User's Guide

Contents

1. Introduction.

1.1 Scope

This document is applicable to the PDF417 Decoding SDK.

SDK is notated as **PDF_PRO_YY**, where **YY**=32|64, and notation "32|64" means 32 bit or 64-bit version.

The Library interface is the same for Windows, Linux, and certain embedded platforms. Both static and dynamic libraries are available.

The library is designed to decode PDF417 Codes in accordance with ISO/IEC 16022 Symbology specification. Symbol quality assessment is provided in accordance with ISO/IEC 15415 standard.

The Quality Parameters option is user-selectable.

Library processes **8-bit** images only.

1.2 Normative references

ISO/IEC 15438:2006 - Symbology specification – PDF417
ISO/IEC 15415 - Symbol quality - Two-dimensional symbols

1.3 SDK package composition

Decoding SDK package contains:

- C++ Windows DLL (**PDF_PRO.DLL**) built in MSVS 2017 and designed to perform PDF417 search, recognition and decoding.
- C++ Demo program **(…/Demo_MSVS2008.exe**) and C# Demo program **(…/Sharp_PDF_EP.exe**) built in MSVS development environment (both come with source code) - to illustrate the DLL usage.
- Current User's Guide.

1.4 Features Description

The Library features are described in the Table below:

PDF417 SDK	
Features	**Features Description**
Direct	allows to decrease image capture time (if barcode orientation is known) by providing for three optional settings – "Left->Right", "Horizontal", "Vertical", "All Direction"
Speed	speed control setting with 3 optional modes: • **Fast Mode** - High speed decoding (~ 2x faster than Robust Mode) - for applications where decoding time is critical and image quality is relatively good; • **Robust Mode** - lower speed but highest accuracy level – for poor quality images; • **Smart Mode** - almost as good as **Robust** Mode in terms of decode rate, but still fast enough
Symbol Quality	Quality Parameters evaluation in accordance with ISO 15415, 15416
Multi-Mode	decodes up to 100 barcodes in one image via variable setting
BW Filter	reduces or enlarges the bar widths with respect to spaces (print gain correction)

All listed features are user-selectable.

1.5 Program session

Typical program session looks as follows:

Step 1. Connect decoder
Step 2. Create and set decoder options
Loop
    Step 3. Capture/read bitmap image
    Step 4. Process image
    Step 5. Request image and symbols info
    … // further application-specific data processing and interaction with user
End Loop
Step 6. Delete decoder options
Step 7. Disconnect decoder.

2. The Basic Interface Structures

The library includes the following structures:

struct TPDF417_OptMode	The set of decoder options
struct TPDF417_ImageInfo	Features of decoded image
struct TPDF417_Info	Features of decoded symbols
struct TPDF417_Quality	Quality Parameters of decoded symbols

2.1 Decoder options.

```
struct TPDF417_OptMode
{
  int MaxSymbolCount;    //
  int SpeedMode;         // SP_Robust, SP_Fast, …
  int CalcGrade;         //
  int DirectMode;        // DM_LeftRight, DM_Horizontal, DM_All
  int BWFilter;          // 0: none, +1 => BW+1,  -1 => BW-1
  int TimeOut;           // milliseconds, zero - not applied
  int ScanStep;          // step of scan lines
};
```

2.2 Image info.

```
struct TPDF417_ImageInfo
{ // averall image properties
    int SymbolCount; // number of well decoded PDF417 symbols within image
    int ErrorCode;   // nonzero, if no one symbols had been decoded
    char* ErrorString; // pointer to error string
};
```

2.3 Symbol info.

Each decoded symbol is described by the following structure:

```
struct TPDF417_Info
{ //result of decoding per each PDF417 symbol

  int ErrorCode;     // Result
  int rowcols[8];    // symbol corners
  bool rcFlag;       // true if a corners of symbol was found
  int RSErrorCount;  // number of Reed Solomon errors
  int ColCount;      // horizontal dimension (from 1 to 30)
  int RowCount;      // vertical dimension   (from 4 to 90)
  char* ErrorString; // pointer to error string
  int ECLevel;       // Error Correction Level (from 0 to 8)
  int pchlen;        // length of decoded byte array
```

```
  unsigned char* pch;// decoded byte array
  char* c_str;       // decoded char array (points to the same allocation in
memory)
  TPDF417_Quality Quality; // parameters of symbol quality (if possible)

};
```

2.4 Symbol Quality

```
struct TDM_Quality
{
  // 0..100                    , -1 = have not been calculated
  float symbol_contrast;            // symbol contrast(SC)  = Rmax-Rmin
  float min_reflectance;            // Rmin
  float max_reflectance;            // Rmax
  float global_threshold;           // global_threshold(GT) = (Rmax+Rmin)/2
  float min_edge_contrast;          // ECmin
  float modulation;                 // modulation (MOD)     = ECmin/SC
  float defects;                    //
  float decodability;               // V
  float width_height_proportion;
  float unused_error_correction;    // UEC

  // grades
  // 0..4,  0 = Worst , 4 = Best, -1 = have not been calculated
  float decode_grade;
  float symbol_contrast_grade;
  float min_reflectance_grade;
  float min_edge_contrast_grade;
  float modulation_grade;
  float defects_grade;
  float decodability_grade;
  float unused_error_correction_grade;

  float overall_grade;

};
```

2.5 The Constants.

```
// TPDF417_SpeedMode
const int
  SP_Fast   = 1,
  SP_Robust = 0,
  SP_Smart  = 2;

// TPDF417_DirectMode
const int
  DM_LeftRight  = 0, // Scan only horizontal barcodes with Start Pattern
                     // on left side of image
  DM_Horizontal = 1, // Scan only horizontal barcodes with Start Pattern
                     // on left or right side of image
  DM_All        = 2; // Scan barcode with any orientation
```

```
// TPDF417_CalcGradeMode
const int
  CG_No         = 0, //
  CG_Yes        = 1; // Calculate Bar code print quality
                     // See ISO/IEC 15415, ISO/IEC 15416


const int// error codes

//
##############################################################################
  PDF417_OK                    =   0,
  PDF417_Rectangle_Error       =  -1, // no Start and Stop Patterns were found
  PDF417_ColCount_Error        =  -4,
  PDF417_RowCount_Error        =  -5,
  PDF417_ECCodeword_Error      =  -6,
  PDF417_ErasuresLimit_Error   =  -7, // too many codewords with improper
clasters
  PDF417_RS_Error              =  -8, // too many errors in Reed Solomon code
  PDF417_CharEncodation_Error  =  -9, // unexpected encodation scheme
  PDF417_StartPattern_Error    = -10, // Stop pattern is not found
  PDF417_StopPattern_Error     = -11, // Stop pattern is not found
  PDF417_Indikators_Error      = -12, // incorrect data in row indicators
  PDF417_Geometry_Error        = -13; // unallowed module size

const int// error codes
  PDF417_NOMEMORY              = -99,
  PDF417_UNKNOWN               = -100,
  PDF417_DISCONNECTED          = -200;
//=======================================
```

3. The Interface Procedures and Functions - C++(Windows DLL)

3.1 Connect_PDF417_Decoder

PPDF417_Decoder Connect_PDF417_Decoder(int maxcolcount, int maxrowcount)

Description.

Function generates new instance of class encapsulating the decoder functionality.

Parameters.

Maximum values of vertical and horizontal image sizes.

Return value.

Pointer to structure TPDF417_Decoder if decoder is created, **null** otherwise

3.2 Disconnect_PDF417_Decoder

void Disconnect_PDF417_Decoder(PPDF417_Decoder& pdecoder)

Description.
Procedure destroys decoder class and frees memory.
Parameters.
pdecoder - the variable containing the pointer on structure TPDF417_Decoder.

3.3 Create_PDF417_Options

PPDF417_Options PDF417_Options_Create(PPDF417_Decoder pdecoder, TPDF417_OptMode optmode)

Description.
Function generates new instance of class encapsulating the decoder with desirable options.
Parameters.
The pointer on structure TPDF417_Decoder.

3.4 Delete_PDF417_Options

void PDF417_Options_Delete(PPDF417_Options& `poptions`)

Description.
Procedure destroys instance of the decoder with options.
Parameters.
The variable containing the pointer on structure TPDF417_Options.

3.5 DecodePDF417_Bits

```
int DecodeBitsF              // base decoding function
               (PPDF417_Options poptions
               ,int rowcount // The number of rows in the image
               ,int colcount // The number of columns
               ,TRow* ppbits // Points to array of pointers:
                             //{pbits[0],...,pbits[rowcount-1]} to bitmap
                  lines
               );
```

Description.
The function processes an image and fills ImageInfo and array of SymbolInfo's.

Parameters.

poptions – pointer on structure TPDF417_Options
rowcount – number of image rows,
colcount – number of image columns,
ppbits – array of pointers to image rows. Every row is a byte array with pixels brightness. (We have **typedef unsigned char* TRow**;)

Return value.

<0, if no one symbol was decoded, >=0 otherwise.

3.6 GetPDF417_ImageInfo

TPDF417_ImageInfo* Get_PDF417_ImageInfo(PPDF417_Options poptions); // inspects image info after decoding

Parameters.

poptions – pointer on structure TPDF417_Options

Return value.

Pointer to Image Info.

3.7 GetPDF417_Info

TPDF417_Info* Get_PDF417_Info(PPDF417_Options poptions, int dmNum); // inspects symbol info after decoding

Parameter.

poptions – pointer on structure TPDF417_Options.
dmNum - number of decoded symbol in image.
If no symbols were decoded then dmNum = 0 extracts Info for most possible symbol location.

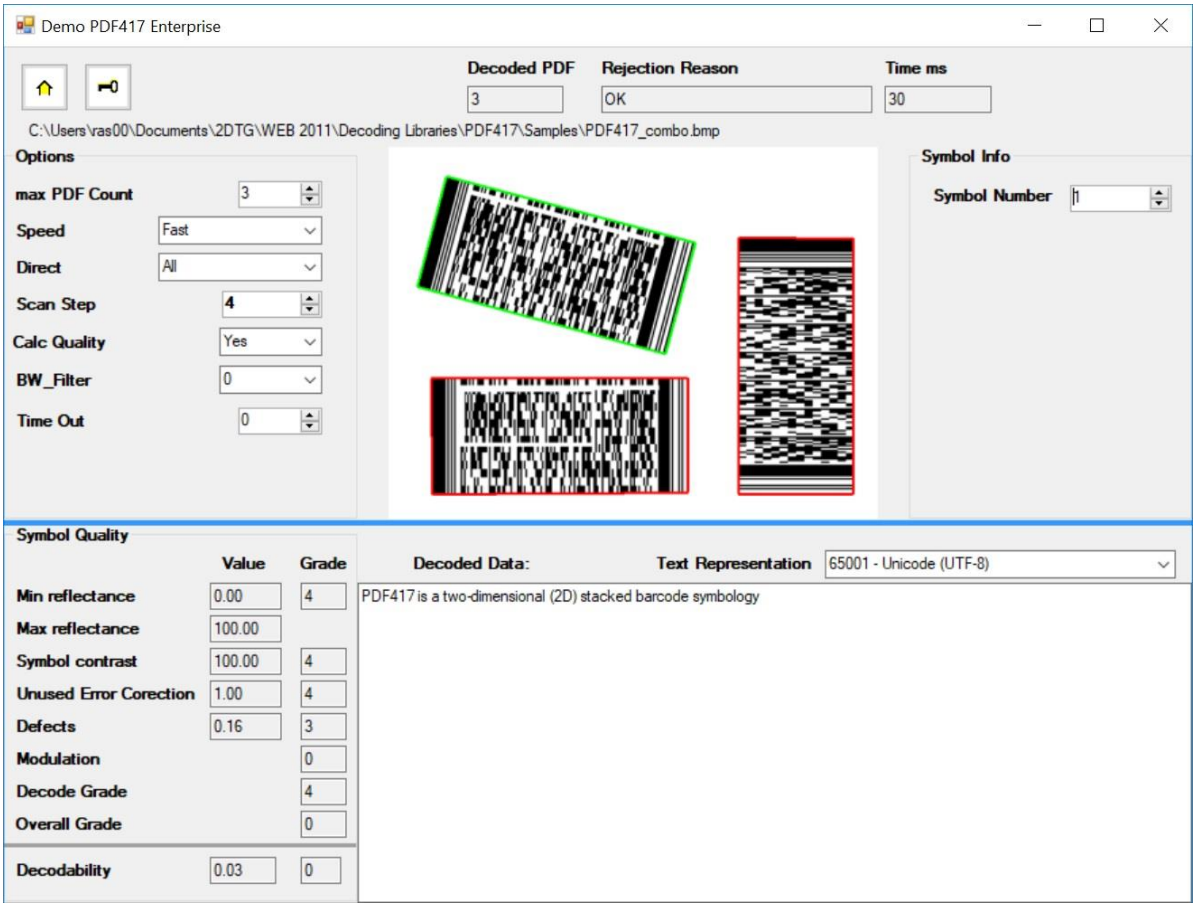
Return value.

Pointer to Symbol Info.

4. Demo application - GUI

Decoding Library comes with a Demo applications built in the C++ - **Demo_MSVS2008 -** and C# - **Sharp_PDF_EP,** built in MSVS environment.

All major Library features are illustrated by the C# application GUI:



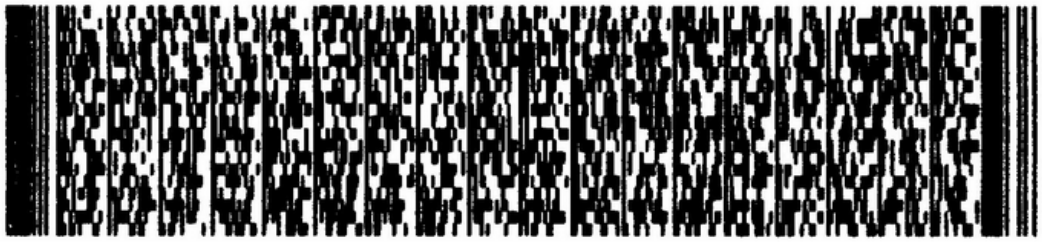
Decode Setting Options:

- **Max. PDF Count** – setting the maximum number of symbols in the image to be decoded – up to 100 (default value – 1). Program will stop when it reaches this number even if there are more symbols in the image.
- **Speed** – decode speed setting with three optional modes:
    - **Fast Mode** (default) - High speed decoding (~ 2x faster than Robust Mode) - for applications where decoding time is critical and image quality is relatively good;
    - **Robust Mode** - lower speed but highest accuracy level – for poor quality images;
    - **Smart Mode** - almost as good as **Robust** Mode in terms of decode rate, but still fast enough.

- **Direct** – setting scan direction (if barcode orientation is known) to minimize symbol search time within the image; default value – "Left->Right"
- **Scan Step** (range: 1-32) – distance between the scan-lines (in pixels); default value - 4 pixels. It might be particularly important for small barcodes - you may want to try 2 pixels in this case though it may increase the decode time.
- **Calc Quality** – turns Quality Assessment option "Yes (ON) – default / No (OFF)"
- **BW_Filter** - reduces or enlarges the bar widths with respect to spaces (before the decoding). Normally, it's particularly useful for decoding the barcodes that have been printed without print gain considerations (see barcode example below).
  Setting values:
  - **"0"** – (default) filter is nor engaged (decoding without Bar/Space change)
  - **"-1"** - Bar Width reduction
  - **"1"** - Bar Width increase.

Important:
Engaging the filter (values "-1" or "1") will change the image. So, if you planning to apply decoding process to the same image both with – and without filter, "0" setting shall be applied first.

Note: the need for **BW_Filter** can be illustrated by the following barcode having the bar widths beyond the ISO standard (it can be decoded only with **BW_Filter = -1)**:



- **Timeout** - allows to set timeout value (in ms); "0" means – no timeout is set (default).

Image Info:

- **Decoded PDF** – number of PDF417 symbols decoded within the image
- **Rejection Reason** - in some cases decoding library can return certain error codes and textual information, associated with the decoding process (this parameter characterizes the whole image if it contains more than 1 symbol). They are as follows:
  - **Rectangle_Error =** **-1** - no Start and Stop Patterns were found
  - **ColCount_Error =** **-4** -
  - **RowCount_Error =** **-5** -
  - **ECCodeword_Error =** **-6** -
  - **ErasuresLimit_Error =** **-7** - too many codewords with improper clusters
  - **RS_Error =** **-8** - too many errors in Reed Solomon code

- **CharEncodation_Error =** **-9** - unexpected encodation scheme
- **StartPattern_Error =** **-10** - Start pattern is not found
- **StopPattern_Error =** **-11** - Stop pattern is not found
- **Indikators_Error =** **-12** - incorrect data in row indicators
- **Geometry_Error =** **-13** - unallowed module size

Important: Successful decoding always returns "**OK**" and "**0**" in these fields.

- **Time** (ms) – total decode time (all PDF417 symbols within the image)

Decode Window – displays all PDF417 Codes found. Decode data on each code are available by "clicking" corresponding symbol within this Window.

Symbol Info:

- **Symbol Number** – Image Number to display (starting from "0") – applicable only for multiple symbols in the image.

Symbol Quality – results of the symbol quality assessment in accordance with ISO/IEC 15415

5. Licensing / Evaluation

Stand-alone license is locked to the computer, on which it was activated, and may not be transferred to another computer. If the computer was upgraded or rebuilt the license may still be valid if its major components had not been changed.

Important:
Licensing mechanism requires two additional files for unlock and operation (in addition to Decoding Library):

- **IP2Lib64.dll** or **IP2Lib32.dll;** and
- XML-file having syntax: **[Product Name].xml**, for example: **DM Decoding Enterprise.xml**.
- Product LOGO file (**ProdLogo_**.bmp**) is also recommended but not strictly required.

By default, 2DTG supplies all these files located in the same folder as demo-application that would call the library.

We recommend activating decoding library by starting our Demo application and following the Activation Instructions below.

If you are planning to call decoding library from your own application, please, make sure to copy those 3 files to the folder where your application is located.