# 光切断3次元計測ライブラリ

(Ver.1.1)

2016 年 5 月

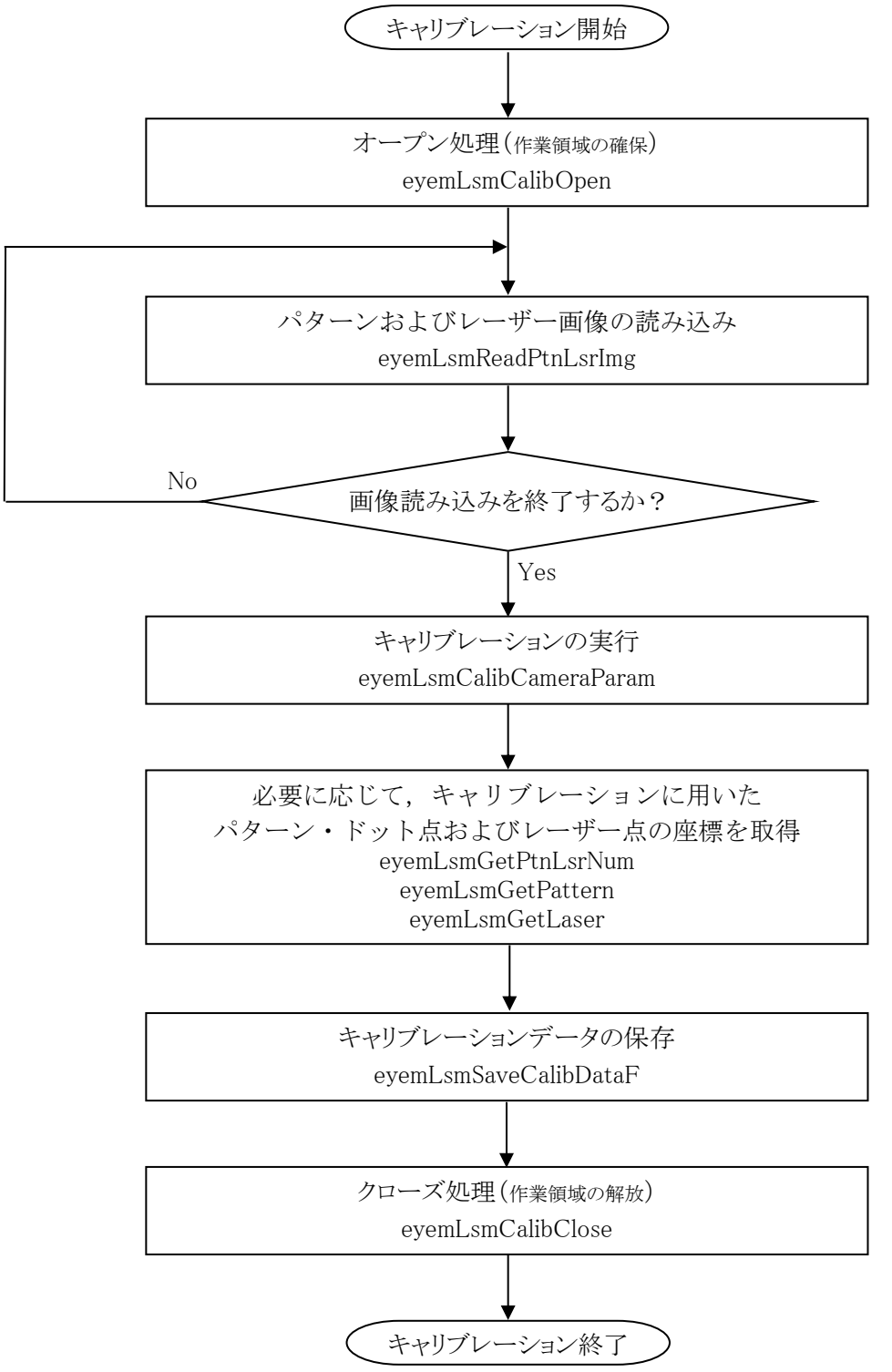
株式会社 アイディール

目次

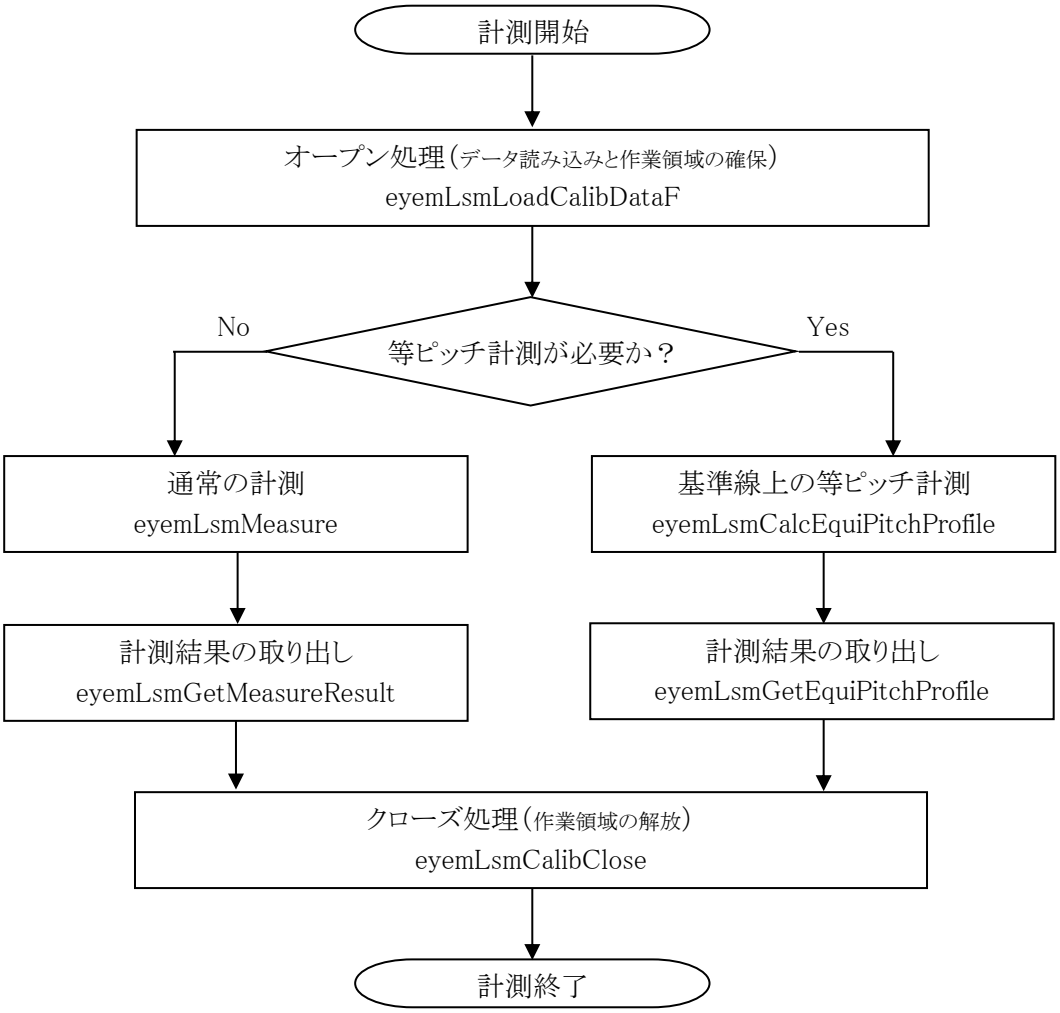
# 1. 光切断3次元計測ライブラリの使用手順

光切断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. 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** に示しましたので, そちらをご参照下さい.

## eyemLsmCalibOpen

**機 能**　　キャリブレーションのオープン処理

**形 式**　　#include “eyemLsm.h”

int　eyemLsmCalibOpen( int iSetNum, EyemLsmJig *tpJig, EyemLsmCam *tpCam,
int iWldFlag, void **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**

iSetNum　［入力］キャリブレーションパターンとレーザーの画像セットの組数．
３組以上で指定します．

*tpJig　［入力］キャリブレーション治具情報．
詳細は **Appendix2** をご参照下さい．

*tpCam　［入力］カメラ情報．
iCam：カメラ種別．以下のいずれかを指定します．

| | |
|---|---|
| EYEM_LSM_CAM_NORMAL | 通常のカメラ |
| EYEM_LSM_CAM_SCHEIMPFLUG | シャインプルーフ構成のカメラ |

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）．

iWldFlag　［入力］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 **Appendix3** をご参照下さい．

| | |
|---|---|
| EYEM_LSM_WLD_PTN | 第1パターン座標系 |
| EYEM_LSM_WLD_CAM | カメラ座標系 |
| EYEM_LSM_WLD_LSR | レーザー平面座標系 |

**vpCalib　［出力］キャリブレーション・ディスクリプタ（データ領域の先頭アドレス）．
なお，*vpCalib 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 | 引数が不適当 |
| 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 eyemLsmCalibClose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．

## eyemLsmSaveCalibDataF

**機 能**　　キャリブレーションデータのファイル保存

**形 式**　　#include “eyemLsm.h”

int　eyemLsmSaveCalibDataF( const char *cpFilePath, void *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**　　*cpFilePath　　［入力］保存先のファイル・パス.

*vpCalib　　［入力］キャリブレーション・ディスクリプタ.

eyemLsmCalibOpen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 | 引数が不適当 |
| 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.

## eyemLsmLoadCalibDataF

**機 能**　　計測のオープン処理

**形 式**　　#include “eyemLsm.h”

int　eyemLsmLoadCalibDataF( const char *cpFilePath, void **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**

| | |
|---|---|
| *cpFilePath | ［入力］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． |
| **vpCalib | ［出力］キャリブレーション・ディスクリプタ（データ領域の先頭アドレス）．<br>なお，*vpCalib 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 | 引数が不適当 |
| 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，計測終了時に eyemLsmCalibClose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．

## eyemLsmCalibClose

**機 能**　　クローズ処理

**形 式**　　#include “eyemLsm.h”
void　eyemLsmCalibClose( void **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**　　**vpCalib　　［入・出力］キャリブレーション・ディスクリプタ
eyemLsmCalibOpen関数 または eyemLsmLoadCalibDataF関数の出力値を指定します. *vpCalib 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.

## eyemLsmReadPtnLsrImg

**機 能**　　パターン・ドットとパターン上レーザー線の座標点の読み込み

**形 式**　　#include “eyemLsm.h”

int　eyemLsmReadPtnLsrImg( void *vpCalib, int iImgNo, EyemLsmImg *tpImgSet,
EyemRect *tpRoi );

**解 説**　　パターン・ドットおよびパターン上レーザー線の画像から，有効な座標点を読み込みます．

**引 数**

*vpCalib　［入・出力］キャリブレーション・ディスクリプタ．
eyemLsmCalibOpen関数の出力値を指定します．

iImgNo　［入力］対象となるパターンとレーザーの画像セット番号．
0番からの通し番号を入力します．

*tpImgSet　［入力］対象となるパターンとレーザーの画像セット．
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，
**Appendix4** をご参照下さい．
*ucpImgPtn：パターン・ドット画像メモリの先頭アドレス．
*ucpImgLsr：パターン上レーザー線画像メモリの先頭アドレス．

*tpRoi　［入力］処理範囲の矩形領域（左上座標およびサイズ）．
NULL を指定すると自動設定となります．
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**　　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 | 引数が不適当 |
| 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_MARK | パターン大ドット点が読めない |
| FUNC_CANNOT_READ_PTN | パターン・ドット点の実座標が読めない |
| FUNC_FEW_LSR_SMPL_NUM | レーザー線の参照点数が足りない |

| | |
|---|---|
| FUNC_FAILED_LSR_LINE | レーザー線の直線近似失敗 |
| FUNC_CANNOT_READ_LSR | レーザー線の実座標が読めない |
| FUNC_FAILED_HOMOGRAPHY | ホモグラフィー行列計算失敗 |

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

## eyemLsmCalibCameraParam

**機 能**　　カメラキャリブレーションの実行

**形 式**　　#include “eyemLsm.h”

int　eyemLsmCalibCameraParam( void *vpCalib, double *dpRms );

**解 説**　　パターン・ドット点およびパターン上レーザー線の座標データ群から，カメラのキャリブレーションを行います．

**引 数**

| | |
|---|---|
| *vpCalib | ［入・出力］キャリブレーション・ディスクリプタ<br>eyemLsmCalibOpen関数の出力値を指定します． |
| *dpRms | ［出力］バンドル調整評価関数値（画素）<br>残差（再投影誤差）の重み付き平均二乗誤差の平方根です． |

**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 | 引数が不適当 |
| FUNC_CANNOT_CALC | 計算不可 |
| FUNC_CANNOT_USE | ライブラリ使用不可 |
| FUNC_CANNOT_READ_LSR | レーザーの実座標が読めない |
| FUNC_FAILED_HOMOGRAPHY | ホモグラフィー行列計算失敗 |
| FUNC_FAILED_CAM_PRM | カメラパラメータ計算失敗 |
| FUNC_FAILED_BUNDLE_ADJ | バンドル調整失敗 |

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

## eyemLsmMeasure

**機 能**　　レーザー線画像座標から3次元座標への変換

**形 式**　　#include “eyemLsm.h”

int　eyemLsmMeasure( void *vpCalib, unsigned char *ucpImage, EyemRect *tpRoi,
EyemLsmLsr *tpParam );

**解 説**　　レーザー線の画像座標を取得し，3次元座標へ変換します．

**引 数**

*vpCalib　［入・出力］キャリブレーション・ディスクリプタ
eyemLsmLoadCalibDataF関数の出力値を指定します．

*ucpImage　［入力］計測対象の画像メモリ先頭アドレス．

*tpRoi　［入力］処理範囲の矩形領域（左上座標およびサイズ）．
NULL を指定すると全画面となります．
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）．

*tpParam　［入力］レーザー点群の選別パラメータ．
NULL を指定するとデフォルト値となります．
dLevelMul： 2値化レベルに対する倍率（デフォルト値：1.0）．
これは，濃度下限値の指定を意味します．
dSigmaCoef： 線幅 σ 値の係数（デフォルト値：3.0）．
これは，線幅のレンジの指定を意味します．

**戻り値**　　終了コードです．

| | |
|---|---|
| 0以上 | （正常終了）座標取得個数 |
| FUNC_NOT_ENOUGH_MEM | ワークメモリ不足 |
| FUNC_ILLEGAL_ARGUMENT | 引数が不適当 |
| FUNC_CANNOT_CALC | 計算不可 |
| 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**　　本関数は，eyemLsmLoadCalibDataF関数でのオープン時に限り使用可能となります．

## eyemLsmGetMeasureResult

**機 能**　　レーザー線の3次元座標の取得

**形 式**　　#include “eyemLsm.h”

int　eyemLsmGetMeasureResult( void *vpCalib, int iDataNo, EyemOcsDXY *tpImgPt,
EyemOcsDXYZ *tpPt3D );

**解 説**　　eyemLsmMeasure 関数による, レーザー線の画像座標値およびその3次元座標値への変換結果を取得します.

**引 数**

| | |
|---|---|
| *vpCalib | ［入力］キャリブレーション・ディスクリプタ<br>eyemLsmLoadCalibDataF関数の出力値を指定します. |
| iDataNo | ［入力］座標データの番号.<br>0番からの通し番号を指定します. |
| *tpImgPt | ［出力］画像座標.<br>dX: X座標(画素).<br>dY: Y座標(画素). |
| *tpPt3D | ［出力］3次元座標.<br>dX: X座標(mm).<br>dY: Y座標(mm).<br>dZ: Z座標(mm). |

**戻り値**　　終了コードです.

| | |
|---|---|
| FUNC_OK | 正常終了 |
| FUNC_ILLEGAL_ARGUMENT | 引数が不適当 |
| FUNC_CANNOT_CALC | 計算不可(データなし) |
| FUNC_CANNOT_USE | ライブラリ使用不可 |

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

## eyemLsmChangeLsr2Dto3D

**機 能**　　画像座標点から3次元座標点への変換

**形 式**　　#include “eyemLsm.h”

int　eyemLsmChangeLsr2Dto3D( void *vpCalib, EyemOcsDXY *tpImgPt,
EyemOcsDXYZ *tpPt3D );

**解 説**　　画像座標点を，ワールド座標系内のレーザー平面上における3次元座標点へ変換します．
なお，ワールド座標系がレーザー平面座標系の場合は，3次元座標のz成分は常にゼロとなります．

**引 数**

| | |
|---|---|
| *vpCalib | ［入力］キャリブレーション・ディスクリプタ<br>eyemLsmCalibOpen関数 または eyemLsmLoadCalibDataF関数の出力値を指定します． |
| *tpImgPt | ［入力］画像座標．<br>dX： X座標（画素）．<br>dY： Y座標（画素）． |
| *tpPt3D | ［出力］3次元座標．<br>dX： X座標（mm）．<br>dY： Y座標（mm）．<br>dZ： Z座標（mm）． |

**戻り値**　　終了コードです．

| | |
|---|---|
| FUNC_OK | 正常終了 |
| FUNC_ILLEGAL_ARGUMENT | 引数が不適当 |
| 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．

## eyemLsmChange3Dto2D

**機 能**　　3次元座標点から画像座標点への変換

**形 式**　　#include “eyemLsm.h”

int　eyemLsmChange3Dto2D( void *vpCalib, EyemOcsDXYZ *tpPt3D,
EyemOcsDXY *tpImgPt );

**解 説**　　ワールド座標系における3次元座標点から，画像座標点への変換を行います．

**引 数**

| | |
|---|---|
| *vpCalib | ［入力］キャリブレーション・ディスクリプタ<br>eyemLsmCalibOpen関数 または eyemLsmLoadCalibDataF関数の出力値を指定します． |
| *tpPt3D | ［入力］3次元座標．<br>dX： X座標（mm）．<br>dY： Y座標（mm）．<br>dZ： Z座標（mm）． |
| *tpImgPt | ［出力］画像座標．<br>dX： X座標（画素）．<br>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 | 引数が不適当 |
| FUNC_CANNOT_USE | ライブラリ使用不可 |

**留意事項**　　特にありません．

## eyemLsmGetHomography

**機 能**　　ホモグラフィ行列の取得

**形 式**　　#include “eyemLsm.h”

int　eyemLsmGetHomography( void *vpCalib, EyemCalibHom *tpHom );

**解 説**　　ホモグラフィ行列を取得します. この行列は, レーザー平面と画像平面との間の射影変換行列 $\mathbf{H}$ です. すなわち, レーザー平面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$(x, \mathrm{y})$ とするとき, $\lambda$ 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**

| | | |
|---|---|---|
| *vpCalib | ［入力］キャリブレーション・ディスクリプタ<br>eyemLsmCalibOpen関数 または eyemLsmLoadCalibDataF関数の出力値を指定します. |
| *tpHom | ［出力］ホモグラフィ行列.<br>daH[3][3]: ホモグラフィー行列 $\mathbf{H}$.<br>daInvH[3][3]: 行列 $\mathbf{H}$ の逆行列 $\mathbf{H}^{-1}$. |

**戻り値**　　終了コードです.

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

**留意事項**　　特にありません.

## eyemLsmGetCamIntParam

**機 能**　　カメラ内部パラメータの取得

**形 式**　　#include “eyemLsm.h”

int　eyemLsmGetCamIntParam( void *vpCalib, EyemCalibInt *tpInt );

**解 説**　　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「焦点距離×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**

*vpCalib　［入力］キャリブレーション・ディスクリプタ
eyemLsmCalibOpen関数 または eyemLsmLoadCalibDataF関数の出力値を指定します.

*tpInt　［出力］カメラ内部パラメータ.
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**　　終了コードです.

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

**留意事項**　　特にありません.

## eyemLsmGetCamExtParam

**機 能**　　カメラ外部パラメータの取得

**形 式**　　#include “eyemLsm.h”

int　eyemLsmGetCamExtParam( void *vpCalib, int iPtnNo, EyemCalibExt *tpExt );

**解 説**　　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**

| | | |
|---|---|---|
| *vpCalib | ［入力］キャリブレーション・ディスクリプタ<br>eyemLsmCalibOpen関数 または eyemLsmLoadCalibDataF関数の出力値を指定します. |
| iPtnNo | ［入力］パターンの番号<br>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. |
| *tpExt | ［出力］カメラ外部パラメータ.<br>daR[3][3]: 回転行列.<br>daT[3]: 平行移動ベクトル.<br>daRdr[3]: 回転ベクトル(ロドリゲスの表現). |

**戻り値**　　終了コードです.

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

**留意事項**　　特にありません.

## eyemLsmGetPtnLsrNum

**機 能**　　パターン・ドット点およびパターン上レーザー線の画像座標データ個数の取得

**形 式**　　#include “eyemLsm.h”

int　eyemLsmGetPtnLsrNum( void *vpCalib, int iDataNo, int *ipPtnNum, int *ipLsrNum );

**解 説**　　キャリブレーションで使用したパターン・ドット点およびパターン上レーザー線の画像座標データの個数を取得します.

**引 数**

| | |
|---|---|
| *vpCalib | ［入力］キャリブレーション・ディスクリプタ<br>eyemLsmCalibOpen関数の出力値を指定します. |
| iDataNo | ［入力］座標データの番号<br>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. |
| *ipPtnNum | ［出力］パターン・ドット点の画像座標データの個数. |
| *ipLsrNum | ［出力］パターン上レーザー線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 | 引数が不適当 |
| FUNC_CANNOT_CALC | 計算不可(データなし) |
| FUNC_CANNOT_USE | ライブラリ使用不可 |

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

## eyemLsmGetPattern

**機 能**　　パターン・ドット点の画像座標データの取得

**形 式**　　#include “eyemLsm.h”

int　eyemLsmGetPattern( void *vpCalib, int iDataNo, EyemOcsDXY taPtn[] );

**解 説**　　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，eyemLsmGetPtrLsrNum 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**

| | | |
|---|---|---|
| *vpCalib | ［入力］キャリブレーション・ディスクリプタ<br>eyemLsmCalibOpen関数の出力値を指定します． |
| iDataNo | ［入力］座標データの番号<br>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． |
| taPtn[] | ［出力］パターン・ドット点の画像座標データ．<br>dX：X座標（画素）．<br>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 | 引数が不適当 |
| FUNC_CANNOT_CALC | 計算不可（データなし） |
| FUNC_CANNOT_USE | ライブラリ使用不可 |

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

## eyemLsmGetLaser

**機 能**　　パターン上レーザー線の画像座標データの取得

**形 式**　　#include “eyemLsm.h”

int　eyemLsmGetLaser( void *vpCalib, int iDataNo, EyemOcsDXY taLsr[] );

**解 説**　　キャリブレーションで使用したパターン上のレーザー線の画像座標データを取得します．予め，eyemLsmGetPtrLsrNum 関数で取得した個数分の配列 taLsr[] を確保してください．

**引 数**

| | | |
|---|---|---|
| *vpCalib | ［入力］キャリブレーション・ディスクリプタ<br>eyemLsmCalibOpen関数の出力値を指定します． |
| iDataNo | ［入力］座標データの番号<br>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． |
| taLsr[] | ［出力］パターン上レーザー線の画像座標データ．<br>dX：X座標（画素）．<br>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 | 引数が不適当 |
| FUNC_CANNOT_CALC | 計算不可（データなし） |
| FUNC_CANNOT_USE | ライブラリ使用不可 |

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

## eyemLsmShiftLsr3D

**機 能**　　レーザー線画像点列の平行移動

**形 式**　　#include “eyemLsm.h”

int　eyemLsmShiftLsr3D( void *vpCalib, EyemOcsDXYZ *tpShift, int iNum,
EyemOcsDXY taSrcPt[], EyemOcsDXY taDstPt[] );

**解 説**　　レーザー線の画像点列を，3次元空間内（ワールド座標系内）にて平行移動し，それを再び画像面へ投影したときのレーザー線画像点列を取得します．なお，ここで用いるワールド座標系は，キャリブレーション時に指定したものです．

**引 数**

| | |
|---|---|
| *vpCalib | ［入力］キャリブレーション・ディスクリプタ<br>eyemLsmCalibOpen関数の出力値を指定します． |
| *tpShift | ［入力］3次元空間内の平行移動量（mm）<br>dX： X方向移動量（mm）．<br>dY： Y方向移動量（mm）．<br>dZ： Z方向移動量（mm）． |
| iNum | ［入力］レーザー線画像点のデータ数．<br>配列 taSrcPt[] の要素数です． |
| taSrcPt[] | ［入力］移動元のレーザー線画像点配列．<br>dX： X座標（画素）．<br>dY： Y座標（画素）． |
| taDstPt[] | ［出力］移動先のレーザー線画像点配列．<br>taSrcPt[] と同じでも構いません．<br>dX： X座標（画素）．<br>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．

## eyemLsmCalcEquiPitchProfile

**機 能**　　レーザー線の等ピッチ3次元座標の計算

**形 式**

#include “eyemLsm.h”

int　eyemLsmCalcEquiPitchProfile( void *vpCalib, unsigned char *ucpImage,
EyemRect *tpRoi, EyemLsmLsr *tpParam, double dPitch );

**解 説**　　≪ワールド座標系:第1パターン座標系　限定≫ 基準線上において等ピッチとなるように，レーザー線の3次元座標を計算します．ここで，基準線とは，レーザー平面と第1パターン座標系 xy 平面との交線です．また，等ピッチとは，隣り合うレーザー点から基準線へ下した垂線の足の間隔が，指定されたピッチであることを意味します．なお，レーザー線の情報がない部分は，線型補間により仮想点を計算します．

**引 数**

| | |
|---|---|
| *vpCalib | ［入・出力］キャリブレーション・ディスクリプタ<br>eyemLsmCalibOpen関数の出力値を指定します． |
| *ucpImage | ［入力］計測対象の画像メモリ先頭アドレス |
| *tpRoi | ［入力］処理範囲の矩形領域（左上座標およびサイズ）．<br>NULL を指定すると全画面となります．<br>iX:　左上X座標（画素）．<br>iY:　左上Y座標（画素）．<br>iWidth:　領域のX方向サイズ（画素）．<br>iHeight:　領域のY方向サイズ（画素）． |
| *tpParam | ［入力］レーザー点群の選別パラメータ．<br>NULL を指定するとデフォルト値となります．<br>dLevelMul:　2値化レベルに対する倍率（デフォルト値:1.0）．<br>これは，濃度下限値の指定を意味します．<br>dSigmaCoef:　線幅 σ 値の係数（デフォルト値:3.0）．<br>これは，線幅のレンジの指定を意味します． |
| dPitch | ［入力］ピッチ（mm）．<br>正の数を指定します． |

**戻り値**　　終了コードです．

| | |
|---|---|
| 0以上 | （正常終了）座標個数 |
| FUNC_NOT_ENOUGH_MEM | ワークメモリ不足 |
| FUNC_ILLEGAL_ARGUMENT | 引数が不適当 |
| FUNC_CANNOT_CALC | 計算不可 |

| | |
|---|---|
| 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**　本関数は，eyemLsmLoadCalibDataF 関数でのオープン時，かつ ワールド座標系が第1パターン座標系の場合に限り使用可能となります．

## eyemLsmGetEquiPitchProfile

**機 能**　　レーザー線の等ピッチ3次元座標の取得

**形 式**　　#include “eyemLsm.h”

int　eyemLsmGetEquiPitchProfile( void *vpCalib, int iDataNo, EyemOcsDXYZ *tpPt3D );

**解 説**　　≪ワールド座標系:第1パターン座標系 限定≫ eyemLsmCalcEquiPitchProfile 関数で求めたレーザー線の等ピッチ3次元座標値を取得します.

**引 数**

| | |
|---|---|
| *vpCalib | ［入力］キャリブレーション・ディスクリプタ<br>eyemLsmCalibOpen関数の出力値を指定します. |
| iDataNo | ［入力］座標データの番号.<br>0番からの通し番号を指定します. |
| *tpPt3D | ［出力］3次元座標.<br>dX: X座標(mm).<br>dY: Y座標(mm).<br>dZ: Z座標(mm). |

**戻り値**　　終了コードです.

| | |
|---|---|
| FUNC_OK | 正常終了 |
| FUNC_ILLEGAL_ARGUMENT | 引数が不適当 |
| FUNC_CANNOT_CALC | 計算不可(データなし) |
| FUNC_CANNOT_USE | ライブラリ使用不可 |

**留意事項**　　本関数は, eyemLsmLoadCalibDataF 関数でのオープン時, かつ ワールド座標系が第1パターン座標系の場合に限り使用可能となります.

# 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)    // 引数が不適当
#define FUNC_CANNOT_CALC                (-100)  // 計算不可
#define FUNC_CANNOT_USE                 (-999)  // ライブラリ使用不可
#define FUNC_FAILED_HOMOGRAPHY          (-150)  // ホモグラフィー行列計算失敗
#define FUNC_FAILED_CAM_PRM             (-151)  // カメラパラメータ計算失敗
#define FUNC_FAILED_BUNDLE_ADJ          (-152)  // バンドル調整失敗
#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_MARK           (-201)  // パターンマーカー点(大ドット点)の認識不可
#define FUNC_CANNOT_READ_PTN            (-202)  // パターン参照点の実座標の認識不可
#define FUNC_FEW_LSR_SMPL_NUM           (-210)  // レーザー線の参照点数が足りない
#define FUNC_CANNOT_READ_LSR            (-211)  // レーザー参照点の実座標の認識不可
#define FUNC_FAILED_LSR_LINE            (-212)  // レーザー線の直線近似失敗
#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


// カメラ種別
enum {
        EYEM_LSM_CAM_NORMAL         = 0,    // 通常のカメラ
        EYEM_LSM_CAM_SCHEIMPFLUG    = 1     // シャインプルーフカメラ
};


// ワールド座標系種別
enum {
        EYEM_LSM_WLD_PTN,                   // 第1パターン座標系
        EYEM_LSM_WLD_CAM,                   // カメラ座標系
        EYEM_LSM_WLD_LSR                    // レーザー平面座標系
};


// キャリブレーション治具情報
typedef struct {
        double      dPatternSizeX;          // パターン横方向(x方向)サイズ(mm)(パターン範囲の幅)
        double      dPatternSizeY;          // パターン縦方向(y方向)サイズ(mm)(パターン範囲の高さ)
        double      dDotSizeSmall;          // 小ドットサイズ(直径)(mm)
        double      dDotSizeLarge;          // 大ドットサイズ(直径)(mm)
        double      dDotPitchSmallX;        // 小ドット横方向(x方向)ピッチ(mm)
        double      dDotPitchSmallY;        // 小ドット縦方向(y方向)ピッチ(mm)
        double      dDotPitchLargeX;        // 大ドット横方向(x方向)ピッチ(mm)
        double      dDotPitchLargeY;        // 大ドット縦方向(y方向)ピッチ(mm)
        int         iLargeDotNum;           // 大ドット個数

        int         iDotColor;              // (2値化)パターンドットの色(黒:EYEM_BIN_BLACK, 白:EYEM_BIN_WHITE)
        int         iDotAreaThrs;           // (2値化)パターンドット点の面積下限値(画素)(ノイズ除去用)
        double      dDotEdgeStrThrs;        // パターンドット点のエッジ強度下限値(0 以上)(ぼけドット除去用, 標準値:0)
        int         iMinPtnPtNum;           // パターン参照点最小数( max( 4, iLargeDotNum ) 以上)

        int         iLsrSmplStep;           // レーザー線のサンプリングステップ(画素)(1 以上, 標準値:10)
        int         iMinLsrPtNum;           // レーザー参照点最小数(標準値:30)
}       EyemLsmJig;
```

```
// カメラ情報
typedef struct {
        int         iCam;                       // カメラ種別(上記 enum 定数)
        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
}       EyemLsmCam;


// パターン&レーザーの画像セット
typedef struct {
        unsigned char       *ucpImgPtn;         // パターン画像メモリの先頭アドレス
        unsigned char       *ucpImgLsr;         // レーザー画像メモリの先頭アドレス
}       EyemLsmImg;


// レーザー点群の選別
typedef struct {
        double      dLevelMul;                  // 2値化レベルに対する倍率(濃度下限値, 標準値:1.0)
        double      dSigmaCoef;                 // 線幅 σ 値の係数(線幅のレンジ, 標準値:3.0)
}       EyemLsmLsr;


// ホモグラフィー行列
typedef struct {
        double      daH[3][3];                  // ホモグラフィー行列 H
        double      daInvH[3][3];               // 行列 H の逆行列
}       EyemCalibHom;


// カメラの内部パラメータ(intrinsic parameter)
typedef struct {
        double      daA[3][3];                  // 内部パラメータ行列
        double      dFx, dFy;                   // 焦点距離×スケール(単位:画素)
        double      dFs;                        // 焦点距離×せん断係数(単位:画素)
        double      dUo, dVo;                   // 画像中心座標(単位:画素)
        double      dTheta;                     // 画像面傾斜角(単位:rad)
        double      dK1, dK2;                   // ラジアル歪み係数
        double      daRsv[5];                   // 予備領域
}       EyemCalibInt;


// カメラの外部パラメータ(extrinsic parameter)
typedef struct {
        double      daR[3][3];                  // 回転行列
        double      daT[3];                     // 並進ベクトル
        double      daRdr[3];                   // 回転ベクトル(ロドリゲスの表現)
}       EyemCalibExt;


// 矩形定義
typedef struct {
        int         iXs;                        // 始点(左上)x座標
        int         iYs;                        // 始点(左上)y座標
        int         iWidth;                     // x方向サイズ(幅)
        int         iHeight;                    // y方向サイズ(高さ)
}       EyemRect;
```

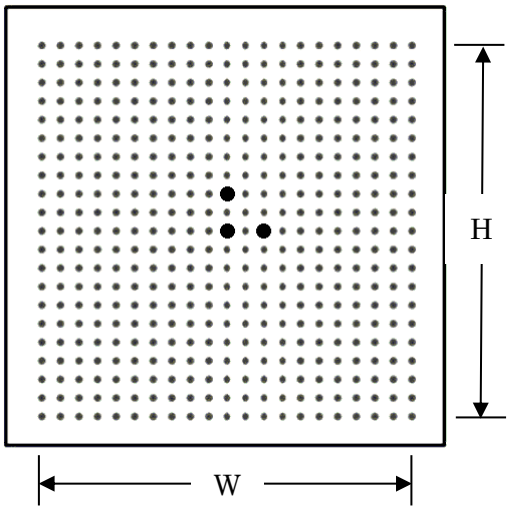
```
// 座標系定義(int 型)
typedef struct{
        int         iX;                 // X座標
        int         iY;                 // Y座標
}       EyemOcsIXY;


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

# Appendix2. キャリブレーション治具情報構造体 EyemLsmJig 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 EyemLsmJig 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）．
左図の W サイズです．

dPatternSizeY：パターン縦方向（y方向）サイズ（mm）．
左図の 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） 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．

以下は，パターン上に照射したレーザーに関する設定です．

iLsrSmplStep：レーザー線のサンプリングステップ（画素）．
レーザー線をサンプリングする間隔を 1 以上の値で指定します（標準値：10）．

iMinLsrPtNum：レーザー線の参照点最小数．
レーザー線のサンプリング個数の最小値を指定します（標準値：30）．

# Appendix3. 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3種類で，すべて右手座標系です．

（1）第1パターン座標系

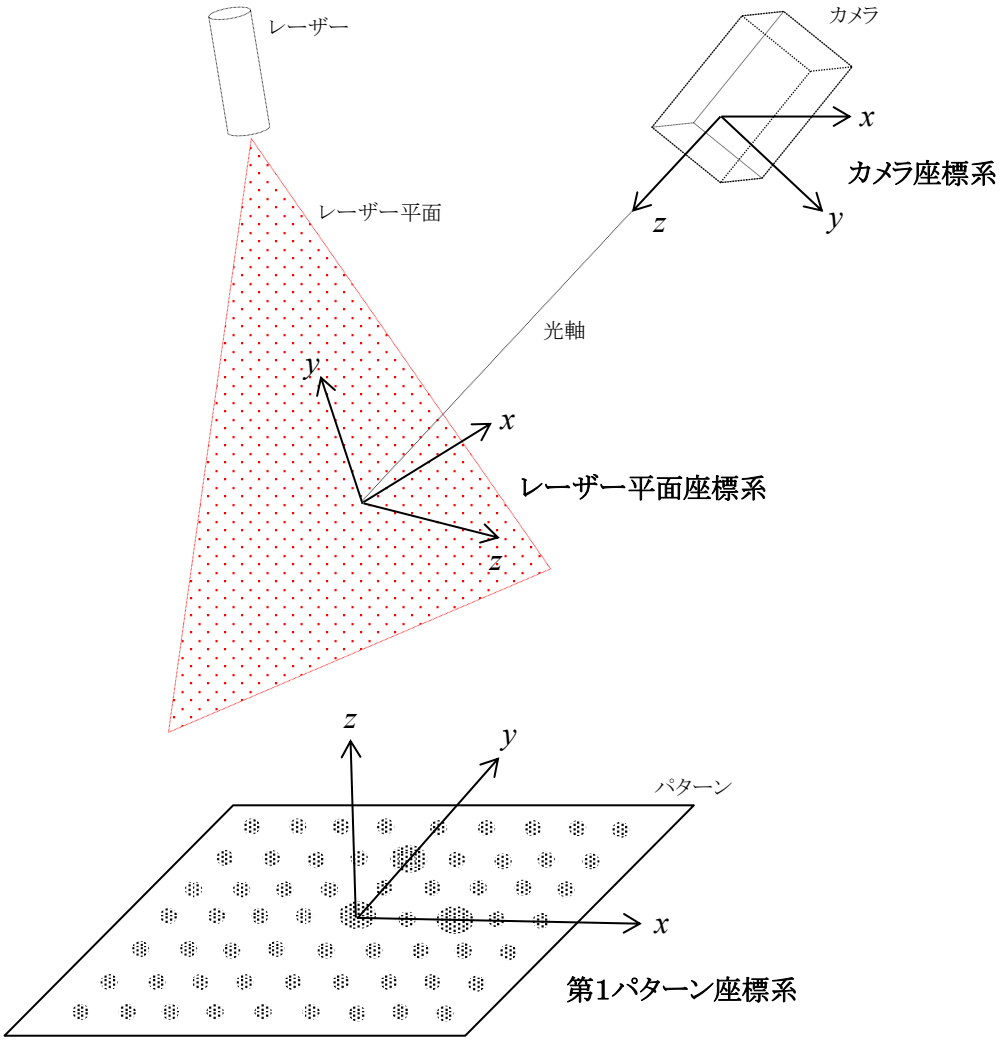
キャリブレーション・パターン画像群における1枚目の画像のパターン位置をワールド座標のxy面とする座標系です．

（2）カメラ座標系

レンズ中心を原点とし，光軸を z 軸とした座標系です．

（3）レーザー平面座標系

レンズ光軸とレーザー平面の交点を原点とし，レーザー平面をxy面とした座標系です．この座標系では，レーザー線の3次元座標のz成分は必ずゼロとなります．




」

# Appendix4. キャリブレーションの方法

キャリブレーションは，レーザーをパターン上に照射し，その画像を読み込むことで行います．具体的な手順は以下のとおりです.

① 最初のパターンを空間３次元座標系ｘｙ面に置き,照射されたレーザー線がパターンの中心付近を通るように位置を調整します（これが，第１パターン・ワールド座標系のｘｙ面となります).
② パターンを固定したまま，レーザーを照射した場合の画像と，照射しない場合の画像を１セットとして読み込みます.
③ パターンとカメラの距離およびパターンの向きを変えながら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.その際,照射されたレーザー線がパターンの中心付近を通るようにします.
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**

（１） 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距離と向きを調整してください.
（２） レーザー線（レーザー平面）は，カメラ座標系ｘ軸またはｙ軸に平行であることを想定しています.

改訂履歴

| Version No. | 日付 | 内　　容 |
|---|---|---|
| 1.0 | 2015.08.14 | ・新規発行 |
| 1.1 | 2015.05.03 | ・エラーコード（FUNC_CANNOT_USE）の新規追加<br>・Appendix4. キャリブレーションの方法の記述修正 |

以 上