

レンズ歪み補正ライブラリ

(Ver.1.0)

2018年7月

株式会社 アイディール

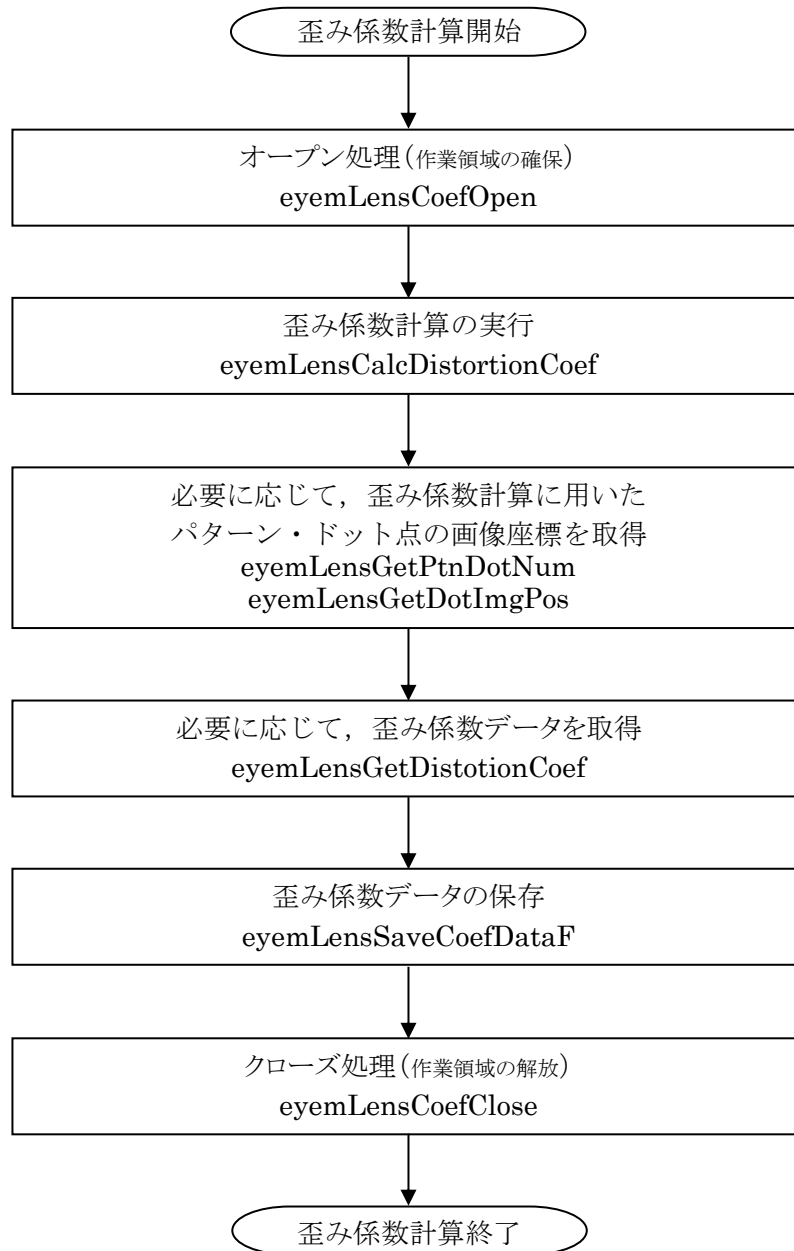
目次

1. レンズ歪み補正ライブラリの使用手順	1
(1) 歪み係数計算フェーズ	1
(2) 歪み補正フェーズ	2
2. ライブラリの詳細説明	2
eyemLensCoefOpen	3
機能 歪み係数計算のオープン処理	3
eyemLensSaveCoefDataF	4
機能 歪み係数データのファイル保存	4
eyemLensLoadCoefDataF	5
機能 歪み係数データのファイル読み込み	5
eyemLensCoefClose	6
機能 クローズ処理	6
eyemLensCalcDistortionCoef	7
機能 レンズ歪み係数の計算	7
eyemLensGetPtnDotNum	8
機能 パターン・ドットの画像座標データ個数の取得	8
eyemLensGetDotImgPos	9
機能 パターン・ドットの画像座標の取得	9
eyemLensGetDistotionCoef	10
機能 歪み係数データの取得	10
eyemLensUnDistImgSize	11
機能 歪み補正画像サイズの取得	11
eyemLensUnDistortImage	12
機能 歪み補正画像の作成	12
eyemLensUnDistortPoint	13
機能 原画像点から原画像サイズの歪み補正画像点への変換	13
eyemLensUnDistortPoint2	14
機能 原画像点から指定サイズの歪み補正画像点への変換	14
eyemLensDistortPoint	15
機能 原画像サイズの歪み補正画像点から原画像点への変換	15
eyemLensDistortPoint2	16
機能 指定サイズの歪み補正画像点から原画像点への変換	16
Appendix1. 本ライブラリで使用している定数および構造体	17
Appendix2. ドット・パターン治具情報構造体 EyemLensJig の設定	18
Appendix3. ドット・パターン画像取得の留意事項	19

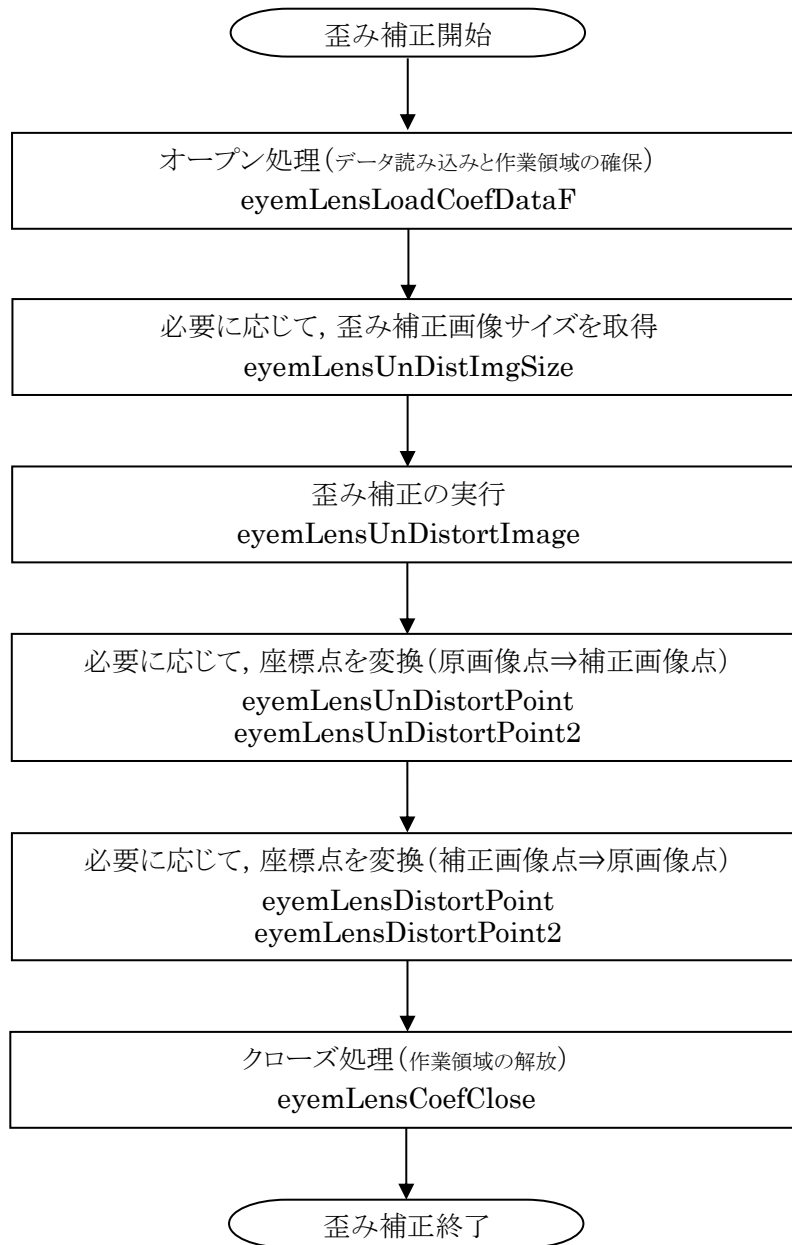
1. レンズ歪み補正ライブラリの使用手順

レンズ歪み補正ライブラリでは、歪み係数計算フェーズと歪み補正フェーズがあります。これらの使用手順は以下の流れとなります。

(1) 歪み係数計算フェーズ



(2) 歪み補正フェーズ



2. ライブラリの詳細説明

次ページ以降に各々のライブラリの説明を行います。なお、ライブラリ内に出てくる定数および構造体の具体的内容は **Appendix1** を参照してください。

eyemLensCoefOpen

機能 歪み係数計算のオープン処理

形式 #include “eyemLens.h”
int eyedLensCoefOpen(EyemLensJig *tpJig, EyemLensCam *tpCam, void **vpCoef);

解説 作業領域の確保を行い、歪み係数ディスクリプタを生成します。

引数

*tpJig	[入力] ドット・パターン治具情報. 詳細は Appendix2 を参照してください.
*tpCam	[入力] カメラ情報(画像サイズ). iWidth: カメラの画像メモリX方向サイズ(画素). iHeight: カメラの画像メモリY方向サイズ(画素).
**vpCoef	[出力] 歪み係数ディスクリプタ(データ領域の先頭アドレス). なお, *vpCoef は 必ず NULL に初期設定してください.

戻り値 終了コードです.

FUNC_OK	正常終了
FUNC_NOT_ENOUGH_MEM	ワークメモリ不足
FUNC_ILLEGAL_ARGUMENT	引数が不適當
FUNC_CANNOT_USE	ライブラリ使用不可

留意事項 本ライブラリを使用した際は、歪み係数計算終了時に eyedLensCoefClose関数 を必ず実行してください。さもないと、作業領域が解放されません。

eyemLensSaveCoefDataF

機能 歪み係数データのファイル保存

形式 #include “eyemLens.h”
int eyemLensSaveCoefDataF(const char *cpFilePath, void *vpCoef);

解説 歪み係数データを指定されたファイルに保存します。

引数 *cpFilePath [入力] 保存先のファイル・パス。
 *vpCoef [入力] 歪み係数ディスクリプタ。
 eyemLensCoefOpen関数の出力値を指定します。

戻り値 終了コードです。

FUNC_OK	正常終了
FUNC_ILLEGAL_ARGUMENT	引数が不適當
FUNC_CANNOT_CALC	計算不可(ファイル保存失敗)
FUNC_CANNOT_USE	ライブラリ使用不可

留意事項 特にありません。

eyemLensLoadCoefDataF

機能 歪み係数データのファイル読み込み

形式 #include “eyemLens.h”
int eyemLensLoadCoefDataF(const char *cpFilePath, void **vpCoef);

解説 保存してあるファイルから歪み係数データを読み込み, 歪み係数ディスクリプタの生成および作業領域の確保を行います.

引数 *cpFilePath [入力] 保存してある歪み係数データのファイル・パス.
**vpCoef [出力] 歪み係数ディスクリプタ(データ領域の先頭アドレス).
 なお, *vpCoef は 必ず NULL に初期設定してください.

戻り値 終了コードです.

FUNC_OK	正常終了
FUNC_NOT_ENOUGH_MEM	ワークメモリ不足
FUNC_ILLEGAL_ARGUMENT	引数が不適當
FUNC_CANNOT_CALC	計算不可(ファイル読み込み失敗)
FUNC_CANNOT_USE	ライブラリ使用不可

留意事項 本ライブラリを使用した際は, 歪み補正終了時に `eyemLensCoefClose`関数 を必ず実行してください. さもないと, 作業領域が解放されません.

eyemLensCoefClose

機能 クローズ処理

形式

```
#include "eyemLens.h"
void  eyedLensCoefClose( void **vpCoef );
```

解説 歪み係数計算または歪み補正で使用したワークメモリの解放を行います。

引数 ****vpCoef** [入・出力] 歪み係数ディスクリプタ
eyemLensCoefOpen関数 または eyedLensLoadCoefDataF関数の出力
値を指定します。*vpCoef は NULL に設定して戻します。

戻り値 ありません。

留意事項 特にありません。

eyemLensCalcDistortionCoef

機能 レンズ歪み係数の計算

形式 #include “eyemLens.h”
int eyedLensCalcDistortionCoef(void *vpCoef, unsigned char *ucpPtnImg);

解説 ドット・パターン画像から、レンズ歪み係数を求めます。なお、歪み係数は、ラジアル歪み(半径方向歪み)のみを求めます。また、パターンの設置方法については、**Appendix3** を参照してください。

引数 *vpCoef [入・出力] 歪み係数ディスクリプタ。
eyemLensCoefOpen関数の出力値を指定します。
*ucpPtnImg [入力] ドット・パターン画像の先頭アドレス。

戻り値 終了コードです。

0以上	(正常終了)バンドル調整反復回数
FUNC_NOT_ENOUGH_MEM	ワークメモリ不足
FUNC_ILLEGAL_ARGUMENT	引数が不適當
FUNC_CANNOT_CALC	計算不可
FUNC_CANNOT_USE	ライブラリ使用不可
FUNC_CANNOT_FIND_PLATE	プレートが見つからない
FUNC_FEW_PTN_SMPL_NUM	パターンのドット点数が足りない
FUNC_CANNOT_READ_PTN	パターン・ドット点の実座標が読めない
FUNC_FAILED_HOMOGRAPHY	ホモグラフィヤ行列計算失敗

留意事項 本関数は、eyemLensCoefOpen関数でのオープン時に限り使用可能となります。

eyemLensGetPtnDotNum

機能 パターン・ドットの画像座標データ個数の取得

形式
#include “eyemLens.h”
int eyemLensGetPtnDotNum(void *vpCoef, int *ipDotNum);

解説 有効なパターン・ドットの画像座標データの個数を取得します。

引数
*vpCoef [入力] 歪み係数ディスクリプタ.
 eyemLensCoefOpen関数の出力値を指定します.
*ipDotNum [出力] パターン・ドットの画像座標データの個数

戻り値 終了コードです.

FUNC_OK	正常終了
FUNC_ILLEGAL_ARGUMENT	引数が不適當
FUNC_CANNOT_CALC	計算不可(データがない)
FUNC_CANNOT_USE	ライブラリ使用不可

留意事項 本関数は、eyemLensCoefOpen関数でのオープン時に限り使用可能となります。

eyemLensGetDotImgPos

機能 パターン・ドットの画像座標の取得

形式 `#include "eyemLens.h"`
`int eyemLensGetDotImgPos(void *vpCoef, EyemOcsDXY taDotImgPos[]);`

解説 歪み係数計算に使用したパターン・ドットの画像座標を取得します。
予め, `eyemLensGetPtnDotNum` 関数で取得した個数分の配列 `taDotImgPos[]` を確保してください。

引 数 `*vpCoef` [入力] 歪み係数ディスクリプタ。
 `eyemLensCoefOpen`関数の出力値を指定します。
`taDotImgPos[]` [出力] パターン・ドットの画像座標データ。

戻り値 終了コードです。

<code>FUNC_OK</code>	正常終了
<code>FUNC_ILLEGAL_ARGUMENT</code>	引数が不適当
<code>FUNC_CANNOT_CALC</code>	計算不可(データがない)
<code>FUNC_CANNOT_USE</code>	ライブラリ使用不可

留意事項 本関数は, `eyemLensCoefOpen`関数でのオープン時に限り使用可能となります。

eyemLensGetDistotionCoef

機能 歪み係数データの取得

形式

```
#include "eyemLens.h"
int  eyedLensGetDistotionCoef( void *vpCoef, EyemLensCoef *tpCoef );
```

解説 eyedLensCalcDistortionCoef 関数で得られた歪み係数データを取得します。取得できるデータは、ラジアル歪み係数 k_1 , k_2 , 歪み中心座標 (c_x, c_y) および スケール s です。これらは、歪みのない理想画像座標 (x, y) および 歪みのある観測画像座標 (\tilde{x}, \tilde{y}) に対する、次の関係式の係数です：

$$\begin{pmatrix} \tilde{x} \\ \tilde{y} \end{pmatrix} = \begin{pmatrix} x \\ y \end{pmatrix} + (k_1 r^2 + k_2 r^4) \begin{pmatrix} x - c_x \\ y - c_y \end{pmatrix},$$
$$r = s \sqrt{(x - c_x)^2 + (y - c_y)^2}.$$

引数

*vpCoef [入力] 歪み係数ディスクリプタ。
eyemLensCoefOpen関数 または eyedLensLoadCoefDataF関数の出力値を指定します。

*tpCoef [出力] 歪み係数データ。
dK1: ラジアル歪み係数 k_1 .
dK2: ラジアル歪み係数 k_2
dCx: 歪み中心 x 座標 c_x .
dCy: 歪み中心 y 座標 c_y .
dScale: スケール s

戻り値 終了コードです。

FUNC_OK	正常終了
FUNC_ILLEGAL_ARGUMENT	引数が不适当
FUNC_CANNOT_USE	ライブラリ使用不可

留意事項 特にありません。

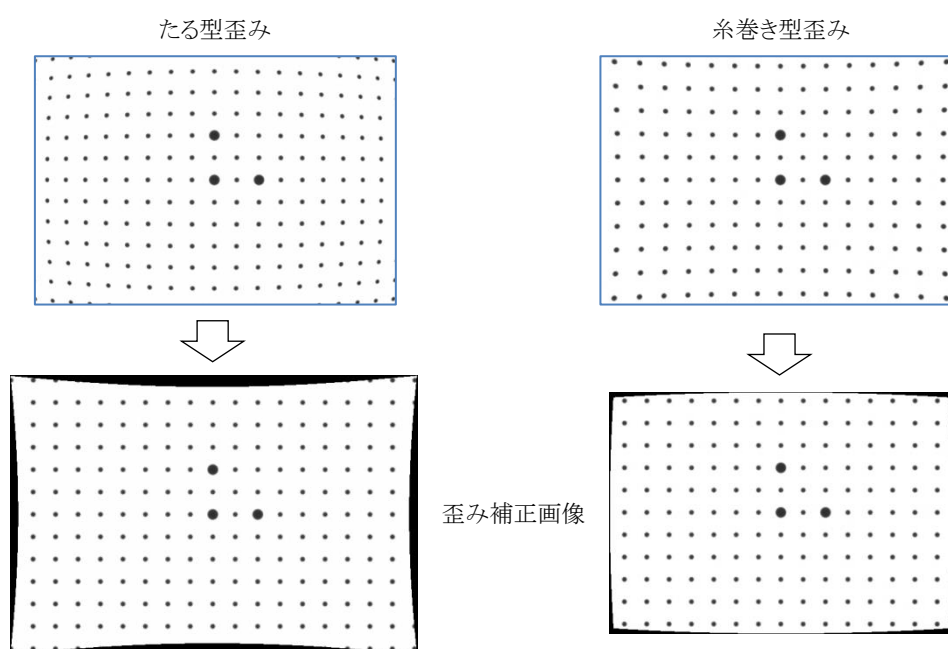
eyemLensUnDistImgSize

機能 歪み補正画像サイズの取得

形式

```
#include "eyemLens.h"
int  eyedLensUnDistImgSize( void *vpCoef, int *ipWidth, int *ipHeight );
```

解説 歪み補正画像の収まる画像サイズを取得します。
歪み補正画像は、元画像のサイズをはみ出したり(たる型歪みの場合)、縮んだり(糸巻き型歪みの場合)します。本関数では、その補正画像のサイズを取得できます。



引数

- *vpCoef [入力] 歪み係数ディスクリプタ。
eyemLensCoefOpen関数 または eyedLensLoadCoefDataF関数の出力値を指定します。
- *ipWidth [出力] 画像の幅(x方向サイズ)。
- *ipHeight [出力] 画像の高さ(y方向サイズ)。

戻り値 終了コードです。

FUNC_OK	正常終了
FUNC_ILLEGAL_ARGUMENT	引数が不適當
FUNC_CANNOT_USE	ライブラリ使用不可

留意事項 特にありません。

eyemLensUnDistortImage

機能 歪み補正画像の作成

形式

```
#include "eyemLens.h"
int  eyedLensUnDistortImage( void *vpCoef, unsigned char *ucpImgSrc, int iDstW,
                             int iDstH, unsigned char *ucpImgDst );
```

解説 歪み補正した画像を作成します。

引数

*vpCoef	[入力] 歪み係数ディスクリプタ。 eyemLensCoefOpen関数 または eyedLensLoadCoefDataF関数の出力値を指定します。
*ucpImgSrc	[入力] 原画像メモリの先頭アドレス。 eyemLensCoefOpen関数で指定した画像サイズとします。
iDstW	[入力] 歪み補正画像を格納する画像メモリの幅(x方向サイズ)。
iDstH	[入力] 歪み補正画像を格納する画像メモリの高さ(y方向サイズ)。
*ucpImgDst	[出力] 歪み補正画像を格納する画像メモリの先頭アドレス。

戻り値 終了コードです。

FUNC_OK	正常終了
FUNC_ILLEGAL_ARGUMENT	引数が不相当
FUNC_CANNOT_USE	ライブラリ使用不可

留意事項 特にありません。

eyemLensUnDistortPoint

機能 原画像点から原画像サイズの歪み補正画像点への変換

形式

```
#include "eyemLens.h"
int  eyedLensUnDistortPoint( void *vpCoef, EyemOcsDXY *tpPtSrc,
                             EyemOcsDXY *tpPtDst );
```

解説 原画像上の座標点を, 歪み補正画像上の座標点に変換します. すなわち, 座標点の歪みを除去します. ただし, 歪み補正画像は, eyedLensUndistortImage 関数において原画像と同サイズで作成された画像とします.

引数

*vpCoef	[入力] 歪み係数ディスクリプタ. eyemLensCoefOpen関数 または eyedLensLoadCoefDataF関数の出力値を指定します.
*tpPtSrc	[入力] 原画像上の座標点. dX: X座標 (画素). dY: Y座標 (画素).
*tpPtDst	[出力] 歪み補正画像上の座標点. tpPtSrc と同じでも構いません. dX: X座標 (画素). dY: Y座標 (画素).

戻り値 終了コードです.

FUNC_OK	正常終了
FUNC_ILLEGAL_ARGUMENT	引数が不適當
FUNC_CANNOT_USE	ライブラリ使用不可
FUNC_FAILED_UNDISTORT	歪み補正失敗

留意事項 特にありません.

eyemLensUnDistortPoint2

機能 原画像点から指定サイズの歪み補正画像点への変換

形式 `#include "eyemLens.h"`
`int eyemLensUnDistortPoint2(void *vpCoef, EyemOcsDXY *tpPtSrc, int iWidth,`
`int iHeight, EyemOcsDXY *tpPtDst);`

解説 原画像上の座標点を、歪み補正画像上の座標点に変換します。すなわち、座標点の歪みを除去します。ただし、歪み補正画像は、`eyemLensUndistortImage` 関数で作成された画像とします。

引数

<code>*vpCoef</code>	[入力] 歪み係数ディスクリプタ。 <code>eyemLensCoefOpen</code> 関数 または <code>eyemLensLoadCoefDataF</code> 関数の出力値を指定します。
<code>*tpPtSrc</code>	[入力] 原画像上の座標点。 dX: X座標(画素). dY: Y座標(画素).
<code>iWidth</code>	[入力] 歪み補正画像メモリの幅(x方向サイズ). <code>eyemLensUndistortImage</code> 関数での指定値です。
<code>iHeight</code>	[入力] 歪み補正画像メモリの高さ(y方向サイズ). <code>eyemLensUndistortImage</code> 関数での指定値です。
<code>*tpPtDst</code>	[出力] 歪み補正画像上の座標点。 <code>tpPtSrc</code> と同じでも構いません。 dX: X座標(画素). dY: Y座標(画素).

戻り値 終了コードです。

FUNC_OK	正常終了
FUNC_ILLEGAL_ARGUMENT	引数が不适当
FUNC_CANNOT_USE	ライブラリ使用不可
FUNC_FAILED_UNDISTORT	歪み補正失敗

留意事項 特にありません。

eyemLensDistortPoint

機能 原画像サイズの歪み補正画像点から原画像点への変換

形式

```
#include "eyemLens.h"
int  eyedLensDistortPoint( void *vpCoef, EyemOcsDXY *tpPtSrc,
                          EyemOcsDXY *tpPtDst );
```

解説 歪み補正画像上の座標点を、原画像上の座標点に変換します。すなわち、座標点に歪みを加えます。ただし、歪み補正画像は、eyemLensUndistortImage 関数において原画像と同じサイズで作成された画像とします。

引数

*vpCoef	[入力] 歪み係数ディスクリプタ。 eyemLensCoefOpen関数 または eyedLensLoadCoefDataF関数の出力値を指定します。
*tpPtSrc	[入力] 歪み補正画像上の座標点。 dX: X座標(画素)。 dY: Y座標(画素)。
*tpPtDst	[出力] 原画像上の座標点。 tpPtSrc と同じでも構いません。 dX: X座標(画素)。 dY: Y座標(画素)。

戻り値 終了コードです。

FUNC_OK	正常終了
FUNC_ILLEGAL_ARGUMENT	引数が不适当
FUNC_CANNOT_USE	ライブラリ使用不可

留意事項 特にありません。

eyemLensDistortPoint2

機能	指定サイズの歪み補正画像点から原画像点への変換										
形式	<pre>#include "eyemLens.h" int eyedLensDistortPoint2(void *vpCoef, EyemOcsDXY *tpPtSrc, int iWidth, int iHeight, EyemOcsDXY *tpPtDst);</pre>										
解説	歪み補正画像上の座標点を、原画像上の座標点に変換します。すなわち、座標点に歪みを加えます。ただし、歪み補正画像は、eyemLensUndistortImage 関数で作成された画像とします。										
引数	<table><tr><td>*vpCoef</td><td>[入力] 歪み係数ディスクリプタ。 eyemLensCoefOpen関数 または eyedLensLoadCoefDataF関数の出力値を指定します。</td></tr><tr><td>*tpPtSrc</td><td>[入力] 歪み補正画像上の座標点。 dX: X座標(画素). dY: Y座標(画素).</td></tr><tr><td>iWidth</td><td>[入力] 歪み補正画像メモリの幅(x方向サイズ). eyemLensUndistortImage関数での指定値です。</td></tr><tr><td>iHeight</td><td>[入力] 歪み補正画像メモリの高さ(y方向サイズ). eyemLensUndistortImage関数での指定値です。</td></tr><tr><td>*tpPtDst</td><td>[出力] 原画像上の座標点。 tpPtSrc と同じでも構いません。 dX: X座標(画素). dY: Y座標(画素).</td></tr></table>	*vpCoef	[入力] 歪み係数ディスクリプタ。 eyemLensCoefOpen関数 または eyedLensLoadCoefDataF関数の出力値を指定します。	*tpPtSrc	[入力] 歪み補正画像上の座標点。 dX: X座標(画素). dY: Y座標(画素).	iWidth	[入力] 歪み補正画像メモリの幅(x方向サイズ). eyemLensUndistortImage関数での指定値です。	iHeight	[入力] 歪み補正画像メモリの高さ(y方向サイズ). eyemLensUndistortImage関数での指定値です。	*tpPtDst	[出力] 原画像上の座標点。 tpPtSrc と同じでも構いません。 dX: X座標(画素). dY: Y座標(画素).
*vpCoef	[入力] 歪み係数ディスクリプタ。 eyemLensCoefOpen関数 または eyedLensLoadCoefDataF関数の出力値を指定します。										
*tpPtSrc	[入力] 歪み補正画像上の座標点。 dX: X座標(画素). dY: Y座標(画素).										
iWidth	[入力] 歪み補正画像メモリの幅(x方向サイズ). eyemLensUndistortImage関数での指定値です。										
iHeight	[入力] 歪み補正画像メモリの高さ(y方向サイズ). eyemLensUndistortImage関数での指定値です。										
*tpPtDst	[出力] 原画像上の座標点。 tpPtSrc と同じでも構いません。 dX: X座標(画素). dY: Y座標(画素).										
戻り値	終了コードです。 <table border="1"><tr><td>FUNC_OK</td><td>正常終了</td></tr><tr><td>FUNC_ILLEGAL_ARGUMENT</td><td>引数が不适当</td></tr><tr><td>FUNC_CANNOT_USE</td><td>ライブラリ使用不可</td></tr></table>	FUNC_OK	正常終了	FUNC_ILLEGAL_ARGUMENT	引数が不适当	FUNC_CANNOT_USE	ライブラリ使用不可				
FUNC_OK	正常終了										
FUNC_ILLEGAL_ARGUMENT	引数が不适当										
FUNC_CANNOT_USE	ライブラリ使用不可										
留意事項	特にありません。										

Appendix1. 本ライブラリで使用している定数および構造体

本ライブラリで使われている定数および構造体の具体的内容は以下のとおりです。

```
// エラーコード
#define FUNC_OK 0 // 正常終了
#define FUNC_NOT_ENOUGH_MEM (-1) // ワークメモリ不足
#define FUNC_ILLEGAL_ARGUMENT (-2) // 引数が不適當
#define FUNC_CANNOT_CALC (-100) // 計算不可
#define FUNC_FAILED_HOMOGRAPHY (-150) // ホモグラフィック行列計算失敗
#define FUNC_FAILED_UNDISTORT (-153) // 歪み補正失敗
#define FUNC_FEW_PTN_SMPL_NUM (-200) // パターンの参照点数が足りない
#define FUNC_CANNOT_READ_PTN (-202) // パターン参照点の実座標の認識不可
#define FUNC_CANNOT_FIND_PLATE (-220) // キャリブレーションプレートが見つからない

// ドット・パターン治具情報
// (注)ドット点群は、正方かつ等間隔であること.
typedef struct {
    int iLineDotNum; // 1ラインのドット個数
    int iDotColor; // (2値化用)ドットの色(黒:EYEM_BIN_BLACK, 白:EYEM_BIN_WHITE)
    int iDotAreaThrs; // (2値化用)ドットの面積下限値(画素)(ノイズ除去用)
} EyemLensJig;

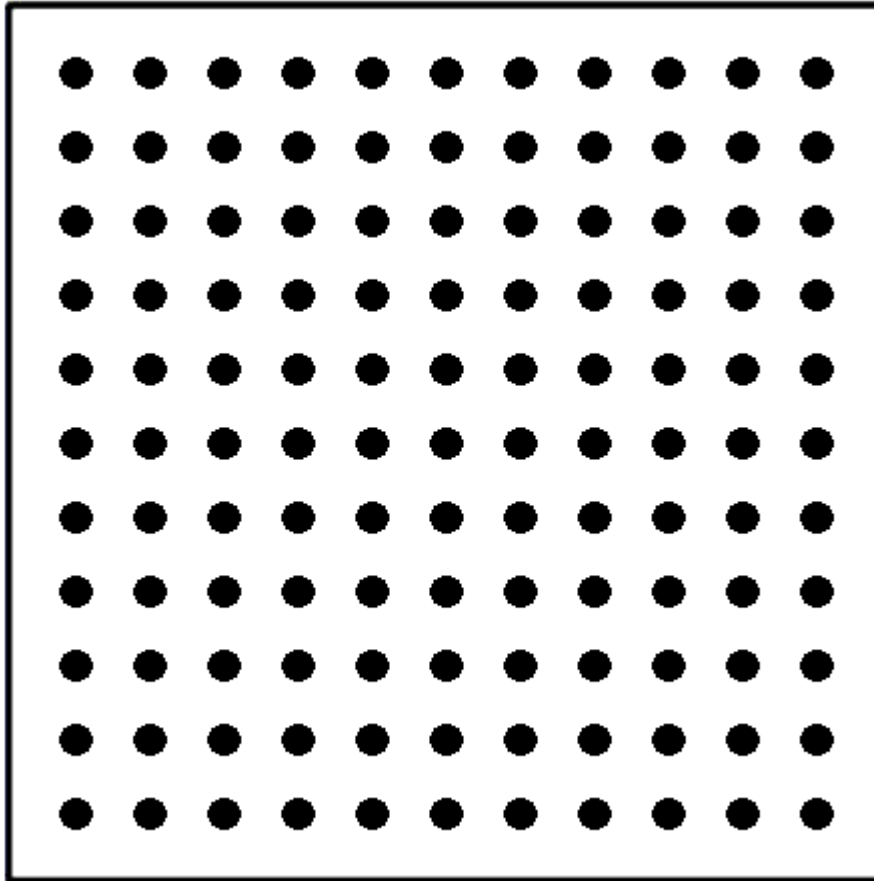
// カメラ情報
typedef struct {
    int iWidth; // カメラの画像メモリx方向サイズ
    int iHeight; // カメラの画像メモリy方向サイズ
} EyemLensCam;

// 歪み係数
typedef struct {
    double dK1; // ラジアル歪み係数 k1
    double dK2; // ラジアル歪み係数 k2
    double dCx; // 歪み中心x座標
    double dCy; // 歪み中心y座標
    double dScale; // スケール s
} EyemLensCoef;

// 座標系定義(double型)
typedef struct {
    double dX; // X座標
    double dY; // Y座標
} EyemOcsDXY;
```

Appendix2. ドット・パターン治具情報構造体 EyemLensJig の設定

本ライブラリでは、下図に示すような円形ドット・パターンで構成されたキャリブレーション治具を想定しています。すなわち、正方等間隔(行および列のドット数が同じ、かつ等間隔)に円形ドットを配置した治具です。



このキャリブレーション治具の情報を設定する EyemLensJig 構造体の具体的内容は以下のとおりです。

iLineDotNum: 1ラインのドット個数.

行および列における1ラインのドット個数です。

iDotColor: ドットの色.

2値化におけるドットの色です. EYEM_BIN_BLACK(黒) または EYEM_BIN_WHITE(白) のいずれかを指定します。

iDotAreaThrs: ドットの面積下限値(画素).

2値化におけるノイズ除去のための値です. 指定値以上の面積をもつ2値ブロップが処理対象となります。

Appendix3. ドット・パターン画像取得の留意事項

レンズ歪み係数の計算 (eyemLensCalcDistortionCoef 関数) に用いるドット・パターン画像は、以下の事項に留意して取得してください。

留意事項

- (1) 計算に用いる画像は、1枚だけです。
- (2) 画面内には、パターンがなるべく全面に映るようにしてください。パターンが映っている範囲が歪み補正の有効範囲と考えてください。
- (3) パターンは「回転」や「傾き」があっても構いませんが、なるべくレンズに正対に設置して、ドットがボケないようにしてください。

改訂履歴

Version No.	日付	内 容
1.0	2018.07.16	・新規発行

以 上