VPAR v.7

Manual de referencia de la DLL

julio de 2016

Tabla de Contenidos

Inicialización / Finalización

vpmrInit

Inicializa el **Vehicle Plates Automatic Reader** (**VPAR**). Carga las Redes Neuronales Artificiales del OCR e inicializa parámetros. Esta función debe llamarse antes de usar cualquiera de las otras funciones de esta librería.

```
long vpmrInit (long lCountryCode,
               long lAverageCharacterHeight,
               long bDuplicateLines = false,
               long lreserved1 = 0,
               long lreserved2 = 0,
               bool bTrace = false  );
```

Parámetros

lCountryCode	Código de país para el que queremos reconocer las matrículas. Referirse al fichero de definición para una lista de todos los países soportados.
lAvCharacterHeight	Altura media aproximada de los caracteres de las matrículas a leer (en *pixels*). Si pasamos un **-1** la librería calcula de manera automática la altura. (de 20 a 50 pixeles) Si se selecciona altura automática, el tiempo de proceso aumenta considerablemente.
bDuplicateLines	Para imágenes capturadas con la mitad de las líneas pasaremos el valor 1 (true). Para los demás casos 0 (false).
lLreserved1	Reordena (cuando el país es España) el resultado de las matrículas de 2 líneas. Por defecto el valor 0 no reordena, con valor 1 se reordena el resultado de la lectura.
lreserved2	Activamos filtro especial para tratamiento de color. Los posibles valores son:

0	Hace la media de los tres canales (**Recomendado, por defecto**)
1	Usa el primer canal (rojo si la imagen/buffer es RGB o azul si es BGR)
2	Usa el segundo canal (en principio, sería siempre el verde)
3	Usa el tercer canal (azul si la imagen/buffer es RGB o rojo si es BGR)
< 0	Error
> 3	Error

bTrace	Traza interna. Este parámetro debe valer false.

Valor de Retorno

0 → Error.
1 → Ok.

MUY IMPORTANTE: DEBE RECORDAR QUE SI INICIALIZA LA LIBRERÍA CON EL PARÁMETRO lAvCharacterHeight a -1 , ESTA BUSCARÁ MATRÍCULAS CON CARACTERES ENTRE 20 y 50 PIXELES DE ALTURA.

SI SUS IMÁGENES TIENE MATRICULAS DE OTRAS ALTURAS USE LA ALTURA FIJA EN EL PARAMETRO lAvCharacterHeight o USE LA FUNCION **vparConfigureAutomaticCharacterHeight** PARA ASIGNAR UN RANGO DE ALTURAS

vpmrEnd

Libera la memoria reservada para el **Vehicle Plates Automatic Reader (VPAR)**. Esta función se llamará al finalizar el programa.

void **vpmrEnd** (void);

vpmrAddCountry

Esta función solo se debe utilizar cuando la librería se ha inicializado en modo **multipais (código 100).** En este caso, AddCountry es usado para añadir países a tener en cuenta para la lectura. Inicialmente (después del Init), ningún país está incluido y debe ser llamado al menos una vez.

long **vpmrAddCountry** (long *lCountryCode*);

Parámetros

lCountryCode Código de país.

Valor de Retorno

0 → Error.
1 → Ok.

Lectura de Matrículas

vpmrRead

Lee la matrícula contenida en una imagen. Esta función recibe como entrada una imagen. **Analiza la imagen** en busca de una matrícula de vehículo y si la encuentra, **lee la matrícula** y retorna el **texto ASCII** de la misma.

Como entrada a esta función debe proporcionarse el *buffer* de la imagen a analizar. Este *buffer* corresponde a los *pixels* de la imagen en **256 niveles de gris** (1 *byte* por *pixel*).

```
long vpmrRead ( long lWidth,
                long lHeight,
                unsigned char * pbImageData );
```

Parámetros

lWidth	Anchura de la imagen que será analizada por el VPAR (en p*ixels*).
lHeight	Altura de la imagen que será analizada por el VPAR (en p*ixels*).
pbImageData	*Buffer* con los datos (*pixels*) de la imagen en 256 niveles de grises (1 *byte* por *pixel*).

Valor de Retorno

0 → Error.
1 → Ok.

vpmrReadRGB24

Lee la matrícula contenida en una imagen. Esta función recibe como entrada una imagen. **Analiza la imagen** en busca de una matrícula de vehículo y si la encuentra, **lee la matrícula** y retorna el **texto ASCII** de la misma.

Como entrada a esta función debe proporcionarse el *buffer* de la imagen a analizar. Este *buffer* corresponde a los *pixels* de la imagen usando **tres bytes por píxel**.

```
long vpmrReadRGB24 ( long lWidth,
                     long lHeight,
                     unsigned char * pbImageData,
                     bool bFlip = false );
```

Parámetros

lWidth	Anchura de la imagen que será analizada por el VPAR (en p*ixels*).
lHeight	Altura de la imagen que será analizada por el VPAR (en p*ixels*).
pbImageData	*Buffer* con los datos (*pixels*) de la imagen usando 3 bytes por píxel (Red, Green, Blue).
bFlip	Este valor deberá ser true, sólo si el buffer RGB contiene primero la línea inferior de la imagen y continúa hacia arriba. Algunos dispositivos obtienen el buffer RGB de esta manera (*bottom-up*).

Valor de Retorno

0 → Error.
1 → Ok.

vpmrReadRGB32

Lee la matrícula contenida en una imagen. Esta función recibe como entrada una imagen. **Analiza la imagen** en busca de una matrícula de vehículo y si la encuentra, **lee la matrícula** y retorna el **texto ASCII** de la misma.

Como entrada a esta función debe proporcionarse el *buffer* de la imagen a analizar. Este *buffer* corresponde a los *pixels* de la imagen usando **cuatro bytes por píxel**.

```
long vpmrReadRGB32 ( long lWidth,
                     long lHeight,
                     unsigned char * pbImageData,
                     bool bFlip = false );
```

Parámetros

lWidth	Anchura de la imagen que será analizada por el VPAR (en p*ixels*).
lHeight	Altura de la imagen que será analizada por el VPAR (en p*ixels*).
pbImageData	*Buffer* con los datos (*pixels*) de la imagen usando 4 bytes por píxel (Red, Green, Blue, Alfa).
bFlip	Este valor deberá ser true, sólo si el buffer RGB contiene primero la línea inferior de la imagen y continúa hacia arriba. Algunos dispositivos obtienen el buffer RGB de esta manera (*bottom-up*).

Valor de Retorno

0 → Error.
1 → Ok.

vpmrReadBMP

Lee la matrícula contenida en una imagen. Esta función recibe como entrada una imagen. **Analiza la imagen** en busca de una matrícula de vehículo y si la encuentra, **lee la matrícula** y retorna el **texto ASCII** de la misma.

Como entrada a esta función debe proporcionarse una imagen en **formato BMP**.

long **vpmrReadBMP** (char * *strFilename*);

Parámetros

strFilename	Nombre del fichero BMP a procesar

Valor de Retorno

0 → Error.
1 → Ok.

vpmrReadJPG

Lee la matrícula contenida en una imagen. Esta función recibe como entrada una imagen. **Analiza la imagen** en busca de una matrícula de vehículo y si la encuentra, **lee la matrícula** y retorna el **texto ASCII** de la misma.

Como entrada a esta función debe proporcionarse una imagen en **formato JPEG**.

long **vpmrReadJPG** (char * *strFilename*);

Parámetros

strFilename	Nombre del fichero JPEG a procesar

Valor de Retorno

0 → Error.
1 → Ok.

Lectura de zonas candidatas

Estructura

La siguiente estructura es utilizada para definir las zonas candidatas a obtener una posible matrícula.

```
struct CandidateRegion
{
	long	left;
	long	top;
	long	right;
	long	bottom;
	long	ach;
};
```

Parámetros

left	Coordenada izquierda de la región dentro de la imagen.(En pixeles)
top	Coordenada superior de la región dentro de la imagen.(En pixeles)
right	Coordenada derecha de la región dentro de la imagen.(En pixeles)
bottom	Coordenada inferior de la región dentro de la imagen.(En pixeles)
ach	Tamaño promedio aproximado del carácter.(En pixeles)

vpmrFindCandidateRegionsFromGreyscale

Encuentra una o más regiones que pueden contener una matrícula. Retorna el número de zonas y sus coordenadas.

```
long  vpmrFindCandidateRegionsFromGreyscale(
                    long lWidth,
                    long lHeight,
                    unsigned char* pbImageData,
                    long* plNumRegions,
                    CandidateRegion* pRegions,
                    long lMaxRegions,
                    bool zPreciseCoordinates = false);
```

Parámetros

lWidth	Anchura de la imagen que será analizada por el VPAR (en p*ixels*).
lHeight	Altura de la imagen que será analizada por el VPAR (en p*ixels*).
pbImageData	*Buffer* con los datos (*pixels*) de la imagen en 256 niveles de grises (1 *byte* por *pixel*).
plNumRegions	Puntero con el numero de regiones encontradas.
pRegions	Vector de *CandidateRegions* donde las coordenadas de las regiones encontradas serán copiadas.
lMaxRegions	Numero de elementos en el vector pRegions.
zPreciseCoordinates	Devuelve las coordenadas precisas sin margen de seguridad.

Valor de Retorno

0 → Error.
1 → Ok.

vpmrFindCandidateRegionsFromRGB24

Encuentra una o más regiones que pueden contener una matrícula. Retorna el número de zonas y sus coordenadas.

```
long  vpmrFindCandidateRegionsFromRGB24(
                    long lWidth,
                    long lHeight,
                    unsigned char* pbImageDataRGB,
                    bool bFlip,
                    long* plNumRegions,
                    CandidateRegion* pRegions,
                    long lMaxRegions,
                    bool zPreciseCoordinates = false);
```

Parámetros

lWidth	Anchura de la imagen que será analizada por el VPAR (en p*ixels*).
lHeight	Altura de la imagen que será analizada por el VPAR (en p*ixels*).
pbImageDataRGB	*Buffer* con los datos (*pixels*) de la imagen usando 3 bytes por píxel (Red, Green, Blue).
bFlip	Este valor deberá ser true, sólo si el buffer RGB contiene primero la línea inferior de la imagen y continúa hacia arriba. Algunos dispositivos obtienen el buffer RGB de esta manera (*bottom-up*).
plNumRegions	Puntero con el numero de regiones encontradas.
pRegions	Vector de *CandidateRegions* donde las coordenadas de las regiones encontradas serán copiadas.
lMaxRegions	Numero de elementos en el vector pRegions.
zPreciseCoordinates	Devuelve las coordenadas precisas sin margen de seguridad.

Valor de Retorno

0 → Error.
1 → Ok.

vpmrFindCandidateRegionsFromRGB32

Encuentra una o más regiones que pueden contener una matrícula. Retorna el número de zonas y sus coordenadas.

```
long  vpmrFindCandidateRegionsFromRGB32(
                    long lWidth,
                    long lHeight,
                    unsigned char* pbImageDataRGB,
                    bool bFlip,
                    long* plNumRegions,
                    CandidateRegion* pRegions,
                    long lMaxRegions,
                    bool zPreciseCoordinates = false);
```

Parámetros

lWidth	Anchura de la imagen que será analizada por el VPAR (en p*ixels*).
lHeight	Altura de la imagen que será analizada por el VPAR (en p*ixels*).
pbImageDataRGB	*Buffer* con los datos (*pixels*) de la imagen usando 4 bytes por píxel (Red, Green, Blue,Alpha).
bFlip	Este valor deberá ser true, sólo si el buffer RGB contiene primero la línea inferior de la imagen y continúa hacia arriba. Algunos dispositivos obtienen el buffer RGB de esta manera (*bottom-up*).
plNumRegions	Puntero con el numero de regiones encontradas.
pRegions	Vector de *CandidateRegions* donde las coordenadas de las regiones encontradas serán copiadas.
lMaxRegions	Numero de elementos en el vector pRegions.
zPreciseCoordinates	Devuelve las coordenadas precisas sin margen de seguridad.

Valor de Retorno

0 → Error.
1 → Ok.

vpmrFindCandidateRegionsFromBMP

Encuentra una o más regiones que pueden contener una matrícula. Retorna el número de zonas y sus coordenadas.

```
long  vpmrFindCandidateRegionsFromBMP (
                    char* strFileName,
                    long* plNumRegions,
                    CandidateRegion* pRegions,
                    long lMaxRegions,
                    bool zPreciseCoordinates = false);
```

Parámetros

strFilename	Nombre del fichero BMP a procesar
plNumRegions	Puntero con el numero de regiones encontradas.
pRegions	Vector de *CandidateRegions* donde las coordenadas de las regiones encontradas serán copiadas.
lMaxRegions	Numero de elementos en el vector pRegions.
zPreciseCoordinates	Devuelve las coordenadas precisas sin margen de seguridad.

Valor de Retorno

0 → Error.
1 → Ok.

vpmrFindCandidateRegionsFromJPG

Encuentra una o más regiones que pueden contener una matrícula. Retorna el número de zonas y sus coordenadas.

```
long  vpmrFindCandidateRegionsFromJPG (
                    char* strFileName,
                    long* plNumRegions,
                    CandidateRegion* pRegions,
                    long lMaxRegions,
                    bool zPreciseCoordinates = false);
```

Parámetros

strFilename	Nombre del fichero JPG a procesar
plNumRegions	Puntero con el numero de regiones encontradas.
pRegions	Vector de *CandidateRegions* donde las coordenadas de las regiones encontradas serán copiadas.
lMaxRegions	Numero de elementos en el vector pRegions.
zPreciseCoordinates	Devuelve las coordenadas precisas sin margen de seguridad.

Valor de Retorno

0 → Error.
1 → Ok.

Consulta de Resultados

vpmrGetNumberOfPlates

Retorna el número de matrículas reconocidas en la última imagen analizada por **Vehicle Plates Automatic Reader**.

long **vparGetNumberOfPlates** (void);

Valor de Retorno

Retorna el número de matrículas leídas.(1 a 8)

NOTA: Este valor será **0** si no se encuentra o no se puede leer la matrícula, **1** en caso de que se encuentre y se lea la matrícula y podrá hasta **8** si la imagen contiene ese número de matrículas

vpmrGetText

Retorna el texto (**ASCII**) de la matrícula reconocida en la última lectura.

En la versión para camiones puede reconocer hasta 2 matrículas (0 y 1). En la versión Standard siempre pediremos que nos retorne la 0.

long **vparGetText** (char * *strResult*,
long *lPlate* = 0);

Parámetros

strResult	Variable donde nos retorna la matrícula reconocida.
lPlate	Número de matrícula que queremos que nos retorne. (0 a 7).

Valor de Retorno

0 → Error.
1 → Ok.

vpmrGetNumberOfCharacters

Retorna el número de caracteres de la matrícula indicada en la última imagen analizada por **Vehicle Plates Automatic Reader**.

long **vparGetNumberOfCharacters** (long *lPlate* = 0);

Parámetros

lPlate	Número de matrícula que queremos que consultar. (0 a 7).

Valor de Retorno

Retorna el número caracteres leídos.

vpmrGetGlobalConfidence

Retorna un valor que expresa la fiabilidad de la lectura de la matrícula de la última imagen analizada por el **Vehicle Plates Automatic Reader**.

Este valor está expresado en tanto por ciento (0% – 100%).

float **vpmrGetGlobalConfidence** (long *lPlate* = 0);

Parámetros

lPlate	Número de matrícula que queremos consultar. (0 a 7).

Valor de Retorno

Retorna la fiabilidad de la lectura.

vpmrGetAverageCharacterHeight

Retorna un valor que expresa la altura de carácter detectada automáticamente por **Vehicle Plates Automatic Reader** en la matrícula indicada

Este valor está expresado en *pixels*.

float **vpmrGetAverageCharacterHeight** (long *lPlate* = 0);

Parámetros

lPlate	Número de matrícula que queremos consultar. (0 a 7).

Valor de Retorno

Retorna la altura en *pixels* de los caracteres de la matrícula analizada.

vpmrGetCharacterConfidence

Retorna un valor que expresa la fiabilidad de un carácter determinado en la matrícula indicada.

Este valor está expresado en tanto por ciento (0% – 100%).

```
float vpmrGetCharacterConfidence ( long lIndex ,
                                   long lPlate = 0 );
```

Parámetros

lIndex	(0..n) carácter cuya fiabilidad deseamos consultar.
lPlate	Número de matrícula que queremos consultar. (0 a 7).

Valor de Retorno

Valor de Fiabilidad del carácter.

vpmrGetRectangle

Retorna las coordenadas del rectángulo que engloba la matrícula indicada tras el último análisis.

```
void vpmrGetRectangle ( long * plLeft,
                        long * plTop,
                        long * plRight,
                        long * plBottom,
                        long lPlate = 0 );
```

Parámetros

plLeft	Coordenada X de la esquina Superior Izquierda del rectángulo
plTop	Coordenada Y de la esquina Superior Izquierda del rectángulo
plRight	Coordenada X de la esquina Inferior Derecha del rectángulo
plBottom	Coordenada Y de la esquina Inferior Derecha del rectángulo
lPlate	Número de matrícula que queremos consultar. (0 a 7).

vpmrGetFormat

Esta propiedad retorna el código del país que se ajusta al formato de matrícula leído. Por ejemplo si la matrícula es española, retornará 101. En caso de no identificar el país se retorna un cero. En caso de identificar continente se retorna por ejemplo 100 para Europa.

long **vpmrGetFormat** (long *lPlate*);

Parámetros

lPlate	Número de matrícula que queremos consultar (0 a 7).

Valor de Retorno

0 → No se detecta el formato.
N → Código de país o continente para el que se detecta.

vpmrGetPolarity

Esta propiedad retorna la polaridad de la matrícula

long **vpmrGetPolarity** (long *lPlate*);

Parámetros

lPlate Número de matrícula que queremos consultar (0 a 7).

Valor de Retorno

1 → Negro sobre blanco.
2 → Otra combinación.

vpmrSavePlateImage

Guarda una imagen BMP de la matricula seleccionada.

```
long vpmrSavePlateImage ( char* strBMPFilename,
                          long lPlate);
```

Parámetros

strBMPFilename	Nombre del fichero a guardar.
lPlate	Si hay más de una matrícula, este el índice de matrícula. (0 si es la primera)

Valor de Retorno

true → Se ha guardado correctamente.
false → Error.

vpmrGetCharRectangle

Retorna las coordenadas del rectángulo que engloba los caracteres de la última imagen leída. **Las coordenadas son relativas a la imagen guardada por la función vpmrSavePlateImage.**

```
void vpmrGetRectangle (long lIndex,
                       long * plLeft,
                       long * plTop,
                       long * plRight,
                       long * plBottom,
                       long lPlate = 0 );
```

Parámetros

lIndex	Índice del carácter. (El primer carácter será el 0)
plLeft	Coordenada X de la esquina Superior Izquierda del carácter
plTop	Coordenada Y de la esquina Superior Izquierda del carácter
plRight	Coordenada X de la esquina Inferior Derecha del carácter
plBottom	Coordenada Y de la esquina Inferior Derecha del carácter
lPlate	Número de matrícula que queremos consultar. (0 a 7).

Control del Tiempo

vpmrGetProcessingTime

Retorna el tiempo de proceso en la última operación de lectura.

Este valor esta expresado en milisegundos.

long **vpmrGetProcessingTime** ( );

Valor de Retorno

Tiempo de proceso en la última operación de lectura (en milisegundos).

vpmrSetTimeOut

Indica al motor de reconocimiento un tiempo de proceso máximo tras el cual retornará con el resultado que haya obtenido hasta el momento.

Este valor está expresado en milisegundos.

void **vparSetTimeOut** (long *lMilliseconds*);

Parámetros

lMilliseconds	Tiempo máximo de proceso (en milisegundos).

Configuración Opcional

vpmrSetCorrectionCoefficients

Esta función establece los coeficientes para el proceso de corrección de la distorsión que será aplicado a las imágenes antes de ser analizadas.

Los tipos de distorsiones que pueden corregirse son: tangencial (perspectiva horizontal y/o vertical) y rotación (matrícula inclinada).

Los coeficientes especificados en los parámetros serán aplicados a todas las imágenes analizadas hasta que esta función sea llamada con diferentes parámetros o se llame a la función ***vpmrSetDistortionCorrectionOff***.

```
void vpmrSetCorrectionCoefficients (    float fDistance,
                                        float fVerticalCoeff,
                                        float fHorizontalCoeff,
                                        float fAngle);
```

Parámetros

fDistance	Distancia aproximada entre la cámara y el objeto (en metros).
fVerticalCoeff	Coeficiente para corregir la Perspectiva Vertical.
fHorizontalCoeff	Coeficiente para corregir la Perspectiva Horizontal.
fAngle	Ángulo para corregir la Rotación (inclinación).

vpmrSetCorrectionCoefficientsEx

Esta función establece los coeficientes para el proceso de corrección de la distorsión que será aplicado a las imágenes antes de ser analizadas.

Los tipos de distorsiones que pueden corregirse son: tangencial (perspectiva horizontal y/o vertical), rotación (matrícula inclinada) y torsión.

Los coeficientes especificados en los parámetros serán aplicados a todas las imágenes analizadas hasta que esta función sea llamada con diferentes parámetros o se llame a la función ***vpmrSetDistortionCorrectionOff***.

```
void vpmrSetCorrectionCoefficientsEx ( float fDistance,
                                       float fVerticalCoeff,
                                       float fHorizontalCoeff,
                                       float fAngle,
                                       float fVerticalSkew,
                                       float fHorizontalSkew);
```

Parámetros

fDistance	Distancia aproximada entre la cámara y el objeto (en metros).
fVerticalCoeff	Coeficiente para corregir la Perspectiva Vertical.
fHorizontalCoeff	Coeficiente para corregir la Perspectiva Horizontal.
fAngle	Ángulo para corregir la Rotación (inclinación).
fVerticalSkew	Coeficiente para corregir la torsión Vertical.
fHorizontalSkew	Coeficiente para corregir la torsión Horizontal.

vpmrSetDistortionCorrectionOff

Esta función desactiva el proceso de corrección de distorsión.

La función ***vpmrSetCorrectionCoefficients*** debe usarse para volver a activar la corrección de distorsión.

void **vpmrSetDistortionCorrectionOff** ();

vpmrConfigureAutomaticCharacterHeight

Esta función configura el intervalo de la lectura en modo de *Altura de Caracteres Automática*.

Cuando ***vpmrInit*** es llamado con el parámetro *lAvCharacterHeight* a -1, se selecciona el modo de *Altura de Caracteres Automática*. Por defecto, el rango de alturas escaneado va de 25 a 60 pixels de altura para los caracteres.

Usando esta función el usuario puede elegir el intervalo de alturas de caracteres que VPAR escaneará.

```
long vpmrConfigureAutomaticCharacterHeight (    long lNumSteps,
                                                long *  plSteps );
```

Parámetros

lNumSteps	SIEMPRE 2.
plSteps	Array con el inicio del intervalo en la primera posición y el final del intervalo en la segunda.pe. 25 , 45

Valor de Retorno

0 → Error.
1 → Ok.

NOTA: Para recuperar la configuración por defecto, el siguiente código debe ejecutarse:

```
long lSteps[8] = { 25, 30, 35, 40, 45, 50, 55, 60 };
vpmrConfigureAutomaticCharacterHeight (8, lSteps );
```

vpmrSetRectangle

Esta función asigna el rectángulo dentro de la imagen donde la inspección tendrá lugar. Sólo la región especificada será inspeccionada para buscar y reconocer la matrícula.

Esta función puede ser usada, por ejemplo, para acelerar el proceso cuando se conoce a priori la posición aproximada de la matrícula dento de la imagen.

```
long vpmrSetRectangle (     long lLeft,
                            long lTop,
                            long lWidth,
                            long lHeight );
```

Parámetros

lLeft, lTop	Coordenadas de la esquina superior izquierda del rectángulo (en *píxels*).
lWidth, lHeight	Dimensiones del rectángulo (en *píxels*).

Valor de Retorno

0 → Error.
1 → Ok.

NOTA:
El rectángulo de inspección especificado se aplicará a todas las inspecciones ejecutadas a partir de entonces, hasta que vparSetRectangle sea llamado de nuevo con parámetros diferentes.
Si se desea volver inspeccionar la totalidad de la imagen, vparSetRectangle debe ser llamado con todos sus parámetros a 0, es decir:

vpmrSetRectangle (0, 0, 0, 0);

Por defecto, tras la inicialización de la librería con vparInit, el rectángulo de inspección corresponde a toda la imagen.

vpmrShadowKillerOn

Activa el preprocesado para tratar las sombras proyectadas sobre la parte superior de las matrículas.

void **vpmrShadowKillerOn** ();

vpmrShadowKillerOff

Desactiva el preprocesado para tratar las sombras proyectadas sobre la parte superior de las matrículas.

void **vpmrShadowKillerOff** ();

vpmrStrictSyntaxOn

Activando esta función, el formato retornado para las matrículas personalizadas (matrículas que no siguen el formato estándar del país) es 0.

void **vpmrStrictSyntaxOn** ();

vpmrStrictSyntaxOff

Activando esta función, el formato retornado para las matrículas personalizadas (matrículas que no siguen el formato estándar del país) es el formato del país. Este es el comportamiento por defecto.

void **vpmrStrictSyntaxOff** ();

vpmrSetScaleFactor

Escala la imagen internamente antes de ser procesada.

void **vparSetScaleFactor** (float *fScale*);

Parámetros

fScale	Factor de escalado. El ancho y alto de la imagen serán multiplicados por este factor.

Nota: El escalado se hará de manera interna, todos los parámetros de configuración conservaran los parámetros relativos a la imagen original (tamaño de carácter, área de interés...).
Los resultados (tamaño medio carácter, rectángulo...) harán referencia también a la imagen original.

vpmrReturnSpaces

Activando esta función después de la función Init, las matrículas reconocidas incluirán los espacios en blanco entre los caracteres.

void **vpmrReturnSpaces** ();

Almacenamiento de datos en el HASP

vpmrWriteHASP

Escribe datos en la memoria interna de la mochila de protección (dongle). Esta capacidad de almacenar datos en el HASP puede ser usada por el usuario para cualquier propósito. Los datos son encriptados automáticamente antes de escribirse en la memoria del HASP.

Pueden almacenarse un máximo de 24 bytes.

```
long vpmrWriteHASP ( unsigned char * pData,
                     long lSize );
```

Parámetros

pData	Buffer con los datos a escribir en la memoria del HASP.
lSize	Tamaño (en bytes) de los datos a escribir (máximo 24).

Valor de Retorno

0 → Error.
1 → Ok.

vpmrReadHASP

Lee los datos almacenados en la memoria interna de la mochila de protección (dongle). Esta capacidad de almacenar datos en el HASP puede ser usada por el usuario para cualquier propósito. Los datos son desencriptados automáticamente después de leerse de la memoria del HASP.

long **vpmrReadHASP** (unsigned char * *pData,*
long *lSize*);

Parámetros

pData	Buffer donde se almacenarán los datos leídos de la memoria del HASP.
lSize	Tamaño (en bytes) de los datos a leer.

Valor de Retorno

0 → Error.
1 → Ok.

vpmrComparePlates

Compara dos matrículas y produce una medida de similitud.

La longitud máxima de las cadenas a comparar es de 15 caracteres. La comparación no distingue mayúsculas de minúsculas. Los espacios se eliminan internamente y no se tienen en cuenta.

long **vpmrComparePlates** (char * *pPlate1*,char * *pPlate2*);

Argumentos

pPlate1	Buffer conteniendo la cadena de la primera matrícula a comparar.
pPlate2	Buffer conteniendo la cadena de la segunda matrícula a comparar.

Valor de retorno

Resultado de similitud entre 0 y 1000 (1000 significa que ambas matrículas son idénticas).

Se retorna un valor negativo en caso de error.

Ejemplo de Uso

```
void func ()
{
    bool ok;

    // Los caracteres tienen aproximadamente 30 pixels de altura, no hay que duplicar
    // líneas horizontales y queremos que ordene las matrículas cuadradas de dos líneas.
    ok = (bool) vpmrInit (101,30, false, true, false, false);
    if (ok)
        {
        unsigned char * buffer;   // buffer de la imagen.
        long numchars;            // número de caracteres leídos.
        char text[32];            // string donde almacenar la matrícula leída.
        float Cf;                 // fiabilidad de la lectura.
        float characterCf[32];    // array para almacenar la fiabilidad para cada carácter.

        // Reserva de memoria para el buffer.
        . . .
        // Obtener el buffer de la imagen en 256 niveles de gris (1 byte por pixel) y
        // almacenarlo en buffer.
        . . .

        ok = (bool) vpmrRead (384, 288, buffer);
        if (ok)
            {
            vpmrGetText (text, 0);
            Cf = vpmrGetGlobalConfidence (0);
            numchars = vpmrGetNumberOfCharacters (0);
            for (long i = 0; i < numchars; i++)
                {
                characterCf[i] = vpmrGetCharacterConfidence (i, 0);
                }
            }
        else
            {
            // Error al leer matrícula.
            }
        // Leeremos las siguientes imágenes.

      // Finalizaremos el motor cuando hayamos leído todas las imagenes
      vpmrEnd ();
      }
    else
      {
      // Error al inicializar el VPAR.
      }
}
```