' ''''
db1 ''''])
        curs = fetch(curs)
        aa = curs.data
        n = numel(curs.data)
        b=strcmp(bb,aa)
        arrayy(epox) = aa
        db1=db(epox+1,:)
        met=1
        epox=epox+ppp;
        ppp=ppp-1;
        end

        else
        curs  =  exec(conn,  ['select
Words from Mid_fix where ID= ' ''''
db1 ''''])
        curs = fetch(curs)
        aa = curs.data
        n = numel(curs.data)
        b=strcmp(bb,aa)
        arrayy(epox) = aa

        spaceb=met-1
        while spaceb>=1
        arrayy(epox-spaceb)=space;
        spaceb=spaceb-1;
        end

        db1=db(epox+1,:)
        met=1
        epox=epox+1;
        end


%try
    elseif  (b==1  &&  n==1)&&  met==1
&& lft==1 && rgt==1
        curs  =  exec(conn,  ['select
Words from Stand_Alone where ID= '
'''' db1 ''''])
        curs = fetch(curs)
        aa = curs.data
        n = numel(curs.data)
        b=strcmp(bb,aa)
        arrayy(epox) = aa
        if epox<charcount
        db1=db(epox+1,:)
        end
        met=1
        epox=epox+1;

    elseif  (b==1  &&  n==1)&&  met==1
&& lft==1 && epox+1>charcount
```

```
        curs = exec(conn, ['select
Words from Stand_Alone where ID= '
'''' db1 ''''])
        curs = fetch(curs)
        aa = curs.data
        n = numel(curs.data)
        b=strcmp(bb,aa)
        arrayy(epox) = aa
        if epox<charcount
        db1=db(epox+1,:)
        end
        met=1
        epox=epox+1;

     elseif (b==1 && n==1)&& met==1
&& rgt==1 && epox-met<1
        curs = exec(conn, ['select
Words from Stand_Alone where ID= '
'''' db1 ''''])
        curs = fetch(curs)
        aa = curs.data
        n = numel(curs.data)
        b=strcmp(bb,aa)
        arrayy(epox) = aa
        if epox<charcount
        db1=db(epox+1,:)
        end
        met=1
        epox=epox+1;

     elseif (b==1 && n==1)&& met==1
&& epox+1>charcount && epox-met<1
        curs = exec(conn, ['select
Words from Stand_Alone where ID= '
'''' db1 ''''])
        curs = fetch(curs)
        aa = curs.data
        n = numel(curs.data)
        b=strcmp(bb,aa)
        arrayy(epox) = aa
        if epox<charcount
        db1=db(epox+1,:)
        end
        met=1
        epox=epox+1;
%not sure


    %query for grade 1 braille(since
grade 2 1 braille char standalone
are unique)
    elseif    (b==1   &&   n   ==   1)   &&
met==1
        curs = exec(conn, ['select
Letter from Grade1 where ID= ' ''''
db1 ''''])
        curs = fetch(curs)
        aa = curs.data
        n = numel(curs.data)
        b=strcmp(bb,aa)


        %query for grade 2 single
braille char standalone(condition is
if
        %there's no match in the
grade 1 DB)
        if (b == 1 && n == 1)
        curs = exec(conn, ['select
Words from Grade2 where ID= ' ''''
db1 ''''])
        curs = fetch(curs)
        aa = curs.data
        n = numel(curs.data)
        b=strcmp(bb,aa)

            if n>1 && epox-met<1 &&
rgt==0
                curs   =   exec(conn,
['select Words from Pre_Fix where
ID= ' ''' db1 ''''])
                curs = fetch(curs)
                aa = curs.data
                n = numel(curs.data)
                b=strcmp(bb,aa)

            elseif n>1 && lft==0 &&
rgt==1 && epox==1
                curs   =   exec(conn,
['select   Words   from   Stand_Alone
where ID= ' ''' db1 ''''])
                curs = fetch(curs)
                aa = curs.data
                n = numel(curs.data)
                b=strcmp(bb,aa)

            elseif n>1 && lft==0 &&
rgt==0
                curs   =   exec(conn,
['select Words from Mid_Fix where
ID= ' ''' db1 ''''])
                curs = fetch(curs)
                aa = curs.data
                n = numel(curs.data)
                b=strcmp(bb,aa)

            elseif n>1 && lft==0 &&
rgt==1
                curs   =   exec(conn,
['select Words from Suffix where ID=
' ''' db1 ''''])
                curs = fetch(curs)
                aa = curs.data
                n = numel(curs.data)
                b=strcmp(bb,aa)

            elseif n>1 && lft==1 &&
rgt==0
                curs   =   exec(conn,
['select Words from Pre_Fix where
ID= ' ''' db1 ''''])
                curs = fetch(curs)
                aa = curs.data
                n = numel(curs.data)
                b=strcmp(bb,aa)
```

```
            elseif n>1 && lft==1 &&
rgt==1
                curs    =    exec(conn,
['select   Words   from   Stand_Alone
where ID= ' '''' db1 ''''])
                curs = fetch(curs)
                aa = curs.data
                n = numel(curs.data)
                b=strcmp(bb,aa)

            end
        end

        %proceed here if match is
already found
        arrayy(epox) = aa
        if epox<charcount
        db1=db(epox+1,:)
        end
        met=1
        epox=epox+1;

    else
    %db1 incrementation if there's
no match in any of the database
    if epox+1<=charcount
    db1=[db1 db(epox+1,:)]
    met=met+1
    epox=epox+1;

    %trial
    else
        curs = exec(conn, ['select
Words from Grade2 where ID= ' ''''
db1 ''''])
        curs = fetch(curs)
        aa = curs.data
        n = numel(curs.data)
        b=strcmp(bb,aa)
        arrayy(epox) = aa

        spaceb=met-1
        while spaceb>=1
        arrayy(epox-spaceb)=space;
        spaceb=spaceb-1;
        end

        epox=epox+1
    end
    %trial


    end

    close(conn)
    close(curs)


end

result = [arrayy(:)]
```

```
ee=strvcat(arrayy(1));
for cc = 2 : charcount
ff=strvcat(arrayy(cc));
ee=[ee ff];
end
ee


set(LOADING,'Visible','off')
set(handles.RecognizedImage,'String'
,ee);
set(handles.BrowseImage,'Enable','on
');
set(handles.CleanImage,'Enable','off
');
set(handles.DeskewImage,'Enable','of
f');
set(handles.SegmentImage,  'Enable',
'off');
set(handles.RecognizeImage,'Enable',
'off');
set(handles.Oneclick,'Enable','off')
;
end

end

% --- Executes on button press in
Showall.
function   Showall_Callback(hObject,
eventdata, handles)
% hObject    handle to Showall (see
GCBO)
% eventdata    reserved - to be
defined in a future version of
MATLAB
% handles    structure with handles
and user data (see GUIDATA)

set(handles.BrowseImage,'Visible','o
n');
set(handles.CleanImage,'Visible','on
');
set(handles.DeskewImage,'Visible','o
n');
set(handles.SegmentImage, 'Visible',
'on');
set(handles.RecognizeImage,'Visible'
,'on');
set(handles.Oneclick,'Visible','Off'
);
set(handles.BrailleAlphabet,'Visible
','on');
set(handles.BrailleContractions,'Vis
ible','on');
set(handles.ResetAll,'Visible','on')
;
set(handles.Exit,'Visible','on');
set(handles.uipanel2,'Visible','on')
;
set(handles.uipanel4,'Visible','on')
;
```

```
set(handles.RecognizedImage,'Visible
','on');
set(handles.uipanel7,'Visible','on')
;
set(handles.uipanel8,'Visible','on')
;
set(handles.text1,'Visible','on');
set(handles.ImageDisplay,'Visible','
on');
set(handles.uipanel6,'Visible','on')
;
set(handles.text3,'Visible','on');
set(handles.Deskew,'Visible','on');
set(handles.uipanel10,'Visible','on'
);
set(handles.text12,'Visible','on');
set(handles.Backside,'Visible','on')
;
set(handles.Showall,'Visible','off')
;
set(handles.Back,'Visible','on');
set(handles.Oneprocess,'Visible','of
f');
set(handles.Banner,'Visible','off');
set(handles.uipanel12,'Visible','on'
);

% --- Executes on button press in
Back.
function       Back_Callback(hObject,
eventdata, handles)
% hObject      handle to Back (see
GCBO)
% eventdata    reserved - to be
defined in a future version of
MATLAB
% handles     structure with handles
and user data (see GUIDATA)
set(handles.BrowseImage,'Visible','o
ff');
set(handles.CleanImage,'Visible','of
f');
set(handles.DeskewImage,'Visible','o
ff');
set(handles.SegmentImage, 'Visible',
'off');
set(handles.RecognizeImage,'Visible'
,'off');
set(handles.Oneclick,'Visible','Off'
);
set(handles.BrailleAlphabet,'Visible
','off');
set(handles.BrailleContractions,'Vis
ible','off');
set(handles.ResetAll,'Visible','off'
);
set(handles.Exit,'Visible','off');
set(handles.uipanel2,'Visible','off'
);
set(handles.uipanel4,'Visible','off'
);
set(handles.RecognizedImage,'Visible
','off');
set(handles.uipanel7,'Visible','off'
);
set(handles.uipanel8,'Visible','off'
);
set(handles.text1,'Visible','off');
set(handles.ImageDisplay,'Visible','
off');
set(handles.uipanel6,'Visible','off'
);
set(handles.text3,'Visible','off');
set(handles.Deskew,'Visible','off');
set(handles.uipanel10,'Visible','off
');
set(handles.text12,'Visible','off');
set(handles.Backside,'Visible','off'
);
set(handles.Showall,'Visible','on');
set(handles.Back,'Visible','off');
set(handles.Oneprocess,'Visible','on
');
set(handles.Banner,'Visible','on');
set(handles.uipanel12,'Visible','off
');

cla(handles.ImageDisplay,'reset');
cla(handles.Deskew,'reset');
cla(handles.Backside,'reset');
cla;
set(handles.text1,'String','Browsed
Image');
set(handles.BrowseImage,'Enable','on
');
set(handles.DeskewImage,'Enable','of
f');
set(handles.CleanImage,'Enable','off
');
set(handles.SegmentImage,'Enable','o
ff');
set(handles.RecognizeImage,'Enable',
'off');
set(handles.RecognizedImage,'String'
,' ');
set(handles.ImageDisplay,'xcolor','w
','ycolor','w','xtick',[],'ytick',[]
);
set(handles.Deskew,'xcolor','w','yco
lor','w','xtick',[],'ytick',[]);
set(handles.Backside,'xcolor','w','y
color','w','xtick',[],'ytick',[]);
set(handles.text1,'visible','on');
set(handles.Oneclick,'Enable','off')
;
set(handles.Angle,'String',' ');
clear

% --- Executes on button press in
Oneprocess.
function
Oneprocess_Callback(hObject,
eventdata, handles)
% hObject      handle to Oneprocess
(see GCBO)
```

```
% eventdata    reserved - to be
defined in a future version of
MATLAB
% handles     structure with handles
and user data (see GUIDATA)

set(handles.BrowseImage,'Visible','o
n');
set(handles.CleanImage,'Visible','of
f');
set(handles.DeskewImage,'Visible','o
ff');
set(handles.SegmentImage, 'Visible',
'off');
set(handles.RecognizeImage,'Visible'
,'off');
set(handles.Oneclick,'Visible','on')
;
set(handles.BrailleAlphabet,'Visible
','on');
set(handles.BrailleContractions,'Vis
ible','on');
set(handles.ResetAll,'Visible','on')
;
set(handles.Exit,'Visible','on');
set(handles.uipanel2,'Visible','on')
;
set(handles.uipanel4,'Visible','on')
;
set(handles.RecognizedImage,'Visible
','on');
set(handles.uipanel7,'Visible','on')
;
set(handles.uipanel8,'Visible','on')
;
set(handles.text1,'Visible','on');
set(handles.ImageDisplay,'Visible','
on');
set(handles.uipanel6,'Visible','off'
);
set(handles.text3,'Visible','off');
set(handles.Deskew,'Visible','off');
set(handles.uipanel10,'Visible','off
');
set(handles.text12,'Visible','off');
set(handles.Backside,'Visible','off'
);
set(handles.Showall,'Visible','off')
;
set(handles.Back,'Visible','on');
set(handles.Oneprocess,'Visible','of
f');
set(handles.Banner,'Visible','off');
```