VGRModeler

0.8.1

作成： Doxygen 1.6.1

Tue Aug 9 17:13:48 2011

Contents

1 データ構造索引 **1**

1.1 データ構造 1

2 ファイル索引 **3**

2.1 ファイル一覧 3

3 データ構造 **5**

3.1 構造体 InsideCylinder 5

3.1.1 構造体 5

3.1.1.1 pointNumber 6

3.2 構造体 LinePoint 7

3.2.1 説明 7

3.2.2 構造体 7

3.2.2.1 x 7

3.2.2.2 y 7

3.2.2.3 z 7

3.3 構造体 ModelParameter 8

3.3.1 説明 8

3.3.2 構造体 8

3.3.2.1 height 8

3.3.2.2 label 8

3.3.2.3 r 8

3.3.2.4 x 8

3.3.2.5 y 9

3.3.2.6 z 9

3.4 構造体 OptionParameter 10

3.4.1 構造体 . . . 10
3.4.1.1 inFileName . . . 10
3.4.1.2 outFileName . . . 10
3.5 構造体 RotationAngle . . . 11
3.5.1 説明 . . . 11
3.5.2 構造体 . . . 11
3.5.2.1 rx . . . 11
3.5.2.2 ry . . . 11
3.5.2.3 rz . . . 11
3.6 構造体 SquareParameter . . . 12
3.6.1 説明 . . . 12
3.6.2 構造体 . . . 12
3.6.2.1 squarePoint . . . 12
3.7 構造体 Vector . . . 13
3.7.1 説明 . . . 13
3.7.2 構造体 . . . 13
3.7.2.1 x . . . 13
3.7.2.2 y . . . 13
3.7.2.3 z . . . 13

4 ファイル **15**

4.1 Model.h . . . 15
4.1.1 列挙型 . . . 16
4.1.1.1 FrameColor . . . 16
4.2 ModelInformation.cpp . . . 17
4.2.1 関数 . . . 17
4.2.1.1 writeCommandInfo . . . 18
4.2.1.2 writeCylinderInfo . . . 18
4.2.1.3 writeGravityInfo . . . 18
4.2.1.4 writeVertex . . . 18
4.3 ModelInformation.h . . . 19
4.3.1 関数 . . . 20
4.3.1.1 writeCommandInfo . . . 20
4.3.1.2 writeCylinderInfo . . . 20

4.3.1.3 writeGravityInfo 20
4.3.1.4 writeVertex 20
4.4 trans.cpp 21
4.4.1 関数 21
4.4.1.1 apply_3D_rotation 21
4.4.1.2 quaternion_rotation 22
4.4.1.3 translate_point 22
4.5 trans.h 23
4.5.1 関数 23
4.5.1.1 translate_point 23
4.6 VGRModeler.cpp 24
4.6.1 関数 25
4.6.1.1 calcNormal 25
4.6.1.2 commandCheck 25
4.6.1.3 createCircle 25
4.6.1.4 main 25
4.6.1.5 outBox 26
4.6.1.6 outCylinder 26
4.6.1.7 parallelSquare 26
4.6.1.8 parseOption 26
4.6.1.9 pointTranslation 26
4.6.1.10 setBox 26
4.7 VGRModeler.h 27
4.7.1 マクロ定義 27
4.7.1.1 EPS 27
4.8 VRMLWriter.cpp 28
4.8.1 関数 29
4.8.1.1 pointDefine 29
4.8.1.2 pointDefineEnd 29
4.8.1.3 writeAxisModel 29
4.8.1.4 writeNormalModel 29
4.8.1.5 writeVRML_Header 29
4.8.1.6 writeVRML_Intro 29
4.8.1.7 writeVRML_Text 29

4.8.1.8 writeWireModelEnd 29
4.8.1.9 writeWireModelStart 29
4.9 VRMLWriter.h 30
4.9.1 関数 31
4.9.1.1 pointDefine 31
4.9.1.2 pointDefineEnd 31
4.9.1.3 writeAxisModel 31
4.9.1.4 writeNormalModel 31
4.9.1.5 writeVRML_Header 31
4.9.1.6 writeVRML_Intro 31
4.9.1.7 writeVRML_Text 32
4.9.1.8 writeWireModelEnd 32
4.9.1.9 writeWireModelStart 32

Chapter 1

データ構造索引

1.1 データ構造

データ構造の説明です。

InsideCylinder 5
LinePoint (点列格納用構造体) 7
ModelParameter (入力されたモデルのパラメーター) 8
OptionParameter 10
RotationAngle (モデルの回転角) 11
SquareParameter (直方体の 1 面) 12
Vector (ベクトル用) 13

Chapter 2

ファイル索引

2.1 ファイル一覧

これはファイル一覧です。

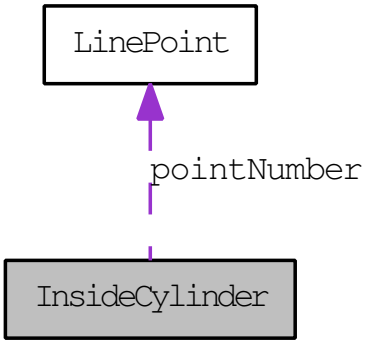
Model.h 15
ModelInformation.cpp 17
ModelInformation.h 19
trans.cpp 21
trans.h 23
VGRModeler.cpp 24
VGRModeler.h 27
VRMLWriter.cpp 28
VRMLWriter.h 30

Chapter 3

データ構造

3.1 構造体 InsideCylinder

`#include <Model.h>`InsideCylinder のコラボレーション図



変数

- LinePoint pointNumber [360]

3.1.1 構造体

3.1.1.1 LinePoint InsideCylinder::pointNumber[360]

この構造体の説明は次のファイルから生成されました:

- Model.h

3.2 構造体 LinePoint

点列格納用構造体

`#include <Model.h>`

変数

- double x
- double y
- double z

3.2.1 説明

点列格納用構造体

3.2.2 構造体

3.2.2.1 double LinePoint::x

3.2.2.2 double LinePoint::y

3.2.2.3 double LinePoint::z

この構造体の説明は次のファイルから生成されました:

- Model.h

3.3 構造体 ModelParameter

入力されたモデルのパラメーター

`#include <Model.h>`

変数

- double x
- double y
- double z
- double r
- double height
- char label [8]

3.3.1 説明

入力されたモデルのパラメーター

3.3.2 構造体

3.3.2.1 double ModelParameter::height

3.3.2.2 char ModelParameter::label[8]

3.3.2.3 double ModelParameter::r

3.3.2.4 double ModelParameter::x

3.3.2.5 double ModelParameter::y

3.3.2.6 double ModelParameter::z

この構造体の説明は次のファイルから生成されました:

- Model.h

3.4 構造体 OptionParameter

`#include <Model.h>`

変数

- char * inFileName
- char * outFileName

3.4.1 構造体

3.4.1.1 char* OptionParameter::inFileName

3.4.1.2 char* OptionParameter::outFileName

この構造体の説明は次のファイルから生成されました:

- Model.h

3.5 構造体 RotationAngle

モデルの回転角

`#include <Model.h>`

変数

- double rx
- double ry
- double rz

3.5.1 説明

モデルの回転角

3.5.2 構造体

3.5.2.1 double RotationAngle::rx

3.5.2.2 double RotationAngle::ry

3.5.2.3 double RotationAngle::rz

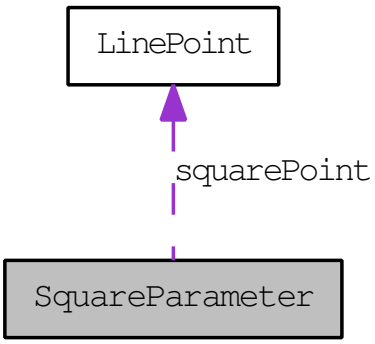
この構造体の説明は次のファイルから生成されました:

- Model.h

3.6 構造体 SquareParameter

直方体の 1 面

`#include <Model.h>`SquareParameter のコラボレーション図



変数

- LinePoint squarePoint [4]

3.6.1 説明

直方体の 1 面

3.6.2 構造体

3.6.2.1 LinePoint SquareParameter::squarePoint[4]

この構造体の説明は次のファイルから生成されました:

- Model.h

3.7 構造体 Vector

ベクトル用

`#include <Model.h>`

変数

- double x
- double y
- double z

3.7.1 説明

ベクトル用

3.7.2 構造体

3.7.2.1 double Vector::x

3.7.2.2 double Vector::y

3.7.2.3 double Vector::z

この構造体の説明は次のファイルから生成されました:

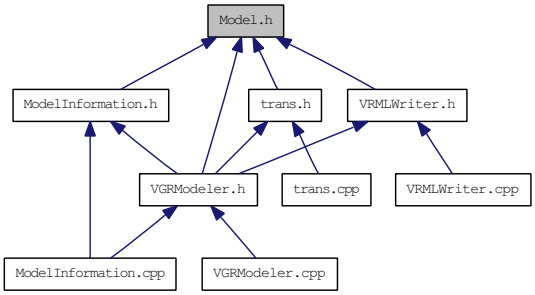
- Model.h

Chapter 4

ファイル

4.1 Model.h

このグラフは、どのファイルから直接、間接的にインクルードされているかを示しています。



データ構造

- struct LinePoint

  点列格納用構造体

- struct ModelParameter

  入力されたモデルのパラメーター

- struct SquareParameter

  直方体の *1* 面

- struct InsideCylinder

- struct RotationAngle
  モデルの回転角
- struct Vector
  ベクトル用
- struct OptionParameter

列挙型

- enum FrameColor { NOT_USE_COLOR, USE_COLOR }

4.1.1 列挙型

4.1.1.1 enum FrameColor

関数 writeWireModelEnd に使用する列挙体 ワイヤーフレームに色を付けている場合は USE_COLOR を選択

列挙型の値:

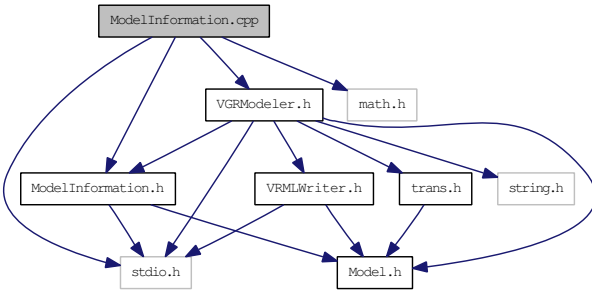
NOT_USE_COLOR

USE_COLOR

4.2 ModelInformation.cpp

```
#include <stdio.h>
#include <math.h>
#include "VGRModeler.h"
#include "ModelInformation.h"
```

ModelInformation.cpp のインクルード依存関係図



関数

- void writeCommandInfo (FILE *fp, const ModelParameter *commandParameter, const RotationAngle *rotation, const Vector *moveBuff)

  入力ファイルのコマンドを出力

- void writeGravityInfo (FILE *fp, const Vector *gravity)

  重心情報を出力

- void writeCylinderInfo (FILE *fp, const ModelParameter *circleParameter, const LinePoint *centerPoint, const Vector *normalVector)

  円筒の情報を出力

- void writeVertex (FILE *fp, const SquareParameter *faceParameter, const int nSquare)

  直方体の頂点情報を出力

4.2.1 関数

4.2.1.1 void writeCommandInfo (FILE * *fp*, const ModelParameter * *commandParameter*, const RotationAngle * *rotation*, const Vector * *moveBuff*)

入力ファイルのコマンドを出力

4.2.1.2 void writeCylinderInfo (FILE * *fp*, const ModelParameter * *circleParameter*, const LinePoint * *centerPoint*, const Vector * *normalVector*)

円筒の情報を出力

4.2.1.3 void writeGravityInfo (FILE * *fp*, const Vector * *gravity*)

重心情報を出力

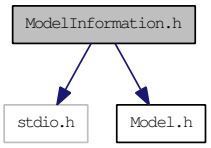
4.2.1.4 void writeVertex (FILE * *fp*, const SquareParameter * *faceParameter*, const int *nSquare*)

直方体の頂点情報を出力

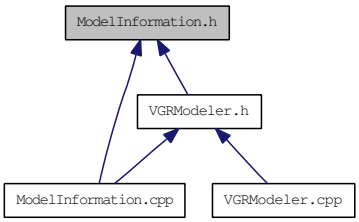
4.3 ModelInformation.h

```
#include <stdio.h>
#include "Model.h"
```

ModelInformation.h のインクルード依存関係図



このグラフは、どのファイルから直接、間接的にインクルードされているかを示しています。



関数

- void writeCommandInfo (FILE *fp, const ModelParameter *commandParameter, const RotationAngle *rotation, const Vector *moveBuff)

  入力ファイルのコマンドを出力

- void writeGravityInfo (FILE *fp, const Vector *gravity)

  重心情報を出力

- void writeCylinderInfo (FILE *fp, const ModelParameter *circleParameter, const LinePoint *centerPoint, const Vector *normalVector)

  円筒の情報を出力

- void writeVertex (FILE *fp, const SquareParameter *faceParameter, const int nSquare)

  直方体の頂点情報を出力

4.3.1 関数

4.3.1.1 void writeCommandInfo (FILE * *fp*, const ModelParameter * *commandParameter*, const RotationAngle * *rotation*, const Vector * *moveBuff*)

入力ファイルのコマンドを出力

4.3.1.2 void writeCylinderInfo (FILE * *fp*, const ModelParameter * *circleParameter*, const LinePoint * *centerPoint*, const Vector * *normalVector*)

円筒の情報を出力

4.3.1.3 void writeGravityInfo (FILE * *fp*, const Vector * *gravity*)

重心情報を出力

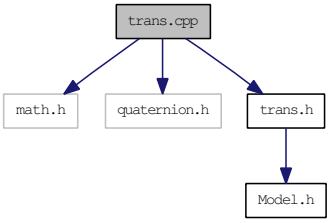
4.3.1.4 void writeVertex (FILE * *fp*, const SquareParameter * *faceParameter*, const int *nSquare*)

直方体の頂点情報を出力

4.4 trans.cpp

```
#include <math.h>
#include "quaternion.h"
#include "trans.h"
```

trans.cpp のインクルード依存関係図



関数

- void quaternion_rotation (quaternion_t q, const double radian, double ax, double ay, double az)

  任意軸周りの回転を表すクォータニオンを求める

- void apply_3D_rotation (double ∗px, double ∗py, double ∗pz, const double ax, const double ay, const double az, const double xoff, const double yoff, const double zoff, const double degree)

  クォータニオンで回転

- void translate_point (const RotationAngle ∗rotation, double xoff, double yoff, double zoff, double ∗returnX, double ∗returnY, double ∗returnZ)

  点を *3* 次元回転させる

4.4.1 関数

4.4.1.1 void apply_3D_rotation (double ∗ *px*, double ∗ *py*, double ∗ *pz*, const double *ax*, const double *ay*, const double *az*, const double *xoff*, const double *yoff*, const double *zoff*, const double *degree*)

クォータニオンで回転

4.4.1.2 void quaternion_rotation (quaternion_t *q*, const double *radian*, double *ax*, double *ay*, double *az*)

任意軸周りの回転を表すクォータニオンを求める

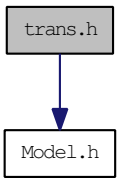
4.4.1.3 void translate_point (const RotationAngle * *rotation*, double *xoff*, double *yoff*, double *zoff*, double * *returnX*, double * *returnY*, double * *returnZ*)

点を 3 次元回転させる クォータニオン計算用

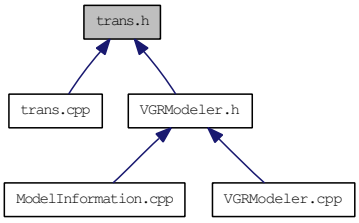
4.5 trans.h

```
#include "Model.h"
```

trans.h のインクルード依存関係図



このグラフは、どのファイルから直接、間接的にインクルードされているかを示しています。



関数

- void translate_point (const RotationAngle *rotation, double xoff, double yoff, double zoff, double *retx, double *rety, double *retz)

  クォータニオン計算用

4.5.1 関数

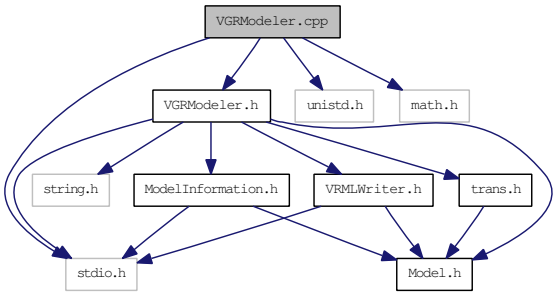
4.5.1.1 void translate_point (const RotationAngle * *rotation*, double *xoff*, double *yoff*, double *zoff*, double * *returnX*, double * *returnY*, double * *returnZ*)

クォータニオン計算用

4.6 VGRModeler.cpp

```
#include <stdio.h>
#include <unistd.h>
#include <math.h>
#include "VGRModeler.h"
```

VGRModeler.cpp のインクルード依存関係図



関数

- int parseOption (int argc, char **argv, OptionParameter &startOption)

  getopt 用設定

- int calcNormal (const LinePoint *point1, const LinePoint *point2, const LinePoint *point3, Vector *normalVector)

  法線ベクトル計算用関数

- void parallelSquare (const LinePoint *originalSquare, LinePoint *copySquare, const ModelParameter *boxParameter, int axis)

  対面している平行な面を作成

- void setBox (const ModelParameter *boxParameter, SquareParameter *faceParameter)

  6 面体の各面の *4* 座標をセットする関数

- void pointTranslation (LinePoint *movePoint, const Vector *moveBuff, const int maxTranslatePoint)

  点の移動

- void createCircle (LinePoint *circlePoints, const ModelParameter *circleParameter, const double radian, int flag)

円の点列初期値作成

- void outCylinder (FILE *fp, const ModelParameter *cylinderParameter, const RotationAngle *rotation, const Vector *moveBuff, int step, int nCircle)

  円筒出力用関数

- void outBox (FILE *fp, const ModelParameter *boxParameter, const RotationAngle *rotation, const Vector *moveBuff)

  直方体出力用関数

- int commandCheck (const int cylinderFlag, const int boxFlag, const ModelParameter *cylinderParameter, const ModelParameter *boxParameter)

  モデル指定のチェック

- int main (int argc, char *argv[ ])

  メイン関数

4.6.1 関数

4.6.1.1 int calcNormal (const LinePoint * *point1*, const LinePoint * *point2*, const LinePoint * *point3*, Vector * *normalVector*)

法線ベクトル計算用関数

4.6.1.2 int commandCheck (const int *cylinderFlag*, const int *boxFlag*, const ModelParameter * *cylinderParameter*, const ModelParameter * *boxParameter*)

モデル指定のチェック

4.6.1.3 void createCircle (LinePoint * *circlePoints*, const ModelParameter * *circleParameter*, const double *radian*, int *flag*)

円の点列初期値作成

4.6.1.4 int main (int *argc*, char * *argv*[ ])

メイン関数

4.6.1.5 void outBox (FILE * *fp*, const ModelParameter * *boxParameter*, const RotationAngle * *rotation*, const Vector * *moveBuff*)

直方体出力用関数

4.6.1.6 void outCylinder (FILE * *fp*, const ModelParameter * *cylinderParameter*, const RotationAngle * *rotation*, const Vector * *moveBuff*, int *step*, int *nCircle*)

円筒出力用関数

4.6.1.7 void parallelSquare (const LinePoint * *originalSquare*, LinePoint * *copySquare*, const ModelParameter * *boxParameter*, int *axis*)

対面している平行な面を作成

4.6.1.8 int parseOption (int *argc*, char ** *argv*, OptionParameter & *startOption*)

getopt 用設定

4.6.1.9 void pointTranslation (LinePoint * *movePoint*, const Vector * *moveBuff*, const int *maxTranslatePoint*)

点の移動

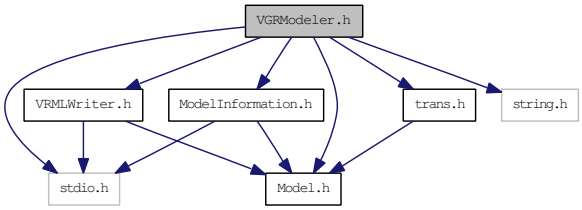
4.6.1.10 void setBox (const ModelParameter * *boxParameter*, SquareParameter * *faceParameter*)

6 面体の各面の 4 座標をセットする関数

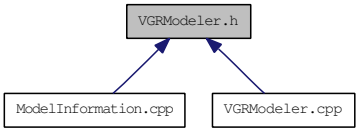
4.7 VGRModeler.h

```
#include <stdio.h>
#include <string.h>
#include "trans.h"
#include "Model.h"
#include "VRMLWriter.h"
#include "ModelInformation.h"
```

VGRModeler.h のインクルード依存関係図



このグラフは、どのファイルから直接、間接的にインクルードされているかを示しています。



マクロ定義

- #define EPS 1.0e-10

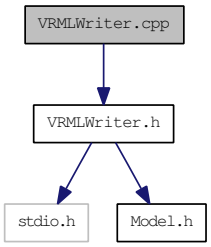
4.7.1 マクロ定義

4.7.1.1 #define EPS 1.0e-10

4.8 VRMLWriter.cpp

```
#include "VRMLWriter.h"
```

VRMLWriter.cpp のインクルード依存関係図



関数

- void writeVRML_Header (FILE *fp)
  VRML のヘッダ出力.
- void writeVRML_Intro (FILE *fp)
  VRML の入れ子の一番外側.
- void writeWireModelStart (FILE *fp)
  ワイヤーフレームモデルの出力、点の定義まで
- void pointDefine (FILE *fp, const LinePoint *defPoint)
  点の定義
- void pointDefineEnd (FILE *fp)
  点の定義を終了し、繋ぐ部分へ進める
- void writeWireModelEnd (FILE *fp, const int colorFlag)
- void writeVRML_Text (FILE *fp, const char outText[128], const double coordX, const double coordY, const double coordZ)
  VRML の文字描画.
- void writeAxisModel (FILE *fp, double axisLength)
  座標軸を *VRML* ファイルに挿入する関数
- void writeNormalModel (FILE *fp, const LinePoint *center, const Vector *normalVector, const double length, const int nFace)
  法線描画用

4.8.1 関数

4.8.1.1 void pointDefine (FILE ∗ *fp*, const LinePoint ∗ *defPoint*)

点の定義

4.8.1.2 void pointDefineEnd (FILE ∗ *fp*)

点の定義を終了し、繋ぐ部分へ進める

4.8.1.3 void writeAxisModel (FILE ∗ *fp*, double *axisLength*)

座標軸を VRML ファイルに挿入する関数

4.8.1.4 void writeNormalModel (FILE ∗ *fp*, const LinePoint ∗ *center*, const Vector ∗ *normalVector*, const double *length*, const int *nFace*)

法線描画用

4.8.1.5 void writeVRML_Header (FILE ∗ *fp*)

VRML のヘッダ出力.

4.8.1.6 void writeVRML_Intro (FILE ∗ *fp*)

VRML の入れ子の一番外側.

4.8.1.7 void writeVRML_Text (FILE ∗ *fp*, const char *outText*[128], const double *coordX*, const double *coordY*, const double *coordZ*)

VRML の文字描画.

4.8.1.8 void writeWireModelEnd (FILE ∗ *fp*, const int *colorFlag*)

立体出力終了 色をつけていない場合 = 0, つけている場合 = 1

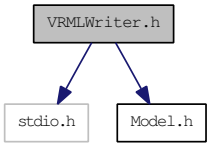
4.8.1.9 void writeWireModelStart (FILE ∗ *fp*)

ワイヤーフレームモデルの出力、点の定義まで

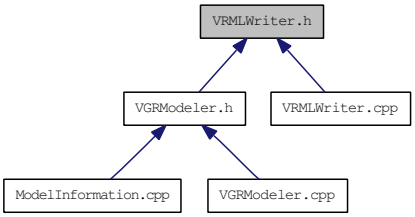
4.9 VRMLWriter.h

```
#include <stdio.h>
#include "Model.h"
```

VRMLWriter.h のインクルード依存関係図



このグラフは、どのファイルから直接、間接的にインクルードされているかを示しています。



関数

- void writeVRML_Header (FILE *fp)
  VRML のヘッダ出力.
- void writeVRML_Intro (FILE *fp)
  VRML の入れ子の一番外側.
- void writeWireModelStart (FILE *fp)
  ワイヤーフレームモデルの出力、点の定義まで
- void pointDefine (FILE *fp, const LinePoint *defPoint)
  点の定義
- void pointDefineEnd (FILE *fp)
  点の定義を終了し、繋ぐ部分へ進める
- void writeWireModelEnd (FILE *fp, const int colorFlag)

- void writeVRML_Text (FILE *fp, const char outText[128], const double coordX, const double coordY, const double coordZ)

  VRML **の文字描画**.

- void writeAxisModel (FILE *fp, double axisLength)

  座標軸を *VRML* **ファイルに挿入する関数**

- void writeNormalModel (FILE *fp, const LinePoint *center, const Vector *normalVector, const double length, const int nFace)

  法線描画用

4.9.1 関数

4.9.1.1 void pointDefine (FILE * *fp*, const LinePoint * *defPoint*)

点の定義

4.9.1.2 void pointDefineEnd (FILE * *fp*)

点の定義を終了し、繋ぐ部分へ進める

4.9.1.3 void writeAxisModel (FILE * *fp*, double *axisLength*)

座標軸を VRML ファイルに挿入する関数

4.9.1.4 void writeNormalModel (FILE * *fp*, const LinePoint * *center*, const Vector * *normalVector*, const double *length*, const int *nFace*)

法線描画用

4.9.1.5 void writeVRML_Header (FILE * *fp*)

VRML のヘッダ出力.

4.9.1.6 void writeVRML_Intro (FILE * *fp*)

VRML の入れ子の一番外側.

4.9.1.7 void writeVRML_Text (FILE * *fp*, const char *outText*[128], const double *coordX*, const double *coordY*, const double *coordZ*)

VRML の文字描画.

4.9.1.8 void writeWireModelEnd (FILE * *fp*, const int *colorFlag*)

立体出力終了 色をつけていない場合 = 0, つけている場合 = 1

4.9.1.9 void writeWireModelStart (FILE * *fp*)

ワイヤーフレームモデルの出力、点の定義まで