２眼ステレオ３次元計測ライブラリ

(Ver.2.3)

2018 年 8 月

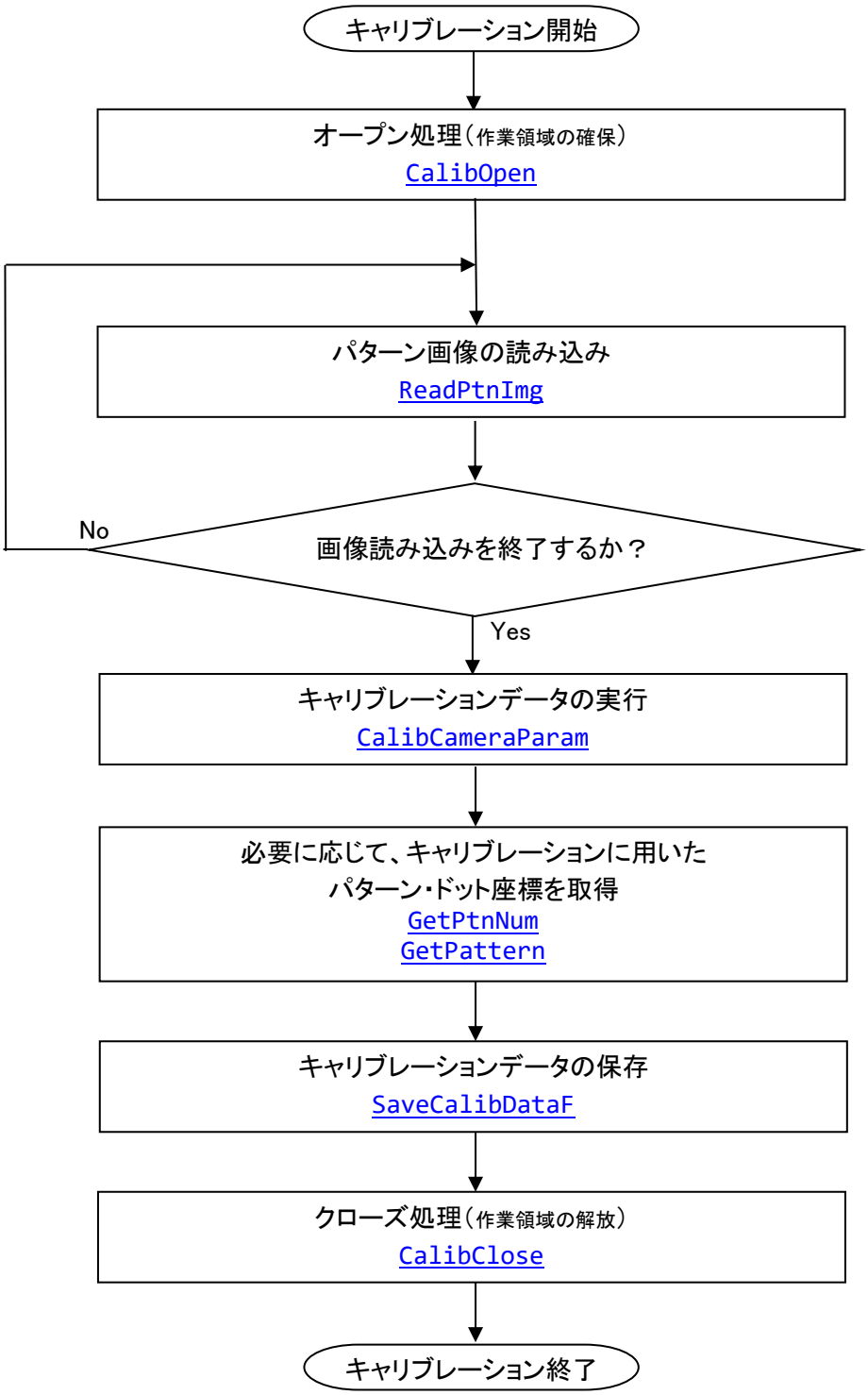
株式会社 アイディール

目次

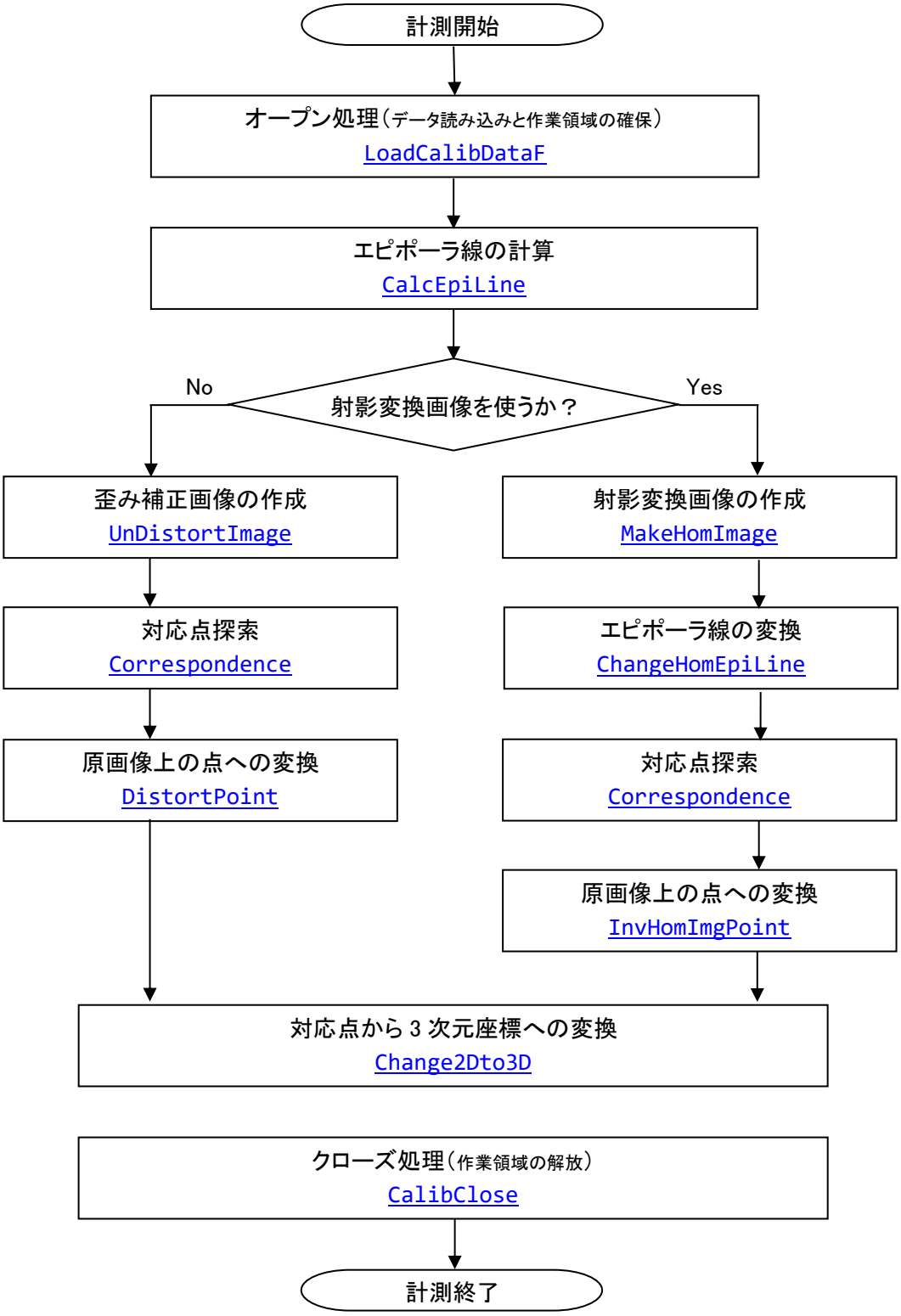
１．２眼ステレオ３次元計測ライブラリの使用手順

2眼ステレオ3次元計測ライブラリでは、キャリブレーションフェーズと計測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）キャリブレーションフェーズ



（２）計測フェーズ



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。

CalibOpen

機　能　　キャリブレーションのオープン処理

クラス　　EyemSharp2View.eyem2View

形　式

```
using EyemSharp;
using EyemSharp2View;

int CalibOpen( int iSetNum, ref Eyem2ViewJig tpJig, ref Eyem2ViewCam tpCam,
ref IntPtr vpCalib )

int CalibOpen( int iSetNum, ref Eyem2ViewJig tpJig, ref Eyem2ViewCam tpCam,
int iWldFlag = (int)EYEM_2VIEW_WLD.PTN,  ref IntPtr vpCalib )
```

解　説　　作業領域の確保を行い、キャリブレーション・ディスクリプタを生成します。

引　数

<table>
<tr><td>入力</td><td>iSetNum</td><td>キャリブレーション・パターンの左右カメラ画像セットの数
3セット以上で指定します。</td></tr>
<tr><td>入力</td><td>tpJig</td><td>キャリブレーション治具情報
詳細はAppendix1をご参照下さい。</td></tr>
<tr><td>入力</td><td>tpCam</td><td>カメラ情報
<table>
<tr><td>iCamL</td><td>左カメラ種別。以下のいずれかを指定します。(※1)</td></tr>
<tr><td>iCamR</td><td>右カメラ種別。以下のいずれかを指定します。(※1)</td></tr>
<tr><td>iWidthL</td><td>左カメラの画像メモリX方向サイズ(画素)</td></tr>
<tr><td>iWidthR</td><td>右カメラの画像メモリX方向サイズ(画素)</td></tr>
<tr><td>iHeightL</td><td>左カメラの画像メモリY方向サイズ(画素)</td></tr>
<tr><td>iHeightR</td><td>右カメラの画像メモリY方向サイズ(画素)</td></tr>
</table>
(※1)
<table>
<tr><td>値</td><td>意味</td></tr>
<tr><td>0</td><td>通常のカメラ</td></tr>
<tr><td>1</td><td>シャインプルーフ構成のカメラ</td></tr>
<tr><td>2</td><td>テレセントリックレンズ使用のカメラ</td></tr>
</table>
(注意1)テレセントリックレンズ使用カメラを含む場合は、必ずその1つを「左カメラ」としてください。
(注意2)カメラが2台ともテレセントリックレンズ使用の場合は、キャリブレーションはできますが、3次元計測はできません。</td></tr>
</table>

<table>
<tr><td>入力</td><td>iWldFlag</td><td>ワールド座標系の種別
以下のいずれかを指定します。なお、座標系の詳細はAppendix2をご参照ください。
<table>
<tr><th>定数</th><th>定義</th><th>意味</th></tr>
<tr><td>0</td><td>EYEM_2VIEW_WLD.PTN</td><td>第1パターン座標系</td></tr>
<tr><td>1</td><td>EYEM_2VIEW_WLD.CAM_LEFT</td><td>左カメラ座標系</td></tr>
<tr><td>2</td><td>EYEM_2VIEW_WLD.CAM_RIGHT</td><td>右カメラ座標系</td></tr>
<tr><td>3</td><td>EYEM_2VIEW_WLDCAM_MIDDLE</td><td>中間点座標系</td></tr>
</table>
（注意）テレセントリックレンズを使用の場合は、第1パターン座標系に限り選択可能となります。</td></tr>
<tr><td>出力</td><td>vpCalib</td><td>キャリブレーション・ディスクリプタ
なお、*vpCalibは必ず初期設定してください。
（例）
IntPtr vpCalib = IntPtr.Zero;</td></tr>
</table>

戻り値

値	定数	意味
0	FUNC_OK	正常終了
－1	FUNC_NOT_ENOUGH_MEM	ワークメモリ不足
－2	FUNC_ILLEGAL_ARGUMENT	引数が不適当
－999	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、キャリブレーション作業終了時にCalibClose関数を必ず実してください。さもないと、作業領域が解放されません。

SaveCalibDataF

機　能　　キャリブレーションデータのファイル保存

形　式

```
using EyemSharp;
using EyemSharp2View;

int SaveCalibDataF( string cpFilePath, IntPtr vpCalib )
```

解　説　　キャリブレーションデータを指定されたファイルに保存します。

引　数

入力	cpFilePath	保存先のファイル・パス
入力	vpCalib	キャリブレーション・ディスクリプタ CalibOpen関数の出力値を指定します。

戻り値

値	定数	意味
0	FUNC_OK	正常終了
－2	FUNC_ILLEGAL_ARGUMENT	引数が不適当
－100	FUNC_CANNOT_CALC	計算不能
－999	FUNC_CANNOT_USE	ライブラリ使用不可

留意事項　　特にありません。

LoadCalibDataF

機　能　　計測のオープン処理

形　式

```
using EyemSharp;
using EyemSharp2View;

int LoadCalibDataF( string cpFilePath, ref IntPtr vpCalib )
```

解　説　　保存してあるファイルからキャリブレーションデータを読み込み、キャリブレーション・ディスクリプタを生成します。また、作業領域の確保を行います。

引　数

入力	cpFilePath	保存してあるキャリブレーションデータのファイル・パス
出力	vpCalib	キャリブレーション・ディスクリプタ なお、*vpCalibは必ず初期設定してください。 （例） IntPtr vpCalib = IntPtr.Zero;

戻り値

値	定数	意味
0	FUNC_OK	正常終了
－1	FUNC_NOT_ENOUGH_MEM	ワークメモリ不足
－2	FUNC_ILLEGAL_ARGUMENT	引数が不適当
－100	FUNC_CANNOT_CALC	計算不可（ファイル読み込み失敗）
－999	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、計測終了時にCalibClose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。

CalibClose

機　能　　クローズ処理

形　式

```
using EyemSharp;
using EyemSharp2View;

void CalibClose( ref IntPtr vpCalib )
```

解　説　　キャリブレーションまたは計測で使用したワークメモリの解放を行います。

引　数

入出力	vpCalib	キャリブレーション・ディスクリプタ CalibOpen関数またはLoadCalibDataF関数の出力値を指定します。 vpCalibは以下の様に設定して戻します。 （例） vpCalib = IntPtr.Zero;

戻り値　　ありません。

留意事項　　特にありません。

ReadPtnImg

機　能　　左右カメラのパターン･ドット座標点の読み込み

クラス　　EyemSharp2View.eyem2View

形　式

```
using EyemSharp;
using EyemSharp2View;

int ReadPtnImg( IntPtr vpCalib, int iImgNo, ref Eyem2ViewImg tpImgSet, int
nAuto,ref EyemRect tpRoiL, ref EyemRect tpRoiR )
```

解　説　　左右カメラのパターン･ドット画像から、有効な座標点を読み込みます。

引　数

入出力	vpCalib	キャリブレーション･ディスクリプタ CalibOpen関数の出力値を指定します。
入力	iImgNo	対象となる左右カメラのパターン画像セットの番号 0番からの通し番号を入力します。
入力	tpImgSet	対象となる左右カメラのパターン画像セット なお、これらの画像の作成（キャリブレーションの方法）については、Appendx3をご参照ください tImgCamL：左カメラ画像メモリ tImgCamR：右カメラ画像メモリ （注）テレセントリックレンズ使用の場合は、（左カメラの）0番目パターン画像のみが使用されます。よって、1番目以降のtImgCamLにはNULLを指定ください。
入力	nAuto	-1のとき、tpRoiL、tpRoiを自動設定
入力	tpRoiL	左カメラ用･処理範囲の矩形領域（左上座標およびサイズ） iX：左上X座標（画素） iY：左上Y座標（画素） iWidth：領域のX方向サイズ（画素） iHeight：領域のY方向サイズ（画素）
入力	tpRoiR	右カメラ用･処理範囲の矩形領域（左上座標およびサイズ） iX：左上X座標（画素） iY：左上Y座標（画素） iWidth：領域のX方向サイズ（画素） iHeight：領域のY方向サイズ（画素）

戻り値

値	定数	意味
0	FUNC_OK	正常終了
－1	FUNC_NOT_ENOUGH_MEM	ワークメモリ不足
－2	FUNC_ILLEGAL_ARGUMENT	引数が不適当
－100	FUNC_CANNOT_CALC	計算不可
－200	FUNC_FEW_PTN_SMPL_NUM	パターンのドット点数が足りない
－201	FUNC_CANNOT_READ_MARK	パターン大ドット点が読めない
－202	FUNC_CANNOT_READ_PTN	パターン・ドット点の実座標が読めない
－220	FUNC_CANNOT_FIND_PLATE	キャリブレーションプレートが見つからない
－999	FUNC_CANNOT_USE	ライブラリ使用不可

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

CalibCameraParam

機　能　　カメラキャリブレーションの実行

クラス　　EyemSharp2View.eyem2View

形　式

```
using EyemSharp;
using EyemSharp2View;

int CalibCameraParam( IntPtr vpCalib, ref Eyem2ViewBndl tpBundle )
```

解　説　　左右パターン画像（3セット以上）から、それぞれのカメラのキャリブレーションを個別に行います。

引　数

入出力	vpCalib	キャリブレーション・ディスクリプタ CalibOpen関数の出力値を指定します。
出力	tpBundle	バンドル調整結果（反復回数、残差RMS） iNumL：左カメラのバンドル調整反復回数 dRmsL：左カメラの残差（再投影誤差）の重み付き平均二乗誤差の平方根（画素） iNumL：右カメラのバンドル調整反復回数 dRmsL：右カメラの残差（再投影誤差）の重み付き平均二乗誤差の平方根（画素）

戻り値

値	定数	意味
0	FUNC_OK	正常終了
－1	FUNC_NOT_ENOUGH_MEM	ワークメモリ不足
－2	FUNC_ILLEGAL_ARGUMENT	引数が不適当
－100	FUNC_CANNOT_CALC	計算不可
－150	FUNC_FAILED_HOMOGRAPHY	ホモグラフィ行列計算失敗
－151	FUNC_FAILED_CAM_PRM	カメラパラメータ計算失敗
－152	FUNC_FAILED_BUNDLE_ADJ	バンドル調整失敗
－210	FUNC_FAILED_FUNDAMENTAL	基礎行列計算失敗
－999	FUNC_CANNOT_USE	ライブラリ使用不可

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

Change2Dto3D

機　能　　左右カメラ画像の対応点から3次元座標点への変換

クラス　　EyemSharp2View.eyem2View

形　式

```
using EyemSharp;
using EyemSharp2View;

int Change2Dto3D( IntPtr vpCalib, ref EyemOcsDXY tpImgPtL, ref EyemOcsDXY
tpImgPtR, ref EyemOcsDXYZ tpPt3D )
```

解　説　　左右カメラの原画像における対応点から、その3次元座標を求めます。

引　数

入力	vpCalib	キャリブレーション・ディスクリプ CalibOpen関数またはLoadCalibDataF関数の出力値を指定します。
入力	tpImgPtL	左カメラの原画像における座標 dX：X座標（画素） dY：Y座標（画素）
入力	tpImgPtR	右カメラの原画像における座標 dX：X座標（画素） dY：Y座標（画素）
出力	tpPt3D	3次元座標 dX：X座標（mm） dY：Y座標（mm） dZ：Z座標（mm）

戻り値

値	定数	意味
0	FUNC_OK	正常終了
－2	FUNC_ILLEGAL_ARGUMENT	引数が不適当
－100	FUNC_CANNOT_CALC	計算不可
－153	FUNC_FAILED_UNDISTORT	歪み補正失敗
－999	FUNC_CANNOT_USE	ライブラリ使用不可

留意事項　（1）2台ともテレセントリックレンズ使用カメラの場合は、使用できません。（計算不可）

（2）射影変換画像（MakeHomImage 関数の出力画像）上の対応点は、必ず原画像上の座標点に変換してから使用してください。この変換は、InvHomImgPoint 関数で行います。

（3）歪み補正画像（UnDistortImage 関数の出力画像）上の対応点は、必ず原画像上の座標に変換してから使用してください。この変換は、DistortPoint 関数で行います。

Change3Dto2D

機　能　　3次元座標点から画像座標点への変換

クラス　　EyemSharp2View.eyem2View

形　式

```
using EyemSharp;
using EyemSharp2View;

int Change3Dto2D( IntPtr vpCalib, char cLR, ref EyemOcsDXYZ tpPt3D, ref
EyemOcsDXY tpPt2D )
```

解　説　　指定された3次元座標点を、指定カメラの画像面上へ投影したときの座標を求めます。

引　数

入力	vpCalib	キャリブレーション・ディスクリプタ CalibOpen関数またはLoadCalibDataF関数の出力値を指定します。
入力	cLR	投影する画像面 左カメラ画像の場合は'L'、右カメラ画像の場合は'R'を指定します。
入力	tpPt3D	3次元座標 dX：X座標（mm） dY：Y座標（mm） dZ：Z座標（mm）
出力	tpPt2D	画像座標 dX：X座標（画素） dY：Y座標（画素）

戻り値

値	定数	意味
0	FUNC_OK	正常終了
－2	FUNC_ILLEGAL_ARGUMENT	引数が不適当
－100	FUNC_CANNOT_CALC	計算不可
－999	FUNC_CANNOT_USE	ライブラリ使用不可

留意事項　　特にありません。

CalcEpiLine

機　能　　エピポーラ線の計算

クラス　　EyemSharp2View.eyem2View

形　式

```
using EyemSharp;
using EyemSharp2View;

int CalcEpiLine( IntPtr vpCalib, char cLR, ref EyemOcsDXY tpImgPt, ref
Eyem2ViewEpi tpEpiLine )
```

解　説　　左(または右)カメラ画像上の指定点に対する、右(または左)カメラ画像上のエピポーラ線を求めます(正確には、歪み補正画像上のエピポーラ線が求まります)。

引　数

入力	vpCalib	キャリブレーション・ディスクリプタ CalibOpen関数またはLoadCalibDataF関数の出力値を指定します。
入力	cLR	エピポーラ線の乗る画像 左カメラ画像の場合は'L'、右カメラ画像の場合は'R'を指定します。
入力	tpImgPt	原画像における指定点の座標 dX：X座標(mm) dY：Y座標(mm)
出力	tpEpiLine	エピポーラ線情報(直線の係数、画像内線分端点) tLine：エピポーラ線(ax + by + c = 0)の係数a、b、c。 taPt[2]：エピポーラ線と画像境界との交点(2点)の座標

戻り値

値	定数	意味
0	FUNC_OK	正常終了
−2	FUNC_ILLEGAL_ARGUMENT	引数が不適当
−100	FUNC_CANNOT_CALC	計算不可
−153	FUNC_FAILED_UNDISTORT	歪み補正失敗
−999	FUNC_CANNOT_USE	ライブラリ使用不可

留意事項　　特にありません。

ChangeHomEpiLine

機　能　　射影変換画像上のエピポーラ線への変換

クラス　　EyemSharp2View.eyem2View

形　式

```
using EyemSharp;
using EyemSharp2View;

int ChangeHomEpiLine( IntPtr vpCalib, char cLR, ref Eyem2ViewHom tpParam,
ref Eyem2ViewEpi tpLineSrc, ref Eyem2ViewEpi tpLineDst )
```

解　説　　指定カメラの原画像上（正確には歪み補正画像上）のエピポーラ線（CalcEpiLine 関数の出力値）を射影変換画像（MakeHomImage 関数の出力画像）上のエピポーラ線に変換します。

引　数

<table>
<tr><td>入力</td><td>vpCalib</td><td>キャリブレーション・ディスクリプタ
CalibOpen関数またはLoadCalibDataF関数の出力値を指定します。</td></tr>
<tr><td>入力</td><td>cLR</td><td>エピポーラ線の乗る画像
左カメラ画像の場合は'L'、右カメラ画像の場合は'R'を指定します。</td></tr>
<tr><td>入力</td><td>tpParam</td><td>画像射影変換パラメータ
必ず、MakeHomImage関数で用いた値を指定します。なお、画像射変換影についてはAppendix4をご参照下さい。
<table>
<tr><td>tCenter</td><td>射影変換画像の中央に対応する、原画像の座標点</td></tr>
<tr><td>dWorldZ</td><td>計測点3次元座標のz座標概略値</td></tr>
</table></td></tr>
<tr><td>入力</td><td>tpLineSrc</td><td>原画像上のエピポーラ線
CalcEpiLine関数の出力値を指定します。
<table>
<tr><td>tLine</td><td>エピポーラ線（ax + by + c = 0）の係数a、b、c</td></tr>
<tr><td>taPt[2]</td><td>エピポーラ線と画像境界との交点（2点）の座標</td></tr>
</table></td></tr>
<tr><td>出力</td><td>tpLineDst</td><td>射影変換画像上のエピポーラ線情報
<table>
<tr><td>tLine</td><td>エピポーラ線（ax + by + c = 0）の係数a、b、c</td></tr>
<tr><td>taPt[2]</td><td>エピポーラ線と画像境界との交点（2点）の座標</td></tr>
</table></td></tr>
</table>

戻り値

値	定数	意味
0	FUNC_OK	正常終了
－2	FUNC_ILLEGAL_ARGUMENT	引数が不適当
－100	FUNC_CANNOT_CALC	計算不可
－999	FUNC_CANNOT_USE	ライブラリ使用不可

留意事項　　特にありません。

Correspondence

機　能　　エピポーラ線上の対応点探索

クラス　　EyemSharp2View.eyem2View

形　式

```
using EyemSharp;
using EyemSharp2View;

int Correspondence( IntPtr vpCalib, int iMethod, char cLR, Bitmap tRefImg,
ref EyemRect tpTmpRect, ref EyemOcsDXY tpMesPt, Bitmap tObjImg, ref
EyemRect tpObjRect, ref Eyem2ViewEpi tpEpiLine, ref EyemOcsDXY tpCrspnd )
```

解　説　　一方のカメラ画像で指定された計測点に対する、もう一方のカメラ画像上のエピポーラ線上で、対応点探索（サーチ）を行います。

引　数

入力	vpCalib	キャリブレーション・ディスクリプタ CalibOpen関数またはLoadCalibDataF関数の出力値を指定します。
入力	iMethod	サーチ方法 EYEM_2VIEW_SEARCH_ZNCC　:ゼロ平均正規化相互相関法 EYEM_2VIEW_SEARCH_SAD　:差分絶対総和法
入力	cLR	探索の対象となる画像 左カメラ画像の場合は'L'、右カメラ画像の場合'R'を指定します。
入力	tRefImg	サーチ・テンプレート用参照画像メモリ CalibOpen関数で指定した画像サイズとします。なお、テンプレート画像は、厳密には歪み補正画像を用いますが、精度上問題がなければ原画像でも構いません。
入力	tpTmpRect	サーチ・テンプレートとする矩形領域（左上座標およびサイズ） iX：左上X座標（画素） iY：左上Y座標（画素） iWidth：領域のX方向サイズ（画素） iHeight：領域のY方向サイズ（画素）
入力	tpMesPt	テンプレート領域内の計測対象点 dX：X座標（画素） dY：Y座標（画素）

<table>
<tr><td>入力</td><td>tObjImg</td><td>探索対象画像メモリ
CalibOpen関数で指定した画像サイズとします。なお、テンプレート画像は、厳密には歪み補正画像を用いますが、精度上問題がなければ原画像でも構いません。また、探索対象画像は、射影変換画像でも構いません（射影変換画像は、歪み補正が施されています）。</td></tr>
<tr><td>入力</td><td>tpObjRect</td><td>探索範囲の矩形領域（左上座標およびサイズ）<table><tr><td>iX</td><td>左上X座標（画素）</td></tr><tr><td>iY</td><td>左上Y座標（画素）</td></tr><tr><td>iWidth</td><td>領域のX方向サイズ（画素）</td></tr><tr><td>iHeight</td><td>領域のY方向サイズ（画素）</td></tr></table></td></tr>
<tr><td>入力</td><td>tpEpiLine</td><td>探索対象画像内のエピポーラ線情報（直線の係数のみ使用）
CalcEpiLine関数またはChangeHomEpiLine関数の出力値を指定します。<table><tr><td>tLine</td><td>エピポーラ線（ax + by + c = 0）の係数a、b、c</td></tr><tr><td>taPt[2]</td><td>エピポーラ線と画像境界との交点（2点）の座標</td></tr></table></td></tr>
<tr><td>出力</td><td>tpCrspnd</td><td>探索対象画像内の対応点<table><tr><td>dX</td><td>X座標（画素）</td></tr><tr><td>dY</td><td>Y座標（画素）</td></tr></table></td></tr>
</table>

戻り値

値	定数	意味
0	FUNC_OK	正常終了
－1	FUNC_NOT_ENOUGH_MEM	ワークメモリ不足
－2	FUNC_ILLEGAL_ARGUMENT	引数が不適当
－100	FUNC_CANNOT_CALC	計算不可
－999	FUNC_CANNOT_USE	ライブラリ使用不可

留意事項　　特にありません。

GetCamHomMat

機　能　　ホモグラフィ行列の取得

クラス　　EyemSharp2View.eyem2View

形　式　　using EyemSharp;
using EyemSharp2View;

int GetCamHomMat(IntPtr vpCalib, char cLR, ref EyemCalibHom tpCalibHom)

解　説　　指定カメラのホモグラフィ行列を取得します。この行列は、0 番目パターン平面（すなわちワールド座標系xy面）と画像平面との間の射影変換行列 H です。すなわち、0 番目パターン上３次元座標を(X, Y, 0)、およびレンズ歪みなしの画像座標を(x, y)とするとき、λ を実数として、

$$\lambda \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \mathbf{H} \begin{pmatrix} X \\ Y \\ 1 \end{pmatrix}$$

となります。

引　数

入力	vpCalib	キャリブレーション・ディスクリプタ CalibOpen関数またはLoadCalibDataF関数の出力値を指定します。
入力	cLR	左右カメラ画像の選択 左カメラ画像の場合は'L'、右カメラ画像の場合は'R'を指定します。
出力	tpCalibHom	指定カメラのホモグラフィ行列 daH[3][3] ホモグラフィ行列H daInvH[3][3] 行列Hの逆行列

戻り値

値	定数	意味
0	FUNC_OK	正常終了
－2	FUNC_ILLEGAL_ARGUMENT	引数が不適当
－999	FUNC_CANNOT_USE	ライブラリ使用不可

留意事項　　特にありません。

GetCamFndMat

機　能　　基礎行列の取得

クラス　　EyemSharp2View.eyem2View

形　式

```
using EyemSharp;
using EyemSharp2View;

int GetCamFndMat( IntPtr vpCalib, ref Eyem2ViewFnd tpFnd )
```

解　説　　左右カメラ間の基礎行列を取得します。この行列は、左右の画像点を結び付ける行列Fであり、次のエピポーラ方程式を満たしています。すなわち、左カメラの歪みなし画像座標を(x_L, y_L)、および右カメラの歪みなし画像座標を(x_R, y_R)とするとき

$$(x_L \quad y_L \quad 1)\mathbf{F}\begin{pmatrix} x_R \\ y_R \\ 1 \end{pmatrix} = 0$$

となります。

引　数

入力	vpCalib	キャリブレーション・ディスクリプタ CalibOpen関数またはLoadCalibDataF関数の出力値を指定します。
出力	tpCalibHom	基礎行列

daF[3][3]	基礎行列$\mathbf{F}$
daFT[3][3]	行列Fの転置行列$\mathbf{F}^{\mathrm{T}}$

戻り値

値	定数	意味
0	FUNC_OK	正常終了
－2	FUNC_ILLEGAL_ARGUMENT	引数が不適当
－999	FUNC_CANNOT_USE	ライブラリ使用不可

留意事項　　特にありません。

GetCamIntParam

機　能　　カメラ内部パラメータの取得

クラス　　EyemSharp2View.eyem2View

形　式

```
using EyemSharp;
using EyemSharp2View;

int GetCamIntParam( IntPtr vpCalib, char cLR, ref EyemCalibInt tpInt )
```

解　説　　指定カメラの内部パラメータを取得します。内部パラメータは、x方向、y方向それぞれの「焦距離×スケール」(f_xおよびf_y)、「焦点距離×せん断係数」(f_s)、「画像中心」(u_0, v_0)および「半径方向の歪み(ラジアル歪み)係数」が得られます。これらを用いて、内部パラメータ行列 **A** は、
次の3×3行列で表されます。

$$\mathbf{A} = \begin{pmatrix} f_x & f_s & u_0 \\ 0 & f_y & v_0 \\ 0 & 0 & 1 \end{pmatrix}.$$

引　数

入力	vpCalib	キャリブレーション・ディスクリプタ CalibOpen関数またはLoadCalibDataF関数の出力値を指定します。
入力	cLR	左右カメラの選択 左カメラの場合は'L'、右カメラの場合は'R'を指定します。
出力	tpInt	カメラの内部パラメータ

daA[3][3]	内部パラメータ行列**A**
dFx、 dFy	焦点距離×スケール(f_xおよびf_y)(画素)
dFs	焦点距離×せん断係数(f_s)(画素)
dUo、 dVo	画像中心座標((u_0, v_0))(画素)
dK1、 dK2	ラジアル歪み係数
dTheta	画像面傾斜角(CCD面x軸周りの角度)(rad) (シャインプルーフ構成カメラの場合に使用可能)

戻り値

値	定数	意味
0	FUNC_OK	正常終了
－2	FUNC_ILLEGAL_ARGUMENT	引数が不適当
－999	FUNC_CANNOT_USE	ライブラリ使用不可

留意事項　　特にありません。

GetCamExtParam

機　能　　カメラ外部パラメータの取得（ワールド座標系基準）

クラス　　EyemSharp2View.eyem2View

形　式

```
using EyemSharp;
using EyemSharp2View;

int GetCamExtParam( IntPtr vpCalib, char cLR, int iPtnNo, ref EyemCalibExt
tpExt )

int GetCamExtParam( IntPtr vpCalib, char cLR, ref EyemCalibExt tpExt )
```

解　説　　指定カメラの外部パラメータを取得します。外部パラメータは、ワールド座標系の3次元座標をカメラ座標へ変換する「回転」および「平行移動」です。なお、ワールド座標系については、Appendix2 をご参照ください。

引　数

入力	vpCalib	キャリブレーション・ディスクリプタ CalibOpen関数またはLoadCalibDataF関数の出力値を指定します。
入力	cLR	左右カメラの選択 左カメラの場合は'L'、右カメラの場合は'R'を指定します。
入力	iPtnNo	パターンの番号 0番からの通し番号を入力します。画像セット番号に対応しています。
出力	tpExt	カメラ外部パラメータ

daR[3][3]	回転行列
daT[3]	平行移動ベクトル
daRdr[3]	回転ベクトル（ロドリゲスの表現）

戻り値

値	定数	意味
0	FUNC_OK	正常終了
－2	FUNC_ILLEGAL_ARGUMENT	引数が不適当
－999	FUNC_CANNOT_USE	ライブラリ使用不可

留意事項　　Ver2.0.0 より引数 iPtnNo は、未使用となります。iPtnNo を使用する場合は、GetCamExtParamPtn メソッドをご利用ください。(パターン座標系基準の時)。

GetCamExtParamPtn

機　能　　カメラ外部パラメータの取得（パターン座標系基準）

クラス　　EyemSharp2View.eyem2View

形　式

```
using EyemSharp;
using EyemSharp2View;

int GetCamExtParamPtn( IntPtr vpCalib, char cLR, int iPtnNo, ref
EyemCalibExt tpExt )
```

解　説　　指定カメラの外部パラメータを取得します。外部パラメータは，各パターンをxy平面とする右手座標系の3次元座標をカメラ座標へ変換する「回転」および「平行移動」です。

引　数

入力	vpCalib	キャリブレーション・ディスクリプタ CalibOpen関数またはLoadCalibDataF関数の出力値を指定します。
入力	cLR	左右カメラの選択 左カメラの場合は'L'、右カメラの場合は'R'を指定します。
入力	iPtnNo	パターンの番号 0番からの通し番号を入力します。画像セット番号に対応しています。
出力	tpExt	カメラ外部パラメータ daR[3][3]　回転行列 daT[3]　平行移動ベクトル daRdr[3]　回転ベクトル（ロドリゲスの表現）

戻り値

値	定数	意味
0	FUNC_OK	正常終了
－2	FUNC_ILLEGAL_ARGUMENT	引数が不適当
－999	FUNC_CANNOT_USE	ライブラリ使用不可

留意事項　　Ver2.0.0 より引数 iPtnNo は、未使用となります。iPtnNo を使用する場合は、GetCamExtParamPtn メソッドをご利用ください。（パターン座標系基準の時）。

GetPtnNum

機　能　　パターン･ドットの画像座標データ個数の取得

クラス　　EyemSharp2View.eyem2View

形　式　　using EyemSharp;
using EyemSharp2View;

int GetPtnNum(IntPtr vpCalib, char cLR, int iDataNo, ref int ipPtnNum)

解　説　　指定カメラのキャリブレーションで使用した、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

入力	vpCalib	キャリブレーション･ディスクリプタ CalibOpen関数の出力値を指定します。
入力	cLR	左右カメラの選択 左カメラの場合は'L'、右カメラの場合は'R'を指定します。
入力	iDataNo	座標データの番号 0番からの通し番号を入力します。画像セット番号に対応しています。
出力	ipPtnNum	パターン･ドットの画像座標データの個数

戻り値

値	定数	意味
0	FUNC_OK	正常終了
－2	FUNC_ILLEGAL_ARGUMENT	引数が不適当
－100	FUNC_CANNOT_CALC	計算不可（データなし）
－999	FUNC_CANNOT_USE	ライブラリ使用不可

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

GetPattern

機　能　　パターン･ドットの画像座標データの取得

クラス　　EyemSharp2View.eyem2View

形　式　　using EyemSharp;
using EyemSharp2View;

int GetPattern(IntPtr vpCalib, char cLR, int iDataNo, ref EyemOcsDXY taPtn)

解　説　　指定カメラのキャリブレーションで使用した、パターン･ドット点の画像座標データを取得します。
予め、GetPtnNum 関数で取得した個数分の配列 taPtn[] を確保してください。

引　数

入力	vpCalib	キャリブレーション･ディスクリプタ CalibOpen関数の出力値を指定します。
入力	cLR	左右カメラの選択 左カメラの場合は'L'、右カメラの場合は'R'を指定します。
入力	iDataNo	座標データの番号 0番からの通し番号を入力します。画像セット番号に対応しています。
出力	taPtn	パターン･ドット点の画像座標データ dX：X座標（画素） dY：Y座標（画素）

戻り値

値	定数	意味
0	FUNC_OK	正常終了
－2	FUNC_ILLEGAL_ARGUMENT	引数が不適当
－100	FUNC_CANNOT_CALC	計算不可（データなし）
－999	FUNC_CANNOT_USE	ライブラリ使用不可

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

MakeHomImage

機　能　　画像の射影変換

クラス　　EyemSharp2View.eyem2View

形　式

```
using EyemSharp;
using EyemSharp2View;

int MakeHomImage( IntPtr vpCalib, char cLR, Bitmap tImgSrc, ref Eyem2ViewHom tpParam, ref Bitmap tImgDst )
```

解　説　　対応点探索用の画像として、指定カメラの原画像を射影変換し、もう一方のカメラから見たような画像にします。なお、画像射影変換については、Appendix4をご参照下さい。

引　数

入力	vpCalib	キャリブレーション・ディスクリプタ CalibOpen関数またはLoadCalibDataF関数の出力値を指定します。
入力	cLR	左右カメラ画像の選択 左カメラ画像の場合は'L'、右カメラ画像の場合は'R'を指定します。
入力	tImgSrc	原画像メモリ CalibOpen関数で指定した画像サイズとします。
入力	tpParam	画像射影変換パラメータ tCenter：射影変換画像の中央に対応する、原画像の座標点 dWorldZ：計測点3次元座標のz座標概略値
出力	tImgDst	射影変換画像を格納する画像メモリ CalibOpen関数で指定した画像サイズとします。

戻り値

値	定数	意味
0	FUNC_OK	正常終了
－2	FUNC_ILLEGAL_ARGUMENT	引数が不適当
－100	FUNC_CANNOT_CALC	計算不可
－999	FUNC_CANNOT_USE	ライブラリ使用不可

留意事項　　特にありません。

InvHomImgPoint

機　能　　射影変換画像点から原画像点への変換

クラス　　EyemSharp2View.eyem2View

形　式

```
using EyemSharp;
using EyemSharp2View;

int InvHomImgPoint( IntPtr vpCalib, char cLR, ref EyemOcsDXY tpPtSrc, ref
Eyem2ViewHom tpParam, ref EyemOcsDXY tpPtDst )
```

解　説　　指定カメラの射影変換画像上の座標点を、原画像上の座標点に変換します。なお、射影変換画像は、MakeHomImage 関数で作成された画像とします。

引　数

<table>
<tr><td>入力</td><td>vpCalib</td><td>キャリブレーション・ディスクリプタ
CalibOpen関数またはLoadCalibDataF関数の出力値を指定します。</td></tr>
<tr><td>入力</td><td>cLR</td><td>左右カメラ画像の選択
左カメラ画像の場合は'L'、右カメラ画像の場合は'R'を指定します。</td></tr>
<tr><td>入力</td><td>tpPtSrc</td><td>射影変換画像上の座標点
<table><tr><td>dX</td><td>X座標（画素）</td></tr><tr><td>dY</td><td>Y座標（画素）</td></tr></table></td></tr>
<tr><td>入力</td><td>tpParam</td><td>画像射影変換パラメータ
必ず、MakeHomImage関数で用いた値を指定します。
なお、画像射影変換については、Appendix4をご参照下さい。
<table><tr><td>tCenter</td><td>射影変換画像の中央に対応する、原画像の座標点</td></tr><tr><td>dWorldZ</td><td>計測点3次元座標のz座標概略値</td></tr></table></td></tr>
<tr><td>出力</td><td>tpPtDst</td><td>原画像上の座標点
tpPtSrcと同じでも構いません。
<table><tr><td>dX</td><td>X座標（画素）</td></tr><tr><td>dY</td><td>Y座標（画素）</td></tr></table></td></tr>
</table>

戻り値

値	定数	意味
0	FUNC_OK	正常終了
－2	FUNC_ILLEGAL_ARGUMENT	引数が不適当
－100	FUNC_CANNOT_CALC	計算不可
－999	FUNC_CANNOT_USE	ライブラリ使用不可

留意事項　　特にありません。

CalcHomImgPoint

機　能　　原画像点から射影変換画像点への変換

クラス　　EyemSharp2View.eyem2View

形　式

```
using EyemSharp;
using EyemSharp2View;

int CalcHomImgPoint( IntPtr vpCalib, char cLR, ref EyemOcsDXY tpPtSrc, ref
Eyem2ViewHom tpParam, ref EyemOcsDXY tpPtDst )
```

解　説　　指定カメラの原画像上の点を、その射影変換画像上の座標点に変換します。なお、射影変換画像は、MakeHomImage 関数で作成された画像とします。

引　数

<table>
<tr><td>入力</td><td>vpCalib</td><td>キャリブレーション・ディスクリプタ
CalibOpen関数またはLoadCalibDataF関数の出力値を指定します。</td></tr>
<tr><td>入力</td><td>cLR</td><td>左右カメラ画像の選択
左カメラの場合は'L'、右カメラの場合は'R'を指定します。</td></tr>
<tr><td>入力</td><td>tpPtSrc</td><td>原画像上の座標点
<table>
<tr><td>dX</td><td>X座標（画素）</td></tr>
<tr><td>dY</td><td>Y座標（画素）</td></tr>
</table></td></tr>
<tr><td>入力</td><td>tpParam</td><td>画像射影変換パラメータ
必ず、MakeHomImage関数で用いた値を指定します。なお、画像射影変換については、Appendix4をご参照下さい。
<table>
<tr><td>tCente</td><td>射影変換画像の中央に対応する、原画像の座標点</td></tr>
<tr><td>dWorldZ</td><td>計測点3次元座標のz座標概略値</td></tr>
</table></td></tr>
<tr><td>出力</td><td>tpPtDst</td><td>射影変換画像上の座標点
tpPtSrcと同じでも構いません。
<table>
<tr><td>dX</td><td>X座標（画素）</td></tr>
<tr><td>dY</td><td>Y座標（画素）</td></tr>
</table></td></tr>
</table>

戻り値

値	定数	意味
0	FUNC_OK	正常終了
－2	FUNC_ILLEGAL_ARGUMENT	引数が不適当
－100	FUNC_CANNOT_CALC	計算不可
－153	FUNC_FAILED_UNDISTORT	歪み補正失敗
－999	FUNC_CANNOT_USE	ライブラリ使用不可

留意事項　　特にありません。

UnDistortImage

機　能　　画像の歪み補正

クラス　　EyemSharp2View.eyem2View

形　式

```
using EyemSharp;
using EyemSharp2View;

int UnDistortImage( IntPtr vpCalib, char cLR, Bitmap tImgSrc, ref Bitmap
tImgDst )
```

解　説　　指定カメラの原画像に対して、歪み補正した画像を作成します。

引　数

入力	vpCalib	キャリブレーション・ディスクリプタ CalibOpen関数またはLoadCalibDataF関数の出力値を指定します。
入力	cLR	左右カメラ画像の選択 左カメラ画像の場合は'L'、右カメラ画像の場合は'R'を指定します。
入力	tImgSrc	原画像メモリ CalibOpen関数で指定した画像サイズとします。
出力	tImgDst	歪み補正した画像を格納する画像メモリ CalibOpen関数で指定した画像サイズとします。

戻り値

値	定数	意味
0	FUNC_OK	正常終了
－2	FUNC_ILLEGAL_ARGUMENT	引数が不適当
－999	FUNC_CANNOT_USE	ライブラリ使用不可

留意事項　　特にありません。

DistortPoint

機　能　　歪み補正画像点から原画像点への変換

クラス　　EyemSharp2View.eyem2View

形　式

```
using EyemSharp;
using EyemSharp2View;

int DistortPoint( IntPtr vpCalib, char cLR, ref EyemOcsDXY tpPtSrc, ref
EyemOcsDXY tpPtDst )
```

解　説　　指定カメラの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。なお、歪み補正画像は、UnDistortImage 関数で作成された画像とします。

引　数

入力	vpCalib	キャリブレーション・ディスクリプタ CalibOpen関数またはLoadCalibDataF関数の出力値を指定します。
入力	cLR	左右カメラ画像の選択 左カメラ画像の場合は'L'、右カメラ画像の場合は'R'を指定します。
入力	tpPtSrc	歪み補正画像上の座標点 dX　X座標（画素） dY　Y座標（画素）
出力	tpPtDst	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

値	定数	意味
0	FUNC_OK	正常終了
－2	FUNC_ILLEGAL_ARGUMENT	引数が不適当
－999	FUNC_CANNOT_USE	ライブラリ使用不可

留意事項　　特にありません。

UnDistortPoint

機　能　　原画像点から歪み補正画像点への変換

クラス　　EyemSharp2View.eyem2View

形　式

```
using EyemSharp;
using EyemSharp2View;

int UnDistortPoint( IntPtr vpCalib, char cLR, ref EyemOcsDXY tpPtSrc, ref
EyemOcsDXY tpPtDst )
```

解　説　　指定カメラの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。なお、歪み補正画像は、UnDistortImage 関数で作成された画像とします。

引　数

入力	vpCalib	キャリブレーション・ディスクリプタ CalibOpen関数またはLoadCalibDataF関数の出力値を指定します。
入力	cLR	左右カメラ画像の選択 左カメラ画像の場合は'L'、右カメラ画像の場合は'R'を指定します。
入力	tpPtSrc	原画像上の座標点 dX：X座標（画素） dY：Y座標（画素）
出力	tpPtDst	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

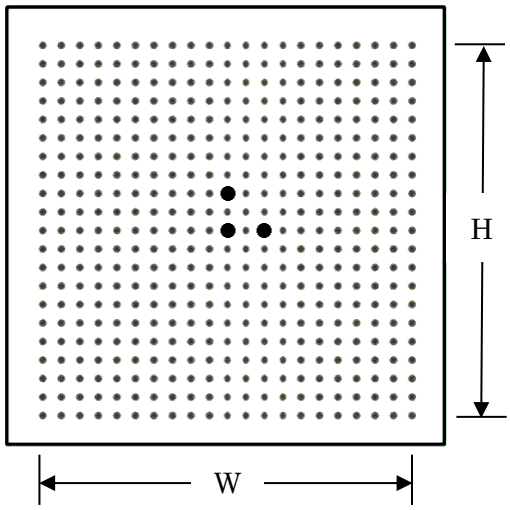
値	定数	意味
0	FUNC_OK	正常終了
－2	FUNC_ILLEGAL_ARGUMENT	引数が不適当
－153	FUNC_FAILED_UNDISTORT	歪み補正失敗
－999	FUNC_CANNOT_USE	ライブラリ使用不可

留意事項　　特にありません。

Appendix1. キャリブレーション治具情報構造体 Eyem2ViewJig の設定

本ライブラリでは、下図に示すような大小の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、格子点に小ドットを配置し、中央部分に座標軸を認識するための大ドットを配置します。大ドットについては、まず座標原点としてプレート中央に1つ配置し、さらにx軸上およびy軸上にそれぞれ1つずつ、合わせて3つを配置します。

このキャリブレーション治具の情報を設定する Eyem2ViewJig 構造体の具体的内容は以下のとおりです。



dPatternSizeX	パターン横方向(x方向)サイズ(mm)。上図の W サイズです。
dPatternSizeY	パターン縦方向(y方向)サイズ(mm)。上図の H サイズです。
dDotSizeSmall	小ドットサイズ(直径)(mm)
dDotSizeLarge	大ドットサイズ(直径)(mm)
dDotPitchSmallX	小ドット横方向(x方向)間隔(mm)
dDotPitchSmallY	小ドット縦方向(y方向)間隔(mm)
dDotPitchLargeX	大ドット横方向(x方向)間隔(mm)
dDotPitchLargeY	大ドット縦方向(y方向)間隔(mm)
iLargeDotNum	大ドット個数。現状は3を指定します。
iDotColor	ドットの色 ２値化におけるドットの色です。EYEM_BIN_BLACK(黒:0)または EYEM_BIN_WHITE(白:1)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値ブロッブが処理対象となります。
dDotEdgeStrThrs	ドットのエッジ強度下限値 ピントぼけしたドットを除去するための値です。0 以上の値を指定します。(標準値:0)

iMinPtnPtNum	ドット最小数 認識すべきドット個数の最小値を max(4、iLargeDotNum)以上の値で指定します。

Appendix2. ワールド座標系

本ライブラリで使用できるワールド座標系は以下の4種類で、すべて右手座標系です。

（1）第1パターン座標系

キャリブレーションに用いたパターン群における1枚目（インデックス 0 番）のパターン面を xy 面とする座標系です。

（2）左カメラ座標系

左カメラのレンズ中心を原点とし、光軸を z 軸とした座標系です。

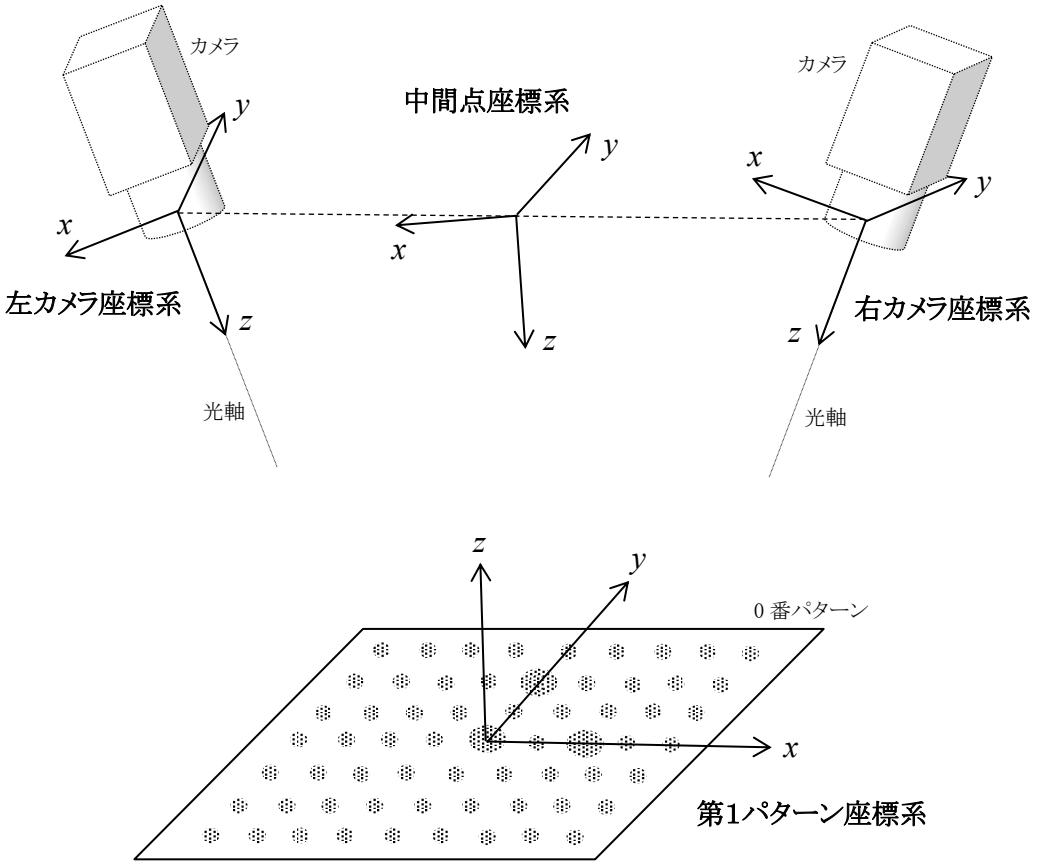
（3）右カメラ座標系

右カメラのレンズ中心を原点とし、光軸を z 軸とした座標系です。

（4）中間点座標系

左右レンズ中心の中間点を原点とした座標系です。座標系の各軸の基底ベクトルは、左右カメラ座標系のそれぞれの軸の基底ベクトルの平均としています。

※この他に、ワールド座標系ではないが、キャリブレーションに用いた各パターンに付随する、第1パター座標系と同じ設定の座標系を、パターン座標系とします。



Appendix3. キャリブレーションの方法

キャリブレーションの具体的な手順は以下のとおりです。

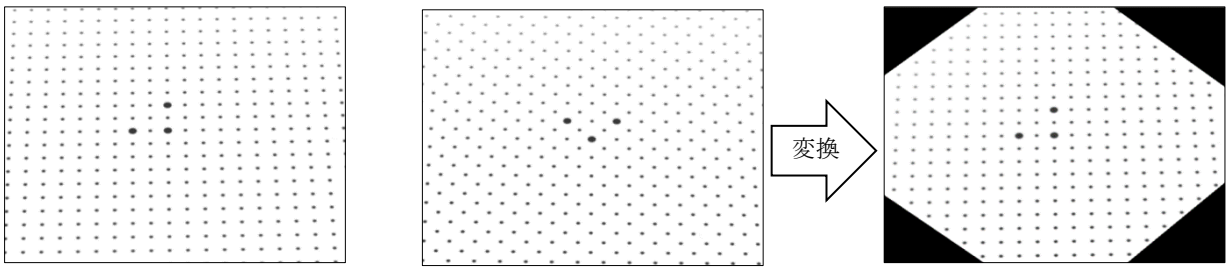
① ワールド座標系のxy 面を設定したい場所にパターンを設置します。
② パターンを固定したまま、左右のカメラそれぞれで画像を読み込みます。これを、1セットとします。
③ パターンの位置および向きを変えながら②の手順を繰り返し、画像セットを3セット以上取得します。
④ 本キャリブレーションライブラリを実行します。

留意事項

画像内に映るパターンが画像内をまんべんなく、さらに、どの2枚も平行とならないようにパターンの位置と向きを調整してください。すなわち、全体として、パターン・ドットが画像内にまんべんなく配置されるようにしてください。これは、歪み補正係数を求めるために必要な措置です。

Appendix4. 画像射影変換について

画像射影変換では、ワールド座標系内の平面 z = h（h は、Eyem2ViewHom 構造体のパラメータ指定値）の見え方を同じにします。具体的には、一方のカメラ（A とします）から見た平面 z = h の画像と同じになるように、他方のカメラ（B とします）から見た平面 z = h の画像を変換します。



カメラ A 原画像　　カメラ B 原画像　　カメラ B 射影変換画像

注意していただきたいのは、あくまで平面 z = h に対して同じ見え方にするということです。例えばカメラ B の画像に平面 z = h 上にない対象 α が映っている場合は、その α が平面 z = h 上にあるとして変換をしてしまいます。よって、平面 z = h 上にない対象物については、カメラ A 画像と変換後のカメラ B 画像において、画像内の対象物の相対的な位置関係が崩れてしまうので、それらを計測対象にすると正確な3次元位置を求めることができません。

画像射影変換構造体 Eyem2ViewHom のパラメータ設定は、以下のとおりです。

●EyemOcsDXY tCenter

射影変換画像の中点座標（画像幅/2、 画像高さ/2）に対応する、原画像の座標点です。

dX: X 座標（画素）

dY: Y 座標（画素）

●double dWorldZ

ワールド座標系における、計測点のz座標概略値です。上記の h です。