

The i2e Image Enhancement Library Version 6

C_INT.dll

API Reference Manual

Document Version 1.0, May 2008

Colour-Science AG
Bettsweilerstrasse 2
CH-8344 Bärestwil
Switzerland

+41 44 979 1070
support@colour-science.com

1. Introduction

1.1 The i2e image enhancement system

I2E is an image processing library intended mainly for fully automatic image enhancement. I2E can be used to process 8bit per color images (e.g. from digital cameras) as well as 12-16bit per color images from film scanners. For 12bit images we provide a special JPEG reader and writer, and for raw images a special reader.

The following processing stages are provided:

- ABE (Adaptive Brightness Enhancement)
- ACE (Adaptive Color Enhancement)
- SHE (Shadow and Highlight Enhancement)
- MCE (Memory Color Enhancement)
- LSE (Local Sharpness Enhancement)
- LNR (Local Noise Reduction)
- RER (Red Eye Reduction)

The individual stages can be activated and deactivated individually, and their impact can be controlled by the user through a set of parameters. The default values of these parameters have been optimized for typical amateur images. For more professional usage, it might be advisable to reduce the strength of some of the stages and to fine-tune some of the parameters. Please contact your system supplier for more details.

Additional parameters are provided to the user, which allow her/him to control the overall color balance, brightness, contrast and color saturation of the processed images. In an interactive environment the user can apply color, brightness and contrast corrections to each individual image.

A separate global sharpening function is provided which can be used to compensate for scanner/printer specific blur.

There are two processing modes supported:

Single image processing: Each image is individually analyzed and processed in one step. This mode is recommended for digital camera and similar images. It avoids the overhead of loading an image twice, once for analysis and once for processing.

Order processing: In a first step, all images of an order are analyzed. It is assumed that an order consists of images stemming from the same film roll or from the same digital camera, flat-bed scanner etc. In a second step the images of an order are processed individually, taking into account the results of both the single image analysis and the order analysis. A factor is provided to control the relative impact of the two contributions on the final result. Order processing is required for film scanner images. It might also be useful in order to achieve more homogeneous results from image to image in situations where the lighting conditions are tightly controlled and the subjects quite similar, as e.g. in the case of school photography. Thumbnails of reasonable size can be used for image analysis in order to minimize the overhead of loading each image twice.

1.2 Hardware and software requirements

OS:	Microsoft Windows 98 and higher (Unix/Linux on request)
Processor:	Pentium IV, 2.5GHz or higher (or equivalent non-Intel processor)
Memory:	min. 512MB

1.3 Files to include in your build

C/C++:	I2E_Types.h (type declarations)
	I2E_C_Interface.h (function declarations)
C#:	I2ELib, contained in the CS sample folder
Basic:	I2E_VB_Interface.bas (on request)

Delphi: I2EIntf.pas (on request)

1.4 **Multithreading**

Library routines do not launch threads and are thread-safe on distinct I2E objects on the following assumptions:

- The same image buffer is not used simultaneously from different threads
- All threads are using the same allocation/free functions

1.5 **Compiler settings, LIB files**

I2E uses quadword alignment, which has an impact on the size of the structures defined in "I2E_Types.h". See the Module "I2E_VB_Interface" on how to adjust the structure sizes in a Visual Basic environment. Enumerated types are treated as integers, for compatibility with MS VC++. The calling convention is "__stdcall".

Two sets of import libraries are provided, I2E_CINT_ms.LIB, I2E_CIIO_ms.LIB for MS Visual C++ and I2E_CINT_bc.LIB, I2E_CIIO_bc.LIB for Borland C++Builder.

1.6 **Library protection system**

The i2e dynamic link library file (I2_CINT.dll) contains the encrypted library along with authorization information, referred to hereafter as "Type A authorization". For compatibility reasons and for special applications the library has a built-in protection mechanism, based on an application ID / password pair, referred to hereafter as "Type B authorization". The application ID is a unique identifier for a particular application, as specified in the corresponding i2e ISV agreement. The password contains encrypted information about the authorized capabilities, like processing speed, throughput limits etc.

Type A authorization:

After a first call to I2E_Initialize (see 3.1.3), the library runs without any restrictions for a trial period of 10 days. Within this time span the library must be registered using the function I2E_Register (see 3.2.1). If the 10 days trial period expires without a valid registration, then the library turns into demo mode, producing watermarked images. For Type A authorization pass the applicationID and password arguments in I2E_Initialize as NULL pointers (or as empty strings) .

Type B authorization:

Pass a valid applicationID / password pair as arguments in I2E_Initialize.

1.7 **Syntax and data types conventions of this document**

Since the i2e library can be used with many different development and programming language environments, the following terms and conventions apply for this document:

integer

signed 32-bit integer variable

Environment	Type	Example	In a function declaration
C/C++	int	int myInt	int myInt
C#	int	int myInt	int myInt
Visual Basic	Long	myInt As Long	ByVal myInt As Long
Delphi	LongInt	myInt:LongInt	myInt:LongInt

integer*

signed 32-bit integer variable reference

Environment	Type	Example	Declared as an argument in a function
C/C++	int*	int* myInt	int* myInt
C#	int	int myInt	ref int myInt
Visual Basic	Long	myInt As Long	ByRef myInt As Long
Delphi	PLongInt=^LongInt	myInt:PLongInt	PLongInt:myInt

float

32-bit floating point variable

Environment	Type	Example	Declared as an argument in a function
C/C++	float	float myFloat	float myFloat
C#	float	float myFloat	float myFloat
Visual Basic	Single	myFloat As Single	ByVal myFloat As Single
Delphi	Single	myFloat:Single	Single:myFloat

float*

32-bit floating point variable reference

Environment	Type	Example	Declared as an argument in a function
C/C++	float*	float* myFloat	float* myFloat
C#	float	float myFloat	ref float myFloat
Visual Basic	Single	myFloat As Single	ByRef myFloat As Single
Delphi	PSingle=^Single	myFloat:PSingle	PSingle:myFloat

float[]

32-bit floating point array

Environment	Type	Example	Remarks
C/C++	float	float myArray[3]	
C#	float	float[] myArray	a)
Visual Basic	(.. To ..) As Single	myArray(0 To 2) As Single	
Delphi	array[..] of Single	myArray:array[0..2] of Single	

a) See I2ELib.cs in the CS sample folder for marshalling definitions.

double

64-bit floating point variable

Environment	Type	Example	Declared as an argument in a function
C/C++	double	double myFloat	double myFloat
C#	double	double myFloat	double myFloat
Visual Basic	Double	myFloat As Double	ByVal myFloat As Double
Delphi	Double	myFloat:Double	Single:myDouble

string

character string

Environment	Type	Example	Declared as an argument in a function
C/C++	char[]	char myString[256]	char* myString
C#	String	String myString	String/IntPtr myString a)
Visual Basic	String	Dim myString As String * 256	ByVal myString As String
Delphi	PChar	myString:PChar	PChar:myString

a) You can pass a String type for input arguments. however, for output arguments, like e.g. version in I2E_GetVersion (), you must pass an allocated IntPtr type argument, and marshal the data from unmanaged to managed memory (Marshal.PtrToStringAnsi, see also the VC# example).

structures

variable composed of a collection of values that can have different types, such as I2E_Image (see 2.6)

Environment	Type example	Example	Declared as an argument in a function
C/C++	typedef struct I2E_Image (see I2E_Types.h)	I2E_Image myImage	I2E_Image* myImage
C#	class I2E_Image (see I2ELib.cs)	myImage = new I2E_Image ()	IntPtr myImagePtr a)
Visual Basic	Type I2E_Image (see I2E_VB_Interface.bas)	Dim myImage As I2E_Image	ByRef myImage As I2E_Image
Delphi	type I2E_Image PI2E_Image=^I2E_Image (see I2EIntf.pas)	myImage:I2E_Image	myImage:PI2E_Image

a) You must pass an IntPtr type variable pointing to an allocated unmanaged memory block of the size of the structure, and marshal the data from managed memory to unmanaged memory and back.

Typical code sequence:

```
I2E_Statistics statistics = new I2E_Statistics ();
IntPtr statisticsPtr = Marshal.AllocHGlobal (Marshal.SizeOf (statistics));
Marshal.StructureToPtr (statistics, statisticsPtr, true);
int rc = I2E.I2E_StartOrder (i2eObject, statisticsPtr);
Marshal.PtrToStructure (statisticsPtr, statistics);
```

(see also the VC# sample code).

1.8 Related documents

The image creation and I/O functions, as well as the ICC profiling functions are part of the I2E_C_ImageIO library. Please refer to the corresponding documentation.

2. Structures and variables

2.1 I2E_SourceType

Description: Type of the image source
Data type: Enumeration

Name	Data type	Value
I2E_FilmScanner	integer	0
I2E_Camera	integer	1
I2E_Other	integer	2

2.2 I2E_ColorOrder

Description: Order of the image colors
Data type: Enumeration

Name	Data type	Value
I2E_BGR	integer	0
I2E_RGB	integer	1
I2E_Gray	integer	2
I2E_BGRA	integer	3

2.3 I2E_DataType

Description: Data type of the image colors (one byte or two bytes)
Data type: Enumeration

Name	Data type	Value
I2E_UChar	integer	0
I2E_UShort	integer	1

2.4 I2E_RenderingIntent

Description: Rendering intent for an ICC-profile based color transform
Data type: Enumeration
Remarks: The related functions are part of the I2E_C_ImageIO library.

Name	Data type	Value
I2E_Perceptual	integer	0
I2E_Relative	integer	1

I2E_Saturation	integer	2
I2E_Absolute	integer	3

2.5 I2E_ObjectType

Description: The type of an object
Data type: Enumeration
Remarks: There is only one object type implemented in the current version.

Name	Data type	Value
I2E_ObjectTypeFace	integer	1

2.6 I2E_Image

Description: Descriptor of an i2e image
Data type: Structure

Name	Data type	Description	Value
sourcetype	I2E_SourceType	type of the image source	one of I2E_SourceType
colororder	I2E_ColorOrder	order of the image colors	one of I2E_ColorOrder
datatype	I2E_DataType	data type of the image colors	one of I2E_DataType
bgrfacs[3]	float array	scanner exposure factors	a)
maxval	integer	maximum color value	b)
width	integer	image pixel width	
height	integer	image pixel height	
bytesperline	integer	number of bytes per line	c)
imagedata	reference	image data buffer reference	
imageID	integer	image registration identifier	d) for internal use

- a) Applicable for data type I2E_UShort only. Scanner exposure factor for blue, green and red. Set to 1.0 if not known or for constant exposure time.
- b) Maximum value of a pixel color:
data type I2E_UChar: 255
data type I2E_UShort: 4095 for 12-bit raw or jpg images
16383 for 14-bit raw images
65535 for 16-bit raw images
- c) Dependent on the data type and padding:
data type I2E_UChar: width * height * 3 + padding bytes
data type I2E_UShort: width * height * 6 + padding bytes
- d) The image creation functions of the I2E_C_ImagelO library (e.g. I2E_CreatelImage, I2E_LoadImage etc.) register the image internally and associate an image identifier to the image. This identifier is used in subsequent calls to I2E_FreelImage and I2E_FreelImages to de-allocate the image data buffer. Set to zero when manually creating an I2E_Image structure.

2.7 I2E_Parameters

Description: Used for parameter settings
Data type: Structure

Name	Data type	Description	Range	Default value
brightness	float	general brightness	-1 (dark) to +1 (bright)	0
hlcontrast	float	highlight contrast	-1 (low) to +1 (high)	0
shcontrast	float	shadow contrast	-1 (low) to +1 (high)	0
saturation	float	general saturation	-1 (gray) to +1 (highly saturated)	0
b	float	general blue color balance	-1 to +1	0
g	float	general green color balance	-1 to +1	0
r	float	general red color balance	-1 to +1	0
ABEfac	float	brightness correction strength	0 to +2	1
ACEfac	float	color correction strength	0 to +2	1
highlights	float	highlights correction strength	0 to +2	1

shadows	float	shadows correction strength	0 to +2	1
MCEfac	float	memory color correction strength	0 to +1	1
LSEfac	float	sharpening strength	0 to +1	1
shadowdesat	float	shadows desaturation	desat.: -1 (no) to +1 (strong)	-1
huerotation	float	rotation (in degrees) of the hues	0 to 359	0
orderfac	float	Rel. contribution of order analysis to corrections	0 to +1	0

2.8 I2E Statistics

Description: Statistics of an image or order
Data type: Structure

Name	Data type	Description	Default value
sourcetype	I2E_SourceType	type of the image source	
datatype	I2E_DataType	data type of the image colors	
imagecount	unsigned long	image count	
bgrmeans[3]	float array	bgr average	
bgrsigmas[3]	float array	bgr standard deviation	
bgrmins[3]	float array	bgr minimum value	
bgrmaxs[3]	float array	bgr maximum value	
satmin	float	minimum saturation	
satmean	float	average saturation	
satmax	float	maximum saturation	
satsigma	float	saturation standard deviation	
lcmin	float	minimum local contrast	
lcmean	float	average local contrast	
lcmax	float	maximum local contrast	
lcsigma	float	local contrast standard deviation	
fstatdat[]	float array	for internal use	
istatdat[]	integer array	for internal use	

2.9 I2E J12Parameter

Description: Statistics of an image or order
Data type: Structure

Name	Data type	Description	Range	Default value
brightness	float	brightness	-1 (dark) to +1 (bright)	0
contrast	float	contrast	-1 (low) to +1 (high)	0
saturation	float	saturation	-1 (low) to +1 (high)	0
ctcmatrix[3][3]	float array	crosstalk matrix	a)	a)
bgrgammas[3]	float array	blue/green/red gammas, b)		0

a) The order of columns and rows is BGR. The default is a unity matrix. The effect is similar to the "Channel-Mixer" in Photoshop, with the following differences: (i) luminance is left unchanged by this transform, (ii) the effect becomes weaker with increasing luminance, in order to avoid colorshifts in the highlights. This matrix should be customized for a particular scanner type either by the system supplier or by the scanner manufacturer.

b) Applied only for exposure level based gamma correction. The values are added to the internally evaluated, exposure level dependent, gamma corrections.

2.10 I2E ImageInfo

Description: Analysis information of an image
Data type: Structure

Name	Data type	Description	Range
bgrfac[3]	float array	scanner exposure factors, see also 2.5	

sourcetype	I2E_SourceType	type of the image source	see 2.1
datatype	I2E_DataType	data type of the image colors	see 2.4
bgrmeans[3]	float array	bgr average	0 - 1.0
bgrsigmas[3]	float array	bgr standard deviation	0 - 1.0
bgrmins[3]	float array	bgr minimum value	0 - 1.0
bgrmaxs[3]	float array	bgr maximum value	0 - 1.0
satmin	float	minimum saturation	0 - 1.0
satmean	float	average saturation	0 - 1.0
satmax	float	maximum saturation	0 - 1.0
satsigma	float	saturation standard deviation	0 - 1.0
lcmin	float	minimum local contrast	0 - 1.0
lcmean	float	average local contrast	0 - 1.0
lcmax	float	maximum local contrast	0 - 1.0
lcsigma	float	local contrast standard deviation	0 - 1.0
isunder	float	indicator for under exposed images	0 - 1.0
haslowcontrast	float	indicator for low contrast images	0 - 1.0
hasfog	float	indicator for fog (for future use)	always 0
imageOK	bool	false for bad images (skip processing/printing)	true/false

2.11 I2E_ImageInfoEx

Description: Extended analysis information of a scanned 12-16bit image
Remarks: Basically the same as I2E_ImageInfo but extended with negative information.
Data type: Structure

Name	Data type	Description	Range
size	integer	size of the I2E_ImageInfoEx structure	
sourcetype	I2E_SourceType	type of the image source	see 2.1
datatype	I2E_DataType	data type of the image colors	see 2.4
bgrmeans[3]	float array	bgr average	0 - 1.0
bgrsigmas[3]	float array	bgr standard deviation	0 - 1.0
bgrmins[3]	float array	bgr minimum value	0 - 1.0
bgrmaxs[3]	float array	bgr maximum value	0 - 1.0
satmin	float	minimum saturation	0 - 1.0
satmean	float	average saturation	0 - 1.0
satmax	float	maximum saturation	0 - 1.0
satsigma	float	saturation standard deviation	0 - 1.0
lcmin	float	minimum local contrast	0 - 1.0
lcmean	float	average local contrast	0 - 1.0
lcmax	float	maximum local contrast	0 - 1.0
lcsigma	float	local contrast standard deviation	0 - 1.0
densshfac	float	indicator for under exposed images	0 - 1.0
denshlfac	float	indicator for low contrast images	0 - 1.0
imageOK	bool	false for bad images (skip processing/printing)	true/false
negbgrfacs[3]	float array	scanner exposure factors, see also 2.5	
negsharpness	float	indicator for sharpness of a scanned image	a)
negExpLevel	float	estimated negative exposure level	variable
negskipStatistic	bool	true if the image is skipped for film statistics	true/false

a) The higher the value (up to 1.0) the better the sharpness of an image. Typical values for sharp images are in the range of 0.8 - 1.0. This value is also internally used to calculate an overall film sharpness for the determination of the sharpening strength.

2.12 I2E_FaceRect

Description: Information about a face rectangle
Remarks: See also 3.7.2, I2E_GetObjects.
Data type: Structure

Name	Data type	Description	Remarks
------	-----------	-------------	---------

X	integer	x coordinate of the face rectangle	
Y	integer	y coordinate of the face rectangle	
Width	integer	x extension of the face rectangle	
Height	integer	y extension of the face rectangle	
Neighbors	integer	number of neighbors within a cluster	a)
DeltaLum	integer	center luminance range	b)
RedEyePixelCount	integer	number of red-eye pixels	
SkinPixelRatio	double	center skin pixel ratio	c)
AvgSkinL	double	average skin Lab L value	
AvgSkinLa	double	average skin Lab a value	
AvgSkinLb	double	average skin Lab b value	
Valid	bool	true for a positive validation, else false	

- a) When scanning the image for faces then the system normally detects multiple occurrences at slightly different positions and extensions, called neighbours. The resulting face rectangle represents the average of the neighbors. The more neighbors the higher the probability of a valid face rectangle.
- b) Shows the difference between the high luminance center area skin pixels and the low luminance of all center area pixels. This value is used for validation purposes.
- c) Shows the ratio between the skin color pixels and the non skin color pixels of the center area of the face rectangle. This value is used for validation purposes.

2.13 I2E_Objects

Description: Information about detected objects
Remarks: See also 3.7.2, I2E_GetObjects.
Data type: Structure

Name	Data type	Description	Remarks
Orientation	integer	orientation of the image	a)
NumberOfObjects	integer	number of elements in ObjectRects	
ObjectRects	reference	points to an array of I2E_FaceRect structures	b)

- a) Image orientation: 0 = upside up, 1 = upside right, 2 = upside left

2.14 I2E_ObjectParameters

Description: Object processing parameters
Data type: Structure

Name	Data type	Description	Default value
Size	integer	size of the structure	
FaceDetectionDisabled	bool	true to disable face detection	false
BottomLineFirst	bool	true if the image data buffer contains the bottom row first	false a)
AutoRotate	bool	true to autorotate processed image	false
CenterSingleFace	bool	true to autocenter single face image	false b)
ScaleSingleFace	bool	true to autoscale single face image	false c)

- a) E.g. if the bitmap data buffer is obtained from a DIB.
b) Automatic centering of single face portrait images (not yet implemented in the current version)
c) Automatic scaling of single face portrait images (not yet implemented in the current version)

3. The API

3.1 Initialization and termination functions

3.1.1 I2E_Create

Purpose: Creation of an i2e instance.
 Parameters: None.
 Returns: A reference to an i2e instance, referred to hereafter as i2eObject. Pass this reference to all subsequent calls to I2E functions.
 Remarks: This function must be called once at the beginning of an i2e session, and the reference returned must be passed to all i2e function as the first argument.

3.1.2 I2E_Destroy

Purpose: Deletion of an i2e instance.
 Returns: Nothing.
 Remarks: This function must be called at the end of an i2e session.
 Parameters:

Name	Data type	Description	Remarks
i2eObject	i2eObject	i2e instance reference	created by I2E_Create

3.1.3 I2E_Initialize (i2eObject, applicationID, password)

Purpose: Initialization of the i2e library.
 Returns: An integer return code.
 Remarks: This function must be called once at the beginning of an i2e session, after calling I2E_Create.

Parameters:

Name	Data type	Description	Remarks
i2eObject	i2eObject	i2e instance reference	created by I2E_Create
applicationID	string	application ID	provided by system supplier a)
password	string	authorization code	provided by system supplier a)

a) Pass a NULL pointer or an empty string for Type A authorization (see1.6).

Return codes:

- 0: success
- 2: memory error
- 3: already initialized
- 4: infile error
- 9: authorisation failure
- +9: authorisation warning

3.1.4 I2E_Exit (i2eObject, applicationID)

Purpose: Termination of an i2e instance.
 Returns: An integer return code.
 Remarks: This function must be called at the end of an i2e session, before I2E_Delete.
 Parameters:

Name	Data type	Description	Remarks
i2eObject	i2eObject	i2e instance reference	created by I2E_Create
applicationID	string	application ID	same as I2E_Initialize

Return codes:

- 0: success
- 1: failure

3.2 Library registration functions

3.2.1 I2E_Register (i2eObject, name, code)

Purpose: Registration of the i2e library.
 Return: An integer return code.

Remarks: This function is required for Type A authorization (see 1.6). This function must be called only once for a particular authorization code, prior to a call to I2E_Initialize.

Parameters:

Name	Data type	Description	Remarks
i2eObject	i2eObject	i2e instance reference	created by I2E_Create
name	string	user name	provided by system supplier
code	string	authorization code	provided by system supplier

Return codes:

- 0: ok
- 1: failure
- 2: trial period expired
- 3: name or code missing

3.2.2 I2E_GetRegistrationCode (i2eObject)

Purpose: Get a registration code for Type B authorization (see 1.6).

Returns: An unsigned integer return code.

Remarks: Pass the return code to the system supplier in order to obtain an authorization code.

Parameters:

Name	Data type	Description	Remarks
i2eObject	i2eObject	i2e instance reference	created by I2E_Create

3.2.3 I2E_GetRegistrationCodeEx (i2eObject, type, name, code)

Purpose: Get an extended registration code for Type A or B authorization (see 1.6).

Returns: An unsigned integer return code.

Remarks: Pass the return code to the system supplier in order to obtain an authorization code.

Parameters:

Name	Data type	Description	Remarks
i2eObject	i2eObject	i2e instance reference	created by I2E_Create
type	integer	library protection system type	1 = Type B, 2 = Type A
name	string [out]	registered user name	a)
code	string [out]	registered authorization code	b)

a) Registered user name if the library has been registered before (only applicable for Type A authorization). The buffer must have a minimum length of 256 characters, or pass a NULL pointer if you don't want to get the information back.

b) Registered authorization code if the library has been registered before (only applicable for Type A authorization). The buffer must have a minimum length of 256 characters, or pass a NULL pointer if you don't want to get the information back.

Return codes:

- >0: valid registration code
- 0: failure

3.3 Image and order processing functions

3.3.1 I2E_StartOrder (i2eObject, statistics)

Purpose: Initialization of the image processing.

Returns: An integer return code.

Remarks: Call this function as the first function for the processing of a set of images. A set of images can contain one or more images, up to a maximum of 2000 images.

Parameters:

Name	Data type	Description	Remarks
i2eObject	i2eObject	i2e instance reference	created by I2E_Create
statistics	I2E_Statistics*	statistics structure reference	a)

- a) The I2E_Statistics structure must be properly initialized before calling the function:
- sourcetype: see 2.1, I2E_SourceType
 - datatype: see 2.3, I2E_DataType
 - imagecount: set to 0 (zero)

Return codes:

- 0: success
- 1: failure (initialisation error or order already started)
- 9: authorisation failure

3.3.2 I2E_AddImage (i2eObject, image, imageindex)

Purpose: Adds an image for the analysis.

Returns: An integer return code.

Remarks: Reiterate this function for each image within a set of images. A set of images can contain one or more images, up to a maximum of 2000 images.

Parameters:

Name	Data type	Description	Remarks
i2eObject	i2eObject	i2e instance reference	created by I2E_Create
image	I2E_Image*	image descriptor reference	a) see 2.6
imageindex	integer* [out]	image index of the analysis results	b)

a) Source type and data type must match the ones of the statistics in I2E_StartOrder.

b) The value returned must be passed to I2E_ProcessImage and optional to I2E_SetCorrection, I2E_GetCorrection, I2E_GetImageInfo and I2E_GetImageInfoEx for the same image.

Return codes:

- 0: success
- +1: image is too small (smaller than 200x250 pixels, warning)
- 1: image is too small (smaller than 40x50 pixels, error)
- 2: source type not consistent with order (not matching with statistics of I2E_StartOrder)
- 3: data type not consistent with order (not matching with statistics of I2E_StartOrder)
- 4: memory allocation error
- 5: not initialised
- 6: order is too large (more than 2000 images)
- 7: invalid bgr factors (applied only for data type I2E_UShort)
- 9: authorisation failure

3.3.3 I2E_EndOrder (i2eObject, statistics)

Purpose: Analysis of a set of image. A set of images can contain one or more images, up to a maximum of 2000 images.

Returns: An integer return code.

Remarks: Call this function after the last call to I2E_AddImage.

Parameters:

Name	Data type	Description	Remarks
i2eObject	i2eObject	i2e instance reference	created by I2E_Create
statistics	I2E_Statistics*	statistics structure reference	same as in I2E_StartOrder

Return codes:

- 0: success
- +1: no images analyzed
- 1: failure (initialization error or order not started)
- 2: inconsistent statistics data (non-matching source type or data type or image count)

3.3.4 I2E_ProcessImage (i2eObject, imageindex, source, destination)

Purpose: Processing of an image based on the image and order analysis.

Returns: An integer return code.

Remarks: Reiterate this function for each image within a set of images. A set of images can contain one or more images, up to a maximum of 2000 images.

Parameters:

Name	Data type	Description	Remarks
i2eObject	i2eObject	i2e instance reference	created by I2E_Create
imageindex	integer	index of the analysis results	returned by I2E_AddImage
source	I2E_Image*	source image descriptor reference	see 2.6
destination	I2E_Image*	destination image descriptor reference	a)

a) The data type must be set to I2E_UChar. The image dimensions (width and height) of destination must be equal or smaller than those of source. Automatic downsampling is performed if the dimensions of destination are smaller than those of source. If source and destination contain the same reference, then an inplace processing is performed.

Return codes:

- 0: success
- +1: failure processing red eye reduction (warning)
- +2: error loading an external memory color lookup table, standard processing applied (warning)
- 1: wrong data type of destination image (not I2E_UChar)
- 2: inconsistent image dimensions
- 3: image index out of range
- 4: memory allocation error
- 5: order not analysed (no I2E_EndOrder executed)
- 6: image data buffer checksum error (for internal use only)
- 9: authorisation failure

3.3.5 I2E_ProcessImageCB (i2eObject, imageindex, source, destination, callback)

Purpose: Same as I2E_ProcessImage, but with a callback function indicating the processing progress.

Returns: An integer return code.

Remarks: Reiterate this function for each image within a set of images. A set of images can contain one or more images, up to a maximum of 2000 images.

Parameters:

Name	Data type	Description	Remarks
i2eObject	i2eObject	i2e instance reference	created by I2E_Create
imageindex	integer	index of the analysis results	returned by I2E_AddImage
source	I2E_Image*	source image descriptor reference	see 2.6
destination	I2E_Image*	destination image descriptor reference	a)
callback	I2E_PPCallback*	callback function reference	progress range 0 ... 100

a) The data type must be set to I2E_UChar. The image dimensions (width and height) of destination must be equal or smaller than those of source. Automatic downsampling is performed if the dimensions of destination are smaller than those of source. If source and destination contain the same reference, then an inplace processing is performed.

Return codes:

- 0: success
- +1: failure processing red eye reduction (warning)
- +2: error loading an external memory color lookup table, standard processing applied (warning)
- 1: wrong data type of destination image (not I2E_UChar)
- 2: inconsistent image dimensions
- 3: image index out of range
- 4: memory allocation error
- 5: order not analysed (no I2E_EndOrder executed)
- 6: image data buffer checksum error (for internal use only)
- 9: authorisation failure

3.3.6 I2E_Execute (i2eObject, statistics, source, destination, imageinfo)

Purpose: Single image processing. Performs I2E_StartOrder, I2E_AddImage, I2E_EndOrder, I2E_ProcessImage, I2E_GetImageInfo.

Returns: An integer return code.

Parameters:

Name	Data type	Description	Remarks
i2eObject	i2eObject	i2e instance reference	created by I2E_Create
statistics	I2E_Statistics*	statistics structure reference	a)
source	I2E_Image*	source image descriptor reference	a) see 2.6
destination	I2E_Image*	destination image descriptor reference	b)
imageinfo	I2E_ImageInfo*	image info structure reference	

a) Source type and data type must be equal in statistics and source.

b) The data type must be set to I2E_UChar. The image dimensions (width and height) of destination must be equal or smaller than those of source. Automatic downsampling is performed if the dimensions of destination are smaller than those of source. If source and destination contain the same reference, then an inplace processing is performed.

Return codes: any of I2E_StartOrder, I2E_AddImage, I2E_EndOrder, I2E_ProcessImage.

3.3.7 I2E_ExecuteCB (i2eObject, statistics, source, destination, imageinfo, callback)

Purpose: Same as I2E_Execute, but with a callback function indicating the processing progress.

Returns: An integer return code.

Parameters:

Name	Data type	Description	Remarks
i2eObject	i2eObject	i2e instance reference	created by I2E_Create
statistics	I2E_Statistics*	statistics structure reference	a)
source	I2E_Image*	source image descriptor reference	a) see 2.6
destination	I2E_Image*	destination image descriptor reference	b)
imageinfo	I2E_ImageInfo*	image info structure reference	
callback	I2E_PPCallback*	callback function reference	progress range 0 ... 100

a) Source type and data type must be equal in statistics and source.

b) The data type must be set to I2E_UChar. The image dimensions (width and height) of destination must be equal or smaller than those of source. Automatic downsampling is performed if the dimensions of destination are smaller than those of source. If source and destination contain the same reference, then an inplace processing is performed.

Return codes: any of I2E_StartOrder, I2E_AddImage, I2E_EndOrder, I2E_ProcessImage.

3.3.8 I2E_UMSharpen (i2eObject, source, destination, radius, factor, threshold, limit)

Purpose: Image sharpening by an unsharp masking technique. This global sharpening function is intended for compensation of scanner and printer specific blur.

Returns: An integer return code.

Parameters:

Name	Data type	Description	Remarks
i2eObject	i2eObject	i2e instance reference	created by I2E_Create
source	I2E_Image*	source image descriptor reference	a) see 2.6
destination	I2E_Image*	destination image descriptor reference	a) see 2.6
radius	integer	pixel radius for unsharp masking	b)
factor	factor	strength of sharpening	c)
threshold	float	used to reduce noise emphasis	d)
limit	float	used to reduce edge overshooting	e)

a) Both, source and destination must be either gray scale or color images, and must have the same width and height. If source and destination contain the same reference, then an inplace processing is performed.

b) Values between 1 and 2 pixels are usually appropriate. If set to zero, then an appropriate radius is calculated internally.

- c) A value near 1.0 should give acceptable results, but this factor has to be adapted to the characteristics of the target scanning and hardcopy devices.
- d) Differences between a pixel and its surrounding below this limit are not emphasized. This parameter is relative to maxval of destination, and thus should be chosen in the range of 0.0 (no thresholding) to 1.0 (no sharpening at all). A typical value is 0.02.
- e) Differences between a pixel and its surrounding exceeding this limit are clipped to this limit. This parameter is relative to the maxval of destination, and thus should be chosen in the range 0.0 (no sharpening at all) to 1.0 (no limitaion). A typical value is 0.2.

Return codes:

- 0: success
- 1: inconsistent color order or image dimensions between source and destination
- 2: memory allocation error

3.3.9 I2E_NoiseRemoval (i2eObject, source, destination, strength, precision)

Purpose: Edge preserving anisotropic noise reduction

Returns: An integer return code.

Parameters:

Name	Data type	Description	Remarks
i2eObject	i2eObject	i2e instance reference	created by I2E_Create
source	I2E_Image*	source image descriptor reference	a) see 2.6
destination	I2E_Image*	destination image descriptor reference	a) see 2.6
strength	float	strength of the noise reduction	b) range 0.0-1.0, typical 0.6
precision	float	precision of the noise reduction	b) range 0.0-5.0, typical 2.0

a) Both, source and destination can be either gray scale or color images. The size of destination (width and height) must be equal or smaller to source. If source and destination contain the same reference, then an inplace processing is performed.

b) Pass 0.0 to apply the default parameter values.

3.4 Parameter and process control functions

3.4.1 I2E_SetProcessing (i2eObject, ABE, ACE, SHE, MCE, LSE, LNR, RER)

Purpose: Selection of the desired process stages.

Returns: Nothing.

Parameters:

Name	Data type	Description	Remarks
i2eObject	i2eObject	i2e instance reference	created by I2E_Create
ABE	integer	apply ABE	0 = disabled, 1 = enabled
ACE	integer	apply ACE	0 = disabled, 1 = enabled
SHE	integer	apply SHE	0 = disabled, 1 = enabled a)
MCE	integer	apply MCE	0 = disabled, 1 = enabled
LSE	integer	apply LSE	0 = disabled, 1 = enabled
LNR	integer	apply LNR	0 = disabled, 1 = enabled b)
RER	integer	apply RER	0 = disabled, 1 = enabled

a) SHE enabled requires ABE enabled.

b) Applied only to underexposed 16-bit images.

3.4.2 I2E_GetProcessing (i2eObject, ABE, ACE, SHE, MCE, LSE, LNR, RER)

Purpose: Get the state of the process stages.

Returns: Nothing.

Parameters:

Name	Data type	Description	Remarks
i2eObject	i2eObject	i2e instance reference	created by I2E_Create
ABE	integer* [out]	apply ABE	0 = disabled, 1 = enabled
ACE	integer* [out]	apply ACE	0 = disabled, 1 = enabled

SHE	integer* [out]	apply SHE	0 = disabled, 1 = enabled
MCE	integer* [out]	apply MCE	0 = disabled, 1 = enabled
LSE	integer* [out]	apply LSE	0 = disabled, 1 = enabled
LNR	integer* [out]	apply LNR	0 = disabled, 1 = enabled
RER	integer* [out]	apply RER	0 = disabled, 1 = enabled

3.4.3 I2E_SetParameters (i2eObject, parameters)

Purpose: Set the processing parameters.

Returns: Nothing.

Parameters:

Name	Data type	Description	Remarks
i2eObject	i2eObject	i2e instance reference	created by I2E_Create
parameters	I2E_Parameters*	parameter structure reference	see 2.7

3.4.4 I2E_GetParameters (i2eObject, parameters)

Purpose: Get the current processing parameters.

Returns: Nothing.

Parameters:

Name	Data type	Description	Remarks
i2eObject	i2eObject	i2e instance reference	created by I2E_Create
parameters	I2E_Parameters*	parameter structure reference	see 2.7

3.4.5 I2E_ResetParameters (i2eObject)

Purpose: Set the processing parameters to the default values (see 2.7).

Returns: Nothing.

Parameters:

Name	Data type	Description	Remarks
i2eObject	i2eObject	i2e instance reference	created by I2E_Create

3.4.6 I2E_SetCorrections (i2eObject, imageindex, brightness, blue, green, red, hlcnt, shcnt)

Purpose: Set the corrections for brightness, color, highlight and shadow contrast for a particular image.

Returns: An integer return code.

Parameters:

Name	Data type	Description	Remarks
i2eObject	i2eObject	i2e instance reference	created by I2E_Create
imageindex	integer	index of the analysis results	see 3.3.2
brightness	float	brightness correction	range -1 to +1
blue	float	blue correction	range -1 to +1
green	float	green correction	range -1 to +1
red	float	red correction	range -1 to +1
hlcnt	float	highlight contrast correction	range -1 to +1
shcnt	float	shadow contrast correction	range -1 to +1

Return codes:

- 0: success
- 1: image index out of range
- 2: order not analysed

3.4.7 I2E_GetCorrections (i2eObject, imageindex, brightness, blue, green, red, hlcnt, shcnt)

Purpose: Get the stored corrections for brightness, color, highlight and shadow contrast for a particular image.

Returns: An integer return code.

Parameters:

Name	Data type	Description	Remarks
------	-----------	-------------	---------

i2eObject	i2eObject	i2e instance reference	created by I2E_Create
imageindex	integer	index of the analysis results	see 3.3.2
brightness	float* [out]	brightness correction	range -1 to +1
blue	float* [out]	blue correction	range -1 to +1
green	float* [out]	green correction	range -1 to +1
red	float* [out]	red correction	range -1 to +1
hlcnt	float* [out]	highlight contrast correction	range -1 to +1
shcnt	float* [out]	shadow contrast correction	range -1 to +1

Return codes:

- 0: success
- 1: image index out of range
- 2: order not analyzed

3.4.8 I2E_ApplyCorrections (i2eObject, image, shcorr, hlcorr, bluegamma, greengamma, redgamma, sat)

Purpose: Apply corrections to an image.

Returns: An integer return code.

Parameters:

Name	Data type	Description	Remarks
i2eObject	i2eObject	i2e instance reference	created by I2E_Create
image	I2E_Image*	image descriptor	see 2.5
shcorr	float	shadow density correction	range >0 (1 = no correction)
hlcorr	float	highlight density correction	range >0 (1 = no correction)
bluegamma	float	blue gamma	range >0 (1 = no correction)
greengamma	float	green gamma	range >0 (1 = no correction)
redgamma	float	red gamma	range >0 (1 = no correction)
sat	float	saturation	range -1 (BW) to +1 (max.)

Return codes:

- 0: success
- 1: image has color order I2E_Gray (is not a color image)
- 2: one or more parameter(s) out of range

3.4.9 I2E_SetJ12Parameters (i2eObject, parameter)

Purpose: Set the processing parameters for 12-16 bit scanner images.

Returns: Nothing.

Parameters:

Name	Data type	Description	Remarks
i2eObject	i2eObject	i2e instance reference	created by I2E_Create
parameter	I2E_J12Parameters*	parameter structure reference	see 2.9

3.4.10 I2E_GetJ12Parameters (i2eObject, parameter)

Purpose: Get the stored processing parameters for 12-16 bit scanner images.

Returns: Nothing.

Parameters:

Name	Data type	Description	Remarks
i2eObject	i2eObject	i2e instance reference	created by I2E_Create
parameter	I2E_J12Parameters*	parameter structure reference	see 2.9

3.4.11 I2E_ResetJ12Parameters (i2eObject)

Purpose: Set the processing parameters for 12-16 bit scanner images to the default values (see 2.9).

Returns: Nothing.

Parameters:

Name	Data type	Description	Remarks
------	-----------	-------------	---------

i2eObject	i2eObject	i2e instance reference	created by I2E_Create
-----------	-----------	------------------------	-----------------------

3.4.12 I2E_SetExtParam (i2eObject, stage, index, par)

Purpose: Set an extended parameter.
Returns: An integer return code.
Remarks: This function is intended for fine-tuning the system for specific applications, and requires the support of the system supplier.

Parameters:

Name	Data type	Description	Remarks
i2eObject	i2eObject	i2e instance reference	created by I2E_Create
stage	integer	parameter stage	range 0 to 13
index	integer	parameter index	range 0 to 49
par	float	parameter value	

Return codes:

0: success
-1: stage or index out of range

3.4.13 I2E_SetExtParamFromFile (i2eObject, filepath)

Purpose: Set extended parameters from an ini-file.
Returns: An integer return code.
Remarks: This function is intended for fine-tuning the system for specific applications, and requires the support of the system supplier.

Parameters:

Name	Data type	Description	Remarks
i2eObject	i2eObject	i2e instance reference	created by I2E_Create
filename	string	full ini-file path name	a)

a) Contact the system supplier for information about the file format.

Return codes:

0: success
-1: file not found
-2: one or more parameters are wrong
-3: memory allocation error
-4: memory color lookup table file not found

3.4.14 I2E_SetRERmode (i2eObject, mode)

Purpose: Set the mode for red eye reduction.
Returns: Nothing.
Remarks: Applicable only if red eye reduction is authorized and activated.

Parameters:

Name	Data type	Description	Remarks
i2eObject	i2eObject	i2e instance reference	created by I2E_Create
mode	integer	processing mode	a)

a) Modes:

+1: image data buffer contains the top row first (default)
0: skip red eye reduction
-1: image data buffer contains the bottom row first
+10: process red eye reduction as the first stage in the image enhancement pipeline (default)
-10: process red eye reduction as the last stage in the image enhancement pipeline

3.4.15 I2E_GetRERmode (i2eObject)

Purpose: Get the current settings for red eye reduction.
Returns: +1: image data buffer contains the top row first
0: skip red eye reduction

-1: image data buffer contains the bottom row first

Remarks: Applicable only if red eye reduction is authorized and activated.

Parameters:

Name	Data type	Description	Remarks
i2eObject	i2eObject	i2e instance reference	created by I2E_Create

3.4.16 I2E_SetColorTable (i2eObject, filename)

Purpose: Load an external customized memory color lookup table.

Returns: An integer return code.

Remarks: This function is intended for fine-tuning the system for specific applications, and requires the support of the system supplier.

Parameters:

Name	Data type	Description	Remarks
i2eObject	i2eObject	i2e instance reference	created by I2E_Create
filename	string	full lookup table file path name	

Return codes:

0: success

-1: failure, the default memory color lookup table is restored

3.4.17 I2E_ResetColorTable (i2eObject)

Purpose: Set the default memory color lookup table.

Returns: Always 0 (success).

Parameters:

Name	Data type	Description	Remarks
i2eObject	i2eObject	i2e instance reference	created by I2E_Create

3.5 Information retrieval functions

3.5.1 I2E_GetImageInfo (i2eObject, imageindex, imageinfo)

Purpose: Get the analysis information of an image.

Returns: An integer return code.

Parameters:

Name	Data type	Description	Remarks
i2eObject	i2eObject	i2e instance reference	created by I2E_Create
imageindex	integer	index of the analysis results	see 3.3.2
imageinfo	I2E_ImageInfo*	image info structure reference	see 2.10

Return codes:

0: success

-1: image index out of range

-2: library not initialised or order not started

3.5.2 I2E_GetImageInfoEx (i2eObject, imageindex, imageinfo)

Purpose: Get the extended analysis information about a 12-16bit scanner image.

Returns: An integer return code.

Parameters:

Name	Data type	Description	Remarks
i2eObject	i2eObject	i2e instance reference	created by I2E_Create
imageindex	integer	index of the analysis results	see 3.3.2
imageinfo	I2E_ImageInfoEx*	image info structure reference	see 2.11

Return codes:

0: success

- 1: image index out of range
- 2: library not initialised or order not started
- 3: the size value is not equal to the size if the I2E_ImagelInfoEx structure

3.5.3 I2E_GetVersion (i2eObject, version)

Purpose: Get the version of the library.

Returns: Nothing.

Parameters:

Name	Data type	Description	Remarks
i2eObject	i2eObject	i2e instance reference	created by I2E_Create
version	string [out]	library version	a)

a) Version format: vv.ss.rr, vv = version, ss = sub-version, rr = revision. The string buffer must have a minimum length of 9 (recommended 256)

3.5.4 I2E_GetLastError (i2eObject, errortext)

Purpose: Get the description of the last occurred error.

Returns: Nothing.

Parameters:

Name	Data type	Description	Remarks
i2eObject	i2eObject	i2e instance reference	created by I2E_Create
errortext	string [out]	error text	a)

a) The string buffer must have a minimum length of 256.

3.6 Miscellaneous functions

3.6.1 I2E_CheckDongle (i2eObject, applicationID)

Purpose: Check if a dongle (hardware key) for a specific application identifier is attached.

Returns: 1 if the dongle is found, else 0.

Parameters:

Name	Data type	Description	Remarks
i2eObject	i2eObject	i2e instance reference	created by I2E_Create
applicationID	string		

3.6.2 I2E_FastMode (i2eObject)

Purpose: Check the authorized throughput limit.

Returns: 1 if the authorized throughput limit is not yet exceeded, otherwise 0. In the latter case processing continues, but is performed in slow mode.

Parameters:

Name	Data type	Description	Remarks
i2eObject	i2eObject	i2e instance reference	created by I2E_Create

3.6.3 I2E_ThroughputRatio (i2eObject)

Purpose: Throughput monitoring.

Returns: A float value representing the ratio of consumed to authorized throughput.

Parameters:

Name	Data type	Description	Remarks
i2eObject	i2eObject	i2e instance reference	created by I2E_Create

3.6.4 I2E_SetWatermark (i2eObject, image, watermark, posx, posy, mode)

Purpose: Add a watermark to an image.

Returns: An integer return code.
 Remarks: In demo mode the library adds automatically the default watermarks to the image. However, customized watermarks might be needed for customer specific software protection systems.

To add watermarks to images of different dimensions, proceed as follows:

- a) Create three B&W bitmaps with the following appr. dimensions:
 - small: 48x32
 - medium: 128x96
 - large: 600x300.
- b) Create three I2E_Image objects of the three bitmaps, e.g. using the image creation functions of the I2E_C_ImageIO library (I2E_CreateImage, I2E_LoadImage).
- c) Pass to the function one of the three watermark images depending on the size (larger dimension of width/height) of the source image:
 - size < 640 pixels: small watermark image
 - size < 1600 pixels: medium watermark image
 - size >= 1600 pixels: large watermark image

Parameters:

Name	Data type	Description	Remarks
i2eObject	i2eObject	i2e instance reference	created by I2E_Create
image	I2E_Image*	source image descriptor reference	see 2.6
watermark	I2E_Image*	watermark image descriptor reference	a)
posx	integer	x position within source image	
posy	integer	y position within source image	
mode	integer	display mode	range 1 to 5, b)

- a) Set sourcetype to I2E_Other, colororder to I2E_Gray and datatype to I2E_UChar.
- b) Defines how the watermark image overlays the source image. In most of the cases mode 1 shows the best results.

Return codes:

- 0: success
- 1: inconsistent watermark image
- 2: memory allocation error
- 3: mode out of range

3.6.5 I2E_SetIniPath (i2eObject, path)

Purpose: Set a non-default path name for the internally maintained system ini-file.
 Remarks: As a default i2e writes system data to the Windows directory. This function allows to change the location of this file, e.g. due to Windows directory access rights problems. For type A authorization this function is obsolete.

Returns: Nothing.

Parameters:

Name	Data type	Description	Remarks
i2eObject	i2eObject	i2e instance reference	created by I2E_Create
path	string	full path name	

3.7 Object detection and manipulation functions

3.7.1 I2E_DetectFaces (i2eObject, image, forceAllOrientation)

Purpose: Find faces in an image.
 Remarks: This function is called automatically within the image processing pipeline. Use this function only if you want to detect faces outside the image enhancement process.

Returns: A reference to an I2E_Objects structure if faces are found, else 0.

Parameters:

Name	Data type	Description	Remarks
------	-----------	-------------	---------

i2eObject	i2eObject	i2e instance reference	created by I2E_Create
image	I2E_Image*	source image descriptor reference	see 2.6
forceAllOrientations	bool	try to find faces in all three orientations	a)

a) The functions checks for faces first in the original orientation of the image, and then in the 90 degrees left and right rotated image. For performance purposes this value is set by default to false, forcing the functions to stop rotation as soon as valid faces have been found.

3.7.2 I2E_GetObjects (i2eObject, objectType)

Purpose: Get an I2E_Objects reference of a previously processed image.
Remarks: Call this function after a call to I2E_ProcessImage or I2E_Execute. The returned I2E_Objects reference must be disposed by a call to I2E_DisposeObjects.
Returns: A reference to an I2E_Objects object if faces are found, else 0.
Parameters:

Name	Data type	Description	Remarks
i2eObject	i2eObject	i2e instance reference	created by I2E_Create
objectType	I2E_ObjectType	type of the object	see 2.5

Code sample:

```
I2E_Objects* objects;
I2E_FaceRect* faceRects;

if ((objects = I2E_GetObjects (i2eObject, I2E_ObjectTypeFace)) != 0)
{
    faceRects = (I2E_FaceRect*) objects->ObjectRects;

    for (int i = 0, i < objects->NumberOfObjects; i++)
    {
        // do something with faceRects[i]...
    }

    I2E_DisposeObjects (i2eObject, objects);
}
```

3.7.3 I2E_DisposeObjects (i2eObject, objects)

Purpose: Dispose an I2E_Objects reference obtained by a previous call to I2E_GetObjects.
Returns: Nothing.
Parameters:

Name	Data type	Description	Remarks
i2eObject	i2eObject	i2e instance reference	created by I2E_Create
objects	I2E_Objects*	objects structure reference	see 2.13

3.7.4 I2E_I2E_SetObjectParameters (i2eObject, parameters)

Purpose: Set the object detection parameters.
Returns: An integer return code.
Parameters:

Name	Data type	Description	Remarks
i2eObject	i2eObject	i2e instance reference	created by I2E_Create
parameters	I2E_ObjectParameters*	parameters structure reference	see 2.14

Return codes:

0: success
-1: wrong structure size

3.7.5 I2E_I2E_GetObjectParameters (i2eObject, parameters)

Purpose: Get the object detection parameters.

Returns: An integer return code.

Parameters:

Name	Data type	Description	Remarks
i2eObject	i2eObject	i2e instance reference	created by I2E_Create
parameters	I2E_ObjectParameters*	parameters structure reference	see 2.14

Return codes:

0: success

-1: wrong structure size