ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA (ICAI)

INGENIERO INDUSTRIAL

VISIÓN ARTIFICIAL DE DOCUMENTOS

Autor: Alberto Arce Arroyo
Director: Juan Antonio Talavera MartÍn

Madrid

Julio, 2016

AUTORIZACIÓN PARA LA DIGITALIZACIÓN, DEPÓSITO Y DIVULGACIÓN EN RED DE PROYECTOS FIN DE GRADO, FIN DE MÁSTER, TESINAS O MEMORIAS DE BACHILLERATO

1º. Declaración de la autoría y acreditación de la misma.

El autor D. ALBERTO ARCE

DECLARA ser el titular de los derechos de propiedad intelectual de la obra: VISIÓN ARTIFICIAL DE DOCUMENTOS

que ésta es una obra original, y que ostenta la condición de autor en el sentido que otorga la Ley de Propiedad Intelectual.

2º. Objeto y fines de la cesión.

Con el fin de dar la máxima difusión a la obra citada a través del Repositorio institucional de la Universidad, el autor **CEDE** a la Universidad Pontificia Comillas, de forma gratuita y no exclusiva, por el máximo plazo legal y con ámbito universal, los derechos de digitalización, de archivo, de reproducción, de distribución y de comunicación pública, incluido el derecho de puesta a disposición electrónica, tal y como se describen en la Ley de Propiedad Intelectual. El derecho de transformación se cede a los únicos efectos de lo dispuesto en la letra a) del apartado siguiente.

3º. Condiciones de la cesión y acceso

Sin perjuicio de la titularidad de la obra, que sigue correspondiendo a su autor, la cesión de derechos contemplada en esta licencia habilita para:

a) Transformarla con el fin de adaptarla a cualquier tecnología que permita incorporarla a internet y hacerla accesible; incorporar metadatos para realizar el registro de la obra e incorporar "marcas de agua" o cualquier otro sistema de seguridad o de protección.
b) Reproducirla en un soporte digital para su incorporación a una base de datos electrónica, incluyendo el derecho de reproducir y almacenar la obra en servidores, a los efectos de garantizar su seguridad, conservación y preservar el formato.
c) Comunicarla, por defecto, a través de un archivo institucional abierto, accesible de modo libre y gratuito a través de internet.
d) Cualquier otra forma de acceso (restringido, embargado, cerrado) deberá solicitarse expresamente y obedecer a causas justificadas.
e) Asignar por defecto a estos trabajos una licencia Creative Commons.
f) Asignar por defecto a estos trabajos un HANDLE (URL *persistente).*

4º. Derechos del autor.

El autor, en tanto que titular de una obra tiene derecho a:

a) Que la Universidad identifique claramente su nombre como autor de la misma
b) Comunicar y dar publicidad a la obra en la versión que ceda y en otras posteriores a través de cualquier medio.
c) Solicitar la retirada de la obra del repositorio por causa justificada.
d) Recibir notificación fehaciente de cualquier reclamación que puedan formular terceras personas en relación con la obra y, en particular, de reclamaciones relativas a los derechos de propiedad intelectual sobre ella.

5º. Deberes del autor.

El autor se compromete a:

a) Garantizar que el compromiso que adquiere mediante el presente escrito no infringe ningún derecho de terceros, ya sean de propiedad industrial, intelectual o cualquier otro.
b) Garantizar que el contenido de las obras no atenta contra los derechos al honor, a la intimidad y a la imagen de terceros.
c) Asumir toda reclamación o responsabilidad, incluyendo las indemnizaciones por daños, que

pudieran ejercitarse contra la Universidad por terceros que vieran infringidos sus derechos e intereses a causa de la cesión.

d) Asumir la responsabilidad en el caso de que las instituciones fueran condenadas por infracción de derechos derivada de las obras objeto de la cesión.

6º. Fines y funcionamiento del Repositorio Institucional.

La obra se pondrá a disposición de los usuarios para que hagan de ella un uso justo y respetuoso con los derechos del autor, según lo permitido por la legislación aplicable, y con fines de estudio, investigación, o cualquier otro fin lícito. Con dicha finalidad, la Universidad asume los siguientes deberes y se reserva las siguientes facultades:

- La Universidad informará a los usuarios del archivo sobre los usos permitidos, y no garantiza ni asume responsabilidad alguna por otras formas en que los usuarios hagan un uso posterior de las obras no conforme con la legislación vigente. El uso posterior, más allá de la copia privada, requerirá que se cite la fuente y se reconozca la autoría, que no se obtenga beneficio comercial, y que no se realicen obras derivadas.
- La Universidad no revisará el contenido de las obras, que en todo caso permanecerá bajo la responsabilidad exclusive del autor y no estará obligada a ejercitar acciones legales en nombre del autor en el supuesto de infracciones a derechos de propiedad intelectual derivados del depósito y archivo de las obras. El autor renuncia a cualquier reclamación frente a la Universidad por las formas no ajustadas a la legislación vigente en que los usuarios hagan uso de las obras.
- La Universidad adoptará las medidas necesarias para la preservación de la obra en un futuro.
- La Universidad se reserva la facultad de retirar la obra, previa notificación al autor, en supuestos suficientemente justificados, o en caso de reclamaciones de terceros.

Madrid, a 17 de Julio de 2016

ACEPTA

Fdo.....

Motivos para solicitar el acceso restringido, cerrado o embargado del trabajo en el Repositorio Institucional:

Autorizada la entrega del proyecto del alumno/a:
(Poner el nombre del alumno/a)

ALBERTO ARCE ARROYO

EL DIRECTOR DEL PROYECTO
(poner el nombre del Director del Proyecto)

Fdo.: JUAN ANTONIO TALAVERA Fecha: 14 / 7 / 2016

V° B° del Coordinador de Proyectos
(poner el nombre del Coordinador de Proyectos)

Fdo.: ..... Fecha: ...../ ...../ .....

VISION ARTIFICIAL DE DOCUMENTOS

Autor: Arce Arroyo, Alberto.

Director: Talavera, Juan Antonio

Entidad Colaboradora: ICAI – Universidad Pontifica Comillas.

RESUMEN DEL PROYECTO

1.- Introducción

El proyecto consiste en estudiar y desarrollar un sistema de visión artificial, se partirá de imágenes de documentos, como DNIs, provenientes de escáner. Se trata de aplicar diferentes técnicas de visión artificial para identificar e interpretar el contenido del documento. En este proyecto nos centramos en técnicas de visión OCR (Reconocimiento óptico de caracteres). El objetivo es implantar la aplicación en un puesto de control de tránsito de personas para poder llevar un registro automático

2.- Estado de la técnica

Existen en la actualidad gran cantidad de programas de OCR comerciales. Están principalmente destinados al reconocimiento de caracteres impresos.

La siguiente tabla muestra el listado de las principales aplicaciones comerciales.

Compañía	**Licencia**	**Sistema operativo**	**Notas**
ExperVision TypeReader & OpenRTK	Comercial	Windows, Mac Os, Linux	ExperVision Inc. fue fundada en 1987, su tecnología OCR y producto, ganó las más altas calificaciones en las pruebas independientes realizadas por UNLV para los años consecutivos que ExperVision participó.
ABBYY FineReaderOCR	Comercial	Windows	Para trabajar con interfaces específicas, se requiere apoyo en el idioma correspondiente.
OmniPage	Comercial	Windows, Mac Os, Linux	Producto de Nuance Communications
Zonal OCR	Comercial	Windows	

Microsoft Office Document Imaging	Comercial	Windows, Mac Os, Linux	Microsoft Office Document Imaging permite a los usuarios escanear documentos en papel e importar los contenidos en los programas de Microsoft Office
Microsoft Office OneNote	Comercial	Windows	
TopOCR	Freeware	Windows	
FreeOCR	Freeware	Windows	
Readiris	Comercial	Windows	Producto de I.R.I.S. Grupo de Bélgica. Ediciones Asia y Oriente Medio
SmartZone (Zonal OCR)	Comercial	Windows	Zonal OCR es el proceso por el cual reconocimiento óptico de caracteres se aplica en zonas específicas de una imagen escaneada.
Computhink's ViewWise	Comercial	Windows	Sistema de gestión de documentos
BrainWare	Comercial	Windows	Plantilla para extracción y procesamiento de datos de documentos en cualquier sistema back-end de datos; facturas, declaraciones de remesas, conocimientos de embarque.
ReadSoft	Comercial	Windows	Escanear, capturar y clasificar los documentos de negocio tales como formularios, facturas y organización de producto.
Scantron Cognition	Comercial	Windows	Para trabajar con interfaces específicas, se requiere apoyo en el idioma correspondiente.
SmartScore	Comercial	Windows	Para partituras musicales.

3.- Objetivos

1-Estudiar y montar el sistema para desarrollar la aplicación.

2- Pruebas aisladas de cargado de imágenes, pre-procesamiento y procesamiento.

3- Implementar las primitivas del lenguaje relacionadas con el cargado de imágenes Implementar las primeras sentencias operativas del lenguaje de usuario.

4- Implementar las primitivas de procesamiento y salida.

5- Integración y pruebas.

4.- Solución

Para montar el sistema se ha utilizado como entorno de programación, Visual Studio 2013, como librería para el tratado de imágenes, OpenCV (C++) y EmguCV (C#),y para el reconocimiento de caracteres, un motor basado en el algoritmo KNN (k vecinos más próximos). En una primera fase del proyecto se ha trabajado en la consola de comandos, programando el proyecto en lenguaje C++ y posteriormente se ha adaptado el código a C#, para así, poder crear una interfaz e integrar todas las funciones.

La digitalización de la imagen se realiza con un escáner y se carga en el programa mediante un botón, el usuario selecciona la imagen en un explorador de archivos.

Antes de procesar la imagen, tenemos que eliminar el ruido y resaltar las características de interés, es decir, el texto. Este proceso se denomina pre-procesado y consiste en realizar una transformación a escala de grises, realizar un difuminado para suavizar los bordes y, por último, aplicar el método del valor umbral.

El primer paso para el procesamiento de la imagen es, segmentar el texto. Nuestra función está basada en el gradiente morfológico, a la sección de pre-procesado añadimos la función de gradiente morfológico para resaltar los bordes. Después de llamar a la función que encuentra los contorno se envía a otra función que comprueba que realmente sea texto.

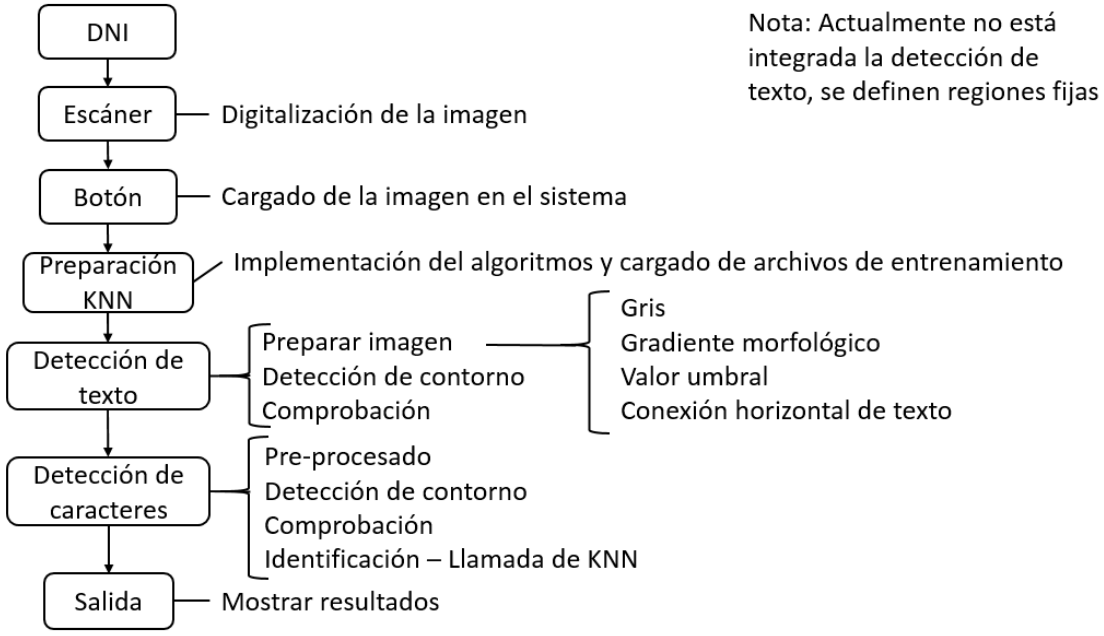
Lo que obtenemos, es la entrada de la detección de caracteres. El proceso es prácticamente el mismo, primero pre-procesamos la imagen, encontramos los bordes y enviamos los datos a una función que comprueba que los posibles caracteres sean realmente caracteres. Dicha función calcula la posición de los caracteres, lo que nos permite, calcular distancia entre caracteres y giro del documento.

Es momento de llamar al motor OCR, pero antes, explicar que para poder utilizarlo hemos de crear un programa aparte. Este programa lo denominados entrenamiento KNN, y consiste en enviar una imagen al sistema con 5 tipografías de número y letras, definiendo cada carácter con el teclado, podemos crear dos archivos que serán la base de nuestra clasificación. Implementando el algoritmo en el sistema y cargando los archivos somos capaces de identificar los caracteres, una vez realizadas las transformaciones de variable necesarias para llamar al algoritmo KNN. Los resultados, ya como texto, son mostrados en la interfaz del programa.

Es importante mencionar que no se ha podido integrar la detección de texto en la segunda parte del proyecto, las áreas de texto de interés las define un operario a mano.

Se ha añadido una función posterior para corregir el reconocimiento entre 0 y O, pues generaba problemas debido a su similitud.

El proceso para obtener información se muestra en el siguiente diagrama.



5.- Resultados y conclusiones

De cara a realizar una recapitulación del proyecto desarrollado, se puede determinar que los objetivos principales se han alcanzado satisfactoriamente gracias a un adecuado análisis y estudio previo al desarrollo. El diseño e implementación de las funciones de visión artificial ha cumplido con las expectativas, obteniendo unos resultados satisfactorios en las pruebas realizadas.

Respecto a las líneas de investigación que podrían abordarse en el futuro sobre la base del trabajo desarrollado, una debería ser, integrar la detección de texto en el programa para automatizar completamente el proceso de extracción de información. Y otra aproximación es, establecer una conexión con una base de datos para comprobar la información obtenida con datos del censo de la población.

Finalmente, podemos decir, que se ha conseguido una aplicación eficaz y eficiente, a la par que fiable y estable, con una curva de aprendizaje mínima y con la funcionalidad más utilizada a la vista, evitando complejos procedimientos para su uso.

ARTIFICIAL VISION FOR DOCUMENTS

Author: Arce Arroyo, Alberto.

Director: Talavera, Juan Antonio

Collaborating Organization: ICAI – Universidad Pontificia Comillas.

PROJECT SUMMARY

1.- Introduction

This Project consist in studying and developing an artificial vision system. We start from digitalizing images from DNIS, in order to apply OCR (Optical character recognition) techniques. The idea is to place this application in a traffic checkpoint to automatically record the information.

2.- Prior Art

Nowadays, we can find different OCR commercial applications. They are focus on recognition of impress characters

The following table shows the most important companies

Company	Licence	Operating system	Notes
ExperVision TypeReader & OpenRTK	Commercial	Windows, Mac Os, Linux	ExperVision Inc. was founding in 1987, its OCR technology and product, won the highest marks in the independent competition done by UNLV
ABBYY FineReaderOCR	Commercial	Windows	Made for working with specific interfaces, it needs support with the language.
OmniPage	Commercial	Windows, Mac Os, Linux	Nuance Communications product
Zonal OCR	Commercial	Windows	
Microsoft Office Document Imaging	Commercial	Windows, Mac Os, Linux	Microsoft Office Document Imaging allows to scan paper documents and send the data to Microsoft Office

Microsoft Office OneNote	Commercial	Windows	
TopOCR	Freeware	Windows	
FreeOCR	Freeware	Windows	
Readiris	Commercial	Windows	I.R.I.S. Grupo de Bélgica. Ediciones Asia y Oriente Medio product
SmartZone (Zonal OCR)	Commercial	Windows	Zonal OCR is a product to recognize characters in specific part of the scanned document.
BrainWare	Commercial	Windows	Template for extracting and processing data in any back-end data; bills, remittance statements
ReadSoft	Commercial	Windows	Scan, capture and classify business documents such as inventory, bills.
Scantron Cognition	Commercial	Windows	Made for working with specific interfaces, it needs support with the language.
SmartScore	Commercial	Windows	For musical papers

3.- Objectives

1- Study and build a system for application developing

2- Isolated tests for uploading images, pre-processing and processing.

3- Implement different functions related with uploading images, pre-processing.

4- Implement functions for processing and showing the results

5- Integrations and optimizations

4.- Solution

For building the system, we have used Visual Studio 2013 as programming environment, for manage images we used OpenCV (C++) and EmguCV (C#), and for extract the information, we have used an OCR motor based on KNN algorithm (K nearest neighbour). As a first step, we start programming the application using C++, afterward we can adapt the code to C#, in order to, integrated all the different functions and develop an interface.

The image digitalization was made with a scanner; we can upload the image into the system using the button shown in the interface.

Before processing the image, we have to clear the noise and show important characteristics, as text. This process is called, pre-processing and it is based on different transformations. In our case, we start changing the image to gray-scale, followed by blur the image and finally use the threshold method.

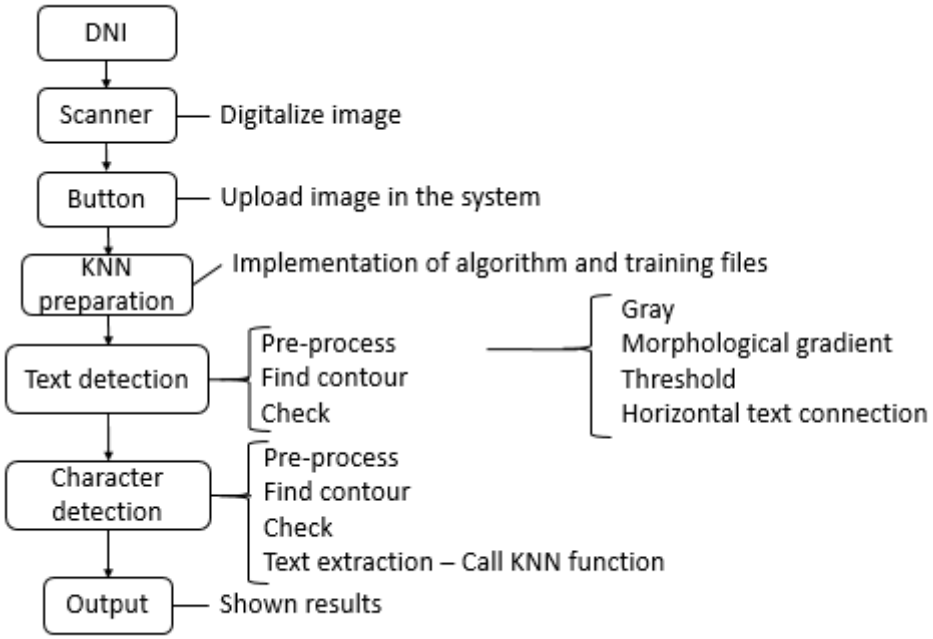
The first step for processing is detect the text. Our function for that, it is based on morphological gradient. To do that, we include the morphological gradient in the pre-processing part, we call the function to find the contours and finally we send the data to a check function.

The output of text detection goes as an input to char detection. This process is almost the same as before, first we pre-process the image, second we find the contours and finally we call the check function. This function, allows us to calculate the exact position of a particular character, so, we can calculate distance between chars, and the rotation angle in the image.

It is time to call the OCR motor, but before, we have to explain how to train it and implement it. In a different program, we send an image with 5 different typographies, we specify via keyboard which image correspond to which character. With this training program we obtain two different files, one contains the images and the other one contains the classification. Once we have the two training files upload in the main system, we can call the function to extract the information, now, the information is a string variable. In order to optimize the results, we implement a function to distinguish between 0 and O. Finally, the results are displayed in the interface.

It is important to say that we cannot integrated de text detection in the second part of the Project (C#), so, the text areas are defined by one worker.

The following diagram shown how it is done the process



5.- Results

To conclude, we have to do a recap of the project. We can say that we success with the main objectives, thank you to have a good preparation, studying and analysing the necessary information. The design and implementation of the functions of artificial vision have reach the expectation, obtaining a successful result in the test done.

On the other hand, focusing on future developments, we should integrate the text detection in order to automatize the process. One important point, is to establish a connection with a database, the state is responsible of the people registration, so the connections should be with a governmental database.

Finally, we can say that we could develop and efficient and stable application, with an easy way to learn and manage, avoiding difficult procedures for the user.

Índice de la memoria

Índice de figuras

Índice de tablas

Parte I MEMORIA

Capítulo 1 INTRODUCCIÓN

El proyecto consiste en estudiar y desarrollar un sistema de visión artificial, se puede definir la "Visión Artificial" como un campo de la "Inteligencia Artificial" que, mediante la utilización de las técnicas adecuadas, permite la obtención, procesamiento y análisis de cualquier tipo de información especial obtenida a través de imágenes digitales.

Se partirá de imágenes de documentos provenientes de cámaras o de escáner (DNIs), se trata de aplicar diferentes técnicas de visión artificial usadas en robots industriales para identificar el contenido del documento fotografiado. En este proyecto nos centramos en técnicas de visión OCR (Reconocimiento óptico de caracteres), a partir de la cual podemos generar un fichero de texto donde nos aparezca la información del propietario, automatizando así su registro.

Se va a desarrollar las primitivas del lenguaje de visión para el reconocimiento de DNIs en un control de entrada en una zona de alta seguridad para que, en un estado posterior del proyecto, el programador de robots, junto con la cámara y robot identifique la información de dicho documento.

Capítulo 2 ESTADO DE LA CUESTIÓN

2.1 HISTORIA

En 1935 se concede la primera patente sobre OCR a Tauschek en EEUU. La máquina de Tauschek era un dispositivo mecánico que utilizaba plantillas. Un foto-detector era colocado de modo que cuando la plantilla y el carácter que se reconocería estuvieran alineados, una luz era dirigida hacia ellos.

En 1950, David Shepard, criptoanalista en la agencia de seguridad de las fuerzas armadas de los Estados Unidos, fue consultado para recomendar los procedimientos de la automatización de los datos de la agencia, es decir, convertir mensajes impresos en lenguajes para almacenarlos en un ordenador. Shepard junto con Harvey, un amigo, deciden construir un prototipo. En este momento, Shepard fundó Intelligent Machines Research Corporation (IMR), comenzando a fabricar el primero de varios sistemas del OCR usados para operaciones comerciales.

El servicio postal de Estados Unidos ha estado utilizando las máquinas de OCR para clasificar el correo desde 1965, mismo año, en el que aparecer el primer uso en Europa, fue en Gran Bretaña, un sistema de actividades bancarias que revolución el sistema de pago de cuentas. El correo postal de Canadá ha estado utilizando sistemas OCR desde 1971. Los sistemas OCR leen el nombre y la dirección del destinatario, e imprimen un código de barras en el sobre basados en el código postal del mismo. [JONA08]

2.2 ESTUDIO DE LOS TRABAJOS EXISTENTES / TECNOLOGÍAS EXISTENTES. [ORDO09]

Existen en la actualidad gran cantidad de programas OCR comercializados. Están principalmente destinados al reconocimiento de caracteres impresos. Su tasa de reconocimiento normalmente se halla entre 80% y 95%, obteniendo desde luego los mejores resultados cuando funcionan con tipos de letra para los que han sido "afinados". Los sistemas comerciales actuales generan un fichero de texto que posteriormente puede ser modificado por el usuario.

La siguiente tabla representa un listado de las principales aplicaciones comerciales que hacen uso de la tecnología OCR.

Compañía	**Licencia**	**Sistema operativo**	**Notas**
ExperVision TypeReader & OpenRTK	Comercial	Windows, Mac Os, Linux	ExperVision Inc. fue fundada en 1987, su tecnología OCR y producto, ganó las más altas calificaciones en las pruebas independientes realizadas por UNLV para los años consecutivos que ExperVision participó.
ABBYY FineReaderOCR	Comercial	Windows	Para trabajar con interfaces específicas, se requiere apoyo en el idioma correspondiente.
OmniPage	Comercial	Windows, Mac Os, Linux	Producto de Nuance Communications
Zonal OCR	Comercial	Windows	
Microsoft Office Document Imaging	Comercial	Windows, Mac Os, Linux	Microsoft Office Document Imaging permite a los usuarios escanear documentos en papel e importar los contenidos en los programas de Microsoft Office
Microsoft Office OneNote	Comercial	Windows	
TopOCR	Freeware	Windows	

FreeOCR	Freeware	Windows	
Readiris	Comercial	Windows	Producto de I.R.I.S. Grupo de Bélgica. Ediciones Asia y Oriente Medio
SmartZone (Zonal OCR)	Comercial	Windows	Zonal OCR es el proceso por el cual reconocimiento óptico de caracteres se aplica en zonas específicas de una imagen escaneada.
Computhink's ViewWise	Comercial	Windows	Sistema de gestión de documentos
BrainWare	Comercial	Windows	Plantilla para extracción y procesamiento de datos de documentos en cualquier sistema back-end de datos; facturas, declaraciones de remesas, conocimientos de embarque.
ReadSoft	Comercial	Windows	Escanear, capturar y clasificar los documentos de negocio tales como formularios, facturas y organización de producto.
Scantron Cognition	Comercial	Windows	Para trabajar con interfaces específicas, se requiere apoyo en el idioma correspondiente.
SmartScore	Comercial	Windows	Para partituras musicales.

Tabla 1. Aplicaciones comerciales que ofertan OCR

2.3 APROXIMACIONES PARA EL RECONOCIMIENTO

2.3.1 KNN

Existe un método muy conveniente, no paramétrico y supervisado, que proporciona resultados muy adecuados para la aplicación que se está tratando, El algoritmo K-NN (K vecinos más próximos). Este método es muy popular debido a su sencillez y a cierto número de propiedades estadísticas bien conocidas que le proporcionan un buen comportamiento para afrontar diversos tipos de problemas de clasificación, siendo uno de ellos el de OCR.

Como se ha dicho anteriormente, es un método de clasificación supervisada no paramétrica, que estima el valor de la función de densidad de probabilidad o

directamente la probabilidad a posteriori, de que un elemento X pertenezca a la clase C_j a partir de la información proporcionada por el conjunto de prototipos.

En nuestro caso, en el reconocimiento de patrones, el algoritmo KNN es usado como método de clasificación de objetos (caracteres) basado en un entrenamiento mediante ejemplos cercanos en el espacio de los elementos. Es tipo "Lazy Learning" (Aprendizaje perezoso), donde la función se aproxima solo localmente y todo el cómputo es diferido a la clasificación.

Proceso de clasificación [SIER07]

1. Se elige un número de vecinos próximos (k).
2. Se elige una métrica, es decir, una función para calcular la distancia entre dos ejemplos.
3. Para cada ejemplo x:
    a. Se calcula la distancia al resto de los ejemplos.
    b. Se seleccionan los k vecinos más cercanos.
    c. La clase de x es la más representada entre estos k.
    d. Resolución de empates. Si coincide el número de vecinos de dos o más clases, se escoge la clase con mayor probabilidad a priori. Si las probabilidades a priori coinciden, se escoge una de las clases en disputa al azar.

La siguiente imagen muestra un ejemplo.

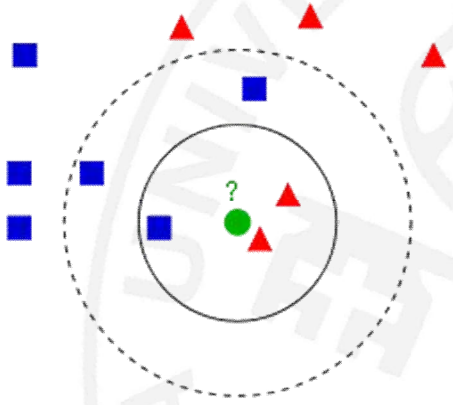


Figura 1. Ejemplo del algoritmo KNN

Para K=3, se clasifica como triangulo, mientras que para K=5, sería cuadrado.

2.3.2 ARBOLES DE DECISIÓN [SA_09]

Los árboles de decisión, al igual que el K-NN, es una técnica de minería de datos que se puede aplicar en el contexto de reconocimiento óptico de caracteres. Su aprendizaje es inductivo y no supervisado. Los patrones o atributos que se quieren evaluar de un carácter determinado constituyen los nodos del árbol, mientras que los resultados finales de los mismos se almacenarán en las hojas del mismo.

La secuencia de aprendizaje del árbol se puede resumir según el siguiente esquema:

1. Documento
2. Proceso de escaneado
3. Segmentación y normalización
4. Obtención de las características binarias
5. Construcción óptima del árbol binario
6. Calculo de patrones

Una vez se tiene construido el árbol hay que detallar como recorrerlo. En definitiva, si durante el recorrido se llega a una hoja, esa serie de patrones responden a un carácter reconocido y por tanto se devuelve. Si no se llega a ninguna hoja, esa característica no ha sido creada y por tanto se debe crear una nueva para guardas el nuevo dato. Un esquema de cómo recorrer un árbol de decisión es el siguiente:

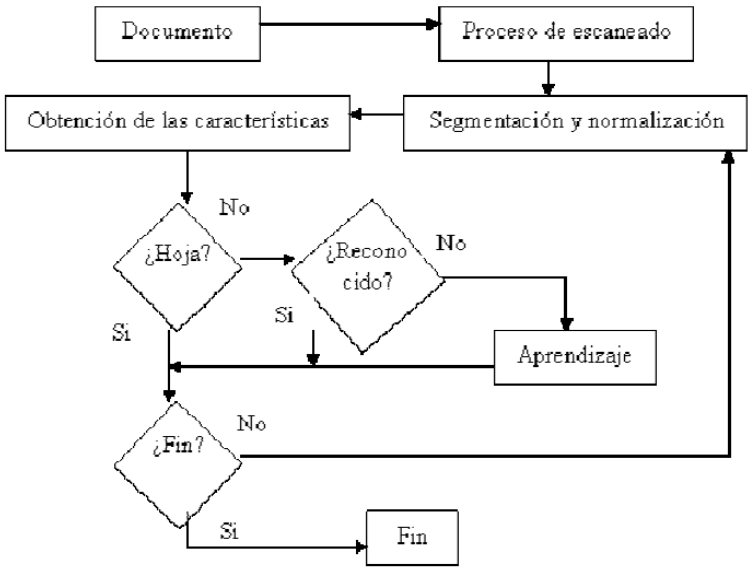


Figura 2. Reconocimiento de caracteres en arboles de decisión

2.3.3 REDES NEURONALES

Las redes neuronales son esquemas que intentan imitar la arquitectura del cerebro. Se componen de una serie de unidades básicas, llamadas neuronas, que básicamente reciben una entrada, la multiplican por unos pesos y presentan una salida con una función de ajuste. Sirven para representar y ajustar muy eficazmente cualquier función que sería muy difícil de definir en términos algebraicos. En el caso de OCR el aprendizaje es supervisado.

La topología de las redes puede ser muy variada y según esta topología las redes se pueden clasificar en:

- Feed Fordward: Red clásica, las salidas alimentan las entradas de la etapa siguiente, en la red no se permite la aparición de ciclos o realimentaciones
- Feed Back: En esta versión si se permiten ciclos cerrados dentro de la red

- Lateral: Adicionalmente, se permite la comunicación vertical entre neuronas de la misma capa, además de en la dirección horizontal, como en las redes clásicas.

2.4 ANÁLISIS DE HERRAMIENTAS

2.4.1 MOTORES OCR [ORDO09]

Antes de iniciar el desarrollo del proyecto es importante realizar un estudio de las herramientas de software y lenguajes de programación existentes en el mercado. Este estudio es importante para conocer las posibilidades existentes en el mercado actual y tener así un conocimiento previo de las herramientas de software con las que se va a trabajar, una visión general, de esta forma se logra implementar de la manera más sencilla y funcional el programa que tratara imágenes.

En este punto se analiza la situación actual de los proyectos SL más importantes encargos de la creación, mantenimiento y evolución de software destinado a realizar labores OCR.

Librería	**Website**	**Última Version**	**Última actualización**	**Lenguaje**	**Licencia**
GOCR	http://jocr.sourceforge.net/index.html	0.50	05/03/2013	C/C++	GPL v2
JavaOCR	http://sourceforge.net/projects/javaocr	1.0	25/10/2012	Java	BSD License
Ocrad	http://www.gnu.org/software/ocrad	0.25	08/04/2015	C/C++	GPL v3
Tesseract	http://github.com/tesseract-ocr	3.02	02/02/2012	C/C++	Apache License 2.0
Cuneiform	http://launchpad.net/cuneiform-linux/	1.1	19/04/2011	C/C++	Simplified BSD

KNN					

Tabla 2. Principales proyectos destinado a labores OCR

A continuación, se describen brevemente las librerías enumeradas en la tabla anterior

1. GOCR.

   GOCR es un programa de OCR desarrollado bajo la Licencia Pública GNU. Convierte las imágenes escaneadas que contienen texto a archivos de texto. GOCR se puede usar con diferentes "front-end", lo que hace que sea muy fácil de portar a diferentes sistemas operativos y arquitecturas.

2. JavaOCR.

   Este es un motor genérico OCR que se puede entrenar. Por defecto no es capaz de realizar la detección y extracción de caracteres. Sin embargo, es capaz de filtrar y limpiar la imagen, convertir a escala de grises, dividir el documento en líneas, dividir las líneas en caracteres, y finalmente comparar cada carácter con los patrones conocidos de las imágenes de capacitación proporcionados por el usuario, y obtener como salida las opciones más cercanas como texto.

3. Ocrad

   Es un programa de OCR basado en un método de extracción de características. Ocrad es capaz de leer imágenes en diferentes formatos como "pbm", "pgm" o "ppm", y produce texto en formato UTF-8. También incluye un analizador de composición capaz de separar las columnas o bloques de texto que forman normalmente las páginas impresas. Ocrad puede ser usado como aplicación autónoma desde la línea de comandos o como complemento de otros programas

4. Tesseract.

   El motor de OCR Tesseract fue uno de los 3 mejores motores en la prueba de Precisión 1995 UNLV. Entre 1995 y 2006 tuvo poca evolución, pero desde entonces se ha mejorado notablemente con la colaboración de

Google y es probablemente uno de los motores más precisos de código abierto. En combinación con la librería de procesamiento de imágenes Leptonica es capaz de leer una amplia variedad de formatos de imagen y convertirlos a texto en más de 40 idiomas.

Proceso

1. Análisis de los componentes conectados y almacenamiento. Reconocer texto invertido, así como texto blanco en fondo negro.
2. Se organizan las líneas de texto según los contornos. Se analizan las regiones a paso fijo.
3. Las líneas de texto se dividen en palabras según el tipo de espacio entre caracteres.
    a. Texto de tamaño fijo se corta inmediatamente por celdas de caracteres
    b. Texto proporcional se divide en palabras usando espacios definidos y difusos
4. El reconocimiento consta de tres pasos.
    a. Se realiza un intento para reconocer las palabras. Cada palabra satisfactoria pasa a un clasificador adaptativo, donde se tiene la oportunidad de realizar el reconocimiento de forma más precisa.
    b. Se realiza una segunda pasada, utilizando lo aprendido por el clasificador adaptativo.

4. Cuneiform.

Cuneiforme es un sistema OCR originalmente desarrollado como código abierto y basado en tecnologías cognitivas. Este proyecto tiene como objetivo crear una versión totalmente portátil de la escritura cuneiforme.

2.4.2 BIBLIOTECAS GENÉRICAS VISIÓN ARTIFICIAL

Este último punto, no trata sobre un software para el reconocimiento óptico de caracteres, sino de una librería general para el tratamiento digital de imágenes, que nos puede ayudar a realizar un pre-procesado de las imágenes previo al tratamiento para OCR. Las aplicaciones que se comentan van más allá del simple reconocimiento óptico de caracteres, por lo que integran diferentes módulos para ofrecer funcionalidades añadidas. Estas aplicaciones integran alguno de los motores comentados anteriormente, sobre los que delegan la labor propia del reconocimiento óptico de caracteres.

Librería	Website	Ultima Version	Ultima actualización	Lenguaje	Licencia
Open CV	http://opencv.org/	3.1	21/12/2015	C/C++	BSD License
Simple CV	http://simplecv.org/	1.3	07/04/2015	C/C++/Python	BSD License

Tabla 3. Principales librerías para visión artificial

1. OpenCV.

   Biblioteca libre de visión artificial originalmente desarrollada por Intel. Desde que apareció su primera versión alfa en el mes de enero de 1999, se ha utilizado en infinidad de aplicaciones, desde sistemas de seguridad con detección de movimiento, hasta aplicativos de control de procesos donde se requiere reconocimiento de objetos. Esto se debe a que su publicación se da bajo licencia BSD, que permite que sea usada libremente para propósitos comerciales y de investigación con las condiciones en ella expresadas.

2. SimpleCV.

SimpleCV es una biblioteca libre de visión artificial, con la cual se tiene acceso a varias librerías como puede ser OpenCV sin tener un gran conocimiento de uso como son, formato de uso, espacios de color, manejo de buffer o auto valores.

2.4.3 HERRAMIENTAS GENÉRICAS VISIÓN ARTIFICIAL

1. Matlab.

MATLAB es una herramienta de software matemático que ofrece un entorno de desarrollo integrado (IDE) con un lenguaje de programación propio (lenguaje M). Está disponible para las plataformas Unix, Windows, Mac OS X y GNU/Linux. El paquete MATLAB dispone de dos herramientas adicionales que expanden sus prestaciones, a saber, Simulink (plataforma de simulación multidominio) y GUIDE (editor de interfaces de usuario - GUI). Además, se pueden ampliar las capacidades de MATLAB con las cajas de herramientas (toolboxes); y las de Simulink con los paquetes de bloques (blocksets).

2.5 ANÁLISIS

2.5.1 COMPARATIVA MOTORES OCR [NAVA13]

En este apartado se comenta el proceso seguido en la elección de la librería de SL, para ello, se realiza un estudio analítico de las diferentes librerías, motores OCR, que se pueden utilizar en función de los siguientes apartados:

1. Eficacia. Lógicamente debemos seleccionar aquella librería que consiga realizar correctamente o con menor margen de error el proceso de OCR.

	Prueba 1	Prueba 2	Prueba 3	Prueba 4
Ocrad	97,51%	98,81%	97,23%	98,90%
GOCR	95,78%	96,23%	95,97%	96,78%

Tesseract	98,34%	99,21%	98,86%	99,04%
JavaOCR	86,38%	85,48%	87,48%	85,96%

Tabla 4. Comparativa de eficacia

2. Adaptabilidad. También es interesante tener una librería que sea fácilmente adaptable para otras finalidades u otros objetivos concretos, por ejemplo, sería interesante poder adaptarse con facilidad a otros idiomas u otras fuentes de caracteres.

	Caracteres	**Tipografias**	**Palabras frecuentes**	**Caracteres permitidos**	**Caracteres prohibidos**
Ocrad	NO	NO	NO	NO	NO
GOCR	NO	SI	NO	NO	NO
Tesseract	SI	SI	SI	SI	SI
JavaOCR	SI	SI	NO	NO	NO

Tabla 5. Comparativa de adaptabilidad

3. Eficiencia. También debemos tener en cuenta la eficiencia de las librerías, es decir, el coste en recursos (CPU, RAM, etc.) que necesita para realizar la tarea de OCR.

	Prueba 1	**Prueba 2**	**Prueba 3**	**Prueba 4**
Ocrad	0,75	0,82	0,79	0,75
GOCR	0,89	0,86	0,92	0,87
Tesseract	2,37	1,98	2,23	2,11
JavaOCR	2,12	1,82	2,25	2,17

Tabla 6. Comparativa de eficiencia

4. Licencia. Este aspecto puede definir la finalidad del software que se desea construir, es decir, si podrá ser utilizada dentro de aplicaciones comerciales o únicamente podremos realizar aplicaciones de SL con ella. Si nuestra librería se basa en software que está licenciado bajo una licencia robusta (por ejemplo, GPL) cualquier aplicación que haga uso de ella estará obligada a ser distribuida bajo esta misma licencia. Si por el contrario deseamos crear una librería que pueda ser utilizada por aplicaciones comerciales debemos basarnos en alguna de las librerías que

estén licenciadas bajo alguna licencia permisiva, por ejemplo, la licencia BSD o licencia Apache 2.0.

	Licencia	Tipo
Ocrad	GPL v3	Robusta
GOCR	GPL v3	Robusta
Tesseract	Apache License 2.0	Permisiva
JavaOCR	BSD License	Permisiva

Tabla 7. Comparativa de licencia

2.5.2 COMPARATIVA LIBRERÍAS DE VISIÓN ARTIFICIAL [SIMPSF]

Se realiza una comparativa entre las principales librerías genéricas de visión artificial y/o herramientas genéricas, las cuales son Matlab, OpenCV y Simple CV. En la siguiente tabla se valora numéricamente (0 -10) las principales características de uso, siendo 10 la mejor valoración.

	Matlab	OpenCV	SimpleCV
Facilidad de uso	9	3	10
Velocidad	2	9	5
Recursos necesitados	4	9	8
Precio	4	10	10
Entorno de desarrollo	8	6	7
Gestión de memoria	9	4	9
Movilidad	3	8	6
Debugging	9	5	9
Comunidad	8	9	10

Tabla 8. Comparativa total

2.5.3 CONCLUSIÓN

Después de valorar todos los aspectos observamos que la librería que mejor se adapta a nuestra necesidad es la librería OpenCV de cara a realizar un primer procesado de imágenes. Comentar que aun obteniendo mejor puntuación la librería Simple CV, según datos de un foro de Simple CV, hemos tomado la decisión de utilizar Open CV porque ofrece más funcionalidad, así como mayor comunidad.

La librería Tesseract se encargará del reconocimiento del texto una vez la imagen haya sido tratada. En el caso de no poder utilizar la librería Tesseract, desarrollamos nuestro propio motor OCR utilizando el algoritmo KNN. Los motivos por los que se ha llevado a cabo dicha decisión se comentan en el siguiente apartado, motivación.

2.6 MOTIVACIÓN

Uno de los sentidos más importantes de los seres humanos es la visión. Ésta es empleada para obtener la información visual del entorno físico. Según Aristóteles, "Visión es saber que hay y donde mediante la vista". De hecho, se calcula que más de 70% de las tareas del cerebro son empleadas en el análisis de la información visual. El refrán popular de "Una imagen vale más que mil palabras" tiene mucho que ver con los aspectos cognitivos de la especie humana. La visión humana es el sentido más desarrollado y el que menos se conoce debido a su gran complejidad. Es una actividad inconsciente y difícil de saber cómo se produce. De hecho, hoy en día, se carece de una teoría que explique cómo los humanos perciben el exterior a través de la vista.

Desarrollar un sistema donde se automatice una tarea que solo pueden hacerlo personas, como es el reconocimiento por visión, puede ayudar a agilizar el tránsito de clientes en controles de acceso de alta seguridad, como aeropuertos, ahorrar

costes de personal y tiempo, evitar errores humanos en el registro de históricos, al mismo tiempo mejora la satisfacción del cliente.

Los motivos que nos han llevado a la decisión del utilizar en primera instancia la librería Tesseract + OpenCV son los siguientes:

- Mayor eficacia.
- Mayor adaptación, la librería Tesseract ofrece las mayores opciones de adaptación y aprendizaje.
- Licencia permisiva que ofrece más margen de maniobra para adaptaciones o futuros usos comerciales.
- Gran comunidad, lo que implica actualizaciones más frecuentes y mayor facilidad para solucionar errores y correcciones.
- Mayor velocidad.
- Menos recursos del equipo utilizado.
- Precio, se desea desarrollar un sistema de identificación de documentos con el menor precio posible.

Debido a problemas de compatibilidad con la versión del sistema operativo utilizado, Windows 10, por consiguiente, problemas de compatibilidad con versiones de Visual Studio, no se pudo continuar una primera versión de la aplicación. En dicho momento se decidió tomar otro camino a la hora de reconocer caracteres, la decisión fue utilizar el algoritmo KNN, los motivos que finalmente nos llevaron a tomar dicha decisión son los siguientes:

- Mayor control de los procesos internos de reconocimiento de caracteres
- Mayor adaptación, desarrollar nuestro propio motor OCR nos permite ser más flexibles
- Licencia permisiva que ofrece más margen de maniobra para adaptaciones o futuros usos comerciales.
- Menos recursos del equipo utilizado.

- Precio, se desea desarrollar un sistema de identificación de documentos con el menor precio posible.

Capítulo 3 OBJETIVOS Y DESARROLLO DEL PROYECTO

1. Estudiar y montar el sistema para desarrollar la aplicación.
    a. Instalación de herramientas y adición de librerías
    b. Toma de imágenes
    c. Entrenamiento KNN
2. Pruebas aisladas de cargado de imágenes y pre-procesamiento.
    a. Cargar imágenes y comprobaciones
    b. Pre-procesamiento
3. Pruebas aisladas de procesamiento de imágenes.
    a. Segmentación de texto
    b. Reconocimiento de caracteres
4. Implementar las primitivas del lenguaje relacionadas con el cargado de imágenes, el pre-procesamiento y sentencias operativas del lenguaje de usuario.
    a. Cargado de imágenes
    b. Pre-procesamiento
    c. Sentencias operativas del lenguaje de usuario
5. Implementar las primitivas de procesamiento y salida.
    a. Detección de texto
    b. Salida
6. Integración y pruebas.
    a. Integración

b. Pruebas y resultados con diferentes imágenes y diferentes configuraciones del programa

7. Optimizaciones.
    a. Integración de la detección de texto

3.1 ESTUDIAR Y MONTAR EL SISTEMA PARA DESARROLLAR LA APLICACIÓN

En este apartado se muestra el procedimiento para montar el sistema sobre la que se programa la aplicación. En nuestro caso, se va a utilizar Visual Studio 2013 Community Edition. Se ha decidido utilizar la versión 2013 del programa debido a la ausencia de compatibilidad con la versión 2010 (versión con mayor comunidad). El procesado de imágenes, como ya se explicó en el capítulo 2, se implementará con la librería Open CV. En nuestro caso se utilizará la versión 3.0.0, tener en cuenta que, la versión que utilicemos no influye significativamente en las instrucciones a realizar, aunque siempre se recomienda hacer uso de la versión más actual.

Para facilitar el desarrollo de la aplicación, se empezará programando un proyecto en lenguaje C++, para posteriormente, programar un proyecto en Visual Basic, C#, donde se integrarán todas las funciones mostradas en una interfaz.

3.1.1 PROCEDIMIENTO PARA PROYECTOS VISUAL C++

1. Descargar e instalar Visual Studio 2013 Community Edition. El programa es gratuito y las opciones de instalación predeterminadas funcionaran para nuestra aplicación.

   Página web de descarga: www.visualstudio.com
2. Descargar la última versión de OpenCV, en nuestro caso la versión 3.0.0.

   Página web de descarga: www.opencv.org/

3. Crear una carpeta "C:/OpenCV-*X.X.X*", donde XXX es la versión de OpenCV. En nuestro caso, por ejemplo, "C:/OpenC-3.0.0".

   Extraemos los archivos de la última versión de OpenCV que hemos descargado.

4. Añadir el directorio **bin** de nuestra versión de OpenCV al **PATH** del sistema operativo.

   Nota: En las carpetas de OpenCV, "vc12" se refiere a Visual Studio 2013.

   En nuestro caso, por ejemplo, utilizando OpenCV 3.0.0 y Visual Studio 2013, la ruta que añadiremos al PATH es:

   "C:\OpenCV-3.0.0\opencv\build\x86\vc12\bin"

   El procedimiento es el siguiente (Windows):

   a. Accedemos al panel de control
   b. Sistema y seguridad
   c. Sistema

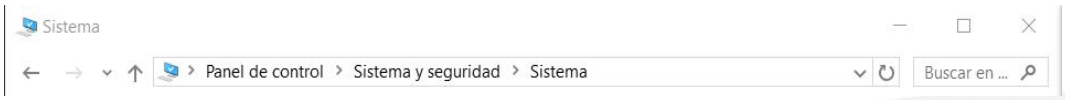


Figura 3. Ruta para acceder a las propiedades avanzadas del sistema

   d. Configuración avanzada del sistema
   e. Variables de entorno

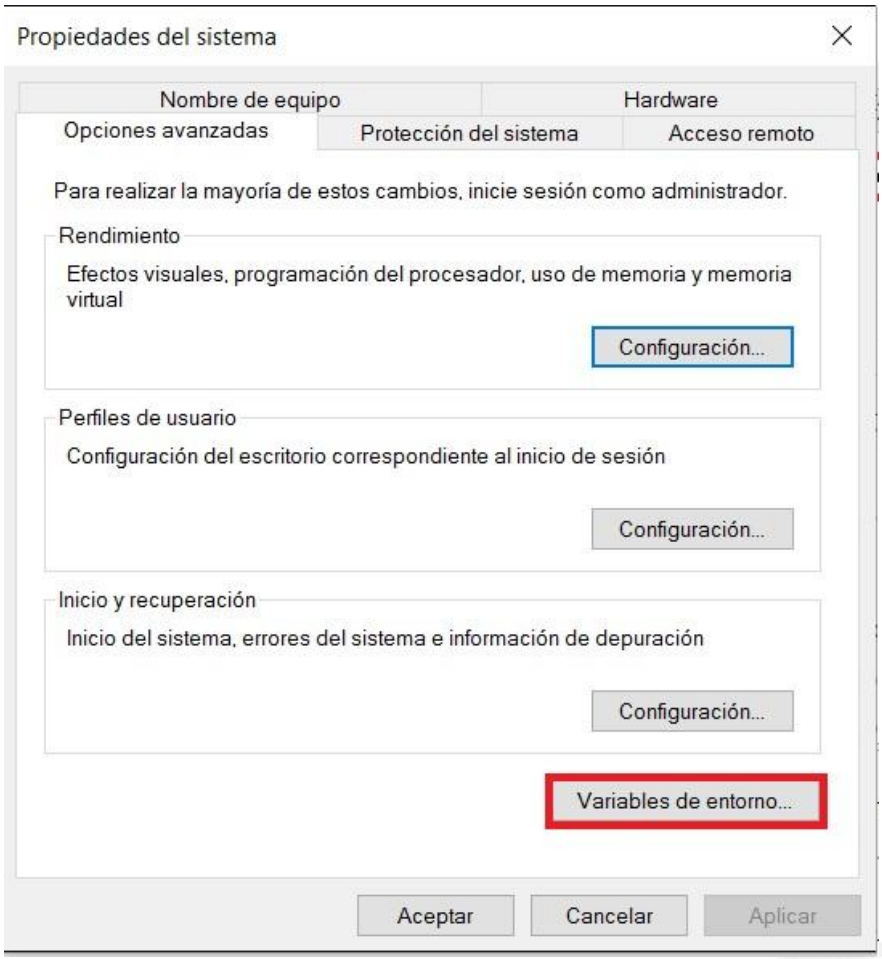


Figura 4. Ruta para acceder a las variables de entorno.

f. Seleccionamos la variable del sistema PATH y damos a editar

g. Finalmente, añadimos la ruta. En nuestro caso

"C:\OpenCV-3.0.0\opencv\build\x86\vc12\bin"

5. Ya podemos crear nuestro primer proyecto y empezar a programar

   a. Abrir Visual Studio, elegir File -> New -> Project

   b. Elegir "C++ File" y seleccionar la locación donde se guardará.

   c. En la barra de herramientas de Visual Studio, verificar que "Solution Configurations" está configurada para "x86"

   d. En VS ir a:

"Project -> Properties -> Configuration Properties -> VC++ Directories -> Include Directories" y añadir la librería. En nuestro caso, por ejemplo, "C:\OpenCV-3.0.0\opencv\build\x86\include"

"Project -> Properties -> Configuration Properties -> VC++ Directories -> Library Directories" y añadir la librería. En nuestro caso, por ejemplo, "C:\OpenCV-3.0.0\opencv\build\x86\vc12\lib"

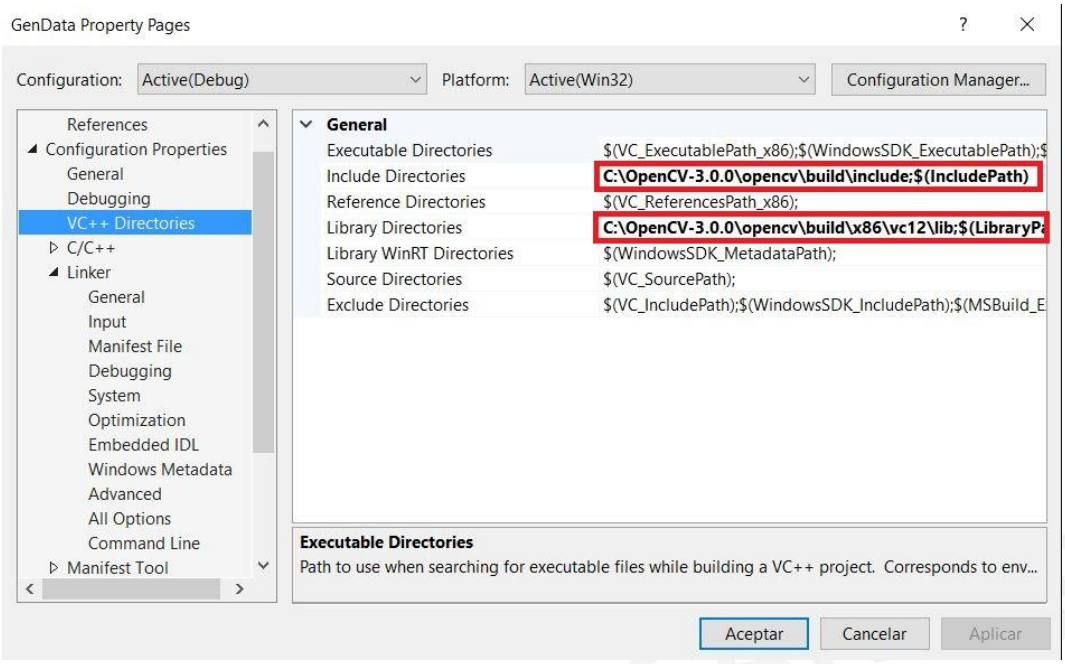


Figura 5. Añadir directorios en Visual Studio

e. Con el explorador de archivos, ir a la dirección lib, ejemplo "C:\OpenCV-3.0.0\opencv\build\x86\vc12\lib"

   En el directorio lib, se puede encontrar las librerías debug (acabadas con una "d"), por ejemplo, en nuestro caso, utilizando la versión 3.0.0, encontramos:

   opencv_ts300d.lib

   opencv_world300d.lib

Copiar y pegar el nombre de las librerías debug en el siguiente directorio de Visual Studio.

"Project -> Properties -> Configuration Properties -> Linker -> Input -> Additional Dependencies"

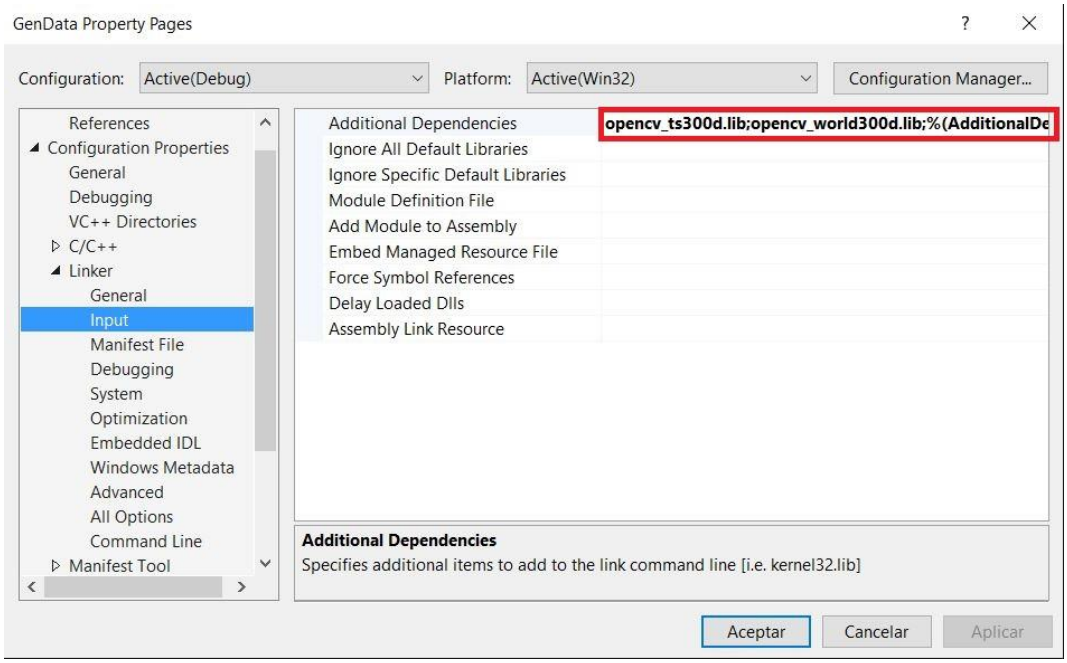


Figura 6. Añadir depencias adicionales para la depuración en Visual Studio

6. Ya se pueden programar proyectos en C++ con Visual Studio.

   Nota: Tener en cuenta que si se quiere depurar mediante "reléase" y no mediante "debug", los archivos seleccionados en el paso 5.e. han de ser sin la última letra "d".

3.1.2 PROCEDIMIENTO PARA PROYECTOS VISUAL C#

1. Descargar e instalar Visual Studio 2013 Community Edition (mismo paso que en 5.1.1)
2.
   a. Descargar la última versión del instalador Emgu CV versión 3

Página web de descarga: www.emgu.com/wiki/index.php/Main_Page

b. Ejecutar el instalador. Se creará una carpeta en el directorio de nuestro disco duro.

3. Añadir el directorio bin al PATH del sistema

   El procedimiento es el mismo realizado en el apartado 4 del punto 5.1.1:

   a. Accedemos al panel de control
   b. Sistema y seguridad
   c. Sistema
   d. Configuración avanzada del sistema
   e. Variables de entorno
   f. Seleccionamos la variable del sistema PATH y damos a editar
   g. Finalmente, añadimos la ruta. En nuestro caso:

      "C:\Emgu\emgucv-windesktop 3.1.0.2282\bin\x86"

   h. Reiniciar

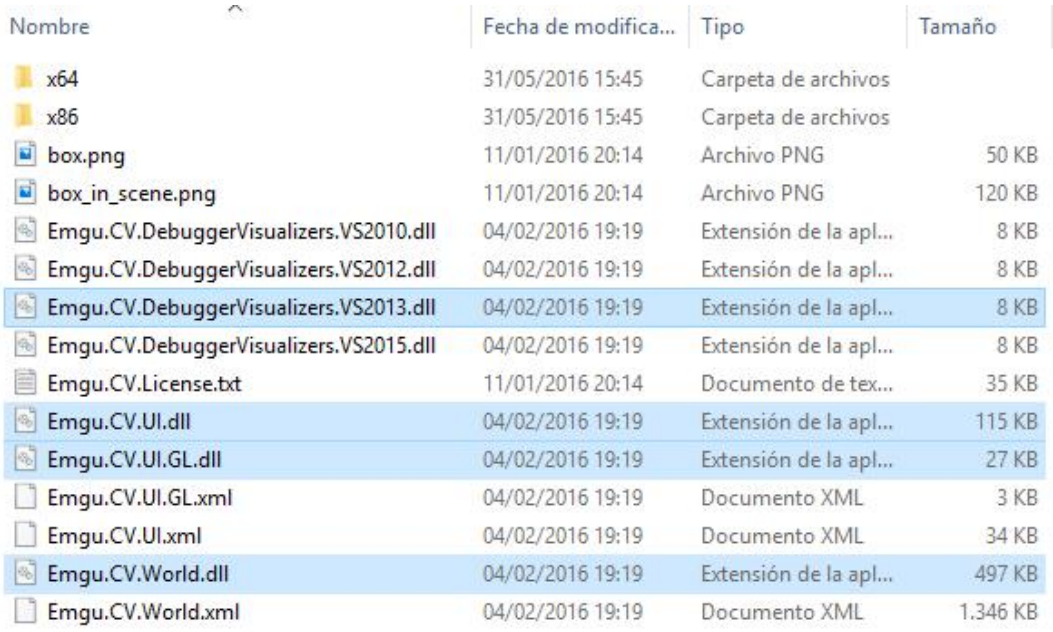
4. Ya podemos crear nuestro primer proyecto

   a. Abrir Visual Studio, elegir File -> New -> Project
   b. Elegir "Visual Basic" o "Visual C#", seleccionar la locación donde se guardará. Recomiendo des-seleccionar "Create directory for solution" y "Add to source control"
   c. En la barra de herramientas de Visual Studio, verificar que "Solution Configurations" está configurada para "x86"
   d. Añadir archivos necesarios para compilar el proyecto I, para ello:

      Ir a "Project -> Add Reference -> Browse -> Browse…"

      Navegar al directorio bin de Emgu, en nuestro caso, por ejemplo, seleccionar "C:\Emgu\emgucv-windesktop 3.1.0.2282\bin"

Seleccionar todos los DLLs que empiezan con "Emgu", a excepción de una versión diferente de nuestro Visual Studio. En nuestro caso resulta:



Nombre	Fecha de modifica...	Tipo	Tamaño
x64	31/05/2016 15:45	Carpeta de archivos	
x86	31/05/2016 15:45	Carpeta de archivos	
box.png	11/01/2016 20:14	Archivo PNG	50 KB
box_in_scene.png	11/01/2016 20:14	Archivo PNG	120 KB
Emgu.CV.DebuggerVisualizers.VS2010.dll	04/02/2016 19:19	Extensión de la apl...	8 KB
Emgu.CV.DebuggerVisualizers.VS2012.dll	04/02/2016 19:19	Extensión de la apl...	8 KB
Emgu.CV.DebuggerVisualizers.VS2013.dll	04/02/2016 19:19	Extensión de la apl...	8 KB
Emgu.CV.DebuggerVisualizers.VS2015.dll	04/02/2016 19:19	Extensión de la apl...	8 KB
Emgu.CV.License.txt	11/01/2016 20:14	Documento de tex...	35 KB
Emgu.CV.UI.dll	04/02/2016 19:19	Extensión de la apl...	115 KB
Emgu.CV.UI.GL.dll	04/02/2016 19:19	Extensión de la apl...	27 KB
Emgu.CV.UI.GL.xml	04/02/2016 19:19	Documento XML	3 KB
Emgu.CV.UI.xml	04/02/2016 19:19	Documento XML	34 KB
Emgu.CV.World.dll	04/02/2016 19:19	Extensión de la apl...	497 KB
Emgu.CV.World.xml	04/02/2016 19:19	Documento XML	1.346 KB

Figura 7. Referencias necesarias para compilar proyecto Visual Basic

Nota: No todos los DLLs son necesarios, pero eligiendo todos nos aseguramos de tener la base cubierta

e. Añadir archivos necesarios para compilar el proyecto II, para ello

Ir a "Project -> Add existing item"

Navegar al directorio bin\x86\ de Emgu, en nuestro caso "C:\Emgu\emgucv-windesktop 3.1.0.2282\bin\x86"

Cambiar opción de vista a "Todos los archivos"

Seleccionar todos los archivos DLLs y dar a añadir

Seleccionar los últimos DLLs en la ventana "Solution Explorer" para definir "Copiar Siempre" en la ventana "Properties"

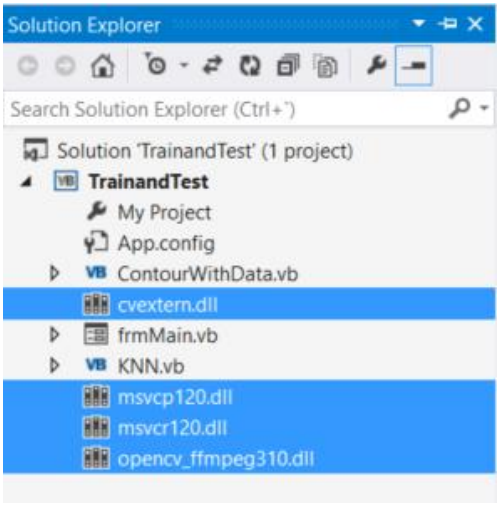


Figura 8. Archivos necesarios para compilar un proyecto en Visual Basic

f. Añadir controles especiales de Emgu a la barra de herramientas

Sobre la pestaña "Design view", click derecho sobre la opción "General" de la barra de herramientas y seleccionar "Choose items"

Elegir "Browse…" y navegar a "C:\Emgu\emgucv-windesktop 3.1.0.2282\bin" para seleccionar "Emgu.CV.UI.dll"

La opción que en este proyecto se ha utilizado, ImageBox, debería aparecer en la lista. Activarla.

5. Ya se puede trabajar con el proyecto creado.

3.1.3 PROCEDIMIENTO PARA OBTENER IMÁGENES A ANALIZAR

En este apartado se evalúa, por un lado, como se han obtenido las imágenes con las que se trabajara durante este proyecto y, por otro lado, el sistema recomendado para obtener imágenes en la aplicación final, en nuestro caso, el puesto de control.

Imágenes utilizadas

En un primer momento se ha decidido utilizar el propio documento de identidad, pero evaluándolo más detenidamente, se ha decidido utilizar una imagen con el

mismo formato, pero sin información personal. Realizando una búsqueda por la web, en particular por la conocida sección "Google imágenes" encontré la imagen que se muestra a continuación, la imagen buscada.

Figura 9. Imagen a utilizar durante el desarrollo del proyecto

Sistema de obtención de imagen

En este proyecto no se evalúa o desarrolla ningún sistema para la obtención de imágenes, pero sí se analizan los diferentes métodos de obtención de imágenes. Básicamente tenemos dos métodos para la obtención de la imagen, fotografiar el documento con una cámara o escanear el documento con un escáner. Debido a las grandes variaciones como: posición, luz, distancia, que pertenecen a la fotografía de una imagen, se recomienda el uso de un escáner donde podemos reducir significativamente las variaciones y futuros problemas implícitos en las cámaras, por ejemplo, obtener una imagen girada.

Actualmente, los sistemas implantados en aeropuertos utilizan escáneres por los motivos comentados anteriormente, un posible ejemplo es mostrado en la siguiente imagen.

Figura 10. Ejemplo de escáner facilitado, compañía Sidytar

3.1.4 ENTRENAMIENTO KNN

Para poder implantar el algoritmo KNN, primero hemos de crear un espacio de estado, es decir, entrenar al sistema mediante ejemplos. Mediante el entrenamiento realizado, obtenemos dos estructuras de datos.

- Conjunto de imágenes
- Conjunto de números indicando el "grupo" o "clasificación" a la que corresponde la imagen

Los caracteres de interés en nuestro proyecto son dígitos del 0 al 9 y letras mayúsculas de la A a la Z. Supongamos que vamos a reconocer dígitos del 0 al 9, podemos tener 5 imágenes diferentes de entrenamiento para cada digito, lo que hace un total de 50 imágenes de entrenamiento para los números. La siguiente imagen muestra las imágenes de entrenamiento.

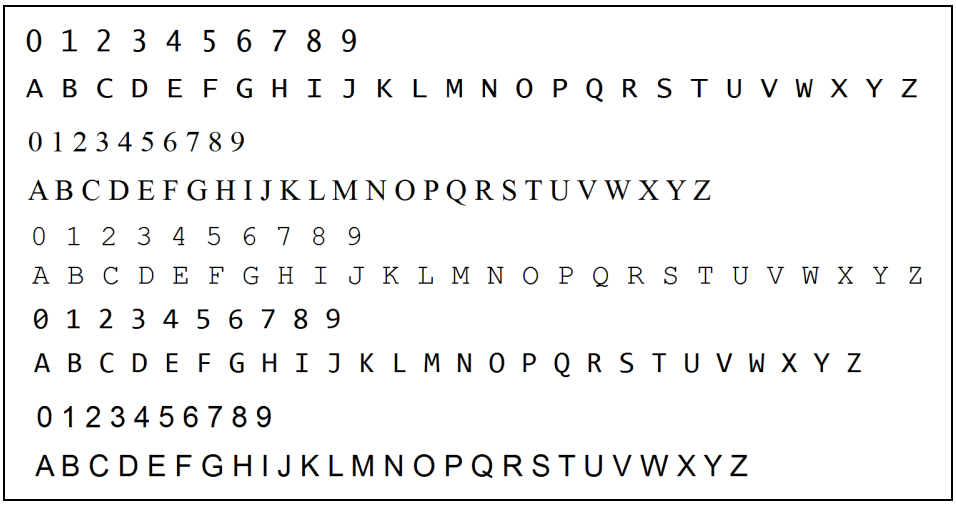


Figura 11. Imágenes de entrenamiento para algoritmo KNN

Sobre la imagen anterior se detectan los caracteres por el contorno y se pregunta al usuario que carácter en particular es, este transmite la información a través del teclado. De ese modo se almacena la imagen del carácter para su posterior comparación.

El código del programa que nos permite crear los dos archivos (classifications.xml y images.xml) es:

```
// Vector de caracteres de interes, digitos del 0 al 9 y letras mayusculas
de A a Z.  p
	std::vector<int> intValidChars = { '0', '1', '2', '3', '4', '5', '6',
'7', '8', '9',
		'A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J',
		'K', 'L', 'M', 'N', 'O', 'P', 'Q', 'R', 'S', 'T',
		'U', 'V', 'W', 'X', 'Y', 'Z' };

	imgTrainingNumbers = cv::imread("training_chars.png");            //
Leer la imagen de entrenamiento
```

```
for (int i = 0; i < ptContours.size(); i++) {                           //
para cada contorno
		if (cv::contourArea(ptContours[i]) > MIN_CONTOUR_AREA) {
// si el contorno es lo suficientemente grande para nuestro interes
			cv::Rect boundingRect =
cv::boundingRect(ptContours[i]);                // obtener rectangulo

			cv::rectangle(imgTrainingNumbers, boundingRect,
cv::Scalar(0, 0, 255), 2);      // dibujar rectanguls para contorno al
preguntar con su entrada (tecla)
```

```
                    cv::Mat matROI = imgThresh(boundingRect);           //
obtener ROI del rectangulo

                    cv::Mat matROIResized;
                    cv::resize(matROI, matROIResized,
cv::Size(RESIZED_IMAGE_WIDTH, RESIZED_IMAGE_HEIGHT));     // redimensionar
imagen, para ser mas consistente en reconocimiento y almacenamiento

                    cv::imshow("matROI", matROI);
// Mostrar ROI
                    cv::imshow("matROIResized", matROIResized);
// Redimensionar ROI
                    cv::imshow("imgTrainingNumbers", imgTrainingNumbers);
// Mostrar imagen con el rectangulo

                    int intChar = cv::waitKey(0);           // Pulsar tecla
correspondiente

                    if (intChar == 27) {        // Si se pulsa Esc,
                           return(0);              // salir del programa
                    }
                    else if (std::find(intValidChars.begin(),
intValidChars.end(), intChar) != intValidChars.end()) {     // sino buscar
el siguiente caracter

                           matClassificationInts.push_back(intChar);
// añadir a la lista de clasificación el caracter

                           cv::Mat matImageFloat;
// para añadir a la imagen de entrenamiento, se necesita conversión
                           matROIResized.convertTo(matImageFloat,
CV_32FC1);       // convertir variable de Mat a float

                           cv::Mat matImageFlattenedFloat =
matImageFloat.reshape(1, 1);       // aplanar


      matTrainingImagesAsFlattenedFloats.push_back(matImageFlattenedFloat);
// añadir a Mat como si fuera un vector

                    }
             }
      }

      std::cout << "training complete\n\n";

      // guardar el archivo de clasificación
//////////////////////////////////////////////////////

      cv::FileStorage fsClassifications("classifications.xml",
cv::FileStorage::WRITE);           // abrir archivo de clasificación

      if (fsClassifications.isOpened() == false) {
// comprobación
            std::cout << "error, unable to open training classifications
file, exiting program\n\n";        // mensaje de error
            return(0);
```

```
	}

	fsClassifications << "classifications" << matClassificationInts;
// escribir la clasificación en el archivo de clasificación
	fsClassifications.release();

	// guardas imagen de entrenamiento
////////////////////////////////////////////////////

	cv::FileStorage fsTrainingImages("images.xml",
cv::FileStorage::WRITE);          // abrir archivo de imagenes de
entrenamiento

	if (fsTrainingImages.isOpened() == false) {
// comprobación
		std::cout << "error, unable to open training images file,
exiting program\n\n";          // mensaje de error
		return(0);
	}

	fsTrainingImages << "images" << matTrainingImagesAsFlattenedFloats;
// escribir imagenes de entrenamiento en el archivo
	fsTrainingImages.release();
```

Nota: Para ver programa completo, ver sección de código, las imágenes se procesan para detectar mejor los caracteres.

3.2 *PRUEBAS AISLADAS DE CARGADO DE IMAGEN Y PRE-PROCESAMIENTO*

3.2.1 CARGADO DE IMÁGENES

En esta sección se explican las primeras instrucciones utilizadas en proyectos C++ para cargar imágenes. Es necesario tener la imagen en el mismo directorio del proyecto, a continuación, se muestra la instrucción para cargar la imagen.

```
#define INPUT_FILE              "dni.jpg"

// 1.- LEER FICHERO QUE CONTIENE LA IMAGEN
imgOriginal = imread(INPUT_FILE, CV_LOAD_IMAGE_COLOR);
```

Antes de ponernos a trabajar con la imagen cargada, realizamos una sencilla comprobación de su contenido.

```
//Comprobación de cargado de imagen
if (imgOriginal.empty()) {
	std::cout << "error: image not read from file\n\n";
	return(0);
}
```

Y ya podemos mostrarla en pantalla.

```
//2.- MOSTRAR IMAGEN
cv::namedWindow("imgOriginal", CV_WINDOW_NORMAL);
imshow("imgOriginal", imgOriginal);
```

3.2.2 PRE-PROCESAMIENTO

El objetivo del pre-procesado no es extraer información, sino realizar mejoras en las imágenes que permitan suprimir distorsiones no deseadas o realzar algunas propiedades importantes para futuros procesos.

En esta fase de pre-procesamiento (o adecuación de la imagen) el objetivo que se persigue es eliminar de la imagen cualquier tipo de ruido o imperfección que no pertenezca al carácter, así como normalizar el tamaño del mismo. Además, para el caso de OCR, la normalización de la imagen también puede implicar un binarizado de la misma.

Los algoritmos que se utilizan en las primeras versiones del proyecto están escritos en C++, se muestran a continuación.

Conversión a gris

```
cv::cvtColor(imgOriginal, imgGrayscale, CV_BGR2GRAY);		// convert to grayscale
```

Método del valor umbral

```
cv::adaptiveThreshold(imgBlurred, imgThresh, 255,
cv::ADAPTIVE_THRESH_GAUSSIAN_C, cv::THRESH_BINARY_INV, 11, 2);
```

Nota: En la sección “primitivas de pre-procesamiento” se realizan mejoras y se muestran las imágenes resultantes, ver más adelante.

3.3 PRUEBAS AISLADAS DE PROCESAMIENTO DE IMÁGENES

La segmentación consiste en dos sub-apartados, segmentación de texto y segmentación de caracteres.

3.3.1 SEGMENTACIÓN DE TEXTO

La detección de texto supone el primer paso del procesamiento, es decir, una vez la imagen ya está pre-procesada, se obtienen automáticamente las secciones de texto que serán enviadas a la detección de caracteres.

Para detectar el texto en la imagen utilizamos el método del gradiente morfológico, la estructura del proceso se muestra a continuación

1. Preparar imagen
    i. Imagen en gris
    ii. Gradiente morfológico
    iii. Valor umbral
    iv. Conectar horizontalmente las regiones
2. Detección de texto
    i. Detección de contorno
    ii. Comprobación

1. Preparar imagen

Partimos de la imagen ya convertida a escala de grises, para llamar a la función que realizara el gradiente morfológico.

```
cv::Mat morphKernel = getStructuringElement(MORPH_ELLIPSE, cv::Size(3, 3));
morphologyEx(rgb, grad, MORPH_GRADIENT, morphKernel);
```

Resultado

Figura 12. Resultado de aplicar el gradiente morfológico

Continuamos el proceso con el valor umbral, aunque pertenece a la sección de pre-procesamiento se muestra para entender mejor el proceso.

```
threshold(grad, bw, 0.0, 255.0, THRESH_BINARY | THRESH_OTSU);
```

Resultado

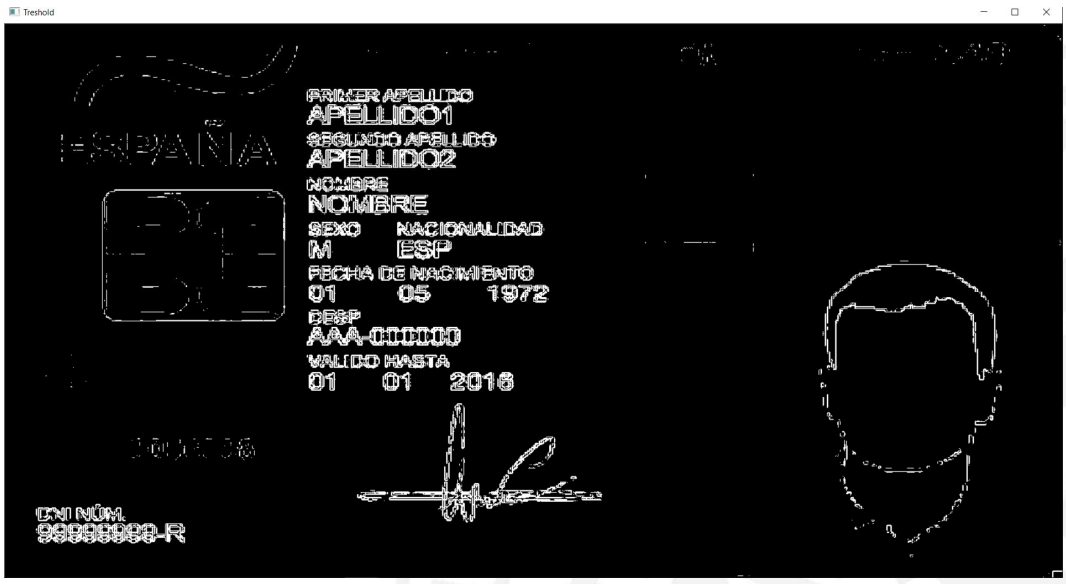


Figura 13. Resultado de aplicar el valor umbral

Conectamos horizontalmente las regiones de texto

```
morphKernel = getStructuringElement(MORPH_RECT, cv::Size(9, 1));
morphologyEx(bw, connected, MORPH_CLOSE, morphKernel);
```

Resultado

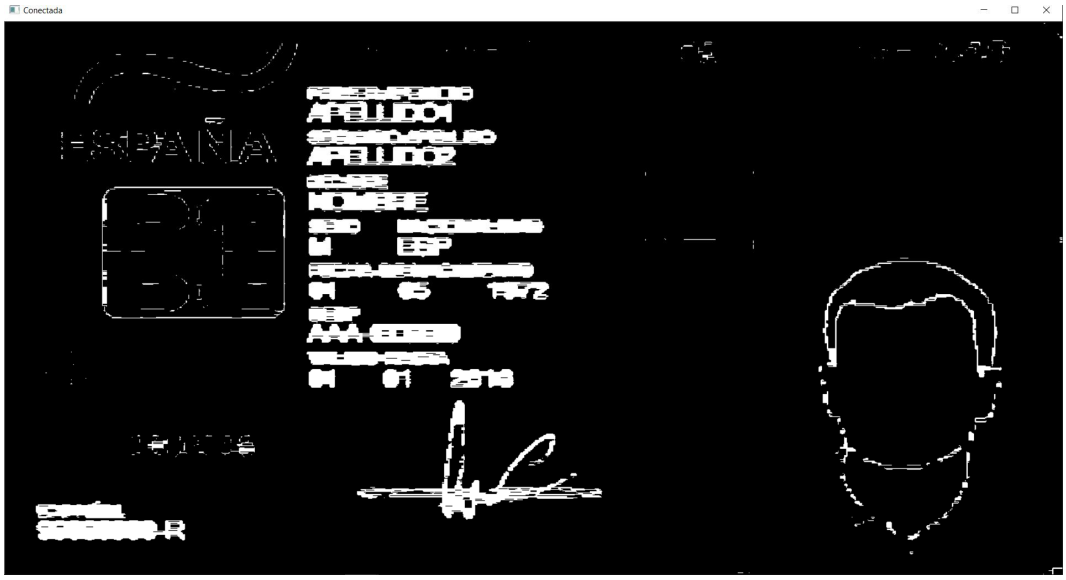


Figura 14. Resultado de conectar las regiones de texto

1. Detección de texto

Encontrar los contornos

```
cv::Mat mask = cv::Mat::zeros(bw.size(), CV_8UC1);
cv::vector<cv::vector<cv::Point>> contours;
cv::vector<cv::Vec4i> hierarchy;
findContours(connected,       contours,       hierarchy,       CV_RETR_CCOMP,
CV_CHAIN_APPROX_SIMPLE, cv::Point(0, 0));
```

Función que comprueba veracidad de las regiones obtenidas

```
for (int idx = 0; idx >= 0; idx = hierarchy[idx][0])
      {
            cv::Rect rect = boundingRect(contours[idx]);
            cv::Mat maskROI(mask, rect);
            maskROI = cv::Scalar(0, 0, 0);
            // Rellenar el contorno
            drawContours(mask, contours, idx, cv::Scalar(255, 255, 255),
CV_FILLED);
            // Ratio de pixeles blancos
            double r = (double)countNonZero(maskROI) /
(rect.width*rect.height);

            if (r > 0.6 /* Al menos el 60% de la región en blanco */
                  &&
                  (rect.height > 8 && rect.width > 8) /* Comprobación de
distancias minimas */

                  )
            {
```

```
                    rectangle(image, rect, cv::Scalar(0, 255, 0), 2);
                    //rectangle(morphKernel, rect, cv::Scalar(0, 255, 0),
2);

                    // Obtener el recorte (ROI)
                    maskROI = image(rect);

                    imshow("Recorte", maskROI);
                    waitKey();
                    destroyWindow("Recorte");
                }
        }
```

Para la imagen de trabajo, los resultados son excelentes. Al implantar esta solución en un programa comercial es necesario mejorar la comprobación de posible texto, actualmente solo se comprueba que el 60 % de la región sea blanca y las distancias mínimas.

Figura 15. Resultado final de la detección de texto

3.3.2 RECONOCIMIENTO DE CARACTERES

El reconocimiento de caracteres está compuesto por:

1. Implementación del algoritmo KNN con los archivos de entrenamiento
2. Detección de caracteres
    i. Detección de contorno

ii. Comprobación de posible carácter

3. Identificación del carácter, llamada de la función KNN

1. Implementación

Para implementar el algoritmo KNN y añadir los archivos de entrenamiento creados anteriormente, hemos de usar el siguiente código.

```
bool loadKNNDataAndTrainKNN(void) {

	// Leer los archivos de clasificación//////////////////////////

	cv::Mat matClassificationInts;

	cv::FileStorage fsClassifications("classifications.xml",
cv::FileStorage::READ);		// abrir el archivo de clasificacion
	// comprobacion
	if (fsClassifications.isOpened() == false) {
		std::cout << "error, unable to open training classifications
file, exiting program\n\n";
		return(false);
	}

	fsClassifications["classifications"] >> matClassificationInts;
// transferir la clasificación a la variable creada anteriormente
	fsClassifications.release();
// cerrar el archivo

	// Leer el archivo de imagenes//////////////////////////

	cv::Mat matTrainingImagesAsFlattenedFloats;

	cv::FileStorage fsTrainingImages("images.xml",
cv::FileStorage::READ);		// abrir

	if (fsTrainingImages.isOpened() == false) {
		std::cout << "error, unable to open training images file,
exiting program\n\n";			return(false);
	}

	fsTrainingImages["images"] >> matTrainingImagesAsFlattenedFloats;
// transmitir datos
	fsTrainingImages.release();
// cerrar

	// realizamos el entrenamiento con los archivos ya cargados
	kNearest->setDefaultK(1);

	kNearest->train(matTrainingImagesAsFlattenedFloats,
cv::ml::ROW_SAMPLE, matClassificationInts);

	return true;
}
```

Es importante mencionar que se ha externalizado la función por lo que hemos de definir la variable kNearest como externa

```
cv::Ptr<cv::ml::KNearest> kNearest = cv::ml::KNearest::create();
```

2. Detección de caracteres

En este punto ya no estamos trabajando con la imagen original, sino que estamos trabajando con imágenes recortadas con el texto de interés, por ejemplo, la siguiente imagen muestra la región de interés para apellido 1.

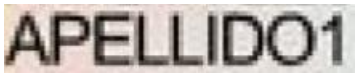


Figura 16. Ejemplo de recorte de interés

Llamamos a una función que encuentra los contornos de las letras una vez se ha pre-procesado, el código se muestra a continuación

```
cv::findContours(matThreshCopy,              // imagen recortada pre-pro
              ptContours,                              // contornos de salida
              v4iHierarchy,
              cv::RETR_EXTERNAL,                       // devolver contornos
externos
              cv::CHAIN_APPROX_SIMPLE);                // dejar solo los
puntos de salida

       for (int i = 0; i < ptContours.size(); i++)
       ContourWithData contourWithData;
              contourWithData.ptContour = ptContours[i];
              contourWithData.boundingRect =
cv::boundingRect(contourWithData.ptContour);       // obtener rectángulo
              contourWithData.fltArea =
cv::contourArea(contourWithData.ptContour);                // calcular área
              allContoursWithData.push_back(contourWithData);
```

Llegados a este punto tenemos una serie de posibles caracteres, están definidos por las esquinas del rectángulo y su área. Llamamos a una función que comprueba cuales de estos posibles caracteres son realmente caracteres

Llamada de la función

```
for (int i = 0; i < allContoursWithData.size(); i++) {
              if (allContoursWithData[i].checkIfContourIsValid()) {
// comprobar validez

       validContoursWithData.push_back(allContoursWithData[i]);
// en ese caso, añadir a la lista
              }
       }
```

Función

```
////////////////////////////////////////////////////////////////////////////
	std::vector<cv::Point> ptContour;		// contorno
	cv::Rect boundingRect;				// rectángulo
	float fltArea;					// área

////////////////////////////////////////////////////////////////////////////
	bool checkIfContourIsValid() {
		if (fltArea < MIN_CONTOUR_AREA) return false;		// 
Comprobación muy básica, estamos en pruebas
		return true;
	}
////////////////////////////////////////////////////////////////////////////
	static bool sortByBoundingRectXPosition(const ContourWithData& 
cwdLeft, const ContourWithData& cwdRight) {	// ordenar de izq a dcha
		return(cwdLeft.boundingRect.x < cwdRight.boundingRect.x);
	}
```

3. Identificación de caracteres

Puesto que ya tenemos definidos el recorte de los posibles caracteres, es momento de enviarlos al motor OCR desarrollado con el algoritmo KNN, para obtener el resultado. Para ello se realizan una serie de transformaciones, se llama a la función y se almacena, el código se muestra a continuación

```
for (int i = 0; i < validContoursWithData.size(); i++) {

		cv::rectangle(matTestingNumbers, 
validContoursWithData[i].boundingRect, cv::Scalar(0, 255, 0), 2);

		cv::Mat matROI = 
matThresh(validContoursWithData[i].boundingRect);		// Obtener el 
recorte de la region
		cv::Mat matROIResized;
		cv::resize(matROI, matROIResized, 
cv::Size(RESIZED_IMAGE_WIDTH, RESIZED_IMAGE_HEIGHT));	// redimensionar 
por consistencia
		cv::Mat matROIFloat;
		matROIResized.convertTo(matROIFloat, CV_32FC1);		// 
Convertir de Mat a float
		cv::Mat matROIFlattenedFloat = matROIFloat.reshape(1, 1);

		cv::Mat matCurrentChar(0, 0, CV_32F);

		kNearest->findNearest(matROIFlattenedFloat, 1, 
matCurrentChar);	// Llamada de la función

	    float fltCurrentChar = (float)matCurrentChar.at<float>(0, 0);

		strFinalString = strFinalString + char(int(fltCurrentChar));
	}
```

El resultado con texto sencillo se puede apreciar en la siguiente imagen:

Figura 17. Resultado reconocimiento de caracteres en proyectos C++

3.4 IMPLEMENTAR PRIMITIVAS DEL LENGUAJE RELACIONADAS CON CARGADO DE IMÁGENES, EL PRE-PROCESAMIENTO Y LAS SENTENCIAS OPERATIVAS DEL LENGUAJE DE USUARIO

3.4.1 CARGADO DE IMÁGENES

Anteriormente se han explicado las funciones para cargar imágenes en proyectos C++, en esta sección se realizará la explicación para proyectos C#. Los motivos del cambio se comentan más adelante, pero básicamente es integrar todas las funciones en una interfaz. Como consecuencia, para el cargado de imágenes en particular, se ha implementado un botón, facilitando la selección y búsqueda de la imagen deseada. La instrucción para cargar imágenes se muestra a continuación

```
    Private Sub btnOpenTestImage_Click(sender As Object, e As EventArgs)
Handles btnOpenTestImage.Click

        ' Limpiar interfaz
        limpiarinicio()

        ' Cargar imagen''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
        Dim blnImageOpenedSuccessfully =
openImageWithErrorHandling(imgOriginal)       ' comprobaciones para abrir
imagen
```

```
        If (Not blnImageOpenedSuccessfully) Then                    'si la
imagen no se abre correctamente
            ibOriginal.Image = Nothing
'interfaz del programa en blanco
            Return
        End If

        ibOriginal.Image = imgOriginal                        ' Actulizar el
hueco de imagen con la imagen
```

En el código anterior aparece una función que comprobara la consistencia de la imagen, igual que en la sección "pruebas aisladas de cargado de imágenes" se realiza una serie de comprobaciones, el código de las comprobaciones se muestra a continuación

```
Function openImageWithErrorHandling(ByRef imgOriginalScene As Mat) As
Boolean
        Dim drChosenFile As DialogResult

        drChosenFile = ofdOpenFile.ShowDialog()                 'abrir
archivo de diálogo

        If (drChosenFile <> DialogResult.OK Or ofdOpenFile.FileName = "")
Then          'si el usuario no elige ningun archivo
            lblChosenFile.Text = "file not chosen"
'actualizar etiqueta
            Return False
        End If

        Try
            imgOriginalScene = CvInvoke.Imread(ofdOpenFile.FileName,
LoadImageType.Color)          'Intento de abrir imagen
        Catch ex As Exception
'comprobación de errores
            lblChosenFile.Text = "unable to open image, error: " +
ex.Message                      'mostrar error
            Return False
        End Try

        If (imgOriginalScene Is Nothing) Then
'si la imagen no se puede abrir
            lblChosenFile.Text = "unable to open image, image was null"
'mostrar error
            Return False
        End If

        If (imgOriginalScene.IsEmpty()) Then
'si la imagen está vacia
            lblChosenFile.Text = "unable to open image, image was empty"
'mostrar error
            Return False
        End If
```

```
        Return True
    End Function
```

3.4.2 PRE-PROCESAMIENTO

Esta sección se puede considerar como una continuación o mejora de la sección "pruebas aisladas de pre-procesamiento", pues se realizan los mismos pasos añadiendo una serie de mejoras. Ahora sí, se pueden ver los resultados.

Conversión a gris

```
imgGrayscale = extractValue(imgOriginal)
```

```
Function extractValue(imgOriginal As Mat) As Mat
        Dim imgHSV As New Mat()
        Dim vectorOfHSVImages As New VectorOfMat()
        Dim imgValue As New Mat()

        CvInvoke.CvtColor(imgOriginal, imgHSV, ColorConversion.Bgr2Hsv)

        CvInvoke.Split(imgHSV, vectorOfHSVImages)

        imgValue = vectorOfHSVImages(2)

        Return imgValue
    End Function
```

Resultado

Figura 18. Resultado del pre-procesamiento - Gris

Método de difuminación

En el procesamiento de imágenes, un desenfoque gaussiano (también conocido como suavizado gaussiano) es el resultado de una imagen borrosa por una función gaussiana. Es un efecto utilizado típicamente para reducir el ruido de la imagen y reducir detalle.

```
Dim imgMaxContrastGrayscale As Mat = maximizeContrast(imgGrayscale)
'maximize contrast with top hat and black hat

Dim imgBlurred As New Mat()
CvInvoke.GaussianBlur(imgMaxContrastGrayscale, imgBlurred, New
Size(GAUSSIAN_BLUR_FILTER_SIZE, GAUSSIAN_BLUR_FILTER_SIZE), 0)
'gaussian blur
```

```
Function maximizeContrast(imgGrayscale As Mat) As Mat
        Dim imgTopHat As New Mat()
           Dim imgBlackHat As New Mat()
           Dim imgGrayscalePlusTopHat As New Mat()
           Dim imgGrayscalePlusTopHatMinusBlackHat As New Mat()

        Dim structuringElement As Mat =
CvInvoke.GetStructuringElement(ElementShape.Rectangle, New Size(3, 3), New
Point(-1, -1))
```

```
        CvInvoke.MorphologyEx(imgGrayscale, imgTopHat, MorphOp.Tophat,
structuringElement, New Point(-1, -1), 1, BorderType.Default, New
MCvScalar())
        CvInvoke.MorphologyEx(imgGrayscale, imgBlackHat, MorphOp.Blackhat,
structuringElement, New Point(-1, -1), 1, BorderType.Default, New
MCvScalar())

        CvInvoke.Add(imgGrayscale, imgTopHat, imgGrayscalePlusTopHat)
        CvInvoke.Subtract(imgGrayscalePlusTopHat, imgBlackHat,
imgGrayscalePlusTopHatMinusBlackHat)

        Return imgGrayscalePlusTopHatMinusBlackHat
    End Function
```

Resultado

Figura 19. Resultado del pre-procesamiento - Difuminación

Método del valor umbral

Los métodos del valor umbral son un grupo de algoritmos cuya finalidad es segmentar gráficos rasterizados, es decir separar los objetos de una imagen que nos interesen del resto. Con la ayuda de los métodos de valor umbral en las situaciones más sencillas se puede decidir qué píxeles conforman los objetos que buscamos y qué píxeles son sólo el entorno de estos objetos. Este método es especialmente útil para separar el texto de un documento del fondo de la imagen y

así poder llevar a cabo el reconocimiento óptico de texto (OCR) con más garantías de obtener el texto correcto.

```
'adaptive threshold to get imgThresh
CvInvoke.AdaptiveThreshold(imgBlurred, imgThresh, 255.0,
AdaptiveThresholdType.GaussianC, ThresholdType.BinaryInv,
ADAPTIVE_THRESH_BLOCK_SIZE, ADAPTIVE_THRESH_WEIGHT)
```

Resultado

Figura 20. Resultado del pre-procesamiento – Valor umbral

3.4.3 LENGUAJE DE USUARIO

Se ha intentado desarrolla la aplicación para un uso sencillo por parte del usuario, limitando complejos procedimientos de uso. El usuario dispone principalmente de dos opciones, la primera ya ha sido comentada anteriormente, se trata del botón para elegir la imagen deseada

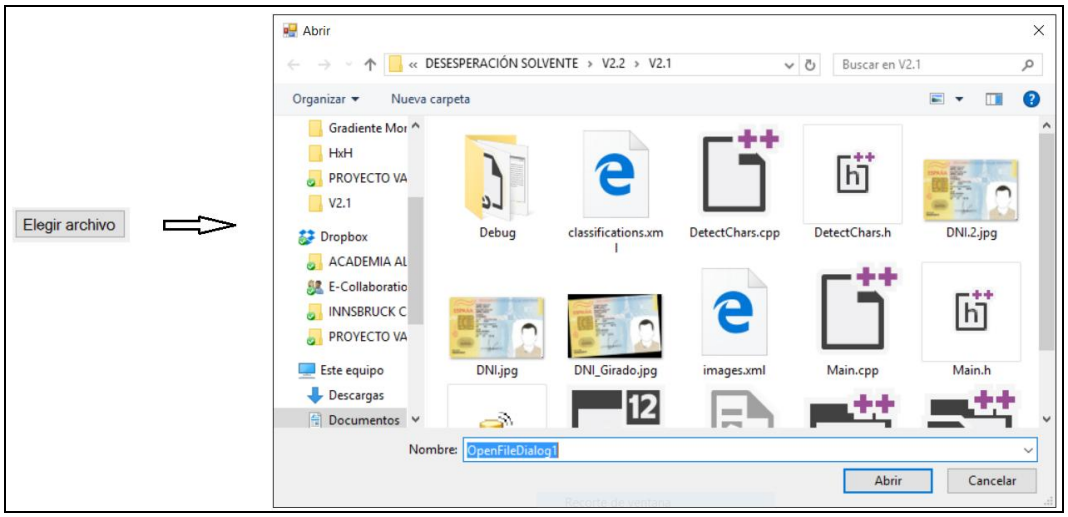


Figura 21. Botón que permite seleccionar archivo en el explorador

La segunda opción, mostrar pasos, nace de la oportunidad de ser conscientes de que está pasando en el proceso. Se trata de una opción que permite al usuario comprobar el error en un instante determinado. A continuación, se muestra un ejemplo de código

```
If (frmMain.cbShowSteps.Checked = True) Then ' mostrar pasos
''''''''''''''''''''''''''''''
            CvInvoke.Imshow("1a", image)
            CvInvoke.Imshow("1b", imgGrayscale)
            CvInvoke.Imshow("1c", imgThresh)
            frmMain.txtInfo.AppendText(vbCrLf + "Obtenemos con el pre-
procesado las siguientes imagenes" + vbCrLf + "Pulse una tecla para
continuar")
            CvInvoke.WaitKey(0)
            CvInvoke.DestroyWindow("1a")
            CvInvoke.DestroyWindow("1b")
            CvInvoke.DestroyWindow("1c")
        End If ' mostrar pasos
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
```

3.5 IMPLEMENTAR PRIMITIVAS DE PROCESAMIENTO Y SALIDA

3.5.1 PROCESAMIENTO

En la anterior sección "pruebas aisladas de procesamiento" se comentaban las dos secciones principales (texto y caracteres), en esta sección solo se implementa la detección de caracteres por falta de tiempo.

Se utiliza el mismo esquema,

1. Implementación del algoritmo KNN con los archivos de entrenamiento
2. Detección de caracteres
    i. Detección de contorno
    ii. Comprobación de posible carácter
3. Identificación del carácter, llamada de la función KNN

A pesar de que ahora estamos trabajando en C# y no en C++, por similitud, se omite el primer y último punto del esquema, comentando así, las mejoras realizadas en el segundo punto.

La comprobación de caracteres se ha mejorado, en vez de simplemente comprobar un área mínima, se comprueban unas dimensiones mínimas de altura y anchura, así como, la pertenencia al rango de ratio-aspecto. La función se muestra a continuación

```
' constantes para la comprobación de caracteres
'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
    Const MIN_PIXEL_WIDTH As Integer = 0
    Const MIN_PIXEL_HEIGHT As Integer = 8

    Const MIN_ASPECT_RATIO As Double = 0.1
    Const MAX_ASPECT_RATIO As Double = 1

    Const MIN_RECT_AREA As Integer = 80
```

```
'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
    Function checkIfPossibleChar(possibleChar As RecCaracteres) As Boolean
        'funcion para comprobar un caracter como válido
        If (intRectArea > MIN_RECT_AREA And _
            boundingRect.Width > MIN_PIXEL_WIDTH And boundingRect.Height > 
MIN_PIXEL_HEIGHT And _
            MIN_ASPECT_RATIO < dblAspectRatio And dblAspectRatio < 
MAX_ASPECT_RATIO) Then
            Return True
        Else
            Return False
        End If
    End Function
```

Para poder realizar dicha comprobación se han realizado una serie de cálculos en cada letra, como se muestra a continuación

```
' formulas para construir variables de la clase 
Contourwithdata''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
''''''''''''''''''''
    Sub New(_contour As VectorOfPoint)
        contour = _contour

        boundingRect = CvInvoke.BoundingRectangle(contour)

        intCenterX = CInt((boundingRect.Left + boundingRect.Right) / 2)
        intCenterY = CInt((boundingRect.Top + boundingRect.Bottom) / 2)

        dblDiagonalSize = Math.Sqrt((boundingRect.Width ^ 2) + 
(boundingRect.Height ^ 2))

        dblAspectRatio = CDbl(boundingRect.Width) / 
CDbl(boundingRect.Height)

        intRectArea = boundingRect.Width * boundingRect.Height
    End Sub
```

El calculo anterior de posición nos permite implementar funciones para calcular distancia y ángulo del documento. Es importante mencionar que utilizando el escáner mostrado en la imagen 10, no tendríamos problemas con imágenes giradas o dadas la vuelta. Las funciones se muestran a continuación

```
'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
    'calcular la distancia entre dos letras utilizando teorema de pitágoras
    Function distanceBetweenChars(firstChar As RecCaracteres, secondChar As 
RecCaracteres) As Double
        Dim intX As Integer = Math.Abs(firstChar.intCenterX - 
secondChar.intCenterX)
        Dim intY As Integer = Math.Abs(firstChar.intCenterY - 
secondChar.intCenterY)

        Return Math.Sqrt((intX ^ 2) + (intY ^ 2))
```

```
    End Function


''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
    'calcular el ángulo
    Function angleBetweenChars(firstChar As RecCaracteres, secondChar As
RecCaracteres) As Double
        Dim dblCon As Double = CDbl(Math.Abs(firstChar.intCenterX -
secondChar.intCenterX))
        Dim dblOpp As Double = CDbl(Math.Abs(firstChar.intCenterY -
secondChar.intCenterY))

        Dim dblAngleInRad As Double = Math.Atan(dblOpp / dblCon)

        Dim dblAngleInDeg As Double = dblAngleInRad * (180.0 / Math.PI)

        Return dblAngleInDeg
    End Function
```

3.5.2 PRIMITIVAS DE SALIDA - INTERFAZ

El motivo por el cuál, anteriormente, se han explicado dos procedimientos para crear diferentes proyectos (C++ y C#) es porque, la primera versión del programa se ha desarrollado en C++ por su facilidad de implementación. En esta primera versión, se trabaja sin ningún tipo de interfaz y los resultados son mostrados en la consola de comandos de Windows.

Durante el desarrollo del proyecto se ha trabajo con un proyecto en C++, donde se desarrolla el núcleo del proyecto, este es, cargar una imagen, pre-procesarla, aplicar reconocimiento de caracteres y mostrar resultado.

La siguiente imagen muestra el resultado obtenido

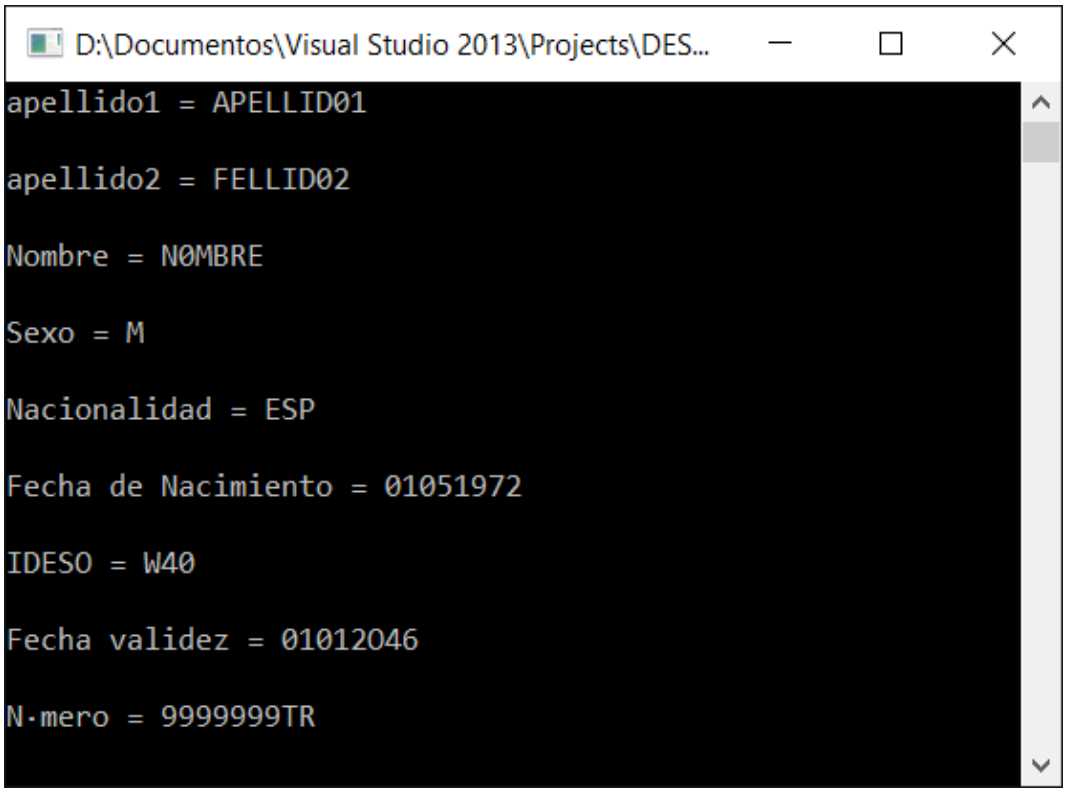


Figura 22. Resultado primer proyecto C++

Llegados a este punto, se decide dotar de una interfaz a la aplicación, para ello se utiliza la librería Emgu CV, que nos permite desarrollar la aplicación en C#. El código anterior se puede fácilmente adaptar manteniendo la base de funcionamiento del primer programa.

La siguiente imagen muestra el diseño que se implementara en la interfaz

Form Design

ID RECOGNITION - frmMain

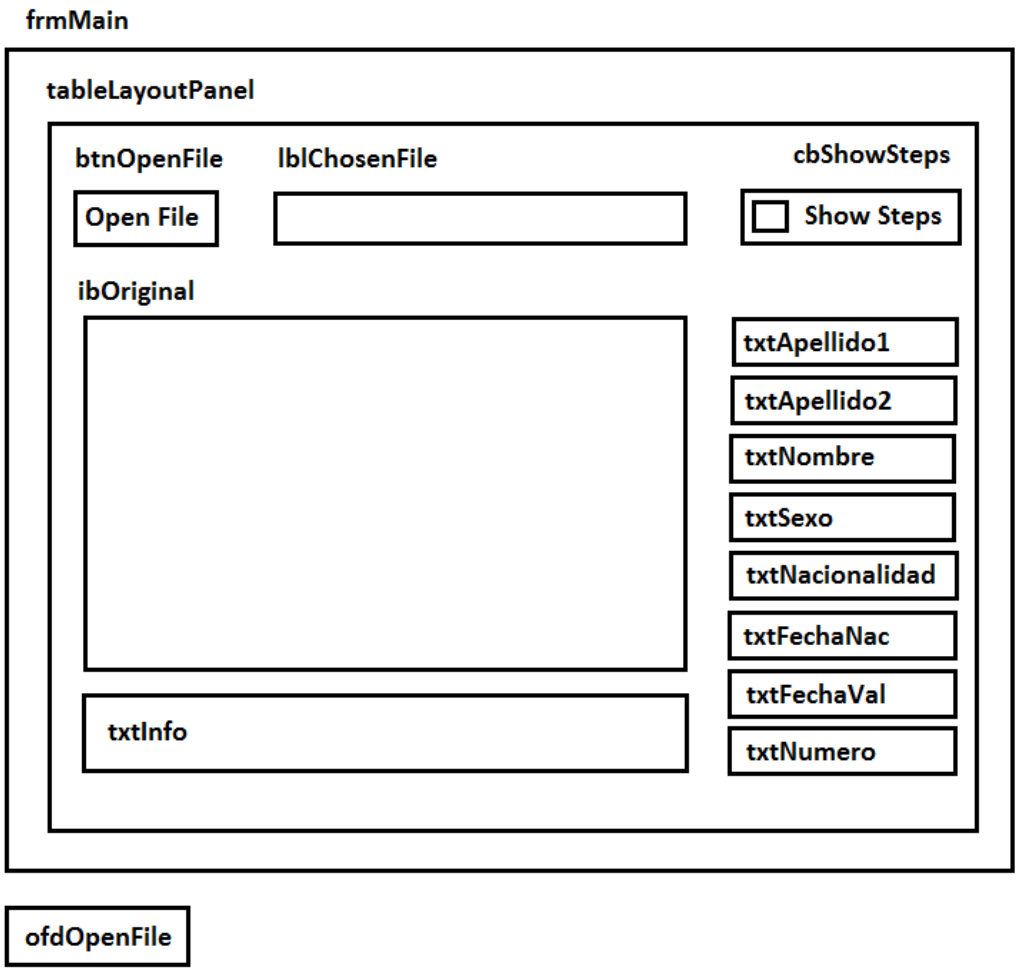


Figura 23. Diseño de la interfaz

Como se puede observar las primitivas de salida son los recuadros de texto donde se muestra la información obtenida. Mención especial tiene el cuadro de texto "txtInfo", en este cuadro de texto se mostrarán mensajes de información del proceso, por ejemplo, desde problemas con el cargado de la imagen hasta el procesado de la misma.

El resultado se muestra a continuación

Figura 24. Resultado del programa utilizando Emgu CV

3.6 INTEGRACIÓN Y PRUEBAS

3.6.1 INTEGRACIÓN – EXPORTAR APLICACIÓN

Anteriormente, se han explicado las diferentes funciones que compondrían el programa, es momento de unificar dichas funciones, pero antes de mostrar el programa completo, es necesario comprender el proceso. A continuación, se muestra un diagrama que muestra dicho proceso.

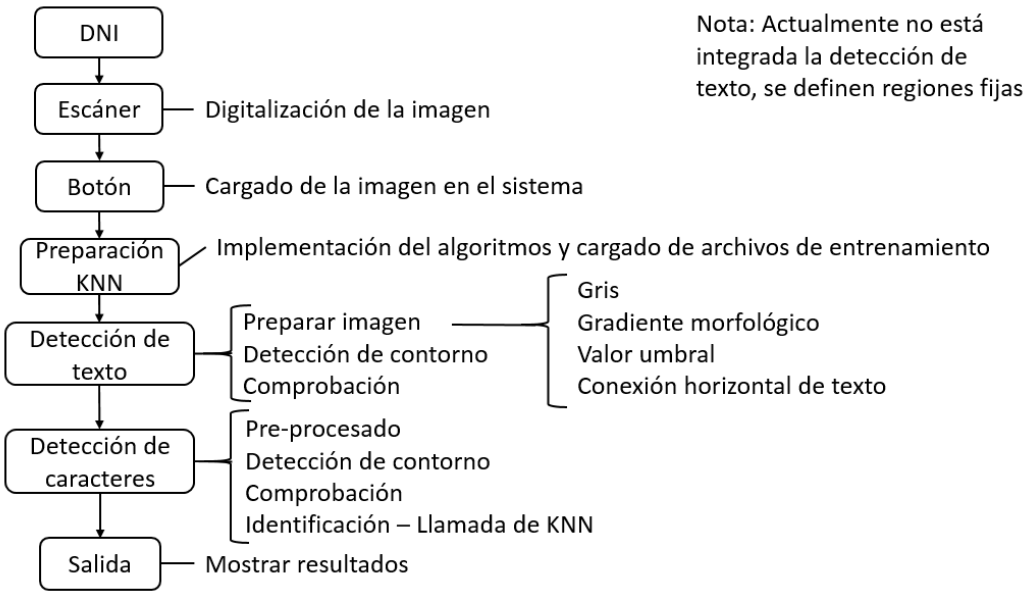


Figura 25. Diagrama del proceso

Se han integrado todas las funciones comentadas anteriormente, salvo la detección de texto, en una interfaz fácil y sencilla de usar. Hasta que se integre la detección de texto, un operario se encargara de definir las regiones de interés en el texto.

3.6.2 PRUEBAS Y RESULTADO CON DIFERENTES IMÁGENES Y CONFIGURACIONES

Prueba con diferentes configuraciones

Hasta este punto se ha desarrollado el grueso del proyecto, pero todavía los resultados no son los esperados. Para los resultados obtenidos con la imagen de prueba, hemos de valorar los siguientes aspectos

1. Problema al distinguir “0” y “O”

Apellido 1 = APELLID01	Nacionalidad = ESP
Apellido 2 = VELLID02	Fecha de nacimiento = 01051972
Nombre = N0MBRE	Fecha de validez = 01012O46
Sexo = M	Numero = 99999999R

Figura 26. Resultado del problema al identificar caracteres ("0" y "O")

Puesto que se trata de dos caracteres muy similares, existen problemas para indentificar cual es cual aun modificando los parametro internos de la aplicación, por lo que, se decide tomar una decisión más agresiva. Una vez el algoritmo nos devuelve la cadena, se envia a una función donde se cambian todos los 0 a O en el caso de texto o bien la inversa en el caso de fechas o numeros.

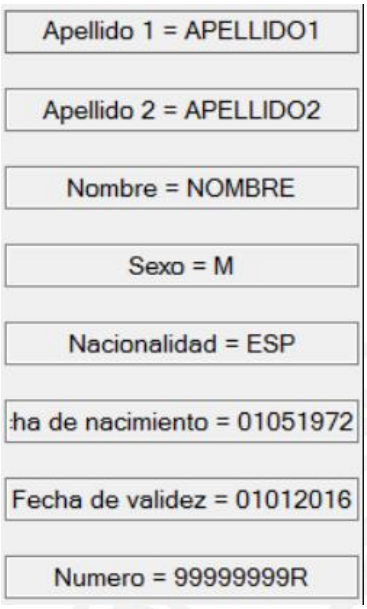


Figura 27. Solución al problema al identificar caracteres ("0" y "O")

2. Dos caracteres juntos se reconocen como uno solo, problema en el reconocimiento y en la clasificación

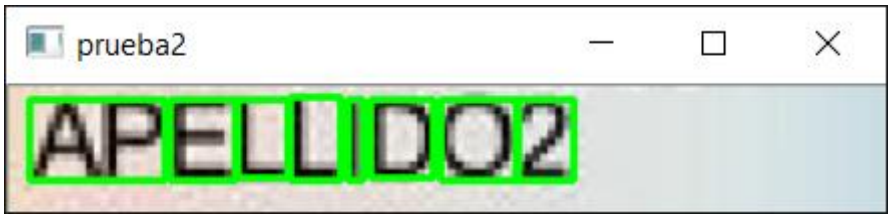


Figura 28. Problema al detectar caracteres muy juntos

Al incluir una mejora en la detección de caracteres, comprobando el ratio mínimo y máximo, dejamos de reconocer los dos caracteres juntos.

Figura 29. Problema al aplicar la comprobación de caracteres si están juntos

Se decide modificar los parámetros del valor umbral para diferenciar la poca distancia entre dos caracteres. El resultado es el deseado como se puede observar en la siguiente imagen.

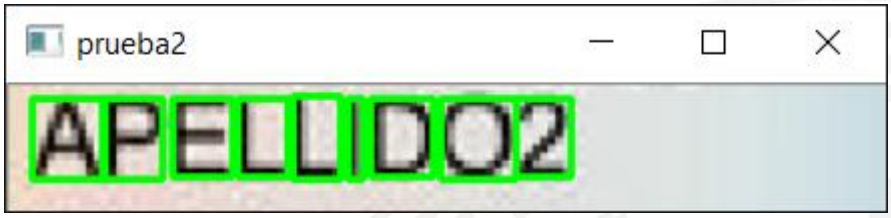


Figura 30. Solución al problema de los caracteres juntos

Pruebas con diferentes imágenes

Esta sección corresponde a las pruebas realizadas con diferentes imágenes, vamos a utilizar imágenes de DNIs reales para comprobar el funcionamiento. Es importante mencionar que antes de cargar la imagen en el programa, se ha normalizado, es decir, tiene el mismo tamaño y la misma densidad de pixeles por pulgada que la imagen de prueba. Tener en cuenta que, puesto que no está integrada la sección de detección de texto, las regiones de interés varian, motivo

por el cual, se tienen que redefinir. Para facilitar la prueba del programa, se han incluido dos opciones para seleccionar que tipo de DNI que se comprueba.

Las imágenes a continuación muestran los resultados.

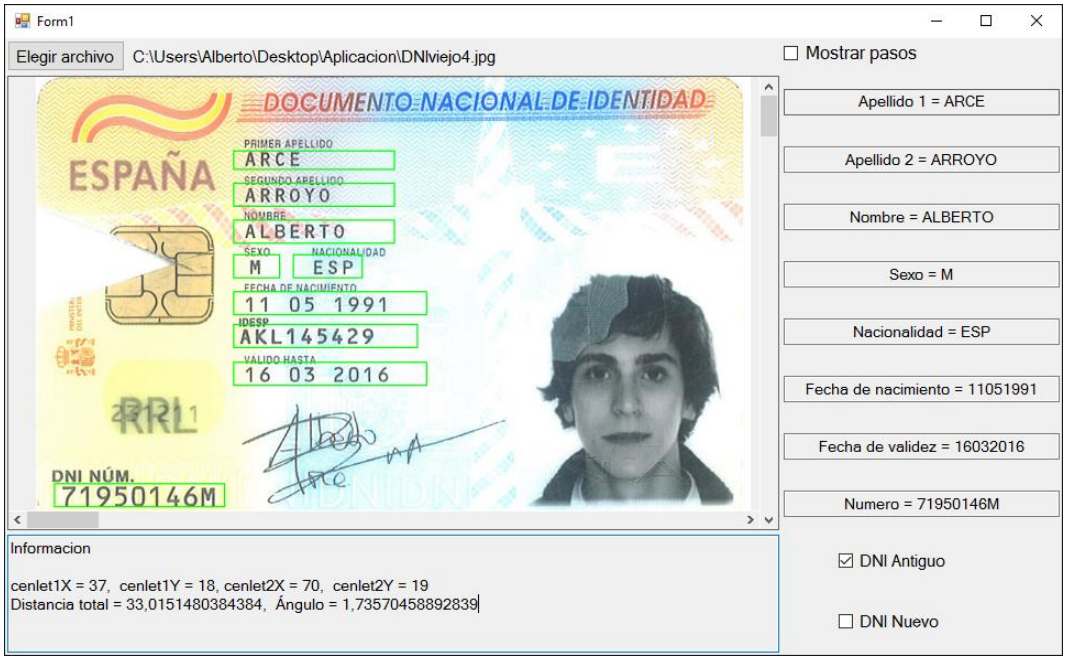


Figura 31. Prueba con DNI real antiguo

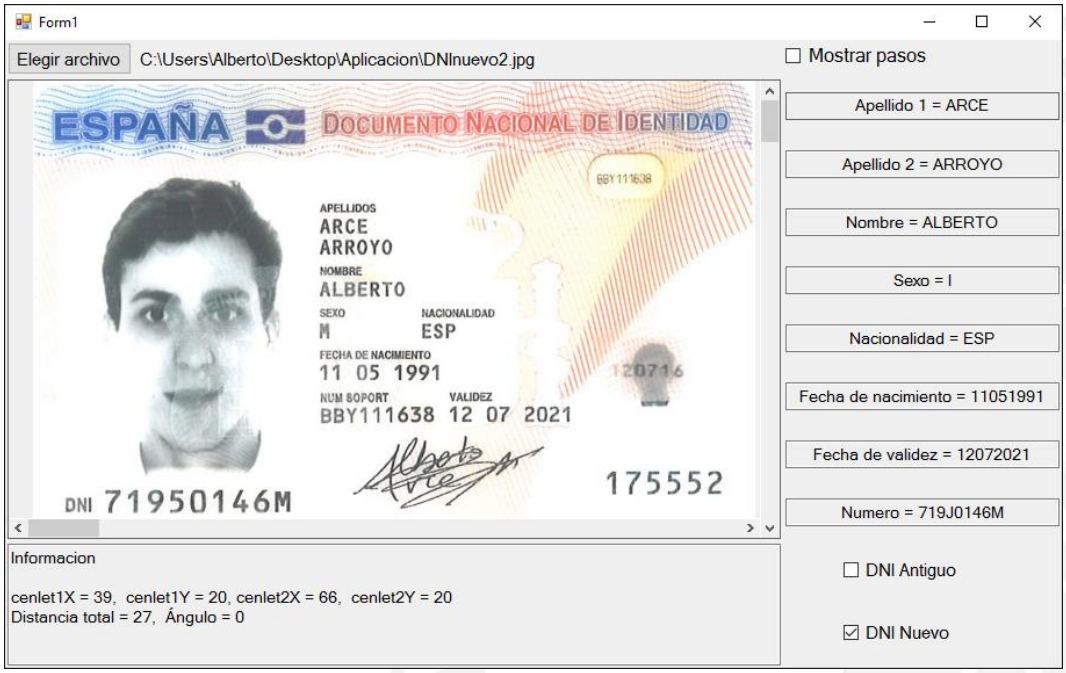


Figura 32. Prueba con DNI real nuevo

Los resultados son satisfactorios, pues para el caso del DNI antiguo, los resultados son excepcionales, mientras que, para el caso del DNI nuevo, podemos encontrar dos detecciones erróneas. El sexo y el cuarto digito del número no se identifican bien.

3.7 OPTIMIZACIONES

3.7.1 SEGMENTACIÓN DE TEXTO

En este proyecto se han desarrollado los primeros pasos para la detección de texto, creando un programa en C++ que automáticamente, obtiene las regiones que contienen el texto de interés. En futuros desarrollos es necesario integrar dicha detección de texto en el programa, en particular, en la detección de caracteres, es decir, los argumentos que obtenemos con la detección de texto (salida) deben ser enviados a la detección de caracteres (entrada). Actualmente un operario define regiones fijas pero el texto no se detecta automáticamente.

Capítulo 4 RECURSOS A EMPLEAR

4.1 ORDENADOR – PC – MICROSOFT VISUAL STUDIO

Microsoft Visual Studio es un conjunto de herramientas de desarrollo diseñadas para facilitar a los diseñadores de software el enfrentamiento a problemas complejos con la máxima comodidad posible. Esta herramienta se trata de un software IDE (Interface Development Enviroment) y nos proporciona un editor de código, un compilador con el que compilar nuestro código y una herramienta de depuración de código entre otras características.

MSVS soporta lenguajes como Visual Basic, Visual C++, Visual C# y Visual J# y dichos lenguajes aprovechan las funciones de .NET Framework que ofrece acceso a tecnologías clave para simplificar el desarrollo de aplicaciones Web ASP y Servicios Web XML. Además, también soporta HTML, XHTML, Javascript y CSS, lenguajes básicos a la hora de realizar páginas web.

Para el proyecto, se utilizarán los lenguajes C++ y C# a pesar de que MSVS ofrece compatibilidad con múltiples lenguajes.

4.2 OPEN CV

OpenCV es una biblioteca libre de visión artificial originalmente desarrollada por Intel. Desde que apareció su primera versión alfa en el mes de enero de 1999, se ha utilizado en infinidad de aplicaciones. Desde sistemas de seguridad con detección de movimiento, hasta aplicaciones de control de procesos donde se requiere reconocimiento de objetos.

Open CV es multiplataforma, existiendo versiones para GNU/Linux, Mac OS X y Windows. Contiene más de 500 funciones que abarcan una gran gama de áreas en el proceso de visión, como reconocimiento de objetos (reconocimiento facial), calibración de cámaras, visión estérea y visión robótica.

4.3 TESSERACT

Tesseract es un motor OCR libre. Fue desarrollado originalmente por Hewlett Packard como software propietario entre 1985 y 1995. Tras diez años sin ningún desarrollo, fue liberado como código abierto en el año 2005 por Hewlett Packard y la Universidad de Nevada, Las Vegas. Tesseract es desarrollado actualmente por Google y distribuido bajo la licencia Apache, versión 2.0. Está disponible para Linux, Widows y Mac OS X y puede detectar textos en inglés, francés, italiano, alemán, español, neerlandés, portugués y brasileño.

4.4 EMGU CV

Emgu es una versión para .NET de la librería OpenCV de procesamiento de imágenes, escrito completamente en C# y puede ser compilado en Mono por lo que puede ser utilizado en Linux, Windows, Mac, IOS y Android. Emgu puede ser utilizada con diferentes lenguajes, incluyendo C#. VB.NET, C++ and IronPython.

Capítulo 5 CONCLUSIONES

Llegados al final, es importante hacer una recapitulación del proyecto desarrollado, viendo los objetivos alcanzados, lo que se ha conseguido por el camino y algunas líneas de investigación que podrían aplicarse en el futuro sobre la base del trabajo realizado.

Como ya se ha discutido a lo largo de la memoria, se puede determinar que los objetivos principales planteados en el inicio se han alcanzado satisfactoriamente gracias a un adecuado análisis y estudio previo al desarrollo.

El diseño e implementación de las funciones de visión artificial ha cumplido con las expectativas, obteniendo unos resultados satisfactorios en las pruebas realizadas. Además de la consecución de estos objetivos, y como parte de lo que debe ser un trabajo de investigación en general y un trabajo de fin de grado en particular, se han adquirido nuevos conocimientos y puesto en práctica otros tantos aprendidos a lo largo de los años previos, en la carrera.

Finalmente, podemos decir, que se ha conseguido una aplicación eficaz y eficiente, a la par que fiable y estable, con una curva de aprendizaje mínima y con la funcionalidad más utilizada a la vista, evitando complejos procedimientos para su uso.

Capítulo 6 FUTUROS DESARROLLOS

6.1 CONEXIÓN A BASE DE DATOS

Es importante mencionar que nuestro sistema trabaja con documentos de identidad, para comprobar la identidad de una persona se realizan comprobaciones cruzando información con los registros de una base de datos. Se requiere por lo tanto una conexión con una base de datos gubernamental, puesto que, el gobierno de cada estado es el responsable del censo de la población. Mediante esta conexión se facilita el control de acceso de alta seguridad en aeropuertos, registro de clientes de un hotel o registro de ciudadanos en oficinas.

Las bases de datos gubernamentales se clasifican en bases de datos relacionales, lo que se asemeja a las bases de datos tradicionales y se caracterizan por ser muy consistentes, organizando la información en tablas y columnas.

Capítulo 7 BIBLIOGRAFÍA

[JONA08] Jonatan (2008), "Proyectos de sistemas informáticos: Historia OCR", obtenido por http://psig5.blogspot.com/2007/08/histori a-ocr.html

[SA_09] Sánchez C.J. y Sandonís V., "Reconocimiento optico de caracteres OCR", Proyecto fin de carrera, Universidad Carlos III, 2009. Obtenido: http://www.it.uc3m.es/jvillena/irc/practicas/08-09/09.pdf

[ORDO09] Ordoñez J.P.(2009), "Reconocimiento óptico de caracteres (OCR) con redes neuronales" Obtenido http://jpordonez.files.wordpress.com/2009/06/estado-del-arte.pdf

[WIKI16] Wikipedia, "Optical carácter recognition, editado en Abril de 2016", obtenido por http://en.wikipedia.org/wiki/Optical_character_recognition

[NAVA13] Navarro Santapau J. (2013), "Software de adquisición de imágenes y reconocimineto óptico de caracteres para android", Trabajo fin de master, Universidad Oberta de Cataluña.

[SABISF] Sabia (s.f.) "Vision artificial e interacción sin mandos". Obtenido del dept. tecnologías de la información y las comunicaciones, Universidad da Coruña http://sabia.tic.udc.es/gc/Contenidos%20adicionales/trabajos/3D/VisionArtificial/index.html

[AINISF] Ainia, "Visión artificial: conozca sus ventajas y aplicaciones". Obtenido:

[RODR09] Rodriguez J.J., "Sistema de navegación locas en entornos urbanos para un vehículo autónomo", Proyecto fin de carrera, Universidad Carlos III Madrid, 2009.

[AREA12] Area E., (2012), "5 herramientas de software gratutito de OCR para convertir imágenes a texto", blog: eduardoarea.blogspot.com.es/2012/08/top-5-herramientas-de-software-gratuito.html

[SIMPSF] "Comparativa entre Simple CV, Open CV y Matlab", obtenido simplecv.tumblr.com/post/19307835766/opencv-vs-matlab-vs-simplecv

[DAHM13] Dahms C.(2013). Repositorio de aplicaciones, obtenido: https://github.com/MicrocontrollersAndMore

[ALERSF] Aler R. (sf), "Clasificadores KNN-I". Obtenido: http://ocw.uc3m.es/ingenieria-informatica/analisis-de-datos/transparencias/KNNyPrototipos.pdf

[SIER07] Sierra A. (2007), "Algoritmo KNN, vecinos próximos", Presentación obtenida: http://arantxa.ii.uam.es/~asierra/tacciii/2006_2007/vecinos.pdf

[JUAR14] Juárez A.,"Desarrollo de un sistema de visión 3D para su integración en un robot", Proyecto fin de carrera, Universidad de Valladolid, 2014.

[MIKE15] Mike, (2015),"Detecting text regions from image using EmguCV"

[DHAN14] Dhanushka (2014) "Extracting text OpenCV". Foro http://stackoverflow.com/questions/23506105/extracting-text-opencv?lq=1

[DHAN15] Dhanushka (2015) "Remove background noise from image to make text more clear for OCR". Foro http://stackoverflow.com/questions/33881175/remove-background-noise-from-image-to-make-text-more-clear-for-ocr/33961545#33961545

Parte II ESTUDIO ECONÓMICO

Capítulo 1 ESTUDIO ECONÓMICO

En este capítulo se realizará el estudio económico que supone la realización del proyecto "Visión artificila de documentos".

Se supone que este proyecto ha sido realizado por un Ingeniero Industrial especializado en Electrónica Industrial, en el que se parte desde cero, donde se tendrá que realizar una inversión inicial para adquirir el software para desarrollar la programación específica.

A continuación, se realizará un desglose de todos los costes asociados al proyecto en función de su naturaleza

1.1 COSTES DIRECTOS

En este apartado se detallarán los costes derivados del desarrollo del material para el curso, incluyendo los costes de personal, los costes amortizables de programas y equipos, y el coste de los materiales directos empleados.

1.1.1 COSTES DE PERSONAL

Para el cálculo de estos costes se considera el trabajo realizado por un Ingeniero Industrial especializado en Electrónica Industrial que será el encargado de realizar elproyecto.

En primer lugar, es necesario considerar el régimen horario de trabajo seguido durante el año laboral. En la siguiente tabla se muestra el total de horas efectivas trabajadas durante el proyecto.

Horas efectivas por año		
Dias por año	270	días
Sabados y Domingos	-72	días
Dias festivos	-13	días
Total días efectivos	185	días/año
Horas trabajadas	4	horas/día
Total horas	740	horas/año

Tabla 9. Horas trabajadas en el proyecto

Los sueldos se calculan teniendo en cuenta los elementos que forman parte del gasto a desembolsar por parte de la empresa: sueldo bruto, incentivos, pagos a la Seguridad Social, vacaciones, días festivos, etc. Se considera un sueldo de 1000€/mes

Concepto	Importe
Sueldo brutto+incentivos	9000 €
Seguridad social (35%)	3150 €
Total coste persona/año	12150 €

Tabla 10. Sueldo anual

Con este valor y el obtenido con las horas efectivas anuales calculadas, se puede calcular el coste horario efectivo.

Coste por hora	
Total coste persona/año	12150
Número de horas de trabajo	740
Total coste/hora	16,4189

Tabla 11. Sueldo efectivo

1.1.2 COSTES AMORTIZABLES DE PROGRAMAS Y EQUIPOS

En este apartado se deberá realizarse el cálculo de la amortización lineal una vez conocida la inversión inicial, según los criterios aconsejados por la ley. En este apartado se presentan los costes de la amortización del material de las aplicaciones informáticas empleadas.

Se considera que la vida media de las aplicaciones informáticas es de 3 años, por lo que el factor de amortización será del 0,33.

Concepto	Importe	Amortización 33,3%
Sistema operativo WINDOWS 10	155,80 €	51,93 €
Paquete informático MICROSOFT OFFICE	678,00 €	226,00 €
PC	900,00 €	300,00 €
Escáner	39,90 €	13,30 €
Total material	1.773,70 €	591,23 €

Tabla 12. Costes de programas y equipos

1.1.3 TOTAL DE LOS COSTES DIRECTOS

El coste total directo será la suma de las partidas anteriormente descritas, costes de personal, amortización de programas y equipos

Concepto	Importe
Coste personal	12.150,00 €
Coste amortización	591,23 €
Total costes directos	12.741,23 €

Tabla 13. Coste directo total

1.2 COSTES INDIRECTOS

Dentro de los costes indirectos se evalúan una serie de factores, que, aunque no revierten de forma directa en el producto, sí que generan unos costes adicionales que hay que tener presentes en el presupuesto total del proyecto. El origen de estos costes es muy variado, desde consumos energéticos de calefacción, refrigeración, iluminación hasta gastos de teléfono o internet.

Concepto	Importe
Consumo electrico	120,00 €
Consumo telefonía+ internet	120,00 €
Total	240,00 €

Tabla 14. Costes indirectos

1.3 COSTES TOTALES

Concepto	Importe
Total costes directos	12.741,23 €
Total costes indirectos	240,00 €
Total costes	12.981,23 €

Tabla 15. Costes totales

Por lo tanto, el coste final del proyecto será de **12.981,23€**

Parte III MANUAL DE USUARIO

1.1 MANUAL DE USUARIO

1.1.1 DISEÑO

El diseño de la aplicación es muy sencillo, con pocas funciones habilitadas para el usuario final. En la siguiente imagen se muestran las opciones que brinda dicha interfaz

Figura 33. Opciones que brinda la interfaz

Parte IV CÓDIGO FUENTE

Capítulo 1 CÓDIGO FUENTE

La estructura de esta sección se muestra en la siguiente imagen

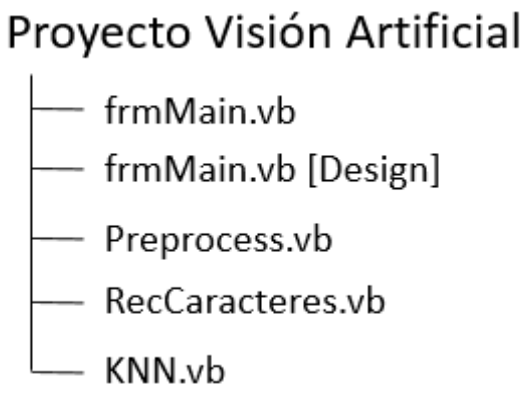


Figura 34. Estructura de los archivos en el proyecto

frmMain

```
'frmMain.vb

'comonentes de diseño de frmMain:
'frmMain
'tableLayoutPanel
'btnOpenTestImage
'lblChosenFile
'txtInfo
'ofdOpenFile
'ibOriginal

Option Explicit On      'requiere declaraciones de variables explicitas
Option Strict On        'restringirr conversion de datos implicitos a
conversiones de ampliacion

Imports Emgu.CV                 'Importaciones típicas de Emgu
Imports Emgu.CV.CvEnum          '
Imports Emgu.CV.Structure       '
Imports Emgu.CV.UI              '
Imports Emgu.CV.Util            '
Imports Emgu.CV.ML              '
```

```
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
''''''''''''''''''''''''
Public Class frmMain


''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
''''''''''''''''''''
    Private Sub frmMain_Load(sender As Object, e As EventArgs) Handles
MyBase.Load                'Segun carga el programa

        Dim blnKNNTrainingSuccessful As Boolean = loadKNNDataAndTrainKNN()
'Intento de entrenamiento KNN

        If (blnKNNTrainingSuccessful = False) Then
'Comprobación
            txtApellido1.AppendText(vbCrLf + "error: KNN traning was not
successful" + vbCrLf)       'Mensaje de error
            MsgBox("error: KNN traning was not successful")
            btnOpenTestImage.Enabled = False
'desactivar boton abrir imagen
            Return
        End If
    End Sub

''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
''''''''''''''''''''
    Private Sub btnOpenTestImage_Click(sender As Object, e As EventArgs)
Handles btnOpenTestImage.Click

        ' Definir variables
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
        Dim datApellido1 As String = ""
        Dim datApellido2 As String = ""
        Dim datSexo As String = ""
        Dim datNacionalidad As String = ""
        Dim datFechaNac As String = ""
        Dim datFechaVal As String = ""
        Dim datIDESP As String = ""
        Dim datNumero As String = ""
        Dim datNombre As String = ""

        Dim cdatApellido1 As String = ""
        Dim cdatApellido2 As String = ""
        Dim cdatSexo As String = ""
        Dim cdatNacionalidad As String = ""
        Dim cdatFechaNac As String = ""
        Dim cdatFechaVal As String = ""
        Dim cdatIDESP As String = ""
        Dim cdatNumero As String = ""
        Dim cdatNombre As String = ""

        Dim imgOriginal As New Mat()

        ' Limpiar interfaz
        limpiarinicio()
```

```
        ' Cargar
imagen''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
        Dim blnImageOpenedSuccessfully =
openImageWithErrorHandling(imgOriginal)       ' comprobaciones para abrir
imagen

        If (Not blnImageOpenedSuccessfully) Then                    'si la
imagen no se abre correctamente
            ibOriginal.Image = Nothing
'interfaz del programa en blanco
            Return
        End If

        ibOriginal.Image = imgOriginal                            ' Actulizar el
hueco de imagen con la imagen

        ' Segmentacion
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''


        'Dim RectApellido1 As New Rectangle(430, 140, 350, 50)
        Dim imgROIApellido1Cloned As New Mat(imgOriginal, New
Rectangle(430, 140, 350, 50))                ' (Posx,PosY,Ancho,Alto)
        CvInvoke.Rectangle(imgOriginal, New Rectangle(430, 140, 350, 50),
New MCvScalar(0.0, 255.0, 0.0), 2)
        Dim imgApellido1 As Mat = imgROIApellido1Cloned.Clone()
        'CvInvoke.Imshow("imgApellido1", imgApellido1)

        Dim imgROIApellido2Cloned As New Mat(imgOriginal, New
Rectangle(430, 219, 350, 50))
        CvInvoke.Rectangle(imgOriginal, New Rectangle(430, 219, 350, 50),
New MCvScalar(0.0, 255.0, 0.0), 2)
        Dim imgApellido2 As Mat = imgROIApellido2Cloned.Clone()
        'CvInvoke.Imshow("imgApellido2", imgApellido2)

        Dim imgROINombreCloned As New Mat(imgOriginal, New Rectangle(430,
299, 350, 50))
        CvInvoke.Rectangle(imgOriginal, New Rectangle(430, 299, 350, 50),
New MCvScalar(0.0, 255.0, 0.0), 2)
        Dim imgNombre As Mat = imgROINombreCloned.Clone()
        'CvInvoke.Imshow("imgNombre", imgNombre)

        Dim imgROISexoCloned As New Mat(imgOriginal, New Rectangle(430,
378, 100, 50))
        CvInvoke.Rectangle(imgOriginal, New Rectangle(430, 378, 100, 50),
New MCvScalar(0.0, 255.0, 0.0), 2)
        Dim imgSexo As Mat = imgROISexoCloned.Clone()
        'CvInvoke.Imshow("imgSexo", imgSexo)

        Dim imgROINacionalidadCloned As New Mat(imgOriginal, New
Rectangle(530, 378, 150, 50))
        CvInvoke.Rectangle(imgOriginal, New Rectangle(530, 378, 150, 50),
New MCvScalar(0.0, 255.0, 0.0), 2)
        Dim imgNacionalidad As Mat = imgROINacionalidadCloned.Clone()
        'CvInvoke.Imshow("imgNacionalidad", imgNacionalidad)

        Dim imgROIFechaNacCloned As New Mat(imgOriginal, New Rectangle(430,
457, 420, 50))
```

```
        CvInvoke.Rectangle(imgOriginal, New Rectangle(430, 457, 420, 50),
New MCvScalar(0.0, 255.0, 0.0), 2)
        Dim imgFechaNac As Mat = imgROIFechaNacCloned.Clone()
        'CvInvoke.Imshow("imgFechaNac", imgFechaNac)

        Dim imgROIIDESPCloned As New Mat(imgOriginal, New Rectangle(430,
536, 400, 50))
        CvInvoke.Rectangle(imgOriginal, New Rectangle(430, 536, 400, 50),
New MCvScalar(0.0, 255.0, 0.0), 2)
        Dim imgIDESP As Mat = imgROIIDESPCloned.Clone()
        'CvInvoke.Imshow("imgIDESP", imgIDESP)

        Dim imgROIFechaValCloned As New Mat(imgOriginal, New Rectangle(430,
616, 420, 50))
        CvInvoke.Rectangle(imgOriginal, New Rectangle(430, 616, 420, 50),
New MCvScalar(0.0, 255.0, 0.0), 2)
        Dim imgFechaVal As Mat = imgROIFechaValCloned.Clone()
        'CvInvoke.Imshow("imgFechaVal", imgFechaVal)

        Dim imgROINumeroCloned As New Mat(imgOriginal, New Rectangle(40,
881, 300, 50))
        CvInvoke.Rectangle(imgOriginal, New Rectangle(40, 881, 300, 50),
New MCvScalar(0.0, 255.0, 0.0), 2)
        Dim imgNumero As Mat = imgROINumeroCloned.Clone()
        'CvInvoke.Imshow("imgNumero", imgNumero)

        ' Preprocesado
'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

        Dim imgGrayscale As New Mat()
        Dim imgBlur As New Mat()
        Dim imgThresh As New Mat()

        Preprocess.preprocess(imgOriginal, imgGrayscale, imgBlur,
imgThresh)      'llamada de la función

        'CvInvoke.Imshow("Original", imgOriginal)
        'CvInvoke.Imshow("Gray", imgGrayscale)
        'CvInvoke.Imshow("Blur", imgBlur)
        'CvInvoke.Imshow("Tresh", imgThresh)

        ' procesado
'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

        lblChosenFile.Text = ofdOpenFile.FileName               'actualizar
etiqueta

        datApellido1 = recognizeCharsInPlate(imgApellido1)
'llamada de la función KNN para reconocimiento de caracteres
        If (datApellido1.Length = 0) Then
'comprobación de contenido
            txtApellido1.AppendText(vbCrLf + "no characters were detected"
+ datApellido1 + vbCrLf)     'mostrar error
            Return
        End If
        cdatApellido1 = comprobacionycorreccion(datApellido1, 0)
'correción de 0 y O (0 para indicar tipo texto)
```

```
        txtApellido1.AppendText(vbCrLf + vbCrLf + "Apellido 1 = " +
cdatApellido1 + vbCrLf)
        'CvInvoke.Imshow("prueba1", imgApellido1)

        datApellido2 = recognizeCharsInPlate(imgApellido2)
        If (datApellido2.Length = 0) Then
            txtApellido1.AppendText(vbCrLf + "no characters were detected"
+ cdatApellido2 + vbCrLf)
            Return
        End If
        cdatApellido2 = comprobacionycorreccion(datApellido2, 0)
        txtApellido2.AppendText(vbCrLf + vbCrLf + "Apellido 2 = " +
cdatApellido2 + vbCrLf)
        'CvInvoke.Imshow("prueba2", imgApellido2)

        datNombre = recognizeCharsInPlate(imgNombre)
        If (datNombre.Length = 0) Then
            txtApellido1.AppendText(vbCrLf + "no characters were detected"
+ cdatApellido2 + vbCrLf)
            Return
        End If
        cdatNombre = comprobacionycorreccion(datNombre, 0)
        txtNombre.AppendText(vbCrLf + vbCrLf + "Nombre = " + cdatNombre +
vbCrLf)
        'CvInvoke.Imshow("imgNombre", imgNombre)

        datSexo = recognizeCharsInPlate(imgSexo)
        If (datSexo.Length = 0) Then
            txtApellido1.AppendText(vbCrLf + "no characters were detected"
+ cdatApellido2 + vbCrLf)
            Return
        End If
        cdatSexo = comprobacionycorreccion(datSexo, 0)
        txtSexo.AppendText(vbCrLf + vbCrLf + "Sexo = " + cdatSexo + vbCrLf)
        'CvInvoke.Imshow("imgSexo", imgSexo)

        datNacionalidad = recognizeCharsInPlate(imgNacionalidad)
        If (datNacionalidad.Length = 0) Then
            txtApellido1.AppendText(vbCrLf + "no characters were detected"
+ cdatApellido2 + vbCrLf)
            Return
        End If
        cdatNacionalidad = comprobacionycorreccion(datNacionalidad, 0)
        txtNacionalidad.AppendText(vbCrLf + vbCrLf + "Nacionalidad = " +
cdatNacionalidad + vbCrLf)
        'CvInvoke.Imshow("imgNacionalidad", imgNacionalidad)

        datFechaNac = recognizeCharsInPlate(imgFechaNac)
        If (datFechaNac.Length = 0) Then
            txtApellido1.AppendText(vbCrLf + "no characters were detected"
+ cdatApellido2 + vbCrLf)
            Return
        End If
        cdatFechaNac = comprobacionycorreccion(datFechaNac, 1)
        txtFechaNac.AppendText(vbCrLf + vbCrLf + "Fecha de nacimiento = " +
cdatFechaNac + vbCrLf)
        'CvInvoke.Imshow("imgFechaNac", imgFechaNac)
```

```
        'datIDESP = recognizeCharsInPlate(imgIDESP)
        'txt.AppendText(vbCrLf + vbCrLf + "IDESP = " + datIDESP + vbCrLf)

        datFechaVal = recognizeCharsInPlate(imgFechaVal)
        If (datFechaVal.Length = 0) Then
            txtApellido1.AppendText(vbCrLf + "no characters were detected"
+ cdatApellido2 + vbCrLf)
            Return
        End If
        cdatFechaVal = comprobacionycorreccion(datFechaVal, 1)
        txtFechaVal.AppendText(vbCrLf + vbCrLf + "Fecha de validez = " +
cdatFechaVal + vbCrLf)
        'CvInvoke.Imshow("imgFechaVal", imgFechaVal)

        datNumero = recognizeCharsInPlate(imgNumero)
        If (datNumero.Length = 0) Then
            txtApellido1.AppendText(vbCrLf + "no characters were detected"
+ cdatApellido2 + vbCrLf)
            Return
        End If
        cdatNumero = comprobacionycorreccion(datNumero, 1)
        txtNumero.AppendText(vbCrLf + vbCrLf + "Numero = " + cdatNumero +
vbCrLf)
        'CvInvoke.Imshow("imgNumero", imgNumero)

    End Sub


''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
''''''''''''''''''''
    Function openImageWithErrorHandling(ByRef imgOriginalScene As Mat) As
Boolean
        Dim drChosenFile As DialogResult

        drChosenFile = ofdOpenFile.ShowDialog()                  'abrir
archivo de diálogo

        If (drChosenFile <> DialogResult.OK Or ofdOpenFile.FileName = "")
Then         'si el usuario no elige ningun archivo
            lblChosenFile.Text = "file not chosen"
'actualizar etiqueta
            Return False
        End If

        Try
            imgOriginalScene = CvInvoke.Imread(ofdOpenFile.FileName,
LoadImageType.Color)            'Intento de abrir imagen
        Catch ex As Exception
'comprobación de errores
            lblChosenFile.Text = "unable to open image, error: " +
ex.Message                        'mostrar error
            Return False
        End Try

        If (imgOriginalScene Is Nothing) Then
'si la imagen no se puede abrir
```

```
            lblChosenFile.Text = "unable to open image, image was null"
'mostrar error
            Return False
        End If

        If (imgOriginalScene.IsEmpty()) Then
'si la imagen está vacia
            lblChosenFile.Text = "unable to open image, image was empty"
'mostrar error
            Return False
        End If

        Return True
    End Function


''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
''''''''''''''''''''
End Class
```

Preprocess

```
'Preprocess.vb


Option Explicit On      'requiere declaraciones de variables explicitas
Option Strict On        'restringirr conversion de datos implicitos a
conversiones de ampliacion

Imports Emgu.CV                     '
Imports Emgu.CV.CvEnum              '
Imports Emgu.CV.Structure           '
Imports Emgu.CV.UI                  '
Imports Emgu.CV.Util                '


Module Preprocess

    '
Preproceso''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'''''''''''''''''''''''''''''''''


    ' constantes
'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
    Const GAUSSIAN_BLUR_FILTER_SIZE As Integer = 5
    Const ADAPTIVE_THRESH_BLOCK_SIZE As Integer = 13 '19
    Const ADAPTIVE_THRESH_WEIGHT As Integer = 9 '10


''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
''''''''''''''''''''

    Sub preprocess(imgOriginal As Mat, ByRef imgGrayscale As Mat, ByRef
imgBlurred As Mat, ByRef imgThresh As Mat)
```

```
        imgGrayscale = extractValue(imgOriginal)
'obtener valor para cambiar a gris
        '''''''''''''''''''''''''''''''''''''''''''''''''''''
        Dim imgMaxContrastGrayscale As Mat = maximizeContrast(imgGrayscale)
'maximizar el contraste con top hat and black hat
        'CvInvoke.CvtColor(Image, imgGrayscale, ColorConversion.Bgr2Gray)


        'Dim imgBlurred As New Mat()

        ' Difuminado gaussiano
        CvInvoke.GaussianBlur(imgMaxContrastGrayscale, imgBlurred, New
Size(GAUSSIAN_BLUR_FILTER_SIZE, GAUSSIAN_BLUR_FILTER_SIZE), 0)

        ' Valor umbral
        CvInvoke.AdaptiveThreshold(imgBlurred, imgThresh, 255.0,
AdaptiveThresholdType.GaussianC, ThresholdType.BinaryInv,
ADAPTIVE_THRESH_BLOCK_SIZE, ADAPTIVE_THRESH_WEIGHT)
    End Sub


''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
''''''''''''''''''
    Function extractValue(imgOriginal As Mat) As Mat
        Dim imgHSV As New Mat()
        Dim vectorOfHSVImages As New VectorOfMat()
        Dim imgValue As New Mat()

        CvInvoke.CvtColor(imgOriginal, imgHSV, ColorConversion.Bgr2Hsv)

        CvInvoke.Split(imgHSV, vectorOfHSVImages)

        imgValue = vectorOfHSVImages(2)

        Return imgValue
    End Function


''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
''''''''''''''''''
    Function maximizeContrast(imgGrayscale As Mat) As Mat
        Dim imgTopHat As New Mat()
           Dim imgBlackHat As New Mat()
           Dim imgGrayscalePlusTopHat As New Mat()
           Dim imgGrayscalePlusTopHatMinusBlackHat As New Mat()

        Dim structuringElement As Mat =
CvInvoke.GetStructuringElement(ElementShape.Rectangle, New Size(3, 3), New
Point(-1, -1))

        CvInvoke.MorphologyEx(imgGrayscale, imgTopHat, MorphOp.Tophat,
structuringElement, New Point(-1, -1), 1, BorderType.Default, New
MCvScalar())
        CvInvoke.MorphologyEx(imgGrayscale, imgBlackHat, MorphOp.Blackhat,
structuringElement, New Point(-1, -1), 1, BorderType.Default, New
MCvScalar())
```

```
        CvInvoke.Add(imgGrayscale, imgTopHat, imgGrayscalePlusTopHat)
        CvInvoke.Subtract(imgGrayscalePlusTopHat, imgBlackHat,
imgGrayscalePlusTopHatMinusBlackHat)

        Return imgGrayscalePlusTopHatMinusBlackHat
    End Function

    '
Postproceso'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

    Function comprobacionycorreccion(cadena As String, c As Integer) As
String
        Dim correccion As String = cadena
        If (c.Equals(0)) Then 'Se refiere que es tipo texto
            correccion = cadena.Replace(CChar("0"), CChar("O"))
        ElseIf (c.Equals(1)) Then 'Se refiere que es tipo numero
            correccion = cadena.Replace(CChar("O"), CChar("0"))
        End If
        Return correccion
    End Function

    Sub limpiarinicio()
        frmMain.txtInfo.Clear()                 'Limpiar casillas de texto
        frmMain.txtApellido1.Clear()
        frmMain.txtApellido2.Clear()
        frmMain.txtNombre.Clear()
        frmMain.txtSexo.Clear()
        frmMain.txtNacionalidad.Clear()
        frmMain.txtFechaNac.Clear()
        frmMain.txtFechaVal.Clear()
        frmMain.txtNumero.Clear()
        CvInvoke.DestroyAllWindows()                        'cerrar todas
las ventanas

        frmMain.txtInfo.AppendText("Informacion" + vbCrLf)
    End Sub

End Module
```

RecCaracteres

```
'ReCaracteres.vb

Option Explicit On      'requiere declaraciones de variables explicitas
Option Strict On        'restringirr conversion de datos implicitos a
conversiones de ampliacion

Imports System.Math

Imports Emgu.CV                     '
Imports Emgu.CV.CvEnum              '
Imports Emgu.CV.Structure           '
Imports Emgu.CV.UI                  '
Imports Emgu.CV.Util                '
```

```
'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
''''''''''''''''''''''''
Public Class RecCaracteres

    ' variables
'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'
    Public contour As VectorOfPoint

    Public boundingRect As Rectangle

    Public intCenterX As Integer
    Public intCenterY As Integer

    Public dblDiagonalSize As Double
    Public dblAspectRatio As Double
    Public intRectArea As Integer

    ' constantes para la comprobación de caracteres
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
    Const MIN_PIXEL_WIDTH As Integer = 0
    Const MIN_PIXEL_HEIGHT As Integer = 8

    Const MIN_ASPECT_RATIO As Double = 0.1
    Const MAX_ASPECT_RATIO As Double = 1

    Const MIN_RECT_AREA As Integer = 80

    ' formulas para construir variables de la clase
Contourwithdata''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'''''''''''''''''''''
    Sub New(_contour As VectorOfPoint)
        contour = _contour

        boundingRect = CvInvoke.BoundingRectangle(contour)

        intCenterX = CInt((boundingRect.Left + boundingRect.Right) / 2)
        intCenterY = CInt((boundingRect.Top + boundingRect.Bottom) / 2)

        dblDiagonalSize = Math.Sqrt((boundingRect.Width ^ 2) +
(boundingRect.Height ^ 2))

        dblAspectRatio = CDbl(boundingRect.Width) /
CDbl(boundingRect.Height)

        intRectArea = boundingRect.Width * boundingRect.Height
    End Sub


'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
''''''''''''''''''''
    Function checkIfPossibleChar(possibleChar As RecCaracteres) As Boolean
        'funcion para comprobar un caracter como válido
        If (intRectArea > MIN_RECT_AREA And _
            boundingRect.Width > MIN_PIXEL_WIDTH And boundingRect.Height >
MIN_PIXEL_HEIGHT And _
```

```
            MIN_ASPECT_RATIO < dblAspectRatio And dblAspectRatio <
MAX_ASPECT_RATIO) Then
            Return True
        Else
            Return False
        End If
    End Function


''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
''''''''''''''''''

''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
''''''''''''''''''''
    'calcular la distancia entre dos letras utilizando teorema de pitágoras
    Function distanceBetweenChars(firstChar As RecCaracteres, secondChar As
RecCaracteres) As Double
        Dim intX As Integer = Math.Abs(firstChar.intCenterX -
secondChar.intCenterX)
        Dim intY As Integer = Math.Abs(firstChar.intCenterY -
secondChar.intCenterY)

        Return Math.Sqrt((intX ^ 2) + (intY ^ 2))
    End Function


''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
''''''''''''''''''
    'calcular el ángulo
    Function angleBetweenChars(firstChar As RecCaracteres, secondChar As
RecCaracteres) As Double
        Dim dblCon As Double = CDbl(Math.Abs(firstChar.intCenterX -
secondChar.intCenterX))
        Dim dblOpp As Double = CDbl(Math.Abs(firstChar.intCenterY -
secondChar.intCenterY))

        Dim dblAngleInRad As Double = Math.Atan(dblOpp / dblCon)

        Dim dblAngleInDeg As Double = dblAngleInRad * (180.0 / Math.PI)

        Return dblAngleInDeg
    End Function


''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
''''''''''''''''''
End Class
```

KNN

```
'KNN.vb


Option Explicit On      'requiere declaraciones de variables explicitas
```

```
Option Strict On        'restringirr conversion de datos implicitos a
conversiones de ampliacion

Imports Emgu.CV                          '
Imports Emgu.CV.CvEnum                   '
Imports Emgu.CV.Structure                '
Imports Emgu.CV.UI                       '
Imports Emgu.CV.ML                       '
Imports Emgu.CV.Util

Imports System.Xml
Imports System.Xml.Serialization    'Necesario para escribir de matrices
objeto a archivo
Imports System.IO

'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
''''''''''''''''''''''''
Module DetectChars

    ' Constantes
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
    Const RESIZED_CHAR_IMAGE_WIDTH As Integer = 20          'La
comprobacion de caracteres se hace al mismo tamaño que en su cargado
    Const RESIZED_CHAR_IMAGE_HEIGHT As Integer = 30

    'variables externa
    Dim kNearest As New KNearest()


'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
''''''''''''''''''''
    Public Function loadKNNDataAndTrainKNN() As Boolean
        'nota: Hemos de leer el primer archivo XML dos veces,
        'la primera vez, leemos el archivo para obtener el número de filas,
no podemos obtener los datos en la primera lectura
        'la segunda vez, definimos las filas en la matriz de clasificación
y en la matriz de entrenamiento para poder leer el archivo


        Dim mtxClassifications As Matrix(Of Single) = New Matrix(Of
Single)(1, 1)
        Dim mtxTrainingImages As Matrix(Of Single) = New Matrix(Of
Single)(1, 1)

        Dim intValidChars As New List(Of Integer)(New Integer() {Asc("0"),
Asc("1"), Asc("2"), Asc("3"), Asc("4"), Asc("5"), Asc("6"), Asc("7"),
Asc("8"), Asc("9"), _
                                                                   Asc("A"),
Asc("B"), Asc("C"), Asc("D"), Asc("E"), Asc("F"), Asc("G"), Asc("H"),
Asc("I"), Asc("J"), _
                                                                   Asc("K"),
Asc("L"), Asc("M"), Asc("N"), Asc("O"), Asc("P"), Asc("Q"), Asc("R"),
Asc("S"), Asc("T"), _
                                                                   Asc("U"),
Asc("V"), Asc("W"), Asc("X"), Asc("Y"), Asc("Z")})
```

```
        Dim xmlSerializer As XmlSerializer = New
XmlSerializer(mtxClassifications.GetType)              'variable para leer
archivo XML
        Dim streamReader As StreamReader

        Try
            streamReader = New StreamReader("classifications.xml")
'intento de abrir el archivo
        Catch ex As Exception
'comprobación de errores
            frmMain.txtInfo.AppendText(vbCrLf + "unable to open
'classifications.xml', error: ")
            frmMain.txtInfo.AppendText(ex.Message + vbCrLf)
            Return False
        End Try

        'Primera lectura del archivo de clasificación
        mtxClassifications = CType(xmlSerializer.Deserialize(streamReader),
Matrix(Of Single))

        streamReader.Close()

        Dim intNumberOfTrainingSamples As Integer = mtxClassifications.Rows
'Obtener número de filas

        'Definir el número de filas en nuestras matrices
        mtxClassifications = New Matrix(Of
Single)(intNumberOfTrainingSamples, 1)
        mtxTrainingImages = New Matrix(Of
Single)(intNumberOfTrainingSamples, RESIZED_CHAR_IMAGE_WIDTH *
RESIZED_CHAR_IMAGE_HEIGHT)

        Try
            streamReader = New StreamReader("classifications.xml")
'reiniciar
        Catch ex As Exception
'para comprobar error
            frmMain.txtInfo.AppendText(vbCrLf + "unable to open
'classifications.xml', error:" + vbCrLf)
            frmMain.txtInfo.AppendText(ex.Message + vbCrLf + vbCrLf)
            Return False
        End Try
        'Segunda lectura para obtener los datos
        mtxClassifications = CType(xmlSerializer.Deserialize(streamReader),
Matrix(Of Single))

        streamReader.Close()

        xmlSerializer = New XmlSerializer(mtxTrainingImages.GetType)

        Try
            streamReader = New StreamReader("images.xml")
        Catch ex As Exception
'Comprobación de errores
            frmMain.txtInfo.AppendText("unable to open 'images.xml',
error:" + vbCrLf)
            frmMain.txtInfo.AppendText(ex.Message + vbCrLf + vbCrLf)
```

```
            Return False
        End Try

        mtxTrainingImages = CType(xmlSerializer.Deserialize(streamReader),
Matrix(Of Single))
        streamReader.Close()

        ' Entrenamiento
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

        kNearest.DefaultK = 1

        kNearest.Train(mtxTrainingImages, MlEnum.DataLayoutType.RowSample,
mtxClassifications)

        Return True
    End Function

    Dim j As Integer = 0

    Function recognizeCharsInPlate(image As Mat) As String 'a la función se
la envia un recorte de la imagen

        Dim imgGrayscale As New Mat()                   '
        Dim imgBlurred As New Mat()                     'imagenes del
preprocesado
        Dim imgThresh As New Mat()                      '
        Dim imgThreshCopy As New Mat()

        Preprocess.preprocess(image, imgGrayscale, imgBlurred, imgThresh)
'preprocesado - gris

        If (frmMain.cbShowSteps.Checked = True) Then ' mostrar pasos
''''''''''''''''''''''''''''
            CvInvoke.Imshow("1a", image)
            CvInvoke.Imshow("1b", imgGrayscale)
            CvInvoke.Imshow("1c", imgThresh)
            frmMain.txtInfo.AppendText(vbCrLf + "Obtenemos con el pre-
procesado las siguientes imagenes" + vbCrLf + "Pulse una tecla para
continuar")
            CvInvoke.WaitKey(0)
            CvInvoke.DestroyWindow("1a")
            CvInvoke.DestroyWindow("1b")
            CvInvoke.DestroyWindow("1c")
        End If ' mostrar pasos
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

        'CvInvoke.Resize(imgThresh, imgThresh, New Size(), 1.6, 1.6)
'redimensionar 160%
        CvInvoke.Threshold(imgThresh, imgThresh, 0.0, 255.0,
ThresholdType.Binary Or ThresholdType.Otsu)    'preprocesar valor umbral
        ''CvInvoke.Imshow("5d", imgThresh)

''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'''''
        '   preprocesado antiguo
```

```
        'CvInvoke.CvtColor(image, imgGrayscale, ColorConversion.Bgr2Gray)
'convertir a gris

        'CvInvoke.GaussianBlur(imgGrayscale, imgBlurred, New Size(5, 5), 0)
'difuminación

        ''valor umbral
        'CvInvoke.AdaptiveThreshold(imgBlurred, imgThresh, 255.0,
AdaptiveThresholdType.GaussianC, ThresholdType.BinaryInv, 11, 2.0)

''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'''''
        'realizar una copia para findcontours
        imgThreshCopy = imgThresh.Clone()

        'crear variable del vector de vectores de puntos
        Dim contours As New VectorOfVectorOfPoint()

        'obtener el contorno exterior
        CvInvoke.FindContours(imgThreshCopy, contours, Nothing,
RetrType.External, ChainApproxMethod.ChainApproxSimple)

        'declarar lista de con los datos de contorno
        Dim listOfContoursWithData As New List(Of RecCaracteres)

        'para cada contorno
        For i As Integer = 0 To contours.Size - 1
            ' Declarar nuevos datos de contorno y relacionarlos entre si
            Dim contourWithData As New RecCaracteres(contours(i))
            If (contourWithData.checkIfPossibleChar(contourWithData)) Then
'llamar a la funcion que comprueba que es un caracter
                listOfContoursWithData.Add(contourWithData)
'añadir a la lista de caracteres
            End If
        Next

        'Ordenar el dato de los contornos de izquierda a derecha
        listOfContoursWithData.Sort(Function(oneContourWithData,
otherContourWithData)
oneContourWithData.boundingRect.X.CompareTo(otherContourWithData.boundingRe
ct.X))
        ''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
        'Declarar cadena final que se devolvera
        Dim strFinalString As String = ""

        'para cada contorno validado
        For Each contourWithData As RecCaracteres In listOfContoursWithData

            'dibujar rectángulo verde alrededor del caracter
            CvInvoke.Rectangle(image, contourWithData.boundingRect, New
MCvScalar(0.0, 255.0, 0.0), 2)

            '   Calcular distancia y
ángulo''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
            '   Variables
            Dim cenlet1X As Integer
            Dim cenlet2X As Integer
```

```
            Dim cenlet1Y As Integer
            Dim cenlet2Y As Integer
            Dim distanciatotal As Double
            Dim dblAngleInDeg As Double

            'Obtener datos para el primer caracter
            If (j.Equals(0)) Then
                cenlet1X = contourWithData.intCenterX
                cenlet1Y = contourWithData.intCenterY
                frmMain.txtInfo.AppendText(vbCrLf + "cenlet1X = " +
CStr(cenlet1X))
                frmMain.txtInfo.AppendText(",  cenlet1Y = " +
CStr(cenlet1Y))
                'CvInvoke.Imshow("uno", image)
                j = 1
                'Obtener datos para el segundo caracter
            ElseIf (j.Equals(1)) Then
                'CvInvoke.Imshow("dos", image)
                cenlet2X = contourWithData.intCenterX
                cenlet2Y = contourWithData.intCenterY
                frmMain.txtInfo.AppendText(", cenlet2X = " +
CStr(cenlet2X))
                frmMain.txtInfo.AppendText(",  cenlet2Y = " +
CStr(cenlet2Y))
                j = 2
            ElseIf (j.Equals(2)) Then
                'calcular distancia
                Dim distanciaX As Integer = Math.Abs(cenlet1X - cenlet2X)
                Dim distanciaY As Integer = Math.Abs(cenlet1Y - cenlet2Y)
                distanciatotal = Math.Sqrt((distanciaX ^ 2) + (distanciaY ^
2))
                'calcular angulo
                Dim dblAdj As Double = CDbl(Math.Abs(cenlet1X - cenlet2X))
                Dim dblOpp As Double = CDbl(Math.Abs(cenlet1Y - cenlet2Y))

                Dim dblAngleInRad As Double = Math.Atan(dblOpp / dblAdj)
                dblAngleInDeg = dblAngleInRad * (180.0 / Math.PI)
                frmMain.txtInfo.AppendText(vbCrLf + "Distancia total = " +
CStr(distanciatotal))
                frmMain.txtInfo.AppendText(",  Ángulo = " +
CStr(dblAngleInDeg))
                j = 3
            End If
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

            'CvInvoke.Imshow("pruebadeberianaparecertodos", image)
            'obtener el recorte del caracter (ROI)
            Dim imgROItoBeCloned2 As New Mat(imgThresh,
contourWithData.boundingRect)

            'clonarlo para no cambiar el tamaño al redimensionar
            Dim imgROI As Mat = imgROItoBeCloned2.Clone()
            Dim imgROIResized As New Mat()

            'Redimensionar la imagen, necesario para su reconocimiento
```

```
            CvInvoke.Resize(imgROI, imgROIResized, New
Size(RESIZED_CHAR_IMAGE_WIDTH, RESIZED_CHAR_IMAGE_HEIGHT))

            'Declarar matriz con las mismas dimensiones que la estructura
de datos de las imagenes
            Dim mtxTemp As Matrix(Of Single) = New Matrix(Of
Single)(imgROIResized.Size())

            'Declarar matriz de ona fila (aplanada) con el mismo tamaño
total
            Dim mtxTempReshaped As Matrix(Of Single) = New Matrix(Of
Single)(1, RESIZED_CHAR_IMAGE_WIDTH * RESIZED_CHAR_IMAGE_HEIGHT)

            'Convertir la imagen a matriz con las mismas dimensiones
            imgROIResized.ConvertTo(mtxTemp, DepthType.Cv32F)

            'Definir el mismo tamaño, utilizar constantes
RESIZED_IMAGE_WIDTH * RESIZED_IMAGE_HEIGHT como numero de columanas
            For intRow As Integer = 0 To RESIZED_CHAR_IMAGE_HEIGHT - 1
                For intCol As Integer = 0 To RESIZED_CHAR_IMAGE_WIDTH - 1
                    mtxTempReshaped(0, (intRow * RESIZED_CHAR_IMAGE_WIDTH)
+ intCol) = mtxTemp(intRow, intCol)
                Next
            Next

            Dim sngCurrentChar As Single

            sngCurrentChar = kNearest.Predict(mtxTempReshaped)
'Podemos reconocer los caracteres

            strFinalString = strFinalString +
Chr(Convert.ToInt32(sngCurrentChar))          'añadir el caracter al vector

            If (frmMain.cbShowSteps.Checked = True) Then ' mostrar pasos
''''''''''''''''''''''''
                CvInvoke.Imshow("2", image)
                frmMain.txtInfo.AppendText(vbCrLf + "Se ha reconocido el
texto: " + strFinalString + vbCrLf + "Pulse una tecla para continuar")
                CvInvoke.WaitKey(0)
                CvInvoke.DestroyWindow("2")
            End If ' mostrar pasos
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
        Next
        If (frmMain.cbShowSteps.Checked = True) Then ' mostrar pasos
''''''''''''''''''''''''
            CvInvoke.Imshow("3", image)
            frmMain.txtInfo.AppendText(vbCrLf + "Pulse una tecla para
continuar" + vbCrLf)
            CvInvoke.WaitKey(0)
            CvInvoke.DestroyWindow("3")
            frmMain.txtInfo.Clear()
        End If ' mostrar pasos
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

        Return strFinalString
    End Function
End Module
```

Alberto
Arce
Arroyo

VISIÓN ARTIFICIAL DE DOCUMENTOS