

2次元幾何計算(2次元キャリパー)

(Ver.1.1)

2013 年 6 月

株式会社 アイディール

目次

eyemClp2dDistanceTwoPoints	1
機能 2点間の距離	1
eyemClp2dCenterTwoPoints	2
機能 2点の中点座標	2
eyemClp2dLineTwoPoints	3
機能 2点を通る直線	3
eyemClp2dMidperpendicularTwoPoints	4
機能 2点の垂直二等分線	4
eyemClp2dVerticalLinePointAndLine	5
機能 指定点を通り, 指定直線に垂直な直線	5
eyemClp2dLinePointAndSlope	6
機能 指定点を通り, 指定の傾きをもつ直線	6
eyemClp2dIntersectionTwoLines	7
機能 2直線の交点	7
eyemClp2dAngleTwoLines	8
機能 2直線の交角	8
eyemClp2dCenterLineOfTwoLines	9
機能 2直線の中央線(交角の2等分線)	9
eyemClp2dDistancePointToLine	10
機能 点と直線の距離	10
eyemClp2dFootOfPerpendicularToLine	11
機能 点から直線へ下した垂線の足	11
eyemClp2dTranslationOfLine	12
機能 直線の平行移動	12
eyemClp2dAreaTriangle	13
機能 3点の作る三角形の面積	13
eyemClp2dSignAreaTriangle	14
機能 3点の作る三角形の符号付き面積	14
eyemClp2dCircleThreePoints	15
機能 3点を通る円	15
eyemClp2dCircleTwoPoints	16
機能 2点を直径の両端とする円	16
eyemClp2dIntersectionLineAndCircle	17
機能 直線と円の交点	17
eyemClp2dTangentPointToCircle	18
機能 指定点から円に引いた接線と接点	18

eyemClp2dClosestToCircle.....	19
機能 指定点から円周への最近点とその距離.....	19

eyemClp2dDistanceTwoPoints

機 能 2点間の距離

形 式 `#include "eyemLib.h"`
`void eyedClp2dDistanceTwoPoints (EyemOcsDXY *tpPoint1,`
`EyemOcsDXY *tpPoint2, double *dpDist);`

解 説 2点 P_1 , P_2 の距離 (ユークリッド距離) を求めます.

引 数 `*tpPoint1` 点 P_1 です.
 `*tpPoint2` 点 P_2 です.
 `*dpDist` 2点 P_1 , P_2 の距離が格納されます.

戻り値 ありません.

留意事項 特にありません.

eyemClp2dCenterTwoPoints

機 能	2点の midpoint 座標	
形 式	<pre>#include "eyemLib.h" void eyedClp2dCenterTwoPoints (EyemOcsDXY *tpPoint1, EyemOcsDXY *tpPoint2, EyemOcsDXY *tpCenter);</pre>	
解 説	2点 P_1 , P_2 の midpoint 座標を求めます.	
引 数	*tpPoint1	点 P_1 です.
	*tpPoint2	点 P_2 です.
	*tpCenter	2点 P_1 , P_2 の midpoint 座標が格納されます.
戻り値	ありません.	
留意事項	特にありません.	

eyemClp2dLineTwoPoints

機 能 2点を通る直線

形 式 `#include "eyemLib.h"`
`int eyedClp2dLineTwoPoints (EyemOcsDXY *tpPoint1, EyemOcsDXY *tpPoint2,`
`EyemOcsDABC *tpLine);`

解 説 2点 P_1 , P_2 を通る直線 $ax + by + c = 0$ を求めます.

引 数 `*tpPoint1` 点 P_1 です.
`*tpPoint2` 点 P_2 です.
`*tpLine` 2点 P_1 , P_2 を通る直線 $ax + by + c = 0$ の係数 a , b , c が格納されます. なお, 法線ベクトル (a, b) は単位ベクトルとなっています.

戻り値 エラー報告です.

FUNC_OK	正常終了
FUNC_CANNOT_CALC	計算が不可 (2点 P_1, P_2 が一致)

留意事項 特にありません.

eyemClp2dMidperpendicularTwoPoints

機 能 2点の垂直二等分線

形 式

```
#include "eyemLib.h"

int      eyemClp2dMidperpendicularTwoPoints ( EyemOcsDXY *tpPoint1,
                                              EyemOcsDXY *tpPoint2, EyemOcsDABC *tpLine );
```

解 説 2点 P_1 , P_2 を結ぶ線分の垂直二等分線 $ax + by + c = 0$ を求めます.

引 数

*tpPoint1	点 P_1 です.
*tpPoint2	点 P_2 です.
*tpLine	2点 P_1 , P_2 を結ぶ線分の垂直二等分線 $ax + by + c = 0$ の係数 a , b , c が格納されます. なお, 法線ベクトル (a, b) は単位ベクトルとなっています.

戻り値 エラー報告です.

FUNC_OK	正常終了
FUNC_CANNOT_CALC	計算が不可 (2点 P_1, P_2 が一致)

留意事項 特にありません.

eyemClp2dVerticalLinePointAndLine

機 能 指定点を通り, 指定直線に垂直な直線

形 式

```
#include "eyemLib.h"

int      eyemClp2dVerticalLinePointAndLine ( EyemOcsDXY *tpPoint,
                                              EyemOcsDABC *tpLine, EyemOcsDABC *tpVertical );
```

解 説 指定された点 P を通り, 指定された直線 L に垂直な直線 $ax + by + c = 0$ を求めます.

引 数

*tpPoint	点 P です.
*tpLine	直線 L です.
*tpVertical	点 P を通り, 直線 L に垂直な直線 $ax + by + c = 0$ の係数 a, b, c が格納されます. なお, 法線ベクトル (a, b) は単位ベクトルとなっています.

戻り値 エラー報告です.

FUNC_OK	正常終了
FUNC_CANNOT_CALC	計算が不可(直線 L が直線でない)

留意事項 特にありません.

eyemClp2dLinePointAndSlope

機 能 指定点を通り, 指定の傾きをもつ直線

形 式

```
#include "eyemLib.h"

void      eyemClp2dLinePointAndSlope ( EyemOcsDXY *tpPoint, double dSlope,
                                         EyemOcsDABC *tpLine );
```

解 説 指定された点 P を通り, 指定された傾き θ をもつ直線 $L: ax + by + c = 0$ を求めます.

引 数

*tpPoint	点 P です.
dSlope	直線 L の傾き θ (単位:rad) です.
*tpLine	直線 L の係数 a, b, c が格納されます. なお, 法線ベクトル (a, b) は単位ベクトルとなっています.

戻り値 ありません.

留意事項 特にありません.

eyemClp2dIntersectionTwoLines

機 能 2直線の交点

形 式 `#include "eyemLib.h"`
`int eyedClp2dIntersectionTwoLines (EyemOcsDABC *tpLine1,`
`EyemOcsDABC *tpLine2, EyemOcsDXY *tpPoint);`

解 説 2直線 L_1 , L_2 の交点座標を求めます.

引 数 `*tpLine1` 直線 L_1 です.
`*tpLine2` 直線 L_2 です.
`*tpPoint` 2直線 L_1 , L_2 の交点座標が格納されます.

戻り値 エラー報告です.

FUNC_OK	正常終了
FUNC_CANNOT_CALC	計算が不可 (直線 L_1 , L_2 が直線でない, または2直線が平行)

留意事項 特にありません.

eyemClp2dAngleTwoLines

機 能 2直線の交角

形 式 `#include "eyemLib.h"`
`int eyedClp2dAngleTwoLines (EyemOcsDABC *tpLine1,`
`EyemOcsDABC *tpLine2, double *dpAngle);`

解 説 2直線 L_1 , L_2 の交角 (小さい方の角) を求めます.

引 数 `*tpLine1` 直線 L_1 です.
`*tpLine2` 直線 L_2 です.
`*dpAngle` 2直線 L_1 , L_2 の交角 (小さい方の角:rad単位) が格納されます.

戻り値 エラー報告です.

FUNC_OK	正常終了
FUNC_CANNOT_CALC	計算が不可 (直線 L_1 , L_2 が直線でない)

留意事項 特にありません.

eyemClp2dCenterLineOfTwoLines

機 能 2直線の中央線(交角の2等分線)

形 式

```
#include "eyemLib.h"

int      eyemClp2dCenterLineOfTwoLines ( EyemOcsDABC *tpLine1,
                                           EyemOcsDABC *tpLine2, EyemOcsDABC *tpLineC );
```

解 説 2直線 L_1 , L_2 の中央線(2直線からの等距離線)を求めます. これは, もし2直線が交わっていれば, 小さい方の交角の2等分線となります.

引 数

*tpLine1	直線 L_1 です.
*tpLine2	直線 L_2 です.
*dpLineC	2直線 L_1 , L_2 の中央線(小さい方の交角の2等分線)が格納されます.

戻り値 エラー報告です.

FUNC_OK	正常終了
FUNC_CANNOT_CALC	計算が不可(直線 L_1 , L_2 が直線でない)

留意事項 特にありません.

eyemClp2dDistancePointToLine

機 能 点と直線の距離

形 式

```
#include "eyemLib.h"

int      eyemClp2dDistancePointToLine ( EyemOcsDXY *tpPoint,
                                         EyemOcsDABC *tpLine, double *dpDist );
```

解 説 点 P と直線 L の距離(ユークリッド距離)を求めます.

引 数

*tpPoint	点 P です.
*tpLine	直線 L です.
*dpDist	点 P と直線 L の距離が格納されます.

戻り値 エラー報告です.

FUNC_OK	正常終了
FUNC_CANNOT_CALC	計算が不可(直線 L が直線でない)

留意事項 特にありません.

機 能	点から直線へ下した垂線の足				
形 式	<pre>#include "eyemLib.h" int eyemClp2dFootOfPerpendicularToLine (EyemOcsDXY *tpPoint, EyemOcsDABC *tpLine, EyemOcsDXY *tpFoot);</pre>				
解 説	点 P から直線 L へ下した垂線の足(交点座標)を求めます.				
引 数	<p>*tpPoint 点 P です.</p> <p>*tpLine 直線 L です.</p> <p>*tpFoot 点 P から直線 L へ下した垂線の足の座標が格納されます.</p>				
戻り値	<p>エラー報告です.</p> <table border="1"> <tr> <td>FUNC_OK</td><td>正常終了</td></tr> <tr> <td>FUNC_CANNOT_CALC</td><td>計算が不可(直線 L が直線でない)</td></tr> </table>	FUNC_OK	正常終了	FUNC_CANNOT_CALC	計算が不可(直線 L が直線でない)
FUNC_OK	正常終了				
FUNC_CANNOT_CALC	計算が不可(直線 L が直線でない)				
留意事項	特にありません.				

eyemClp2dTranslationOfLine

機 能 直線の平行移動

形 式 `#include "eyemLib.h"`
`int eyedClp2dTranslationOfLine (EyemOcsDABC *tpSrcL, EyemOcsDXY *tpTrans,`
`EyemOcsDABC *tpDstL);`

解 説 直線 L_s を移動量 T だけ平行移動した直線 L_D を求めます.

引 数 `*tpSrcL` 元の直線 L_s です.
`*tpTrans` 平行移動量 T です. x 方向移動量, y 方向移動量を指定します.
`*tpDstL` 平行移動後の直線 L_D が格納されます. `tpSrcL`と同じでも構いません.

戻り値 エラー報告です.

FUNC_OK	正常終了
FUNC_CANNOT_CALC	計算が不可 (直線 L_s が直線でない)

留意事項 特にありません.

eyemClp2dAreaTriangle

機 能	3点の作る三角形の面積								
形 式	<pre>#include "eyemLib.h" void eyemClp2dAreaTriangle (EyemOcsDXY *tpPoint1, EyemOcsDXY *tpPoint2, EyemOcsDXY *tpPoint3, double *dpArea);</pre>								
解 説	3点 P_1 , P_2 , P_3 の作る三角形の面積を求めます.								
引 数	<table><tr><td>*tpPoint1</td><td>点 P_1 です.</td></tr><tr><td>*tpPoint2</td><td>点 P_2 です.</td></tr><tr><td>*tpPoint3</td><td>点 P_3 です.</td></tr><tr><td>*dpArea</td><td>3点 P_1, P_2, P_3 の作る三角形の面積が格納されます.</td></tr></table>	*tpPoint1	点 P_1 です.	*tpPoint2	点 P_2 です.	*tpPoint3	点 P_3 です.	*dpArea	3点 P_1 , P_2 , P_3 の作る三角形の面積が格納されます.
*tpPoint1	点 P_1 です.								
*tpPoint2	点 P_2 です.								
*tpPoint3	点 P_3 です.								
*dpArea	3点 P_1 , P_2 , P_3 の作る三角形の面積が格納されます.								
戻り値	ありません.								
留意事項	特にありません.								

eyemClp2dSignAreaTriangle

機 能	3点の作る三角形の符号付き面積								
形 式	<pre>#include "eyemLib.h" void eyemClp2dSignAreaTriangle (EyemOcsDXY *tpPoint0, EyemOcsDXY *tpPoint1, EyemOcsDXY *tpPoint2, double *dpArea);</pre>								
解 説	3点 P_0 , P_1 , P_2 の作る三角形の面積を符号付きで求めます. 点 P_0 と点 P_1 を結ぶ線分 $\overline{P_0P_1}$, と, 点 P_0 と点 P_2 を結ぶ線分 $\overline{P_0P_2}$ に対して, 線分 $\overline{P_0P_1}$ から見た線分 $\overline{P_0P_2}$ が正の(負の)回転方向にあれば, 正の(負の)面積となります.								
引 数	<table><tr><td>*tpPoint0</td><td>点 P_0 です.</td></tr><tr><td>*tpPoint1</td><td>点 P_1 です.</td></tr><tr><td>*tpPoint2</td><td>点 P_2 です.</td></tr><tr><td>*dpArea</td><td>3点 P_0, P_1, P_2 の作る三角形の符号付き面積が格納されます.</td></tr></table>	*tpPoint0	点 P_0 です.	*tpPoint1	点 P_1 です.	*tpPoint2	点 P_2 です.	*dpArea	3点 P_0 , P_1 , P_2 の作る三角形の符号付き面積が格納されます.
*tpPoint0	点 P_0 です.								
*tpPoint1	点 P_1 です.								
*tpPoint2	点 P_2 です.								
*dpArea	3点 P_0 , P_1 , P_2 の作る三角形の符号付き面積が格納されます.								
戻り値	ありません.								
留意事項	特にありません.								

eyemClp2dCircleThreePoints

機 能 3点を通る円

形 式 `#include "eyemLib.h"`
`int eyedClp2dCircleThreePoints (EyemOcsDXY *tpPoint1,`

`EyemOcsDXY *tpPoint2, EyemOcsDXY *tpPoint3,`
`EyemOcsDXYR *tpCircle);`

解 説 3点 P_1, P_2, P_3 を通る円 $(x-a)^2 + (y-b)^2 = r^2$ を求めます. なお, 3点は三角形を作る位置になければいけません.

引 数 `*tpPoint1` 点 P_1 です.
`*tpPoint2` 点 P_2 です.
`*tpPoint3` 点 P_3 です.
`*tpCircle` 3点 P_1, P_2, P_3 を通る円のパラメータ (中心座標 (a, b) および半径 r) が格納されます.

戻り値 エラー報告です.

FUNC_OK	正常終了
FUNC_CANNOT_CALC	計算が不可 (3点が三角形を作らない)

留意事項 特にありません.

eyemClp2dCircleTwoPoints

機 能 2点を直径の両端とする円

形 式 `#include "eyemLib.h"`
`void eyedClp2dCircleTwoPoints (EyemOcsDXY *tpPoint1, EyemOcsDXY *tpPoint2,`
`EyemOcsDXYR *tpCircle);`

解 説 2点 P_1 , P_2 を直径の両端とする円 $(x-a)^2 + (y-b)^2 = r^2$ を求めます.

引 数 `*tpPoint1` 点 P_1 です.
`*tpPoint2` 点 P_2 です.
`*tpCircle` 2点 P_1 , P_2 を直径の両端とする円のパラメータ(中心座標 (a, b) および半径 r) が格納されます.

戻り値 ありません.

留意事項 特にありません.

eyemClp2dIntersectionLineAndCircle

機 能 直線と円の交点

形 式

```
#include "eyemLib.h"

int      eyemClp2dIntersectionLineAndCircle ( EyemOcsDABC *tpLine,
                                              EyemOcsDXYR *tpCircle, EyemOcsDXY *tpPoint1,
                                              EyemOcsDXY *tpPoint2, );
```

解 説 直線 L と円 C の二つの交点 P_1 と P_2 を求めます.

引 数

*tpLine	直線 L です.
*tpCircle	円 C です.
*tpPoint1	直線 L と円 C の交点 P_1 が格納されます.
*tpPoint2	直線 L と円 C の交点 P_2 が格納されます.

戻り値 エラー報告です.

FUNC_OK	正常終了
FUNC_CANNOT_CALC	計算が不可(直線 L と円 C が交わらない場合を含む)

留意事項 特にありません.

eyemClp2dTangentPointToCircle

機 能 指定点から円に引いた接線と接点

形 式

```
#include "eyemLib.h"

int      eyemClp2dTangentPointToCircle ( EyemOcsDXY *tpPoint,
                                           EyemOcsDXYR *tpCircle,
                                           EyemOcsDABC *tpTangent1, EyemOcsDXY *tpContact1,
                                           EyemOcsDABC *tpTangent2, EyemOcsDXY *tpContact2 );
```

解 説 指定された点 P から円 C へ引いた二つの接線 L_1 と L_2 およびそれぞれの接点 P_1 と P_2 を求めます.

引 数

*tpPoint	点 P です.
*tpCircle	円 C です.
*tpTangent1	点 P から円 C へ引いた接線 $L_1 : a_1x + b_1y + c_1 = 0$ が格納されます. なお, 法線ベクトル (a_1, b_1) は単位ベクトルとなっています.
*tpContact1	接線 L_1 の接点 P_1 が格納されます.
*tpTangent2	点 P から円 C へ引いた接線 $L_2 : a_2x + b_2y + c_2 = 0$ が格納されます. なお, 法線ベクトル (a_2, b_2) は単位ベクトルとなっています.
*tpContact2	接線 L_2 の接点 P_2 が格納されます.

戻り値 エラー報告です.

FUNC_OK	正常終了
FUNC_CANNOT_CALC	計算が不可

留意事項 特にありません.

eyemClp2dClosestToCircle

機 能 指定点から円周への最近点とその距離

形 式

```
#include "eyemLib.h"

int      eyemClp2dClosestToCircle ( EyemOcsDXY *tpPoint, EyemOcsDXYR *tpCircle,
                                     EyemOcsDXY *tpClosest, double *dpDist );
```

解 説 指定された点 P から円 C の円周への最近点とそこまでの距離を求めます.

引 数

*tpPoint	点 P です.
*tpCircle	円 C です.
*tpClosest	点 P から円 C の円周への最近点が格納されます.
*dpDist	点 P と最近点との距離が格納されます.

戻り値 エラー報告です.

FUNC_OK	正常終了
FUNC_CANNOT_CALC	計算が不可 (円 C が円でない)

留意事項 特にありません.

改訂履歴

Version No.	内 容
1.0	<ul style="list-style-type: none"> • 新規発行
1.1	<ul style="list-style-type: none"> • 3関数 (eyemClp2dLinePointAndSlope, eyedClp2dTranslationOfLine および eyedClp2dCenterLineOfTwoLines) の追加. • 項目順の入れ替え.