Software Development Kit

Recogniform Layout Analysis

Copyright 2006

Recogniform Technologies SpA

HOW TO CONTACT US

Recogniform Technologies SpA
Via Edison, 22
87036 Rende (CS), Italy

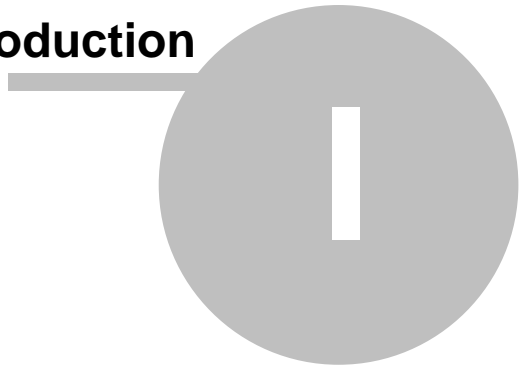<u>Phone</u>   : +39 0984 404174

<u>Fax</u>   : +39 0984 830299

<u>Internet</u> : www.recogniform.com

<u>E-Mail</u>   : info@recogniform.com

Tavola dei contenuti

Introduction

I

1   Introduction

1.1   Copyright

The software and the documentation are property of:

Recogniform Technologies SpA
Via Edison, 22
87036 Rende (CS)
Italy

www.recogniform.com
info@recogniform.com

1.2   License

It is illegal to copy or reproduce this manual, or any part thereof, in any shape or form.

The information contained in this manual is subject to change without notice and does not present a commitment on the part of Recogniform Technologies SpA.

Recogniform Technologies SpA shall not be held liable for technical or editorial errors and/or omissions made here, nor for incidental or consequential damages resulting from the furnishing, performance, or use of the software and documentation.

Recogniform Technologies SpA reserves the right to make changes to the software and documentation without notice.

Product names mentioned here are used for identification purposes only and may be tradenames and/or registered trademarks of their respective companies.

YOU CANNOT DISTRUBUTE SOFTWARE INCLUDING THIS SDK LIBRARY UNLESS YOU HAVE A WRITTEN AGREEMENT (ROYALTIES FREE OR ROYALTIES BASED) WITH RECOGNIFORM TECHNOLOGIES SPA

1.3   Overview

The Recogniform Layout Analysis Library 1.0 allows to automatically analyze the layout of images detecting zones of:
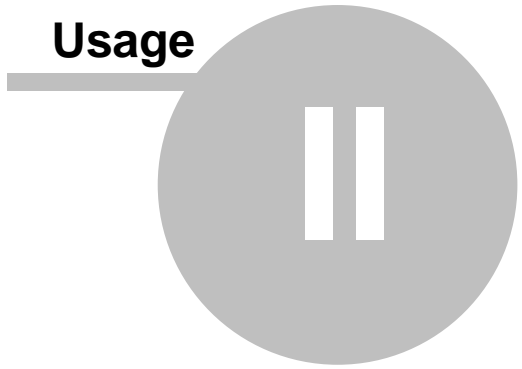
- normal text

- inverted text (white on dark background)
- images and graphics
- lines (horizontal and vertical)
- tables
- table columns
- table cells
- noise

It can work on bitonal images passed as DIB (Device Independent Bitamp)

For each zone are returned the coordinates of rectangle circubscribing the zone as well as the zone type.

Usage

II

2   Usage

2.1   Visual C++

You have to include the RECOLA.C in your program (see sample application).

Before to execute your application make sure the *RECOLA.DLL* is available in your same .exe directory or in windows\system directory.

2.2   Visual Basic

You have to include the module RECOLA.BAS in your program (see sample application).
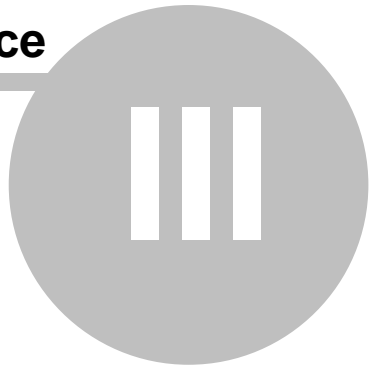
Before to execute your application make sure the *RECOLA.DLL* is available in your same .exe directory or in windows\system directory.

2.3   Delphi

You have to include the unit RECOLA.PAS in your program (see sample application).

Before to execute your application make sure the *RECOLA.DLL* is available in your same .exe directory or in windows\system directory.

API Reference

III

3   API Reference

3.1   LoadLALibrary

`RECOIO_API void LoadLALibrary(void)`

Description

Load the Layout Analysis DLL library: you have to use this function one time before to use other API functions from Visual C++
This function is not required using the API from Delphi or Visual Basic

3.2   FreeLALibrary

`RECOIO_API void FreeLALibrary(void)`

Description

Unload the Layout Analysis DLL library: you have to use this function one time before to exit from your Visual C++ application.
This function is not required using the API from Delphi or Visual Basic

3.3   LA_Init

`RECOLA_API long LA_Init(char *User, char *Password);`

Description

This is the first function to call: initialize the library and returns a session handle to use in next calls. When you buy the library you receive an "user" and a "password" string necessary to initialize the library in normal mode: without this value or with wrang values the library is initialized in evaluation mode. The

evaluation mode works exactly as normal mode but some time, when you call a function, is displayed a warning dialog box remembering the evaluation state: you can close it and continue to work with no problems.

Parameters

User (in) - then user name string
Passwors (in) - then password string

Return values

The session handle if the library is initialized, 0 otherwise.

3.4   LA_Done

RECOLA_API void LA_Done(long SessionHandle);

Description

This is the last function to call when you don't need more services from the library: deinitialize the library and free all used resources.

Parameters

SessionHandle (in) - the session handle to free

Return values

n/a

3.5   LA_Execute

```
RECOLA_API long LA_Execute(long
SessionHandle, long DIBHandle);
```

Description

This is the function performing the layout analysis processing on an image.

Parameters

SessionHandle (in) - the session handle to use
DIBHandle(in) - the memory handle of the bitonal DIB (Device Indipentend Bitmap) do process

Return values

"0" if all goes ok else an error code.

Notes

Refer to Microsoft documentation for additional info about Device Indipendent Bitmaps.

If you don't know how to handle DIB you can evaluate our Recogniform Imaging Library allowing to load images from file obtaining a DIB handle to use in other SDKs.

3.6   LA_GetZonesCount

```
RECOLA_API long LA_GetGetZonesCount(long
SessionHandle);
```

Description

This is the function returning the number of zones found after

analysis. You can call this function upon a LA_Execute.

Parameters

SessionHandle (in) - the session handle to use

Return values

The number of zones found.

3.7   LA_GetZoneRect

```
RECOLA_API long LA_GetZoneRect(long
SessionHandle, long ZoneIndex, long *Left,
long *Top, long *Right, long *Bottom);
```

Description

This is the function to call to obtain the rectangle coordinates of one of zones found. You can call this function upon a LA_Execute.

Parameters

SessionHandle (in) - the session handle to use
ZoneIndex(in) - the index of the zone to retrieve the coordinates. First value is 0 while last value is the result of LA_GetZonesCount less 1.
Left(out) - the left coordinate of zone rectangle
Top(out) - the top coordinate of zone rectangle
Right(out) - the right coordinate of zone rectangle
Bottom(out) - the bottom coordinate of zone rectangle

Return values

"0" if all goes ok else an error code.

3.8 **LA_GetZoneContent**

```
RECOLA_API long LA_GetZoneContent(long
SessionHandle, long ZoneIndex, long
*ContentCode);
```

Description

This is the function to call to obtain the content type of one of zones found. You can call this function upon a LA_Execute.

Parameters

essionHandle (in) - the session handle to use
ZoneIndex(in) - the index of the zone to retrieve the coordinates. First value is 0 while last value is the result of LA_GetZonesCount less 1.
ContentCode(out) - the code indicating the zone content. Possible values are:

  0 = line
  1 = table
  2 = column inside a table
  3 = cell inside a table
  4 = inverse text
  5 = image
  6 = text
  7 = noise

Return values

"0" if all goes ok else an error code.

3.9   LA_GetParameter

```
RECOLA_API long LA_GetParameter(long
SessionHandle, long ParameterIndex, long
*Value);
```

Description

This is the function to get the value of user parameters
changing the analysis mode.

Parameters

SessionHandle (in) - the session handle to use
ParameterIndex(in) - the index of parameter to retrieve the
value
Value(out) - the buffer to store the retrieved parameter value

Return values

"0" if all goes ok else an error code.

3.10   LA_SetParameter

```
RECOLA_API long LA_SetParameter(long
SessionHandle, long ParameterIndex, long
Value);
```

Description

This is the function to set the value of user parameters
changing the analysis mode.

Parameters

SessionHandle (in) - the session handle to use
ParameterIndex(in) - the index of parameter to set the value

Value(in) - the parameter new value

Return values

"0" if all goes ok else an error code.

Samples

IV

4 Samples

4.1 Visual C++

```c
#include "stdafx.h"
#include <stdio.h>
#include "recola.c"

int APIENTRY WinMain(HINSTANCE hInstance,
                     HINSTANCE
hPrevInstance,
                     LPSTR      lpCmdLine,
                     int        nCmdShow)
{

    // Load dynamically the library
    LoadLALibrary();

    // Init the session
    int Session= LA_Init("demo", "demo");

    long hBitmap;

    // Check if a DIB is available in
clipboard
    bool bAvail=
IsClipboardFormatAvailable(CF_DIB);

    hBitmap = 0;



    if (bAvail)
    {

        // Open the Clipboard
        ::OpenClipboard(NULL);

        // Retrieve the DIB from
clipboard
```

```
         hBitmap = (long)
GetClipboardData(CF_DIB);

         // Perorm layout analysis
         long RemovedPoints=
LA_Execute(Session, (long) hBitmap);

         // Close the Clipboard
         ::CloseClipboard();


         long i, Zones, Left, Top, Right,
Bottom, ZoneType;

         Zones=LA_GetZonesCount(Session);

         for (i=0;i<Zones;i++)
          {
               LA_GetZoneRect(Session, i,
&Left, &Top, &Right, &Bottom);
               LA_GetZoneContent(Session,
i, &ZoneType);

               char buffer[200]

               sprintf( buffer, "Zone %d -
Left=%d, Top=%d, Right=%d, Bottom=%d -
Content=&d", i, Left, Top, Right, Bottom,
ZoneType)

               MessageBox(NULL, buffer,
"RESULT", MB_OK);

          }


     }
     // Show an error message
     else MessageBox(NULL, "Unable to
```

```
paste DIB", "ERROR", MB_OK);


    // Close the session
    LA_Done(Session);

    // Unload the library
    FreeLALibrary();

    return 0;
}
```

4.2   Visual Basic

To-Do

4.3   Delphi

To-Do

Annotazioni