３．外部ファイル形式

３－１．概要

（1）ＬＳＳ－Ｓ形式とＬＳＳ－Ｇ形式

景観シミュレーション・システムでファイル入出力や通信に用いるデータは、独自のデータフォーマットであるＬＳＳフォーマット[補注)]となっており、適用事例や、共通性の高い景観構成要素を登録したデータベース等の蓄積保存には、この形式を用いている。

初版から、データ形式に関しては全く変更しておらず、1996 年に配布開始した初版のシミュレータで作成したデータを最新版で利用することができ、また最新版で作成したデータを初版で開くことができる。各分野で実務的に必要とされる各種のデータ形式とのデータ交換のためには、ファイル・コンバータを増補することにより対応してきた（第８章)。

三次元的な地物を表現するためのフォーマットには、ＬＳＳ－Ｓ（シーン）とＬＳＳ－Ｇ（ジオメトリ）がある。地物に固有の属性と、表示に必要な環境条件を別ファイルに分離している点に特徴がある。

どちらもテキスト･ファイルであり､エディタ等で開いて内部を解読することができる。開発に着手した 1993 年頃､データ形式の検討段階で､OpenGL をベースとした Open Inventor を参考としたため、その後開発された VRML とは親和性が高い。

記法は「;」（セミコロン）を記述の区切りとするＣ言語の記法となっている。一つのコマンドを複数行にまたがって記述することができる。ラベルの大文字小文字は区別する。ラベルを構成する文字はトークン、禁則文字以外すべて許される。

ラベルの途中で改行があっても、接続して一つの文字列として処理する。コメント行、改行、スペース、タブコードはロードの際に取り除かれ、保存する際には再現しない。Ver.2.09 においては、ラベルには ASCII コードから派生した、例えばシフトＪＩＳによる日本語などの２バイト系言語の使用を排除していないが、異なる言語のＯＳ上への可搬性は保証されない。UNICODE による２バイト系文字列を用いたファイルはサポートしていない。

リスト３－１：コマンドの基本形

```
(1)コメント行
#文字列
(2)ラベル付のコマンド
ラベル=コマンド(引数 1, 引数 2, 引数 3, .... );
(3)ラベルの無いコマンド
コマンド(引数 1, 引数 2, 引数 3, .... );
```

ＬＳＳ－Ｓ形式は、シーンファイルを構築するデータフォーマットで、ＬＳＳ－Ｇ形式で定義された地物を、二次元画像として一意的に表示するために必要となる各種環境条件（光源、時刻、視点位置等）の定義をしている。ＬＳＳ－Ｓ形式を構成する各コマンドに

ついては、３－２で解説する。

ＬＳＳ－Ｇは、環境に依存しない地物固有の性質（幾何学的形状、表面仕上などの各種属性）を記述するジオメトリファイルを構築するデータフォーマットである。また環境条件の一部である時間の関数として表現できる、地物の規則的な経年変化（時間の関数として表現できる形状や表面仕上げ）も、ＬＳＳ－Ｇの中で定義する。ＬＳＳ－Ｇ形式を構成する各コマンドについては、３－３で解説する。

ＬＳＳ－Ｓ形式とＬＳＳ－Ｇ形式に共通するコマンドについては、３－４で解説する。

通常の景観検討の作業は、まず地形や設計対象物などを定義するＬＳＳ－Ｇ形式のファイルを構築し、次に複数のＬＳＳ－Ｇファイル（計画案選択肢）を参照すると共に、視点場や光源条件などを設定するＬＳＳ－Ｓファイルを作成する手順で進められる。ＬＳＳ－Ｇファイルの構築に際しても、部品を定義するＬＳＳ－Ｇファイル群をまず作成し、次にこれらを配置し組み合わせることにより、より大きな、複雑な地物構成を表現する上位のＬＳＳ－Ｇファイルを作成する、という手順で作業が進められる（３－５）。

ＬＳＳ－Ｓファイル及びＬＳＳ－Ｇファイルは、通常は、本システムの各種機能を用いて画面操作を行い、結果をファイル保存することにより作成される。また、テキスト・エディタで作成・編集することも、あるいは、ファイル・コンバータや形状生成を伴うシミュレーションを行うアプリケーションの出力として作成することもできる。

ＬＳＳ－Ｓファイル及びＬＳＳ－Ｇファイルの入出力に関する機能は、スタティック・リンク・ライブラリの一つである IP.lib により提供している（４－２）。

なお、編集ダイアログ等においてシステムが自動的に生成するラベルには、パターンが存在する（例えば、時間のラベルは「T00X」といったスタイル）。しかし、ユーザーが手入力するラベル、あるいはコンバータ等が生成するラベルが、この慣習に従う必要は全くない。以下のような人を惑わす記述であっても、ラベル名に関わりなく、用いられたコマンドの規約に従って、整合性がある限り正しく処理される。

リスト３－２：紛らわしいが許されるコマンドの例

```
MODEL1 = COLOR(0,0,3);
TEXTURE1 = VERTEX(COLOR1,COLOR2,COLOR3);
MATERIAL1 = FACE(TEXTURE1,TEXTURE2,TEXTURE3);
COLOR = FILE(TRANSPARENT.GEO);
MESSAGE = GROUP(URGENT);
```

（２）各種画像ファイル

景観シミュレーションを行うために、現況写真などの画像を背景・前景として使用する場合、及び樹木や構造物の仕上げ面を画像で表現するためのテクスチャに各種ファイルを使用する。本システムの初期のバージョンにおいては、ＳＧＩ形式を標準としていたが、その後デジタルカメラや、インターネットによる画像交換の普及に伴い、画像ファイル形

式が普及したため、これらも利用できるようにした。これらについては、３－８で解説する。ファイルの入出力処理には、既存のオープン・ソースを使用したもの(SGI, JPEG)、Windowsに関連したAPI関数を使用したもの(BMP)、仕様書などから独自に作成したもの(その他）がある。

（3）マテリアル・ファイル

ＬＳＳ－Ｇファイルを構成する各種地物（地形や、住宅・社会資本等）の材質を記述するためには、ＬＳＳ－Ｇファイルの中で、各種オブジェクトの色彩やテクスチャを直接記述することもできるが、頻繁に使用される各種材料に関して、あらかじめ共通のマテリアル・ファイルに、各種材料（鉄、コンクリート、木材、塗料、水面、芝生等）毎に、表示に必要な光学的属性を定義しておき、これを間接的に参照することにより、ＬＳＳ－Ｇファイルの中での材質に関する記述をシンプルにすることができる。

各種マテリアルには、色彩やテクスチャ以外に、鏡面反射率、輝度、テクスチャと、その経年変化を記述することができる。将来、グラフィクスの表現技術が高度化した場合であっても、マテリアル・ファイルによる定義を詳細化することにより、ＬＳＳ－Ｇファイルは変更修正することなしに対応することができることを配慮している。

マテリアル・ファイルの形式については、３－６で解説する。マテリアル・ファイルの読み込みは、DBILライブラリ（４－２(3)参照）のdbLoadMaterial関数により行う。マテリアル・ファイルを構築するための編集ダイアログ等は、Ver.2.09の段階においてもまだ存在しない。テキスト・エディタにより入力・編集している。外部関数として実装されたファイル・コンバータ VRML2LL.exe の中に、変換元のファイルにおいて適用された鏡面反射率や発光体等の属性をマテリアル・ファイル形式で出力する機能が組み込まれている。

（4）データベース関連ファイル

景観シミュレーション・システムの操作に際して使用する、①事例、②構成要素、及び③材料の３種類データベースの登録データを記述するためのファイル(com.txt)と、分類体系を記述するためのファイル(XXX.cls)を使用している。これらについては、１４－２、１４－３で解説する。

Com.txtについては、DBILライブラリ（４－２(3)参照）に入出力の機能を用意している(dbMakeData関数、dbSave関数)。また、３種類のデータベースに対応する検索機能(yuu.exe、kou.exe、zai.exe)のほかに、入力のために、専用エディタ editor.exe を用意している。

（5）環境設定ファイル

景観シミュレーション・システムの操作環境を定義するために、環境設定ファイル(標準名kdbms.set)を用意している。これについては第12章で解説している。

この他に、ステレオ表示の条件を記述する小さなファイル(eye_param.set)がある（７－

４参照)。

（６）エラーメッセージ定義ファイル

エラーメッセージは、初期から外部ファイル err_msg.txt により定義してきた。Ver.2.09 においては、これを継承し、言語別にファイルを用意することとしている。詳細は、３－９で解説する。

（７）ヘルプ・ファイル

ヘルプ・ファイルは、テキスト・ファイルであり、特に形式を定めていない。表示された状態でユーザーが編集・追記し、上書き保存することができる。拡張子は、XXX.txt である。ファイル名称 XXX は通常、基幹部分およびプラグインＤＬＬの各ダイアログの名称を用いている。Ver.2.09 においては、多言語の管理のために、言語別のディレクトリに置くと共に、ファイル名称を XXX.yy.txt としている(yy は言語コード、例えば、haichi.ko.txt は韓国語による配置操作のヘルプ)。表示処理について１１－６(4)で解説する。

（８）選択項目定義ファイル

システムの拡充に伴い、プログラムを修正することなく選択肢を拡充できるように、いくつかのコントロール・ファイルを用意している。例えば、頻繁に使用するテクスチャを登録する autotex.set、ユーザーが追加することができるパラメトリックな形状を生成する外部関数を登録する ext.tab、道路断面を登録する roadsec.set や河川断面を登録するファイル riversec.set 等である。これらについては、３－１０において解説する。

（９）多言語対応のための訳語格納ファイル

基幹部分(sim.exe)の実行に際して、使用する言語の変更が行われた場合に、メニュー項目や、各種ダイアログ中のコントロール（操作ボタン等）の表示に用いる文字列を各国語の訳語に変更するために、訳語格納ファイルを xml 形式で用意する必要がある。

外部関数およびプラグインＤＬＬに関しても、多言語に対応する場合には、それぞれについて、同様の xml 形式の訳語格納ファイルを用意する。

プログラムが改良・修正され、必要となる表示文字列が増加した場合には、基幹部分の多言語対応機能により、リソースファイルと xml のタイムスタンプを比較し、自動的にバージョン管理を行っている。これらについては、第 11 章で解説する。

（１０）その他の一時的ファイル

また、関連する実行形式の間でデータを交換するために一時的に作成されるファイルがいくつか存在する。これらについては、３－１１において解説する。

（補注：LSS は Landscape Simulation の略である。開発初期の構想の中で、景観データベースを構築するにあたり、

優良景観事例にＬＳＳ－Ｓ、景観構成要素データベースにＬＳＳ－Ｇ、景観材料データベースにマテリアルを登録する、という概念が存在していたが、開発はそのようには進展しなかった。現在は、いずれのデータベースにおいても、ＬＳＳ－Ｇファイルが登録されている。ソースコード中の変数名等に一部そのような概念の名残がある。）

Ver.2.09 のインタープリタにおいては、コマンドに関して、以下のように処理している。

(1) 名称の前後のスペース、タブコード(0x07)、改行文字(0x0d)は無視する。

(2) 名称に２バイト系の文字が含まれていても、処理を行う。

(3) コマンドのないトークン「;」：

エラー（コマンドがない）

(4) 最後のトークン「;」の後の記述（トークンの無いファイル終了）：無視

(5) トークン「;」の間にスペース、タブ、改行しかない場合：無視

(6) 名称に含まれる日本語のホワイトスペース：文字列の一部として処理

(7) 名称に含まれるスペース：詰めて処理(「a b c d e」→「abcde」

(8) 名称が”引用符”で囲まれていた場合：文字列に含まれるスペースを詰めない

(9) 名称に”引用符”で囲まれた部分が複数あった場合：連結。”1 “ “ 2 “ “3” →「 1 2 3」

(10) 始まりの欠落した“引用符”:無視。「 abcd”」 →「abcd」

(11) 文字列の途中にある”引用符” :「ab”cd”ef」→「abcdef」

(12) 名称が数字であった場合：名称として処理。 [例]0 = GROUP(); は受け付ける。

(13) 数値の中にスペースがあった場合 3. 14 →3.14

(14) 数値が引用符「”」で囲まれていた場合：数値として解釈

(15) 数値が引用符「”」で囲まれていて、その中にスペースが含まれる場合：スペース以前までを評価

(16) 数値が引用符「”」で囲まれていて、その冒頭にスペースがあった場合：スペースを無視(“ 3.0” →3.0)

(17) シングル引用符「’」:文字列の一部として解釈

(18) 小数点以上の省略 .3 → 0.3 として解釈

(19) 小数点以下の省略 3. →3.0 として解釈

(20) 小数点のみ . →0.0 として解釈

(21) 小数点が複数あった場合：「1.2.3.4.5」→1.2

(22) 指数表現：可 1e3 →1000.0

(23) 引数の数値が省略された場合（,,,）→(0.0, 0.0, 0.0, 0.0)として解釈

(24) コマンドとして定義されていない文字列があった場合：エラー（コマンドが未定義）

(25) LSS-S ファイルの中に、LSS-G 系列のコマンドが使用された場合：

解釈するが表示には反映しない。エラーがあればメッセージを作成。

(26) LSS-G ファイルの中に、LSS-S 系列のコマンドが使用された場合：

解釈するが表示には反映しない。エラーがあればメッセージを作成。

(27) コマンド文字列の中に空白があった場合：詰めて解釈 「G 1 = FI LE(SA MPLE);」→「G1=FILE(SAMPLE);」

(28) コマンド文字列の中に改行があった場合：詰めて解釈

(29) 引数部() が欠落した場合：エラー [例] G1 = FILE;

(30) 引数の括弧閉じ「)」が欠落した場合：エラー

(31) 引数が不足する場合：エラー　但し、コマンドにより、省略を許容する場合がある。

(32) 引数が超過する場合：インタープリタでは全て拾い処理に渡す。処理は各コマンドによる。

(33) 定義されていない引数（文字列）が引用された場合：エラー。前方参照は行わない。

３－２．ＬＳＳ－Ｓコマンド

ＬＳＳ－Ｓ（Landscape Simulation-Scene）ファイルは、画面の背景画像、前景画像、表示する三次元モデル（３－３で解説するＬＳＳ－Ｇ形式のファイル)、経過年数（建築後年数)、視点、太陽位置などの光源、及びその他の表示に係る効果を記述するファイルである。テキスト・ファイルであり、インタプリタ（ＩＰ）により解釈され、シーンに関するデータを扱うＳＭＬライブラリ（４－２参照）を介してメモリ上に格納される。

ＬＳＳ－Ｓの拡張子は".scn"である。ＬＳＳ－Ｓファイルは、ＬＳＳ－Ｓコマンドにより記述されている。

開発初期においては、ＬＳＳ－ＳおよびＬＳＳ－Ｇのコマンドをオペレータ（プログラマ）が１行ずつコマンドとして入力し、テスト・デバッグを行っていたようであるが、システムとして完成したシステムの中には、１行ずつ入力するコマンドラインのコンソールなどのインターフェースは残されていない。従って、通常は外部ファイル（ＬＳＳ－Ｓ、ＬＳＳ－Ｇ)の中に記述されたコマンドとして､インタープリタによりバッチ処理される。

ただし、ファイル経由で１行ずつハンドシェークしながら入力・実行する機能は残されており､「成熟都市シミュレータ」はこの機能を利用して、市街地の変化（建物の追加・削除）をシミュレーションし、その結果を、リアルタイムで景観シミュレータに出力・表示するインターフェースを用いている。逐次コマンドが処理され変化する表示は、アニメーションとなる。

リスト３－３：ＬＳＳ－Ｓファイルに用いられるコマンドの一覧

```
TIME
CAMERA
LIGHT
LIGHTGROUP
EFFECT
EFFECTGROUP
MODEL
IMAGE
SCENE
OUTPUT_SCENE
```

（１） 時間 TIME

時間データを作成する。時間は、日数を単位として、浮動小数点値で示す。景観シミュレータは、現在の時間データを保持しており、表示に際しては、材料の経年劣化や熟成など、時間に依存して変化する性質を有する地物に関して、このシステム時間を適用した表示を行う。ＬＳＳ－Ｓ形式の記述の中では、後述するSCENEコマンドの中で、時間を数値で直接指定するのではなく、TIMEコマンドで定義されたラベルを使用する。

[記法]　時間名称＝TIME(日数);

　日数　　経年変化を表す（浮動小数点値）

[例]　　T1=TIME(730.0);

①日数は、負であっても構わない。

②日数が空欄または数値として解析できない文字列の場合にはゼロとする。

③引数が２以上あった場合には、最初の引数を日数として解釈する。

④同一ラベルで再定義が行われた場合、パラメータ（日数）だけを更新する。この更新は、そのラベルを用いて時間を定義している全てのシーンに遡って適用される。

⑤時間が設定、修正された場合には、時間に依存するモデルの再構築が行われる。

（２）カメラ情報 CAMERA

視点データを作成する。三次元で表現された地物を、二次元のディスプレイに透視図として表示（レンダリング）するためには、視点位置が必要である。景観シミュレータにはユーザーが視点位置を移動することにより様々な位置・視角で地物を表示する機能が備わっているが、地物自体には、「見どころ」の視点場などを保存する機能はなく、ＬＳＳ－Ｓで記述する。

[記法]視点名称＝CAMERA(EYE_X, EYE_Y, EYE_Z,
　　CENTER_X, CENTER_Y, CENTER_Z,
　　UP_X, UP_Y, UP_Z,
　　FOVY, ASPECT, ZNEAR, ZFAR);

EYE_X	視点のＸ座標値
EYE_Y	視点のＹ座標値
EYE_Z	視点のＺ座標値
CENTER_X	注視点のＸ座標値
CENTER_Y	注視点のＹ座標値
CENTER_Z	注視点のＺ座標値
UP_X	天頂方向ベクトルのＸ座標値

UP_Z	天頂方向ベクトルのＺ座標値
FOVY	焦点距離
ASPECT	アスペクト比
ZNEAR	視点からの見え始める距離
ZFAR	視点からの見えなくなる距離

パラメータはＯpenＧＬの gluLookAt()及び gluPerspective()関数に適用する。

カメラ位置は、視点位置（カメラの位置)、注視点（中心に見えるポイント)、アップベクトル（カメラの視点－注視点を軸とする回転角）を含む。注視点は、視点から注視点に結ぶ半直線上のどこに移動しても表示に影響は無いが、景観シミュレータにおける視点移動における「回転」は、注視点を中心として視点位置を回転させているため、以後の視点移動の動作に違いが顕われる。

F（焦点距離）は、35mm カメラの焦点距離に対応するパラメータである。OpenGL で用いる視野角 fovy とは、以下の換算式で関連づけている。

fovy = atan(17.5 / F) * 114.591559026164641753596309628 2

F = 17.5 / tan(fovy * 0.008726646259971647884618453842443 06

焦点距離を小さくとると、広角の表示となり、同じ対象物は中心に小さく描画される。大きくするとズームアップ状態となり、遠くの物体の局部が大きく表示される。

ASPECT は、表示画面の横／縦比率で、通常は画面のプロポーションに合わせる。画面のサイズ変更に際して、この値を正しく追随する必要がある。

zNear－zFar は、表示において、手前にある地物によってその裏側の地物が隠される関係を計算するＺバッファの有効範囲（距離）を定義している。近くの物体、遠くの物体をあえて表示しないために､意図的に設定する場合がある｡全ての地物を表示する場合には、zNear を十分小さく、また zFar を十分大きく設定しなければならない。しかし、グラフィックス能力が低いマシン上では､zNear-zFar のレンジを不必要に大きくとると､前後判定の精度が下がり、辺が鋸線状に表示されるような異常が生じる場合がある。

（３）光源 LIGHT, LIGHTGROUP

光源を定義する。光源は、光源ユニットと、それを組み合わせた光源グループを用いて定義する。ひとつのシーンに対して、ひとつの光源グループが定義される。ひとつの光源グループは最大８の光源から成る(三次元表示に使用している OpenGL ライブラリの制約条件に由来)。

（光源ユニット）

[記法]　光源ユニット名称＝LIGHT(光源タイプ, X, Y, Z, W, R, G, B);

光源タイプ	光源タイプの種類

一般の光源　　：０
緯度経度、月日、時分から自動計算した４光源の場合：
　　太陽光源　　：１
　　副光源太陽の反対側　：２
　　副光源東側　　：３
　　副光源西側　　：４

X　　位置のＸ座標値

Y　　位置のＹ座標値

Z　　位置のＺ座標値

W　　位置のＷ座標値

(X, Y, Z, W)は同次座標。４の値が１コマンドでOpenGLに渡される。その際に、Wが0.0であれば、座標点から原点に向かう平行光源が、また、Wが1.0であれば、座標点から八方に発せられる点光源が定義される。

R　　色：RED

G　　色：GREEN

B　　色：BLUE

(R, G, B)は色で、0.0≦≦1.0とする。

Ver.2.09において、
①パラメータの数が不足していた場合：エラー。そのラベルを用いたカメラを生成するが、内容は保証されない。
②パラメータの数が過剰の場合：余分なパラメータ記述を無視する。
③同じラベルを用いて、光源が再定義された場合、その光源を含む光源グループを用いた全てのシーンに遡って適用する。

（光源グループ）

［記法］　光源グループ＝LIGHTGROUP(光源ユニット名称, 光源ユニット名称, ...);
光源ユニット名称　　定義済みの光源ユニットの名前を参照する
光源グループを作成する。
　最大８個まで光源ユニットを登録することができる

Ver.2.09において、
①光源の数がゼロだった場合：デフォルト光源を一つ設定
②光源の数が８以上定義された場合：警告を出し、８までを採用
③光源グループの定義において、光源に同じ名称が繰り返し使用された場合：同じ名称の光源ユニットを複数生成する（危険な状況）。

［例］L1 = LIGHTGROUP(L1,L1,L1);

④同じラベルを用いた光源グループの再定義はエラーとして処理する。既に定義済みの光源グループは影響を受けない。但し、それを構成する光源ユニットの内部のパラメータは、光源ユニットの再定義によって変化しうる。

（４）効果 EFFECT, EFFECTGROUP

OpenGL で使用することのできる効果等を記述する。シーンを定義するために必須ではなく、何も定義されていなければ設定されない。効果タイプの後の引数の数に制限はない。現在の段階までで実装している機能には次のようなものがある。

```
EFFECT(STDFOG,int 種別);  種別は、霧の濃度を示す１～５の値
EFFECT(GRID, float 間隔);  平面図等の表示におけるグリッド
EFFECT(MEASURE);    平面図等の表示におけるメジャー
EFFECT(STEREO, float 眼距, int SWAP); ステレオ表示モード
EFFECT(SHADOW, int mode, float 影の長さ);  影の表示を行う。
EFFECT(ANTIAREASING, int value);アンチエリアシング
EFFECT(TRIQUAD); 三角四角高速表示
EFFECT(DISPLIST); ディスプレイ・リストの使用
EFFECT(SIZE,x,y); 画面サイズ（背景・前景画像が存在する場合）
EFFECT(VISUAL,x1,y1,x2,y2,NAME); 可視範囲解析結果の表示
```

［記法］ 効果ユニット名称＝EFFECT(効果タイプ、...);
効果タイプ　　効果の種類の指定
効果ユニットを作成する。

定義した EFFECT は、EFFECTGROUP コマンドの中で指定され、その EFFECTGROUP があるシーンに適用されて初めて意味を持つ。EFFECTGROUP から引用されなかった EFFECT や、シーンに引用されなかった EFFECTGROUP から引用された EFFECT は、システム状態や表示には反映されず、ＬＳＳ－Ｓ形式のファイルのロードが終了し、インタープリタがリセットされた時点で、メモリから削除・解放される。

()の中に記述される効果タイプおよび引数の数、内容関して、形式が守られている限りインタープリタによる制約はなく、エラー処理も行わない。形式的な解析結果が、シーン管理ライブラリ(SML)に渡される。それが意味のないものであれば、実際の動作はない。表示段階で未定義の EFFECT コマンドが検出されれば、その段階でエラー表示が行われるが、データ自体は削除されず、再保存に際してもＬＳＳ-Ｓファイルに反映される。一方、ユーザーが「シャッター」を操作して、ひとつのシーンを更新した場合には、システムが現在の表示状態から自動的に EFFECTGROUP を生成するため、そこに元々あった未定義の EFFECT コマンドは記録されるシーンには含まれない。言いかえると、読み込むファイルに未定義の EFFECT コマンドが含まれていた場合、シャッター操作によるデータの更新が行われない限り、再保存される LSS-S ファイルにもそれは継承される。

例：　E0 = EFFECT();　→空のエフェクトが作成される。動作上意味がない。

E1 = EFFECT(NONSENSE,,,,,,,,,,,,,nonsense,,,,,);
->形式的に解釈し、システムに渡される。

[記法] 効果グループ＝EFFECTGROUP(効果ユニット名称、効果ユニット名称、...);
効果ユニット名称　　効果を指定する
効果グループを作成する。複数の効果を組み合わせて一つのシーンに対して指定することができる。効果が一つしかない場合であっても、EFFECTGROUP コマンドを通じてしか、シーンに対して効果を指定することはできない。

Ver.2.07 以前においては、無視される。必要があれば、シーン切換時の挨拶音や、表示中の BGM 等を指定することに利用できるかも知れない。後述の GROUP 定義における属性と同様、将来拡張性のあるコマンドである。

Ver.2.09 における EFFECT は、全てシステムが自動的に生成する。シーンの編集において、あるシーンをシャッターで確定する時点で、画面に設定されていた表示条件を EFFECT コマンドの形式を用いて、そのシーンのデータに対して自動的に記録する。表示するシーンを切り換えた時点や、LSS-S ファイルをロードした時点で、表示すべきシーンに EFFECT コマンドが設定されていた場合には、それに対応する表示方法に切り替える。EFFECT が設定されていなかった場合には、全ての EFFECT に関連した項目を、デフォルト設定とする。

[Ver.2.09 インタープリタの処理]

①インタープリタは、基本的な文法を守った EFFECT 記述を、意味の有無にかかわらず解釈して、結果をシーン管理ライブラリ(４－２(1)参照) に引き渡す。あるシーンを表示する段階で、未定義の記述が検出されれば、そこでエラーを表示する。定義されたコマンドで引数等に不適切な値が設定されていた場合に、エラー処理するか、適当な値を仮定して処理を続行するか等は、それぞれのエフェクトの処理機構に依存している。

②保存段階では、インタープリタは、シーン毎の効果構造体として渡されたデータをそのまま出力する。Ver.2.09 においては、シャッターを操作する段階で、既存の効果構造体をクリアし、その瞬間における表示に係る各種設定を効果構造体として記述する。従って、表示設定に関係のない効果に係る未定義のコマンドは、この段階でメモリ上の効果構造体から削除され、ファイル保存に際しても出力されなくなる。

③同じラベルを用いた効果、効果グループの再定義はエラーとして処理する。既に定義された効果、効果グループは影響を受けない。

（５）モデル MODEL

地物の幾何形状を記述したＬＳＳ－Ｇファイルを指定する。LSS-S ファイルは、異なるシーンに異なるモデルを指定することにより、シーンを切り換えることにより、物体が置

換・変形したような表現を行うことができる。

［記法］　モデル名称＝MODEL(ＬＳＳ－Ｇファイル名称);
ＬＳＳ－Ｇファイル名称　ＬＳＳ－Ｇファイル
モデルデータを作成する。

異なるモデル名称を用いて同じ LSS-G ファイルが再度参照された場合、あるいはロード済みの LSS-G ファイルが、別の LSS-G ファイルから部品として再度参照された場合、インタープリタは繰り返して同じ LSS-G ファイルをロードせずに、ロード済みのデータを再利用する。

2001 年度に、まちづくり・コミュニケーション・システムの一部として、景観シミュレータの機能拡張を行った。その際に、WEB サーバーから公開されるＬＳＳ－Ｓファイルを、HTML ページからリンクできるようにした。景観シミュレータがセットアップされた環境においては、この操作により景観シミュレータが起動し、ＬＳＳ－Ｓファイルをダウンロードし開く。そのＬＳＳ－Ｓファイルの中では、モデルは、例えば以下のように記述されている。

MDL1 = MODEL("http://sim.nilim.go.jp/niigata/geometry/3ddb/08P0487P211200040003.geo");

この情報に基づき、景観シミュレータは、”http://sim.nilim.go.jp/niigata/geometry/3ddb/”というプレフィックスを用いながら、このＬＳＳ－Ｇファイルと、それが更に引用する部品等のＬＳＳ－Ｇファイルをダウンロードして表示する。

［エラー処理の現状］

①モデルは、MODEL コマンドが実行された時点でロードされる。ロードに失敗した場合には、インタープリタのエラーとして処理する。

②同じラベルを用いたモデルの再定義はエラーとして処理する。定義済みのモデルは影響を受けない。

（６）背景・前景画像 IMAGE

背景・前景画像は写真合成に使用する。写真合成において、背景は、設計対象物や地形を記述した三次元オブジェクト(MODEL)が存在しない画面領域にのみ表示される。前景は、常に表示されるが、アルファ値（不透明度）が１ではない画像領域に関しては、モデルや背景と混合し、アルファ値がゼロの場合は完全に透明となる。写真の中に、三次元オブジェクトよりも手前にある被写体が映っている場合、その部分以外の部分を透明（アルファ値ゼロ）となるように加工した画像を前景として設定する。定義した画像を前景とするか背景とするかは、SCENE コマンド（後述）の引数列内の位置により指定する。よく用いら

れる JPEG ファイルではアルファ値が設定できないため、前景として利用する場合は、アルファ値が設定できる SGI 形式などに変換する必要がある。

［記法］ 画像名称＝IMAGE(画像ファイル名称);
画像ファイル名称　　SGI 形式、JPEG 形式、BMP 形式のファイル名。
画像データを作成する。

［例］ I1 = IMAGE(sample.sgi);

Ver.2.09 における処理
①ファイル名称がフルパスで記載されていない場合には、作業環境が設定されている場合には、その場所を、そこになければ環境設定ファイルで定義された画像ファイル格納ディレクトリから検索する。
②画像ファイルをロードする時点は、IMAGE コマンドが実行される時点である。画像ファイルのロードに失敗すると、インタープリタはエラー処理を行う。
③同じ画像名称を用いた画像の再定義が行われた場合には、エラーとして処理する。
定義済みのイメージは影響を受けない。

（7）シーンの生成 SCENE

以上の各コマンドにより定義した情報を用いて、シーンを合成する。

［記法］ シーン名称＝SCENE(シーン・タイプ、背景画像名称、前景画像名称、
モデル名称、光源グループ名称、効果グループ名称、視点名称、時間名称);

シーン・タイプ　図法、表示モードを指定する３桁のコード（表３－１）
背景画像名称　IMAGE コマンドで定義した画像名称
前景画像名称　IMAGE コマンドで定義した画像名称
モデル名称　　MODEL コマンドで定義したモデル名称
光源グループ名称　　LIGHTGROUP コマンドで定義した光源グループ名称
効果グループ名称　　EFFECTGROUP コマンドで定義した効果グループ名称
視点名称　　CAMERA コマンドで定義した視点名称
時間名称　　TIME コマンドで定義した時間名称
シーンを生成するコマンドは、実行されるたびに、新たなシーンを生成し、メモリ上の配列に蓄積される。景観シミュレータでの表示においては、左下のシーン送り／戻しボタンにより、蓄積されているシーンを次々と表示することができる。
最上位の定義である SCENE においては、ファイル読み込み終了後も、その各種設定を名

称で管理しているため、同じラベルの再定義が行われると、編集操作などによりシーンの順序が変更となった場合などに、定義の解釈が難しくなる。そこで、LSS-S の記述に用いる名称は、同じ名称を再定義するとエラーとして処理する。

規則違反のデータに関して、Ver.2.09 では以下のように対処している。

①引数が不足する場合：エラー

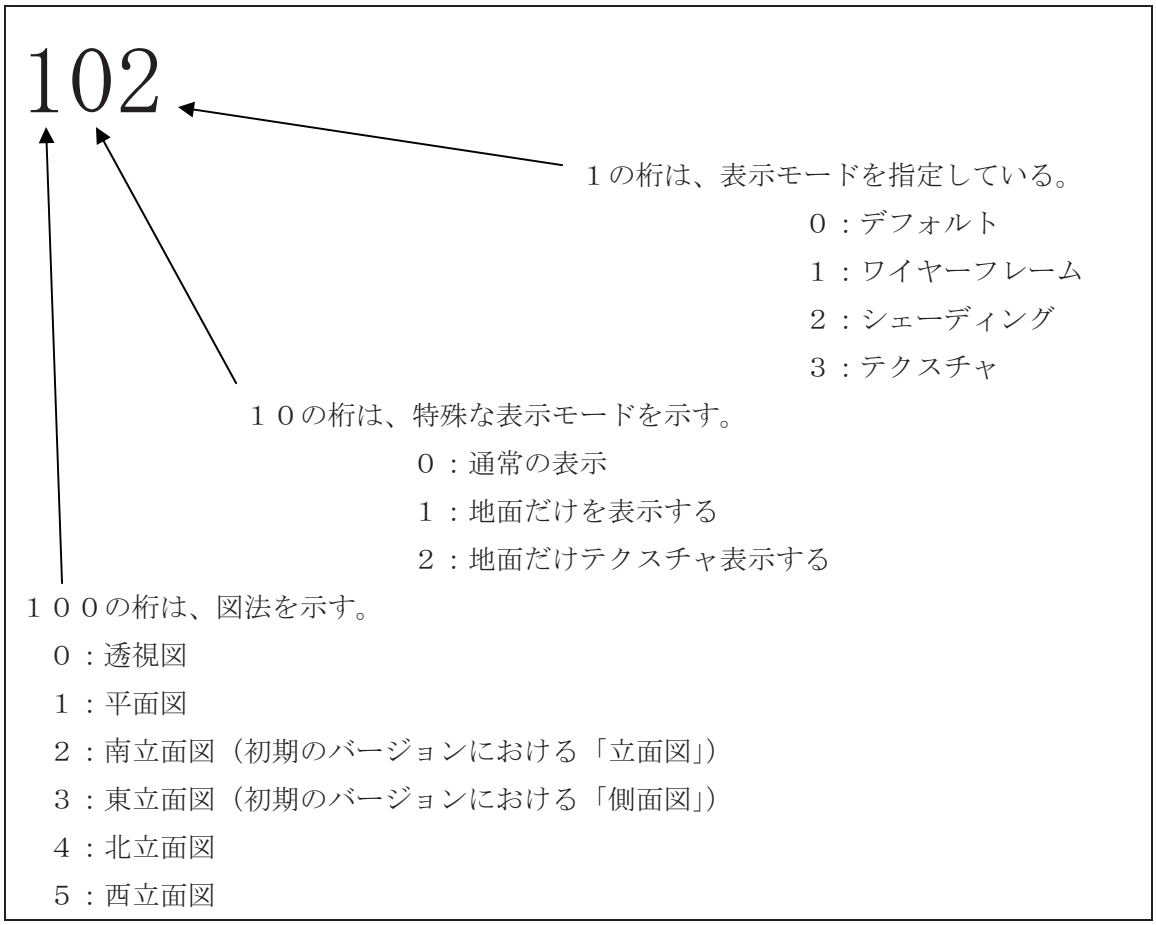
②引数が省略された場合：

1. タイプ：　0 とみなす
2. 背景画像名称：　　背景なし（積極的な意味）
3. 前景画像名称：　　前景なし（積極的な意味）
4. モデル名称：　モデルなし（積極的な意味）
5. 光源グループ名称：光源なし（表示にはデフォルト光源を使用）
6. 効果グループ名称：効果なし（積極的な意味）
7. 視点名称：　エラーではないが、モデルが表示されない。
8. 時間名称：　　エラーメッセージを出し、０を適用。

③同じラベルを用いたシーンの再定義が行われた場合にはエラーとして処理する。

④SCENE コマンドが一つも存在しない場合、エラーとして処理する。

表３－１：シーン・タイプの意味



（8）シーンのファイル出力 OUTPUT_SCENE

シーンの出力を行う。

［記法］ ファイル名称=OUTPUT_SCENE（）；

例） sample＝OUTPUT_SCENE();

３－３ ＬＳＳ－Ｇコマンド

ＬＳＳ－Ｇ（ＬandＳcape Simulation-Geometry）ファイルは、景観構成要素データベースにおける３Ｄ形状データを具現化したファイルである。テキスト・ファイルであり、インタプリタ（ＩＰ、４－２(6)参照）により解釈され、ＤＭＬライブラリ（４－２(2)参照）を介してメモリ上に格納される。

ＬＳＳ－Ｇの拡張子は".geo"である。

表３－２：ＬＳＳ－Ｇファイルに用いられるコマンドの一覧

COORD
NORMAL
TCOORD
COLOR
VERTEX
FACE
LINE
GROUP
LINK
MATERIAL
TEXTURE
FILE
FILE_SUMMIT
FACE_COLOR
FACE_NORMAL
FACE_MATERIAL
FACE_TEXTURE
LINE_COLOR
LINE_NORMAL
LINE_MATERIAL
LINE_TEXTURE
GROUP_FACE

GROUP_MATERIAL
GROUP_TEXTURE
LINK_XFORM
CONCAVE
OUTPUT

３－３－１．データ構築コマンド

（１）座標 COORD

［記法］　座標名称＝COORD(X, Y, Z);

Ｘ　　Ｘ座標値

Ｙ　　Ｙ座標値

Ｚ　　Ｚ座標値

座標データを作成する。

例）　ＶＯＯＯＯ１＝COORD(10.0,20.0,30.0);

（２）法線 NORMAL

［記法］　法線名称＝NORMAL(X, Y, Z);

Ｘ　　法線方向のＸ座標値

Ｙ　　法線方向のＹ座標値

Ｚ　　法線方向のＺ座標値

法線データを作成する。正規化されている必要はない。

例）　ＮＯＯＯＯ１＝NORMAL(10.0,0.0,0.0);

（３）テクスチャ座標 TCOORD

［記法］　テクスチャ座標名称＝TCOORD(S,T);

Ｓ　　テクスチャ座標のＳ値

T　　テクスチャ座標のＴ値

テクスチャ座標データを作成する。

テクスチャ領域は0.0≦ ≦1.0に対応。

例）　ＴＯＯＯＯ１＝TCOORD(0.5,0.5);

テクスチャを面に貼り付ける場合には、面に対して適用するテクスチャの画像ファイル名を定義するだけではなく、各頂点を、画像ファイルのどこに対応づけるかを示すためにテクスチャ座標を指定する必要がある（テクスチャについては、２－９参照)。

（４）色 COLOR

［記法］　色名称＝COLOR(R,G,B,A);

R　　色データの RED 値

G　　色データの GREEN 値

B　　色データの BLUE 値

A　　色データの ALPHA 値

色データを作成する。各値は 0.0≦ ≦1.0 の浮動小数点値とする。

例)　ＣＯＯＯＯ１＝COLOR(1.0,0.5,0.0,1.0);

色は、面や線に対して一括して指定することも、頂点毎に指定することもできる。一つの面や線を構成する点に異なる色を指定すると、その中間部は補間した色となる。

（５）頂点 VERTEX

最大で、座標、法線、テクスチャ、色の４属性[1)]を有する頂点を生成する。同じ位置にあっても、異なる法線ベクトル等をもつ頂点は、別の頂点として定義する必要がある。

[記法]　頂点名称＝VERTEX(座標名称、法線名称、テクスチャ座標名称、色名称);

座標名称　　COORD コマンドで定義された座標名称

法線名称　　ＮＯＲＭＡＬコマンドで定義された法線名称

テクスチャ座標名称　　TCOORD コマンドで定義されたテクスチャ座標名称

色名称　COLOR コマンドで定義された色名称

頂点データを作成する。

例)　ＰＯＯＯＯ１＝VERTEX(V0001,N0001,T0001,C0001);

法線名称、テクスチャ座標名称、色名称は省略可能である。

例)　ＰＯＯＯＯ１＝VERTEX(V0001,,T0001);

例)　ＰＯＯＯＯ１＝VERTEX(V0001);

1)：Ver.2.07 までのインタープリタ（ＩＰライブラリ）では、５以上の属性が定義されていた場合においても、最初の４属性のみを解釈し、以降の属性は無視している。Ver.2.09 においては、仮想線の出発点となる頂点を明記する必要がある場合に、ファイル出力時点で第５引数に、その仮想線の終点の座標名称を第５引数として記述する。入力時点では、面を定義するコマンドにおいて、次の頂点の座標名称が、第５引数で指定された頂点の座標名称と一致する場合には、当該頂点に仮想線の出発点であることを示すフラグを立てる。このフラグが無い場合で、仮想線が存在しうる場合（８頂点以上の面）に関しては、従来通り面の定義が行われた時点で、使用されている頂点名称の比較分析により仮想線を判定する。この属性が意味を有するのは、図形演算が途中で失敗ないし中断し、図形の内部に、外周との接続がなく、面積の無い線状の切り込みが残された状態で、検査等の目的のためにこれをファイル保存する場合である。

（６）面 FACE

[記法]　面名称＝FACE(頂点名称、頂点名称、...);

頂点名称　　VERTEX コマンドで定義された各属性名称

面データを作成する。

例)　ＳＯＯＯＯ１＝FACE(P0001,P0002, P0003);

面は無制限の数の頂点を持つことができる。最後の頂点と最初の頂点は自動的につながれるため、例えばｎ角形の場合にはｎ個の頂点を指定する。最初の頂点と同じ頂点を最後に定義すると、表示は同等であっても長さゼロの辺が一つ余分に定義された「病的な」面が生成するので注意を要する。

面の場合、次に述べる線の場合とは異なり、頂点の順序を巡回的に変更しても、生成される面は同じものとなる。

（７）線 LINE

［記法］ 線名称＝LINE(頂点名称、頂点名称、...);

頂点名称　　VERTEX コマンドで定義された各属性名称

線データを作成する。

例)　ＳＯＯＯ１＝LINE(P0001,P0002, P0003);

線は、無制限の数の頂点を持つ折れ線である。面とは異なり、閉じたループを作る場合には、最後の点と最初の点の頂点は同じものでなければならない。折れ線は、面とは異なり、同一平面上にない場合でも支障はない（例えば、ジェットコースター線路の中心線軌跡)。

（８）グループとその属性の定義 GROUP

［記法］ グループ名称＝GROUP(属性 1,属性２,・・・);

グループを生成する。

属性：属性を記述する文字列。

引数なし　　通常のグループ定義

&GROUND グループに地面属性を付ける

&TREE3　３枚の面で作られた近似的な樹木であることを示し表示を制御

&INFO:文字列情報　（グループを選択したときに情報の表示を行う）

&EXFO:ファイル名　（そのグループを選択したときに、テキスト・ファイルが表示される

&HOUSE:住宅情報　（住宅情報を見る際に表示される構造化された情報）

&PHISI:物理的属性

&CHEMI:化学的属性

&DB:データベース参照情報（景観データベースの登録情報へのタグ）

例)　ＧＯＯＯ１＝GROUP();

同じグループ名称が二度定義されると、属性部分が追加される。例えば、（１２）ＦＩＬＥコマンドを用いて構築されたグループに属性情報を付加する場合には、同じグループ名称を用いたＧＲＯＵＰコマンドを用いて、属性情報を付加することができる。

G01=FILE(物体 1);

G01=GROUP(属性 1);

G01=GROUP(属性 2);

というコマンドが実行された場合、これをファイル保存すると、次のように保存される。

G01=FILE(物体 1);

G01=GROUP(属性 1,属性 2);
というコマンド列として保存される。

（9）リンクの宣言 LINK

二つのグループを関連づける

[記法]　リンク名称＝LINK(グループ名称、グループ名称);

グループ名称（第１引数）　　親となるグループの名称

グループ名称（第２引数）　　子となるグループの名称

グループ間の階層を定義する。第一パラメータで示されるグループが親となり、第二パラメータで示されるグループが子となる。

例)　L0001＝LINK(G0001.G0002);

リンクには、マトリクスが定義されており、リンクが定義された時点では単位マトリクスに初期化されている。続くコマンドでこのマトリクスが変更されると、子グループの位置・スケールが変化する。

同じ親グループと子グループの間には、複数のリンクを定義することができる。それぞれのリンク・マトリクスが異なっていれば、リンクの数だけ子グループが異なる位置に表示される。この機能は、街路樹など、同じ物体を、メモリ消費を増大させることなく、多数配置する場合に有用である。言い換えると、画面に表示されているオブジェクトは、実はリンクを見ている、と理解することができる。

（１０）マテリアルＩＤ　MATERIAL

[記法]　マテリアル ID＝MATERIAL(マテリアル名称);

マテリアル名称　マテリアルファイル（３－６参照）に定義されているマテリアルの名称（ラベル）で、以下その内容に関する記述（カラー値、テクスチャ等）が続く。マテリアル・ファイルは、ＤＢＩＬライブラリの dbLoadMaterial 関数により行われる。

例)　M001 = MATERIAL(SILVER);

（１１）テクスチャＩＤ　TEXTURE

[記法]　テクスチャ ID＝TEXTURE(テクスチャファイル名);

テクスチャファイル名　　テクスチャとして適用する画像ファイル名（３－８参照)。

画像ファイル名がフルパスで指定されている場合を除き、環境設定ファイル kdbms.set(１２－１参照)で指定したディレクトリから取得する。

例)　I001 =TEXTURE (renga) ;

I002 = TEXTURE(renga.sgi);

I003 = TEXTURE(renga.jpg);

I004 = TEXTURE(“D:¥MyProject¥image3.bmp”)

Ver.209 においては、
①拡張子が省略された場合には、.sgi を補う。SGI 形式(RGB)形式はマイナーであるが、景観シミュレータの開発開始頃(1993 年)に利用可能であった、透明度（アルファ値）まで表現しうる唯一の画像であった。構造は RGB または RGBA をバイナリでシリアルに並べただけのシンプルな形式である（３－８参照)。
②テクスチャは、TEXTURE コマンドが実行された時点でロードされる。同じファイルが別のラベルで参照された場合には、同じデータで多重にメモリを消費しないようにバッファリングを行っている。

（１２）外部関数、外部ファイル参照 FILE

[記法] グループ名称＝FILE(ファイル名称、パラメータ、パラメータ、...);

ファイル名称　ＬＳＳ－Ｇファイルまたは、EXT.TAB ファイルに登録されている外部コマンドファイル

パラメータ　EXT.TAB ファイルで定義された引数リスト。ＬＳＳ－Ｇファイルを指定したときは、パラメータはない。

ファイル名称として固定的なＬＳＳ－Ｇファイル名が指定された場合、データファイルを読み込む。引数は無い。

ファイル名称として、EXT.TAB ファイル（３－１０、５－４参照）に登録してある外部関数を指定した場合は、そのコマンドが実行される。その場合引数として指定するパラメータの数と形式は EXT.TAB に定義されている引数と同じでなければならない。

例１）　G0001 = FILE(BRG_ARCH.geo);

例２）　G0001 = FILE(SPHERE, 10.0, 20.0, 30.0, 3.5);

半径 3.5、中心点座標が(10.0, 20.0, 30.0)の球。

パラメトリックな図形を外部関数により作成し、あるいはＬＳＳ－Ｇ形式のファイル(固定的図形）を読み込み、このグループ名称を有するグループを構築する。ＦＩＬＥコマンドに使用するグループ名称が既に使用されているものである場合、エラーとなる。

ファイル名称で参照されたファイル（上記の例１）の場合、BRG_ARCH.geo)の中で使用されているグループ名称（例えば）G0001 と、参照元で直接定義するグループ名称との重複・一致・衝突を避けるために、FILE コマンドで参照されたファイルの中で使用されるグループ名称には、G0001.G0001 のように、親グループ名称をプレフィックスとして追加している。

（１３）ルートグループにおける外部関数、ファイル参照 FILE_SUMMIT

強制的にグループ名プレフィックスを付加しないFILE コマンド。ファイル保存に際して、必要な場合には、通常のＦＩＬＥコマンドに代わり、システムが自動的に使用する。

［記法］ グループ名称＝FILE_SUMMIT(ファイル名称, パラメータ, パラメータ,...);

ファイル名称　ＬＳＳ－Ｇファイルまたは、EXT.TAB ファイルに登録されている外部コマンドファイル

パラメータ　EXT.TAB ファイルで定義された引数リスト。ＬＳＳ－Ｇファイルを指定したときは、パラメータはない。

ファイル名称として固定的なＬＳＳ－Ｇファイル名が指定された場合、データファイルを読み込む。引数は無い。

ファイル名称として、EXT.TAB ファイル（３－１０、５－４参照）に登録してある外部関数を指定した場合は、そのコマンドが実行される。その場合引数として指定するパラメータの数と形式はEXT.TAB に定義されている引数と同じでなければならない。

FILE_SUMMIT コマンドでは、ロードに際して、指定したファイルの中で定義されたグループの名称に関して、親グループ名称のプレフィックスを付けない。

最上位の親グループとして、FILE コマンドで定義されたグループしかないような場合、保存・読み込み等を繰り返す度に無用なプレフィックスが追加されて長いグループ名となってしまうような事態を防ぐために用いる。

例）G0001.G0001.G0001.G0001

３－３－２．データ定義／更新コマンド

面、線関連

（１）色の面への適用　FACE_COLOR

面データに色を割り当てる。

［記法］ FACE_COLOR（面名称、色名称）；

面名称　FACE コマンドで定義された面名称

色名称　COLOR コマンドで定義された色名称

例）　FACE_COLOR（S0001、N0001);

内部処理においてはd3Face 構造体のパラメータを設定する。

（２）法線の面に対する適用 FACE_NORMAL

面データに法線を割り当てる。

［記法］ FACE_NORMAL（面名称、法線名称）；

面名称　FACE コマンドで定義された面名称

法線名称　NORMAL コマンドで定義された法線名称

例）　FACE_NORMAL（S0001、N0001);

内部処理においては d3Face 構造体のパラメータを設定する。

（3）マテリアルの面への適用 FACE_MATERIAL

［記法］ FACE_MATERIAL（面名称、マテリアル ID）；

面名称　　FACE コマンドで定義された面名称

マテリアル ID　MATEREAL コマンドで定義されたマテリアルＩＤ

面にマテリアルを割り当てる。

例）　FACE_MATERIAL（S0001、M0001)；

ＬＳＳ－Ｇコマンドにおいては、面にマテリアルを定義する際に、マテリアル名称（鉄、コンクリート、あるいは色見本帳番号など）を直接指定するのではなく、マテリアルＩＤを一度定義した上で、これを用いて間接指定しているため、単純なケースでは、冗長な定義となっている。

しかし、例えば複雑な形状の壁を有する建物の場合、同一の仕上げの面が多数存在する。これらに対して、同一のマテリアル「M-kabe」を定義しておくと、個別の面へのマテリアル適用に先行して M-kabe を定義している１行を、例えば「M-kabe=MATERIAL(リシン吹付);」から「M-kabe=MATERIAL(レンガタイル);」に変更するだけで、全ての壁の仕上げを一括変更することができる。

内部処理においては d3Face 構造体のパラメータを設定する。

（4）面へのテクスチャの適用　FACE_TEXTURE

［記法］ FACE_TEXTURE（面名称、テクスチャ ID）；

面名称　　FACE コマンドで定義された面名称

テクスチャ ID　TEXTURE コマンドで定義されたテクスチャＩＤ

面にテクスチャを割り当てる。

例）　FACE_TEXTURE（S0001、I0001)；

面にテクスチャを適用する場合、あらかじめ面を構成する頂点の定義(VERTEX コマンド)に際して、テクスチャ座標が定義されていなければならない。

内部処理においては d3Face 構造体のパラメータを設定する。

（5）線への色の適用　LINE_COLOR

線分データに色を割り当てる。

［記法］ LINE_COLOR（線分名称、色名称）；

線分名称　　LINE コマンドで定義された線分名称

色名称　　COLOR コマンドで定義された色名称

例）　LINE_COLOR(LN0001、C0001)；

線分に適用した色は、光源の条件に関わりなく、例えば闇夜であっても、その色で画面

表示される。指定されない場合であっても、明るい灰色で表示される。従って、電線などの細長い物体を線としてモデリングすると、不自然な表示となる場合がある。むしろ、補助線・寸法線などの標示に用いるのに適している。

一方、線分として定義された高架道路などの長尺物の断面を用いて実際の三次元形状を生成する道路生成機能や外部関数「掃引体１面」においては、線に適用されていた色を生成された立体に適用する。このような断面の定義には線に適応した色が有用である。

内部処理においては d3Face 構造体のパラメータを設定する。

（６）線への法線の適用 LINE_NORMAL

［記法］ LINE_NORMAL（線分名称、法線名称）；

線分名称　　LINE コマンドで定義された線分名称

法線名称　　NORMAL コマンドで定義された法線名称

線分データに法線を割り当てる

例）　LINE_NORMAL(LN0001、N0001)；

線に適用された法線は、その線自体を表示する際には意味を持たない。しかし、長尺物の断面を定義する線に法線を定義することは、線が掃引されて生成される面の表裏を区別する上で意味をもつ。

内部処理においては d3Face 構造体（２－２参照）のメンバ変数（法線ベクトル）に値を設定する。

（７）線へのマテリアルの適用 LINE_MATERIAL

［記法］ LINE_MATERIAL（線分名称、マテリアル ID）；

線分名称　　LINE コマンドで定義された線分名称

マテリアル ID　　MATEREAL コマンドで定義されたマテリアルＩＤ

線分データにマテリアルを割り当てる

例）　LINE_MATERIAL(LN0001、M0001)；

線が断面を定義するデータの場合、これを用いて生成した立体に反映される。

内部処理においては d3Face 構造体(２－２参照)のメンバ変数 material を設定する。

（８）線へのテクスチャの適用 LINE_TEXTURE

［記法］ LINE_TEXTURE（線分名称、テクスチャ ID）；

線分名称　　LINE コマンドで定義された線分名称

テクスチャ ID　　TEXTURE コマンドで定義されたテクスチャＩＤ

線分データにマテリアルを割り当てる

例）　LINE_TEXTURE(LN0001、T0001)；

表示に反映するためには､線の各頂点 VERTEX にテクスチャ座標が定義されている必要が

ある。

内部処理においては d3Face 構造体（２－２参照）のメンバ変数 texture を設定する。

（9）グループへの面の割り当て GROUP_FACE

［記法］ GROUP_FACE(グループ名称、面名称、面名称、...);

グループ名称　　GROUP コマンドで定義されたグループ名称

面名称　　FACE コマンドで定義された面名称

例）　GROUP_FACE(G0001,S0001);

一つのグループには無制限個の面を割り当てることができる。グループに既に面が割り当てられている場合には、これに追加される。

実装しているインタープリタにおいては、GROUP_FACE コマンドが実行された後に、その面を構築する際に定義された面の属性､面を構成する頂点､各頂点の属性が変更されても、一度グループに対して割り当てられた面に遡って影響は及ばない。

内部処理においては このコマンドが実行された時点で、インタープリタの解析木の上にそれまでに定義されてきた頂点やカラー等の構成要素を用いて、面を記述するＤＭＬ空間（４－２(7)参照）の d3Face 構造体（メモリ・ブロック）を構築し、これを、ＤＭＬ空間に定義された d3Group 構造体に追加する。この追加は、d3Group 構造体のメンバ変数 f（d3Face*）を起点としてポインタつなぎで定義する､グループに帰属する一群の面のリストの末尾に、この面を加えている。

この面を定義する構成要素がこの時点で確定することから、以後、インタープリタの中で、この面の定義に用いられた各要素の変更・再定義が行われても、ＤＭＬ空間に既に移管・構築された面に遡っては影響が及ばない。

この仕様を利用して、例えば定義済みの面を利用して、それを構成する頂点の座標だけを変更し、再度同じ名称の面としてグループに割り当てることにより、効率的に複数の面を定義することができる。このことはコンバータの作成等においてきわめて有用である。

（１０）グループへの線分の割り当て GROUP_LINE

［記法］ GROUP_LINE(グループ名称、線分名称、線分名称、...);

グループ名称　　GROUP コマンドで定義されたグループ名称

線分名称　　LINE コマンドで定義された線分名称

例）　GROUP_LINE(G0001,LN0001,LN0002);

機能や仕様は、GROUP_FACE コマンドと同等である。グループに面と線を混在させて割り当てることができる。

（１１）グループへのマテリアルの定義 GROUP_MATERIAL

［記法］ GROUP_MATERIAL(グループ名称、マテリアル ID);

グループ名称　GROUP コマンドで定義されたグループ名称

マテリアル ID　MATEREAL コマンドで定義されたマテリアルＩＤ

例)　GROUP_MATERIAL(G0001,M0001);

このマテリアルは､グループ内のデフォルトのマテリアルとなる｡グループに属する面、このグループにリンクする子グループにマテリアルが定義されていない場合、このマテリアルが適用される。

内部処理においては d3Group 構造体のパラメータを設定する。グループは、DML 空間に既に登録されているが、そのグループへのリンク（ポインタ）はインタープリタの解析木に残されているので、これを用いてグループの属性を変更することができる。

グループを定義している d3Group 構造体のメンバである int material メンバに ID 番号として登録される。実際のマテリアルのデータは、インタープリタがこのコマンドを実行するに際して、d3RegisterMaterial(マテリアル名称)関数を起動することにより、DML 空間のスタティックなテーブル mtltab に登録される。但し、この段階では、マテリアルの名称だけが登録されており、実際のデータはまだ登録されていない。実際にマテリアルのロードが行われるのは表示段階である。一度ロードされたマテリアルは、このテーブルから取得されることとなる。なお、マテリアルには経年変化するものがあることから、時間が変更された場合には、mtltab は見直される。

（１２）グループへのテクスチャの定義 GROUP_TEXTURE

グループにテクスチャを割り当てる。

［記法］ GROUP_TEXTURE(グループ名称、テクスチャ ID);

グループ名称　GROUP コマンドで定義されたグループ名称

テクスチャ ID　TEXTURE コマンドで定義されたテクスチャＩＤ

例)　GROUP_TEXTURE(G0001,I0001);

このテクスチャは、グループ内のデフォルトのテクスチャとなる。

内部処理においては d3Group 構造体のパラメータを設定する。グループは、DML 空間に既に登録されているが、そのグループへのリンク（ポインタ）はインタープリタの解析木に残されているので、これを用いてグループの属性を変更することができる。

グループを定義している d3Group 構造体のメンバである int texture メンバに ID 番号として登録される。実際のテクスチャのデータは、インタープリタがこのコマンドを実行するに際して、d3RegisterTexture(テクスチャ名称)関数を起動することにより、DML 空間のスタティックなテーブル textab に登録される。但し、この段階では、テクスチャの名称だけが登録されており、実際のデータはまだ登録されていない。実際のテクスチャのロードは表示段階で g3LoadMateiral によって果たされる。一度ロードされたテクスチャは、このテーブルから取得されることとなる。

（１３）リンク・マトリクスの定義 LINK_XFORM

リンクに変換マトリクスを定義する。

［記法］ LINK_XFORM(リンク名称、変換タイプ、変換名称、パラメータ、...);

変換タイプ

LOAD　それまでのマトリクスを初期化する。

PRE　それまでのマトリクスの前に乗ずる。

POST　それまでのマトリクスの後に乗ずる。

変換名称及びそれに付随するパラメータは以下の通り

IDENTITY　パラメータ=なし（対角だけ１の単位マトリクス）

TRANSLATE　パラメータ=X、Y、Z（平行移動）

ROTATE_X　パラメータ=X 軸周り回転角(deg)

ROTATE_Y　パラメータ=Y 軸周り回転角(deg)

ROTATE_Z　パラメータ=Z 軸周り回転角(deg)

ROTATE_A　パラメータ=回転軸ベクトル（X、Y、Z)、回転角

SCALE　パラメータ=X、Y、Z

MATRIX　パラメータ=行列要素 0、1、...、１５

例）　LINK_XFORM（L001、LOAD、ROTATE_X､30.0）

LINK_XFORM コマンドが実行されるまでは、リンクが定義された時点で単位マトリクスが定義されている。

マトリクスの詳細については、４－２（３）マトリクス／ベクトル関数の、d3SetLinkMatrix 関数の項を併せて参照されたい。上記の MATRIX により１６個の倍精度実数値で記述する場合には、

0　4　8 12

1　5　9 13

2　6 10 14

3　7 11 15

の順でマトリクスの要素を記述し、変換されるベクトルは列ベクトルである。頂点座標の変換計算は、XYZ に１を加えた同次座標に変換マトリクスを乗じることにより、回転と移動を一挙に計算する。法線ベクトルについては、平行移動はないため、３×３の部分だけが計算に用いられるのでなければならない、ということはデバッグ、検査に際して留意すべき点である。

（１４）凹ポリゴンの指定 CONCAVE

［記法］ CONCAVE(パラメータ);

ＯＮ　凹多角形として表示処理を行う

ＯＦＦ　凹多角形表示処理を行わない

凹多角形処理の開始、終了を指示する（２－２（６）参照)。
例)　CONCAVE(ON);
CONCAVE スイッチが ON になっている間、インタープリタは、入力されメモリ空間に構築する面に凹ポリゴンの属性(D3_SHP_CONCAVE)を付ける。表示段階では、この属性が付いた面に関しては、三角形に分割して表示を行うため、無い筈の場所に現れる意味のない黒影で惑わされることが防がれる。

（１５）グループのファイル出力 OUTPUT

データファイルに書き込む
［記法］ ファイル名称=OUTPUT（グループ名称）;
例)　sample＝OUTPUT(G0001);
指定されたグループ名称から下位の情報が書き込まれる

３－４．　共通コマンド

LSS-G と LSS-S に共通に用いられる、システム的なコマンドである。

表３－３：共通のコマンド

CLEAR
DELETE
RESET
COMMENT
UNIT

（1）インタープリタのリセット RESET

インタープリタ空間をリセットし、ラベルの辞書等に占有していたメモリを解放する。
［記法］ RESET();

（2）データの初期化 CLEAR

未実装（廃止)。1998 年の仕様書には記されていたが、使われたことがなく、今後も使う予定はない。

（3）データの削除 DELETE

［記法］DELETE（*名称）;
名称で指定されたデータを削除する
例)　DELETE（S0001); //この例では面 S0001 を削除する
例)　DELETE（G0001); //この例ではグループ G0001 を削除する
例)　DELETE（L0001); //この例ではリンク L0001 を削除する

（4）注釈領域の始まりと終わり COMMENT

コメントの開始、終了を指定する。

[記法] COMMENT（パラメータ）；

ON　　コメントの開始

OFF　　コメントの終了

COMMEN(ON)以降はCOMMENT(OFF)が現れるまでコメントとして扱う。ファイル中、情報としては残しておたいが、表示には使用したくないようなデータに用いる。システム開発者が、モデリング機能やコンバータの開発途上でのテスト・デバッグにおいて、データと表示の対応を検査する際に有用である。

例)　COMMENT(ON);

（5）長さ精度の単位 UNIT

座標値の精度、即ち最小単位を、メートルを尺度として指定する。単位よりも小さな値は四捨五入される。従って、データの誤差の下限を示している。単位よりも小さい距離にある二つの異なる点は、同じ点として理解される。図形の内外判定において、単位は線の太さを示し、線上の判定に用いられる。例えば1,000ｍを指定すると、線の太さは１km となり、多くの建物は点に包摂される。

[記法] UNIT（長さの単位）；

（長さの単位）；長さの最小単位を指定する。0.001＝1mm がデフォルト有効である。

例)　UNIT(1e-3)

３－５　ＬＳＳデータ構築の実際

（1）ＬＳＳ－Ｓファイル

ＬＳＳ－Ｓデータを構築するには、まず、シーンコマンドで設定するのに必要な各要素（時間、視点、光源、光源グループ、モデル、イメージ）の設定をはじめに行う。ただし、すべての要素を設定する必要はなく、最低限必要な要素は、時間・視点・光源・光源グループである。

①┌まず、必須である要素の設定を最初に行う。

リスト３－４：必須要素の設定

```
例)
time = TIME( 1 );
camera = CAMERA(
        15, 105, 18,
        28, 10, 18,
        0, 0, 1,
        60, 1.566, 1, 1000);
```

```
l = LIGHT(
        0,
        20, 10, 7, 0,
        1, 1, 1);
lg = LIGHTGROUP( l );
```

②次に、モデルやイメージデータがある場合は、それらの設定を行う。

リスト３－５：モデルとイメージデータの設定

```
例)
img = IMAGE(haikei1.sgi);
model1 = MODEL(BRG_ARCH.geo);
```

③最後に、シーンとして先程設定した要素を割り付ける。

リスト３－６：シーンへの各要素の割り付け

```
例)
scene_1 = SCENE( 0, img, , model1, lg, , camera, time );
```

これで少なくとも一つのシーンを持った、ＬＳＳ－Ｓファイルとなる。

④ひとつのＬＳＳ－Ｓファイルの中に、複数の要素、複数のシーンデータを設定する事ができる。

リスト３－７：複数シーンの設定

```
例)
time = TIME( 1 );
camera1 =CAMERA(
        15, 105, 18,
        28, 10, 18,
        0, 0, 1,
        60, 1.566, 1, 1000);
camera2 =CAMERA(
        20, 80, 30,
        28, 10, 18,
        0, 0, 1,
        60, 1.566, 1, 1000);
l = LIGHT(      0,
        20, 10, 7, 0,
```

```
        1, 1, 1);
lg = LIGHTGROUP( 1 );
img = IMAGE(haikei1.sgi);
model1 = MODEL(BRG_ARCH.geo);

scene_1 = SCENE( 0, img, , model1, lg, , camera1, time );
scene_2 = SCENE( 0, img, , model1, lg, , camera2, time );
```

この例では、視点情報を２つ設定している。これを用い、シーンも２つ定義し、それぞれに違う視点情報を設定している。つまり、背景や光源情報、時間情報、モデル情報は同一だが、視点情報だけ異なる２つのシーンが設定されたことになる。プレゼンテーションにおいては、シーンを切り換えることにより、予め設定してある視点場から別の視点場へと景観表示をすばやく切り換えることができる。

あるいは、上の例と同様に、視点情報以外の要素でも複数設定できる。例えば、異なるモデルを同じ視点から見たシーンを用意しておき、現況と建て替え後の、同じ視点から見た比較表示を行うことができる。

（２）ＬＳＳ－Ｇファイル

①グループの作成

ＬＳＳ－Ｇデータの形状定義は、大きく４段階から成る。

座標設定（図４）→面設定（図３）→グループ設定（図２）→リンク設定（図１）

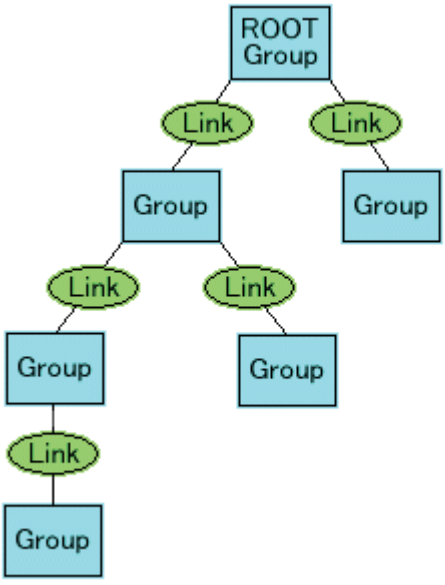
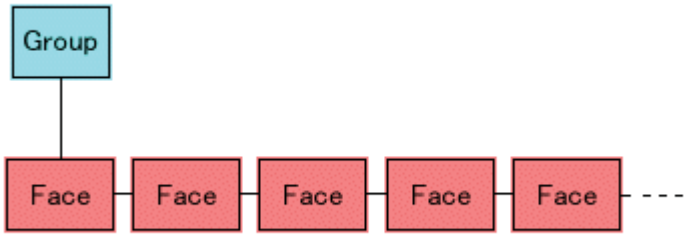
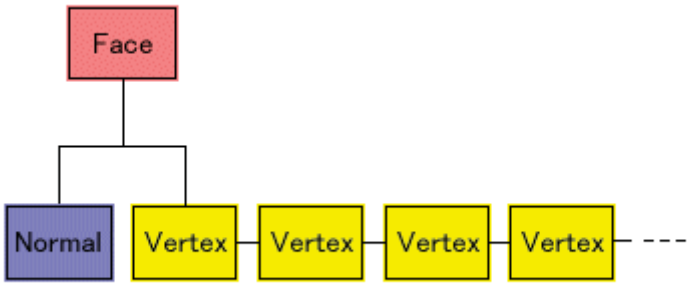


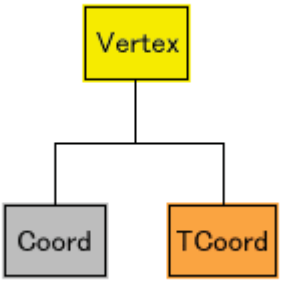
図３－１：リンク設定



·図３－２：グループ設定



・図３－３：面設定



・図３－４：座標設定

最も簡単な例を挙げながら、各設定の流れを説明する。

例）立方体を作る

①まず、座標を任意の座標名称に設定する。

立方体は、8 頂点あるので８個分設定する。

リスト３－８：頂点座標の設定

```
v01=COORD(0, 0, -5);
```

```
v02=COORD(5, 0, -5);
v03=COORD(5, 5, -5);
v04=COORD(0, 5, -5);
v05=COORD(0, 0, -10);
v06=COORD(5, 0, -10);
v07=COORD(5, 5, -10);
v08=COORD(0, 5, -10);
```

②次に、先程設定した座標名称を使って頂点を設定する。

リスト３－９：頂点の設定

```
p01=VERTEX(v01);
p02=VERTEX(v02);
p03=VERTEX(v03);
p04=VERTEX(v04);
p05=VERTEX(v05);
p06=VERTEX(v06);
p07=VERTEX(v07);
p08=VERTEX(v08);
```

この場合、テクスチャ座標等がないため、VERTEX は座標値しかもたない。

③先程設定した頂点名称を使って面データを設定する。

面数は６面なので６面分設定し、面は頂点が４つなので４つの頂点名称を設定する。

リスト３－１０：面の設定

```
s01=FACE(p01, p02, p03, p04);
s02=FACE(p02, p06, p07, p03);
s03=FACE(p06, p05, p08, p07);
s04=FACE(p05, p01, p04, p08);
s05=FACE(p04, p03, p07, p08);
s06=FACE(p05, p06, p02, p01);
```

④ここで、法線を設定する。

６面あるので、６つの法線を設定する。

リスト３－１１：法線の設定

```
n01=NORMAL(1, 0, 0);
n02=NORMAL(-1, 0, 0);
n03=NORMAL(0, 1, 0);
n04=NORMAL(0, -1, 0);
n05=NORMAL(0, 0, 1);
n06=NORMAL(0, 0, -1);
```

⑤先程設定した各面に法線名称を登録する。

リスト３－１２：面への法線の登録

```
FACE_NORMAL(s01, n05);
FACE_NORMAL(s02, n01);
FACE_NORMAL(s03, n06);
FACE_NORMAL(s04, n02);
FACE_NORMAL(s05, n03);
FACE_NORMAL(s06, n04);
```

⑥グループを設定する。

リスト３－１３：グループの設定

```
g01=GROUP();
```

⑦そのグループに先程設定した面データを登録する。

６面体なので、６面登録する。

リスト３－１４：グループへの面の登録

```
GROUP_FACE(g01, s01, s02, s03, s04, s05, s06);
```

⑧以上で、一つのグループが出来た。

これをファイルに保存し、景観シミュレータで表示すると以下のように表示される。

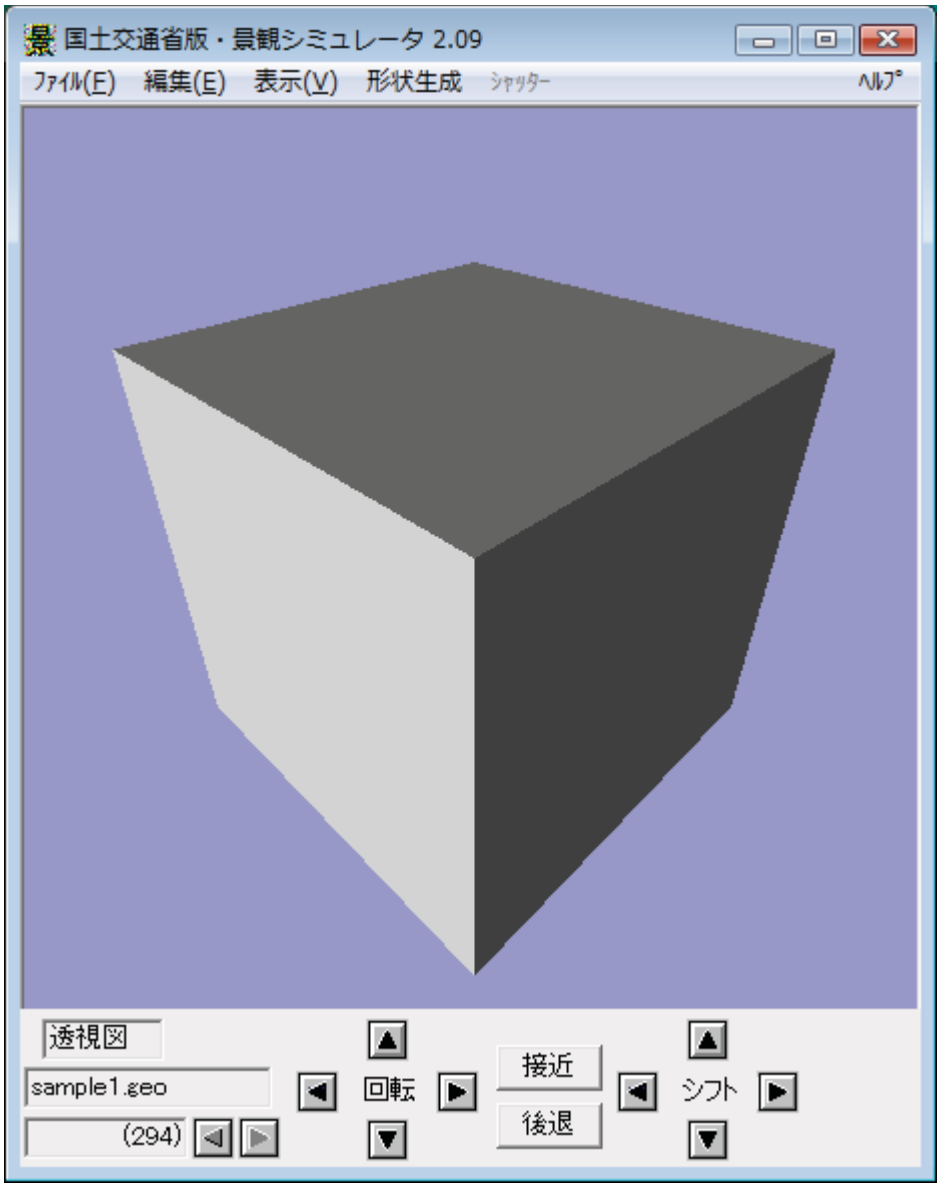


図３－５：６面から成る立方体の表示

（2）マテリアル

今度は、このデータにマテリアルを設定する。

①マテリアルを設定したい物体に適用するに先立って、予め MATERIAL コマンドを使って、マテリアル ID を定義しておく。

リスト３－１５：マテリアルＩＤの定義

```
m01 = MATERIAL(RED);
m02 = MATERIAL(WHITE);
```

②面にマテリアルを付けたい場合は、以下のように FACE_MATERIAL コマンドで登録する。

リスト３－１６：マテリアルＩＤの面への登録

```
FACE_MATERIAL(s01, m01);
FACE_MATERIAL(s02, m02);
FACE_MATERIAL(s03, m02);
FACE_MATERIAL(s04, m02);
FACE_MATERIAL(s05, m02);
FACE_MATERIAL(s06, m02);
```

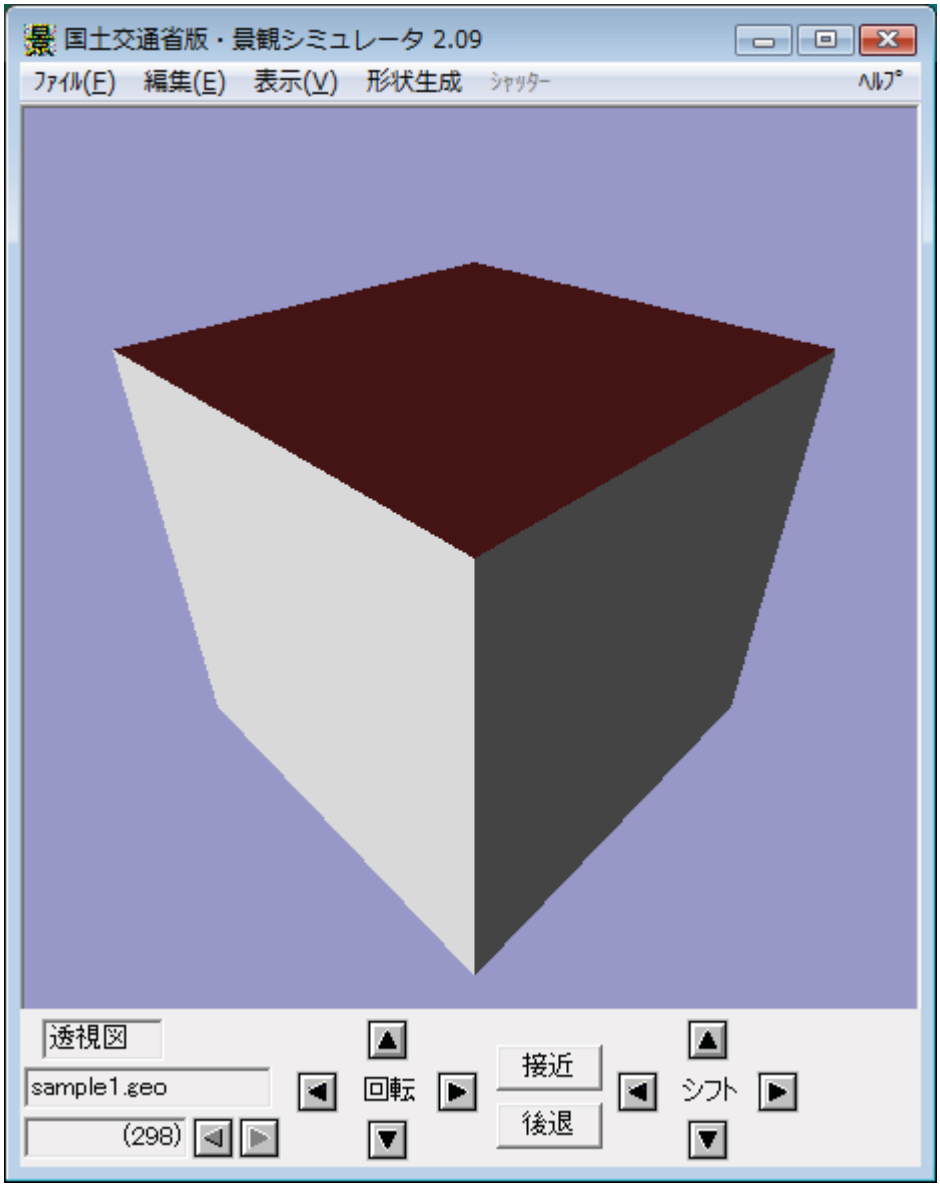


図３－６：面にマテリアルを設定する

③グループにマテリアルを付けたい場合は、以下のように GROUP_MATERIAL コマンドで設定する。

リスト３－１７：マテリアルＩＤのグループへの登録

```
GROUP_MATERIAL(g01, m01);
```

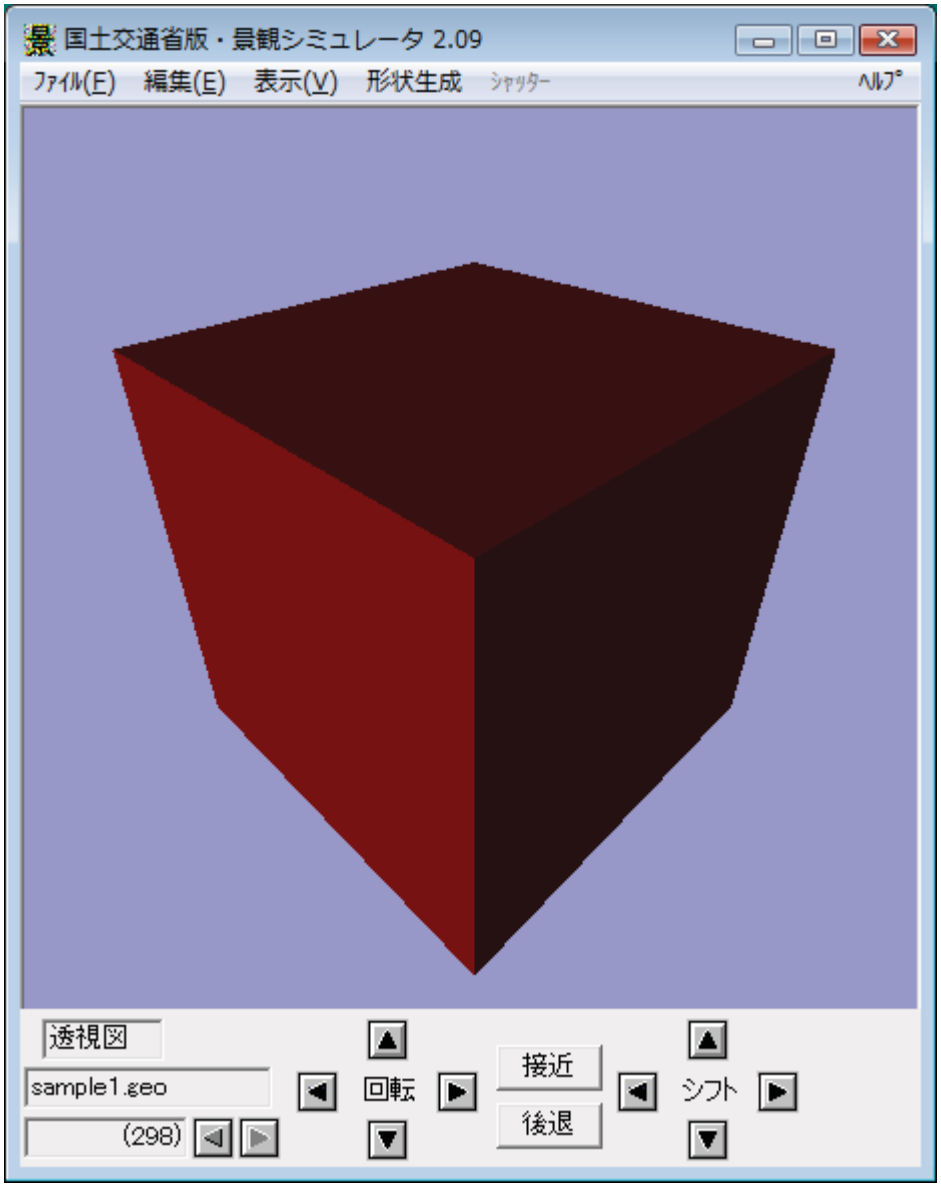


図３－７：グループへのマテリアルの設定

（４）テクスチャ

今度は、このデータにテクスチャを設定する。

①┌グループを設定する。

リスト３－１８：グループの設定

```
g01 = GROUP();
```

②座標、テクスチャ座標を１面分（４点分）設定する（この面にテクスチャを貼りつける）。

リスト３－１９：頂点座標とテクスチャ座標の設定

```
V000000 = COORD(0.000000, 0.000000, -10.000000);
T000000 = TCOORD(0.000000, 0.000000);
V000001 = COORD(5.000000, 0.000000, -10.000000);
T000001 = TCOORD(5.000000, 0.000000);
V000002 = COORD(5.000000, 0.000000, -5.000000);
T000002 = TCOORD(5.000000, 5.000000);
V000003 = COORD(0.000000, 0.000000, -5.000000);
T000003 = TCOORD(0.000000, 5.000000);
```

③先程設定した、座標名称・テクスチャ座標名称を使って、頂点を設定する。

リスト３－２０：頂点の設定

```
P000000 = VERTEX(V000000, , T000000, );
P000001 = VERTEX(V000001, , T000001, );
P000002 = VERTEX(V000002, , T000002, );
P000003 = VERTEX(V000003, , T000003, );
```

④頂点名称を使って面を設定する。

リスト３－２１：面の設定

```
F000000 = FACE(P000000, P000001, P000002, P000003);
```

⑤法線を設定する。

リスト３－２２：法線の設定

```
N000004 = NORMAL(0.000000, -1.000000, 0.000000);
```

⑥面に法線を登録する。

リスト３－２３：面への法線の登録

```
FACE_NORMAL(F000000, N000004);
```

⑦テクスチャＩＤを設定する。

リスト３－２４：テクスチャＩＤの設定

```
I000004 = TEXTURE("renga_1.sgi");
```

⑧面にテクスチャＩＤを登録する。

リスト３－２５：面へのテクスチャＩＤの登録

```
FACE_TEXTURE(F000000, I000004);
```

⑨この面をグループに登録する。

リスト３－２６：面のグループへの登録

```
GROUP_FACE(g01, F000000);
```

これで１面分を作成し、グループに登録した。以降も同様に１面ずつ作成して、グループに登録する。またこの時、同じ頂点名称（VERTEX）を使用したまま、中味の情報を更新する。更新しても今までこの頂点名称を用いて設定し、GROUP_FACE を実行した面には影響

しない。

一度設定した頂点名称を各面で共用した場合、例えば、テクスチャ座標が設定された頂点座標を共用した場合、テクスチャを貼り付けたくない面にまでテクスチャ座標が設定されてしまい、そのデータを表示した時、他の面にまでテクスチャが貼られてしまうという現象が出る。そこで、同じ名称の情報を、テクスチャの無い座標値だけの頂点定義に更新して使用する（⑩の例）か、別の頂点名称に新たな情報を設定してその名称の方を使用する事により、この現象を回避できる。

⑩以降の面は、テクスチャを貼り付けない設定を行う。

リスト３－２７：テクスチャのないその他の面の設定

```
g01 = GROUP();

V000000 = COORD(0.000000, 5.000000, -5.000000);
V000001 = COORD(5.000000, 5.000000, -5.000000);
V000002 = COORD(5.000000, 5.000000, -10.000000);
V000003 = COORD(0.000000, 5.000000, -10.000000);
P000000 = VERTEX(V000000, , , );//テクスチャ座標の定義をしない
P000001 = VERTEX(V000001, , , );
P000002 = VERTEX(V000002, , , );
P000003 = VERTEX(V000003, , , );
F000001 = FACE(P000000, P000001, P000002, P000003);
N000004 = NORMAL(0.000000, 1.000000, 0.000000);
FACE_NORMAL(F000001, N000004);
GROUP_FACE(g01, F000001);

V000000 = COORD(0.000000, 0.000000, -10.000000);
V000001 = COORD(0.000000, 0.000000, -5.000000);
V000002 = COORD(0.000000, 5.000000, -5.000000);
V000003 = COORD(0.000000, 5.000000, -10.000000);
P000000 = VERTEX(V000000); //二番目以降のパラメータを省略
P000001 = VERTEX(V000001);
P000002 = VERTEX(V000002);
P000003 = VERTEX(V000003);
F000002 = FACE(P000000, P000001, P000002, P000003);
N000004 = NORMAL(-1.000000, 0.000000, 0.000000);
FACE_NORMAL(F000002, N000004);
```

```
GROUP_FACE(g01, F000002);

V000000 = COORD(5.000000, 0.000000, -10.000000);
V000001 = COORD(0.000000, 0.000000, -10.000000);
V000002 = COORD(0.000000, 5.000000, -10.000000);
V000003 = COORD(5.000000, 5.000000, -10.000000);
//VERTEX の定義を省略（定義済み）
F000003 = FACE(P000000, P000001, P000002, P000003);
N000004 = NORMAL(0.000000, 0.000000, -1.000000);
FACE_NORMAL(F000003, N000004);
GROUP_FACE(g01, F000003);

V000000 = COORD(5.000000, 0.000000, -5.000000);
V000001 = COORD(5.000000, 0.000000, -10.000000);
V000002 = COORD(5.000000, 5.000000, -10.000000);
V000003 = COORD(5.000000, 5.000000, -5.000000);
N000004 = NORMAL(1.000000, 0.000000, 0.000000);
FACE_NORMAL(F000003, N000004);//F000003 を流用し、面の定義を省略
GROUP_FACE(g01, F000003);

V000000 = COORD(0.000000, 0.000000, -5.000000);
V000001 = COORD(5.000000, 0.000000, -5.000000);
//座標値が変化しない頂点に関して座標の定義を省略
V000003 = COORD(0.000000, 5.000000, -5.000000);
GROUP_FACE(g01, F000003);//面に垂直な法線を省略
```

上記の、テクスチャを貼らない面の定義において、順次記述を省略する方法を例示した。座標値やテクスチャ座標、その他の属性設定は、GROUP_FACE コマンドの実行により、確定する。以後、それ以前に定義したラベルを再定義しても、確定した面の定義は変化しない。このことを利用して、COORD から始めて FACE を構築するのに用いた以前の諸定義を、いわばテンプレートのように活用し、変化した数値（例えば座標値）だけを書き変えて、同じ面を用いて GROUP_FACE コマンドを実行することにより、効率的にデータを構築することができる。このことは、ファイル・コンバータを開発する際に有用である。

データのパターンに応じて、LSS－G ファイルの冒頭部分にいくつかのテンプレートとなる面を定義しておき、変化する部分だけを更新しながら、GROUP_FACE を繰り返していくと

いう方法が、コンパクトなデータに変換するために、よく用いられている。

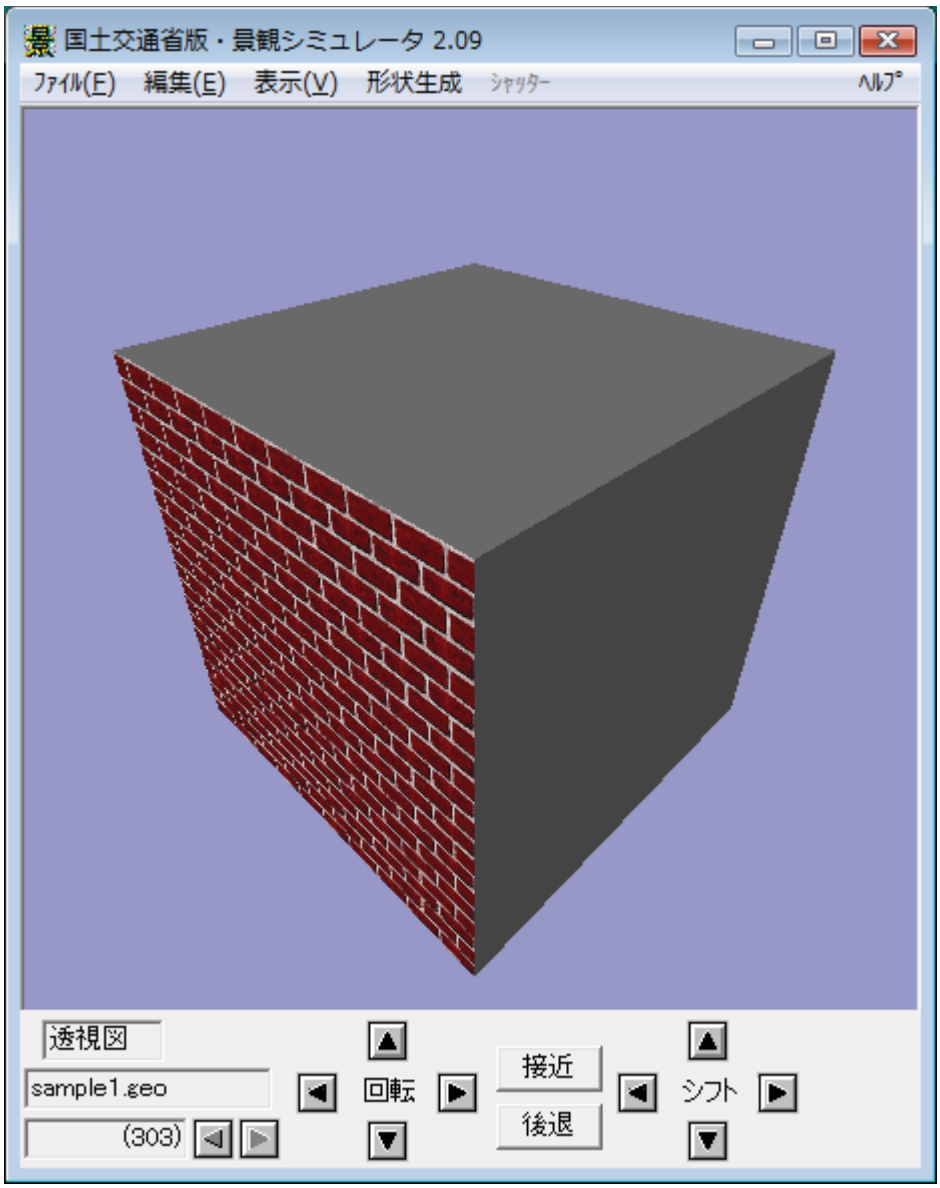


図３－８：一面だけにテクスチャを貼った立方体

（５）グループの作成（FILE コマンド）

グループを作成するもう１つの方法について説明する。

①グループを作成する方法は、以上のような作成方法の他に、すでにあるＬＳＳ－Ｇファイルを読み込んで一つのグループとして扱う方法がある。

リスト３－２８：ＦＩＬＥコマンドによるグループの設定

```
g10 = FILE(BRG_ARCH.geo);
```

FILE()コマンドは、group()コマンドの機能を一部含んでいるので、FILE()コマンドの前に

```
g10 = GROUP();
```

という設定をしておく必要はない。

（6）リンクの設定

今までは、グループを１つだけ定義した。次に、複数のグループを設定し、親子関係を設定する方法について説明する。

①┌グループを複数設定する。

リスト３－２９：リンクする複数グループの設定

```
g01 = GROUP();
g02 = GROUP();
g03 = GROUP();
g04 = GROUP();
g05 = GROUP();
```

②g01 の子供として g02 を登録する。

リスト３－３０：g01 を g02 の親とするリンクの設定

```
L000000 = LINK(g01, g02);
```

③g01 の子供として g03 を登録する。

リスト３－３１：g01 を g03 の親とするリンクの設定

```
L000001 = LINK(g01, g03);
```

④g02 の子供として g04 を登録する。

リスト３－３２：g02 を g04 の親とするリンクの設定

```
L000002 = LINK(g02, g04);
```

⑤g02 の子供として g05 を登録する。

リスト３－３３：g02 を g05 の親とするリンクの設定

```
L000003 = LINK(g02, g05);
```

以上のようなリンクを設定すると、以下のような親子関係となる。

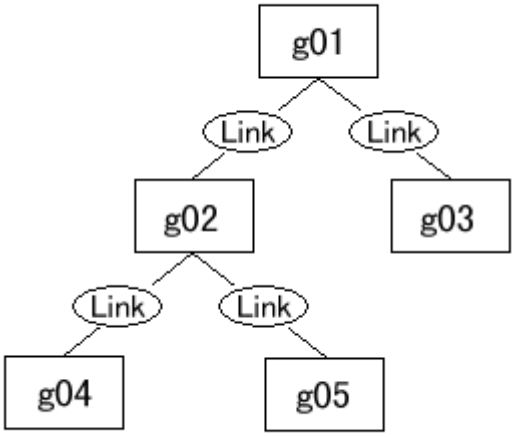


図３－９：リンク

これらのリンクによる親子の座標関係は、子の座標系が、グローバル座標系ではなく、親の座標系に準拠する。つまり、親の座標系の原点と子供の座標系の原点が一致する。

また、このリンクに変換マトリクスを定義する事も出来る。この場合、親の座標系に対する子の座標系の変換（移動・回転・拡大縮小）がマトリクスにより記述される。親がグローバル座標に関して、最上位グループとのリンク・マトリクスにより移動した場合、子はこれに付いて動く。リンクが設定されただけで、リンク・マトリクスが明示的に定義されていない場合には、デフォルト値として単位マトリクス（移動・回転・拡大縮小なし）が設定されている。

リスト３－３４：リンク・マトリクスの設定

```
例)
    LINK_XFORM(L000000, LOAD, MATRIX,
            0.707107, 0.707107, 0.000000, 0.000000,
           -0.707107, 0.707107, 0.000000, 0.000000,
            0.000000, 0.000000, 1.000000, 0.000000,
            2.500000, 2.500000, 0.000000, 1.000000);
```

LINK_XFORM コマンドの詳細に就いては、＜３－３．ＬＳＳ－Ｇコマンド＞で解説する。

（７）異なる設定が同時に行われた場合の優先順位

マテリアルおよびテクスチャの優先順位について説明する。

①┌同一属性が異なるレベルで設定された場合の優先順位

マテリアルおよびテクスチャは、面・グループそれぞれに設定できる。また、カラーと法線は、頂点・面それぞれに設定できる。

これらは、下位の方（マテリアル・テクスチャ：面、法線：頂点）の設定が、優先順位

が高く処理される。つまり、マテリアルの例を挙げると、面・グループのどちらにも同時に設定されている時には、面の設定が優先され有効となる。

②異なる属性が同一レベルで設定された場合の優先順位

面に色を設定する方法には、MATERIAL と COLOR の 2 種類がある。

カラーの定義を含むマテリアルとカラーが同じ面に同時に定義された場合、カラーが優先される。ただし、このことはマテリアル定義の内の色(RGB)についてのみであって、マテリアルの中にテクスチャ、輝度、反射率等が別途定義されている場合には、それらはそのまま適用され、カラーだけが無視されて、別途単独で COLOR 設定された色情報が用いられる。

また、MATERIAL の中には、テクスチャを含むものが存在する。同じグループまたは面に対して、テクスチャ付きマテリアルとテクスチャが同時に設定された場合、マテリアルのテクスチャが優先される。

これらのコマンドを上手に使う事により、より本物に近い質感で表示をさせる事ができる。例えば、TEXTURE コマンドで材質感を表現し、COLOR コマンドで色を微調整するといった使い方である。

（8）マテリアル・テクスチャのデフォルト値と継承

マテリアル・テクスチャが定義されていない場合には、デフォルトのマテリアル（色が{0.8,0,8,0.8,1.0}で、テクスチャその他の設定なし）が適用される。

親グループにマテリアル･テクスチャが定義された場合､そのグループ自身に属する面、これとリンクで結ばれた子グループ（更にその下の孫グループ以下)、及びその子グループに属する面に継承され､それぞれにマテリアル･テクスチャが定義されなかった場合のデフォルトの設定となる。

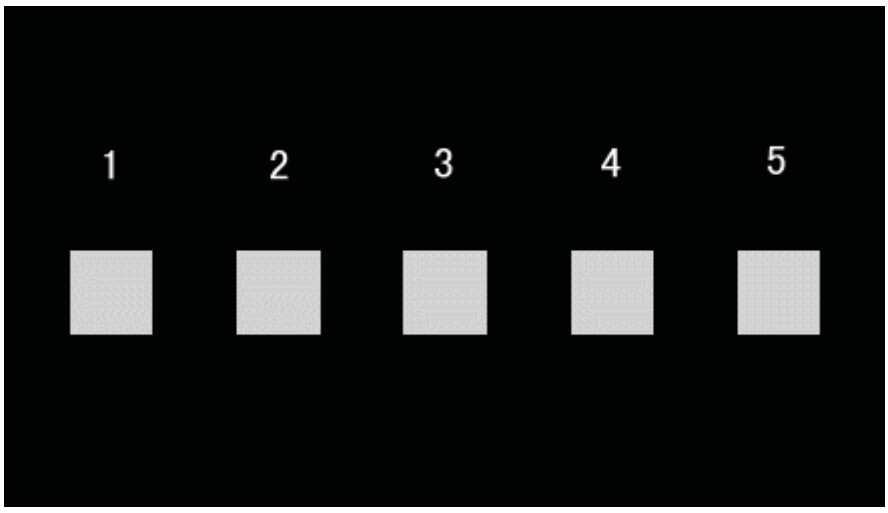


図３－１０：デフォルト状態

マテリアル・テクスチャが設定されない場合、デフォルト設定で表示される。

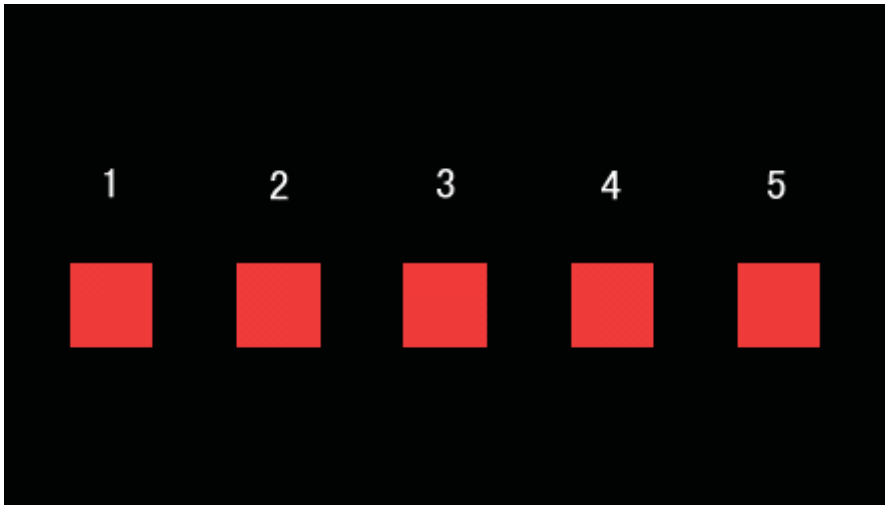


図３－１１：親への設定

親グループにマテリアル「RED」が設定されると、その下に属する、マテリアル等が特記されていない一般の面や子グループ全てに影響が及ぶ。

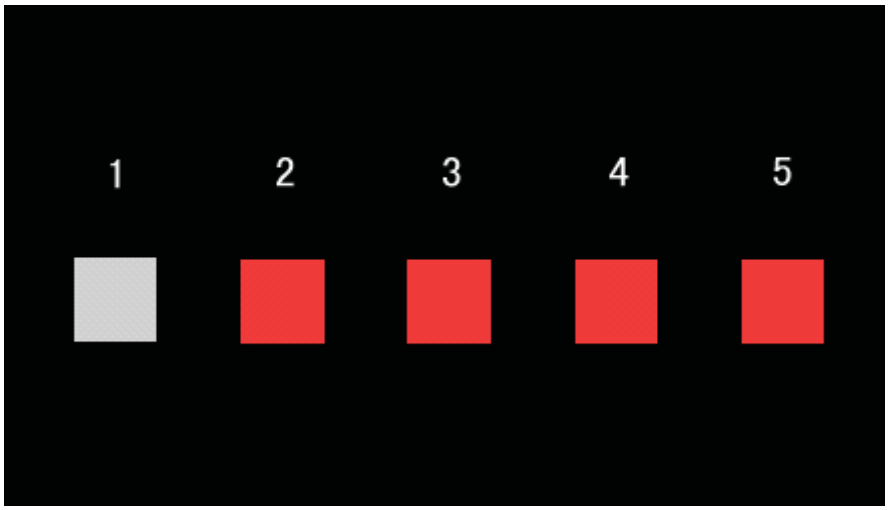


図３－１２：ユニークな設定

親グループから継承されたマテリアル「RED」が存在している状況下で特定の面またはグループ１だけ別のマテリアル「WHITE」を設定すると、その面だけユニークなマテリアルとなり、それ以外の面は親から継承した設定となる。

３－６．ファイル参照とリンクを併用したデータ構築

サンプルデータ１と、サンプルデータ２（これらは長いためデータ自体のリストは省略する）を FILE コマンドで参照して組み合わせることにより、サンプルデータ３を構成した例を示す。

（１）サンプルデータ１：街灯の支柱部分

・　　005_1.geo 表示例

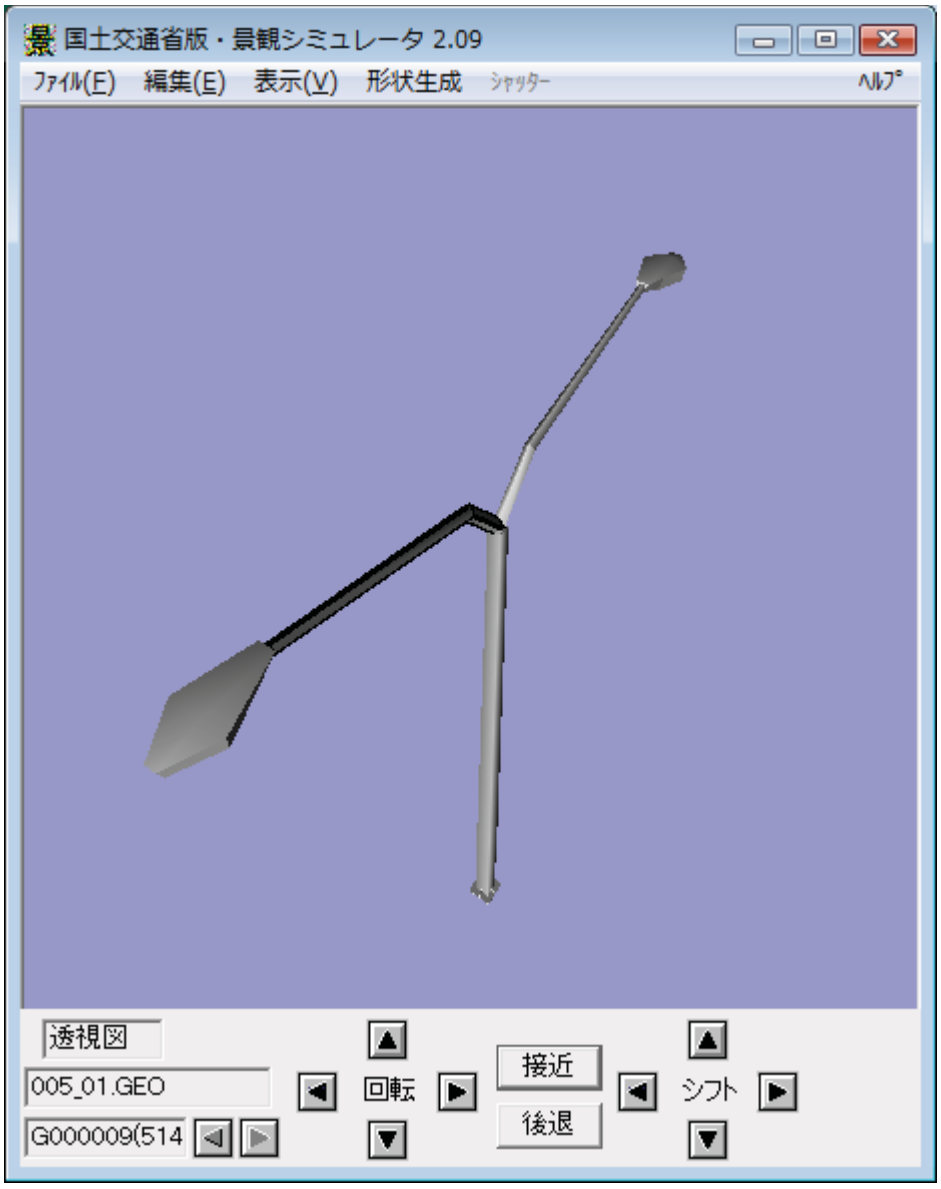


図３－１３：サンプル・データ１

（２）サンプルデータ２：街灯のランプ部分

・　　005_2. geo

表示例）

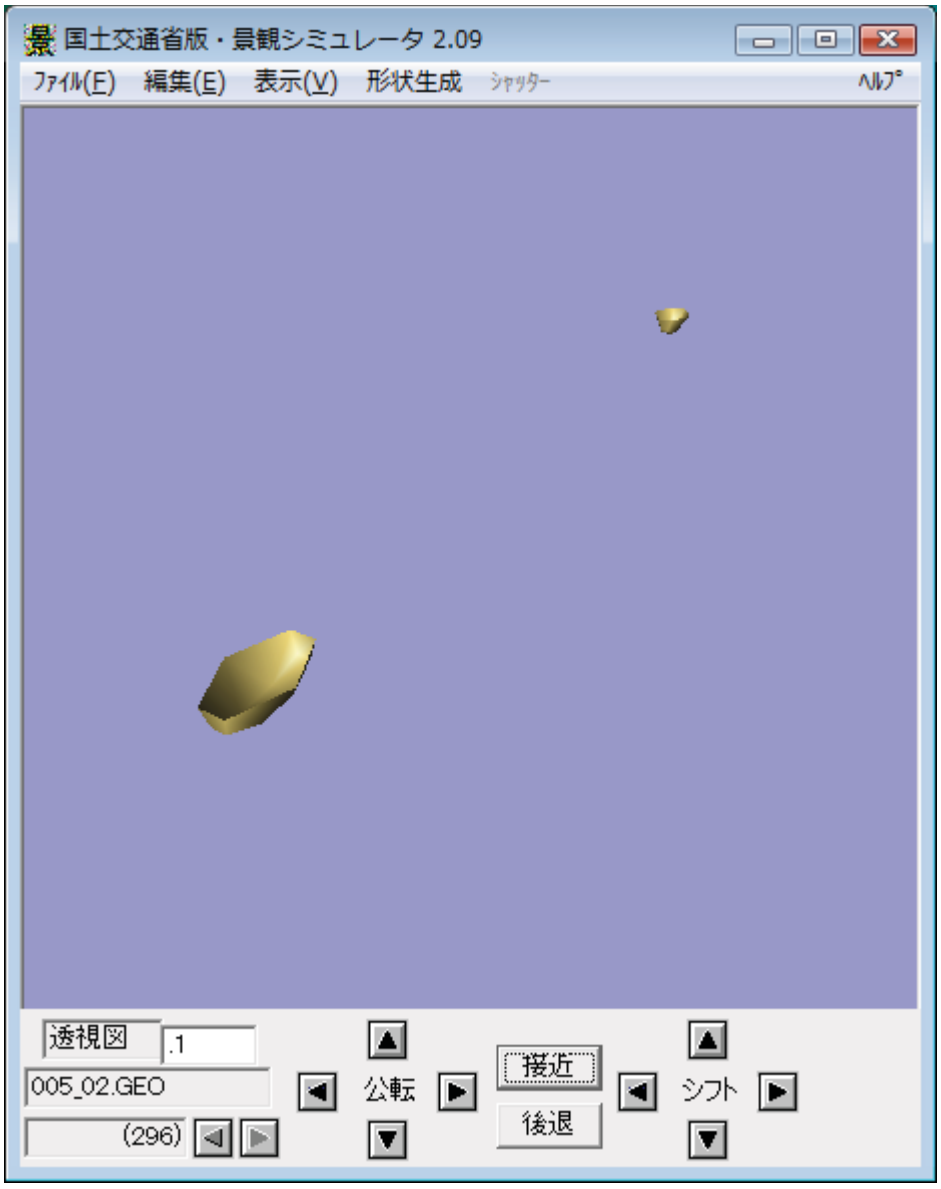


図３－１４：サンプル・データ２

（３）サンプルデータ３：街灯全体

リスト３－３５：total005.geo

```
#
    ROOT = GROUP();
#
#    Child model link
#
    G000000 = FILE(005_01.geo);
    L000000 = LINK(ROOT, G000000);
```

```
#
    G000001 = FILE(005_02.geo);
    L000001 = LINK(ROOT, G000001);
```

表示例)

図３－１５：ファイル参照とリンクによる合成

この、FILE コマンドと LINK コマンドで構築することにより、複雑な景観構成をコンパクトなファイルで記述することができる。データベースとして予め用意してある構成要素データを配置しながら構築したデータは、一般にこのような構成を有しており、景観検討のためのデータ構築作業としては極めてよく用いられる方法である。

３－７．マテリアル・ファイル

（1）　マテリアル・ファイル

マテリアル・ファイルは、景観材料データベース内のマテリアル情報を格納したデータとする構想が開発初期には存在し、一部の変数名などに名残を留めているが、実際に開発されたシステムにおいては環境設定ファイルの「FILE_PATH_MATERIAL」エントリーで定義されたディレクトリ（標準： kdb/material）にテキスト・ファイルとして並列的に置くこととなり、Ver.2.09まで継承している。

その役割は、鉄鋼、コンクリート、舗装、芝生、壁材料など、固有的形状を持たず使用される場所・状況により形状が変化する材料の固有の光学特性を記述するためのファイルである。

ひとつのマテリアル・ファイルには、一群のマテリアルが定義されている。例えば、日塗工の色見本帳は、色コードをマテリアル名とするマテリアル定義の集合体として、一つのファイル「nittoko.mtl」にまとめられている。

マテリアル・ファイルは、テキスト・ファイルであり、マテリアル名称をキーとし、材料の表面特性に関するパラメータが記述される。

マテリアル・ファイルにおいては、LSS-G ファイルの中で直接記述することができるカラー、テクスチャの他に、鏡面反射率と輝度（発光）も定義することができる。LGG-G ファイルにおいては、カラーは４つの値（RGB と透明度を表すα）で指定するが、マテリアルファイルにおいては、色彩を表す RGB または Lab 値と、透明度を表す TRANSPARENCY を分離している。

全てのパラメータに、期間を定義することができる。色彩、鏡面反射率およびテクスチャを期間毎に定義することにより、経年変化のシミュレーションを簡単に実行することができる。

マテリアル・ファイルの仕様を以下に示す。

表３－４：マテリアル・ファイルの仕様

･１カラム目が’#’の行はコメント行である。 ･マテリアル名称の宣言は１カラム目から始まる。 ･マテリアルの定義内容を関するキーワード行は１カラム目が空白またはタブであり、これにキーワード、期間の定義、パラメータが続く。 ・キーワードには次の種類がある。 ①RGB 拡散反射係数（通常のカラー）を直接ＲＧＢの３値で指定する（各 0.0≦≦1.0)。 ②Lab 拡散反射係数をＬａｂの３値で記述し、内部で変換して適用する。 ②SPECULAR 鏡面反射係数は、正反射により、表面が平滑な材料のツヤを表現する（各 0.0≦≦1.0)。

指定されなかった場合のデフォルトは0.0である。

④TRANSPARENCY

透明度は0.0≦≦1.0の範囲で指定する。0.0が透明で、指定されなかった場合のデフォルトは1.0（不透明）である。

⑤TEXTURE

テクスチャを記述する画像ファイル名称を指定する。

⑥EMISSION

放射光係数は０≦≦１の範囲で指定する（EMISSION)。

・期間の定義は、適用される時刻の範囲を指定できる（省略可)。
・期間指定には0.0以上の浮動小数点値を用い、[開始時点-終了時点]の形式で記述する。
・期間範囲がオーバーラップしないキーワードを任意個数記述できる。

リスト３－３６：マテリアルの記述例

```
ORENGE
        RGB [0-365]      1.0  0.5  0.0
        RGB [366-1000]   0.8  0.4  0.0
        SPECULAR [0-1000]        0.0
        TRANSPARENCY [0-1000]    1.0
SPECULAR_96
        RGB[0-500000]            1.0      1.0      1.0
        SPECULAR[0-500000]       0.96
        TRANSPARENCY[0-500000]   1.0
NORI
        RGB[0-99]                1.0      1.0      1.0
        RGB[100-10000]           1.0      1.0      1.0
        TEXTURE[0-99]            slop_1.sgi
        TEXTURE[100-10000]       slop_2.sgi
WHITE3
        RGB                      1.0      1.0      1.0
        SPECULAR                 0.0
        TRANSPARENCY             1.0
        EMISSION                 1.0
```

マテリアル・ファイルのロードは、ＤＢＩＬライブラリのdbLoadMaterial関数(dbms.c)により行われる。マテリアルのロードは、期間の定義が、現在のシステムの時刻に合致するキーワードに関してのみ実行される。

LSS-G ファイルのロードに際しては、インタープリタにより、LSS-G ファイルから参照されたマテリアルのリストが作成され、ロードの最終段階にリストされた全てのマテリアルが、dbLoadMaterial 関数によりロードされる。

マテリアル編集ダイアログ(４－４(21),(24))で、あるマテリアルが選択された場合には、そのマテリアルがロードされる。また、グラフィックなマテリアル編集ダイアログ(４－４(25))では、選択したマテリアルファイルで定義される全てのマテリアルをロードし、一覧表示する。

経年変化ダイアログ（４－４(32)）の操作や、シーンの切換に伴い、システムの時間が変更された場合にも、新たに設定されたシステム時間を用いてマテリアルの再ロードを行う。

Ver.2.09 のマテリアル・ファイルのロードに際して、以下のようにエラー処理している。
・名称の中にキーワードが一つもない場合：警告(3345)を発し、マテリアルは適用しない。
・無効なキーワードが宣言された場合：警告(3343)を発し、有効キーワードのみを適用。
・期間の定義がない場合：システム時間にかかわりなく常にロードする。
・同じマテリアル名称が重複して登録されていた場合：最初に検出したものを採用する。
・有効なキーワードが複数存在する場合：最後の有効なキーワードが適用される。
・拡散反射係数に係のキーワードが存在しない場合：デフォルト値 0.8 0.8 0.8 を適用。
・パラメータが必要数に不足する場合：存在する分のみ適用し、残りはデフォルト値。
・パラメータが必要数を超える場合：必要な数のみ適用し、残りは無視する。
・期間の開始時点が負数であった場合には、警告(3338)を発し、そのマテリアルを適用しない。

LSS-G 形式のファイルにおいて、木・石・鉄・アスファルト・コンクリート・ガラスなどの基本的な素材の光学特性を直接数値定義するのではなく、マテリアルとして間接定義しておくことにより、将来の表示処理技術の高度化に際して、LSS-G 形式のファイルを変更することなく対応が可能となる。また、表示に直接関係しない、重量や物質含量や価格等の集計分析も可能なデータとなる。

３－８．画像ファイル

景観シミュレーション・システムの開発に着手した 1993 年当時は、TIFF、BMP 形式などもされていたが、基本的な画像保存形式として、SGI 形式（グラフィック・ワークステーションと OpenGL を提供していた Silicon Graphics 社による公開性の高い単純な形式）を採用した。RGB という拡張子も用いられる場合がある。IMGSGI.lib ライブラリで入出力処理を行っている。

Ver.2.03 以降、Windows 上で背景画像等を利用するために、BMP 形式のファイルを読み込めるように拡充した。Windows 系の開発環境で入出力関数が用意されている。

TIFF 形式は、Aldus 社から仕様書を入手することができる公開性の高い形式であったが、

圧縮方法を含め様々なバリエーションを含む形式であるため、フルスペックでは対応していない。しかし、衛星画像を配布する際に用いられている GeoTiff 形式は、この内の単純な部分仕様に従っているため、支障なく使用することができる。

その後、圧縮効果の高い JPEG 形式が、デジタルカメラ等の普及に伴い広く利用されるようになったため、背景画像やテクスチャとして取り込めるようにした。また、表示画面をファイル保存し、報告書等に使用する際にも便利であるため、出力ファイル形式としても選択可能としている。しかし、圧縮に伴い、ファイルは小さくなるものの、画像データとしての同一性は失われるため、画像解析的な目的には適さない。JPEG.lib で入出力を行っている。

GIF 形式は、入出力プログラムにライセンスの制約があるため採用していない。景観シミュレーションに使用するためには、画像編集ソフトなどを用いて別途変換する必要がある。PNG 形式は国土地理院による電子国土等に用いられている。サーバー側の機能開発では、画像の WEB 配信に使用したが、クライアント側で動作する本システムでは、まだ対応していない。これも別途変換する必要がある。

SGI 形式は、現在では殆ど流通に用いられておらず、ファイル形式は単純で固定されているため、将来的に仕様変更される危険性が小さく、データベースに登録した基本的構成要素のためのテクスチャなどを変更することなく長期に保存するためには最も適していると言える。最大の特徴は、アルファ値を有する画像を記述できる点であり、樹木等を長方形にテクスチャを貼ったオブジェクトとして表現することは、JPEG などの形式では不可能である。SGI 形式における圧縮は、同じビットパターンが連続する場合に、そのパターンと繰り返し数に置き換えるという単純な方法であるが、面や背景が、同じカラーまたは透明部分を有するような CG 画像をコンパクトに保存するためには効果的である。

３－９．エラーメッセージ定義ファイル

テキスト・ファイルであり、ASCII コードまたはこれを拡張したシフト JIS コード等により記述する。システム起動時、または言語切換時に z3LoadMessage()関数によりロードされ、以後メモリ上に置かれ、各種メッセージ表示が必要となる場合に利用される。メッセージは、種別と番号によりアクセスされ、z3Message(メッセージ番号,引数・・・)等の関数により表示に使用される。ファイルのフォーマットは以下の通りである。

① 行単位でエラーメッセージを定義する。

② 行の最初の１文字が、アルファベット大文字の E、W、I、C で始まる行のみをデータとして解釈し、それ以外の文字で始まる行や、単に改行のみの行はコメント解説等として読み飛ばす。

③ 上記の１文字コードは、メッセージの種類を区別する。

E:エラー(Error)

W:警告(Warning)

I:情報(Information)

C:確認(Confirmation)

原則として、システムの側に責任が帰する障害の場合に E を、またユーザーの誤操作等に帰する障害の場合に W を用いる。C(確認)は、通常は z3Confirmation 関数で用い、「はい」または「いいえ」の選択結果(戻り値)を続く処理に反映させる。

④　1 文字コードの後にスペース区切りで、メッセージ番号を記述する。

メッセージ番号は、メッセージの種類に関わらずユニークでなければならない。整理のために、エラーは 1000 番台、警告は 3000 番台、情報は 5000 番台、確認は 7000 番台としているが、必ずしもこの慣習に従う必要はない。また番号が昇順に並んでいる必要もない。

⑤　メッセージ番号の後にスペース区切りで改行まで続く文字列をエラーとして表示する。

この文字列は、printf 文のフォーマット文字列と同様に、引数で指定する整数値(%d)、浮動小数点値(%f)、文字列(%s)を含めることができる。④　W　の場合、警告として表示する。ERR_MSG.TXT ファイル中におけるメッセージの定義例をリストに、またこれを用いたエラーメッセージの表示のプログラム例をリストに示す。

リスト３－３７：ERR_MSG.txt による定義

```
W 3006 %s のセットに失敗しました
```

リスト３－３８：エラーメッセージのプログラム例

```
CString s;
int i;
s.Format("EFFECT(%s",e->params[0]);
for(i=1;i<e->count;i++){
        s += _T(",");
        s += _T(e->params[i]);
}
s += _T(")");
z3Message(3006,s);
```

(3)実際の表示

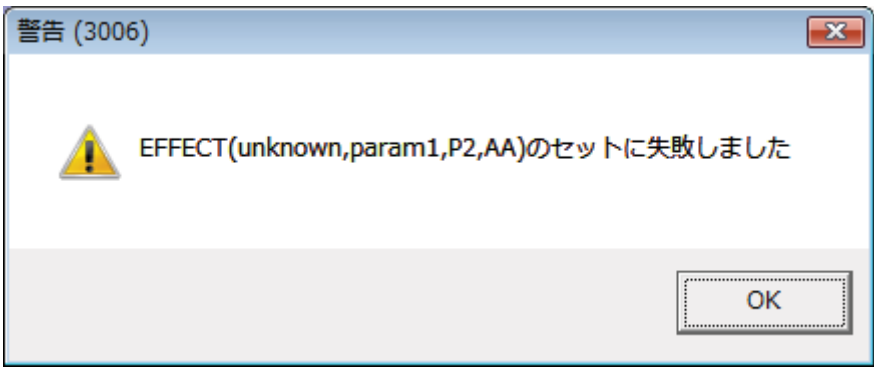


図３－１６：警告の表示例

３－１０．選択項目定義ファイル

ユーザーが選択するアイテムを、プログラム中に固定的に記述するのではなく、外部テキスト・ファイルにおいて定義しておき、これを用いて選択ダイアログやメニューの表示を行うものである。これには以下のものがある。

（1）ＥＸＴ．ＴＡＢ

外部関数の一覧を格納する。ksim/bin に置かれる。選択可能な外部関数の範囲を定めると共に、引数の型を指定する。新たな外部関数を作成し追加する場合には、このテーブルに登録する。

改行を単位とするレコードである。#から始まる行はコメントとして無視される。

行のはじめはFILE から始め、()の中に、外部関数名とこれに続く引数をデータ型で記述し、「;」（セミコロン）で終了する。

データ型は、表３－４に示す通りである。

表３－５：EXT.TAB の引数のデータ型

INT	整数
FLOAT	浮動小数点
DOUBLE	倍精度実数
STRING	文字列
FILE	ファイル名称
FACE	指定されたＬＳＳ－Ｇファイルを開き、最初のグループの最初の面を取得する。
LINE	指定されたＬＳＳ－Ｇファイルを開き、最初のグループの最初の線を取得する。
TIME	現在の経年（日数）を倍精度実数の形式でシステムから受け取る。

Ext.tab ファイルは、システム起動時に読み込まれ、メニューの[形状生成][オプション]が選択された時に、選択ダイアログを開き、各関数の名称の一覧を表示する。

リスト３－３９：EXT.TAB

```
#外部関数一覧
FILE(CUBE,DOUBLE,DOUBLE,DOUBLE,DOUBLE,DOUBLE,DOUBLE);
FILE(SPHERE,DOUBLE,DOUBLE,DOUBLE,DOUBLE);
FILE(CYLINDER,DOUBLE,DOUBLE,DOUBLE,DOUBLE,DOUBLE,DOUBLE,DOUBLE);
FILE(CONE,DOUBLE,DOUBLE,DOUBLE,DOUBLE,DOUBLE,DOUBLE,DOUBLE,DOUBLE);
FILE(FLATCYLI,DOUBLE,DOUBLE,DOUBLE,DOUBLE,DOUBLE,DOUBLE,DOUBLE,INT);
FILE(FLATCONE,DOUBLE,DOUBLE,DOUBLE,DOUBLE,DOUBLE,DOUBLE,DOUBLE,DOUBLE,INT);
#FILE(SWEEP,FACE,DOUBLE,DOUBLE,DOUBLE);
FILE(STEEL,DOUBLE,DOUBLE,DOUBLE,DOUBLE,DOUBLE);
FILE(SWEEP1,FACE,LINE);
FILE(SWEEP2,FACE,FACE);
#FILE(SWEEP1,FILE,FILE);
#FILE(SWEEP2,FILE,FILE);
FILE(HSTEEL,DOUBLE,DOUBLE,DOUBLE,DOUBLE,DOUBLE);
FILE(CSTEEL,DOUBLE,DOUBLE,DOUBLE,DOUBLE,DOUBLE);
FILE(TSTEEL,DOUBLE,DOUBLE,DOUBLE,DOUBLE,DOUBLE);
FILE(LSTEEL,DOUBLE,DOUBLE,DOUBLE,DOUBLE,DOUBLE,DOUBLE,DOUBLE);
FILE(SAMPLE,DOUBLE,DOUBLE,DOUBLE);
FILE(STRING,STRING,STRING,STRING);
```

```
FILE(STAIR,DOUBLE,DOUBLE,DOUBLE,DOUBLE);
FILE(TIMESPHERE,DOUBLE,DOUBLE,DOUBLE,TIME);
FILE(PERIOD,FILE,DOUBLE,DOUBLE,TIME);
FILE(VRML2LSS,FILE);
FILE(URL,STRING);
FILE(HAKOBUIL,DOUBLE,DOUBLE,DOUBLE,FILE);
FILE(BS2LSS,STRING);
FILE(TAMENTAI,INT,DOUBLE);
FILE(CONE,DOUBLE,DOUBLE,DOUBLE,DOUBLE,DOUBLE,DOUBLE,DOUBLE,DOUBLE);
FILE(STEEL);
FILE(SEGITIGA,DOUBLE,DOUBLE,DOUBLE);
FILE(FIRE2LSS,FILE);
FILE(SCADEC2,STRING,STRING);
```

Ver.2.09 においては、原始図形とユーザー定義の外部関数の内部処理を一元化したが、メニュー構成においては、従前の操作環境を維持している。このため、上記におけるメニューのオプションから起動するダイアログにおいては、ext-tab に登録されている関数の全てを表示するのではなく、この内、原始図形及び基本構成要素：型鋼以外のものについてのみ、表示している。

（2）ＰＬＵＧＩＮ．ＴＡＢ

プラグインＤＬＬの一覧を格納する。ksim/bin に置かれる。選択可能なプラグインＤＬＬの一覧を記述する。新たな外部関数を作成し追加する場合には、このテーブルに登録する。コンマ「,」区切りで、表示する名称と、DLL の名称を１行単位で記述する。

リスト３－４０：PLUGIN.TAB

```
地形編集,land.dll
トンネル,TUNNEL.dll
道路法面生成,nori.dll
園路生成,ParkRoad.dll
日本語版,test.dll
```

基幹部分のメニューから[形状生成][プラグイン]を選択すると、このファイルに登録されている見出し文字列が、サブメニューとして一覧表示される。

Ver.2.09 の多言語環境においては、メニュー項目として表示する文字列の部分のみ各言語に翻訳し、ファイル名称は、autotex.set のままで、各言語ディレクトリ下に置く。

（3）ＲＯＡＤ＿ＳＥＣ．ＳＥＴ

道路生成ダイアログ（[形状生成]［基本構成要素］[道路］で起動）で選択する道路断面ファイルの一覧を記述する。道路断面ファイルはＬＳＳ－Ｇ形式で、断面の構成要素をＬＩＮＥコマンドで定義している。ROAD_SEC.SET ファイルは、kdb/geometry ディレクトリに置かれる。平面生成ダイアログ（[形状生成]［原始図形］[平面]）で作成した図形を道路断面として保存した場合には、ファイル名が、ROAD_SEC.SET ファイルの末尾に自動的に追加される。ファイルの内部は、改行区切りで名称を列挙している。

リスト３－４１：ROAD_SEC.SET

```
roadsection1.geo
roadsection2.geo
roadsection3.geo
1117doro.geo
Sarijadi.geo
```

```
senro.geo
```

（4）ＲＩＶＥＲ＿ＳＥＣ．ＳＥＴ

河川生成ダイアログ（[形状生成[]基本構成要素]［河川]で起動）で選択する河川断面ファイルの一覧を記述する。河川断面ファイルはＬＳＳ－Ｇ形式で、断面の構成要素をＬＩＮＥコマンドで定義している。RIVER_SEC.SET ファイルは、kdb/geometry ディレクトリに置かれる。平面生成ダイアログ([形状生成][原始図形[]平面]で起動)で作成した図形を河川断面として保存した場合には、ファイル名が、RIVER_SEC.SET ファイルの末尾に自動的に追加される。ファイルの内部は、改行区切りで名称を列挙している。

リスト３－４２：RIVER_SEC.SET

```
riversection1.geo
riversection2.geo
riversection3.geo
riversection4.geo
1117kawa.geo
Sarijadi.geo
senro.geo
```

（5）ＴＵＮＮＥＬ＿ＳＥＣ．ＳＥＴ

トンネル生成ダイアログ（Ver.2.09 においてはプラグインＤＬＬとして分離）で選択するトンネル断面ファイルの一覧を記述する。トンネル断面ファイルはテキスト形式で、断面の断面形を独自形式で定義している。TUNNEL_SEC.SET ファイルは、kdb/geometry ディレクトリに置かれる。ファイルの内部は、改行区切りで名称を列挙している。

リスト３－４３：TUNNEL_SEC.SET

```
riversection1.geo
riversection2.geo
riversection3.geo
riversection4.geo
1117kawa.geo
Antapani.geo
Ankyo.geo
```

（6）ＥＮＲＯ＿ＳＥＣ．ＳＥＴ

園路生成ダイアログ（Ver.2.09 においてはプラグインＤＬＬとして分離）で選択する園路断面ファイルの一覧を記述する。園路断面ファイルはＬＳＳ－Ｇ形式で、断面の構成要素をＬＩＮＥコマンドで定義している。ENRO_SEC.SET ファイルは、kdb/geometry ディレクトリに置かれる。ファイルの内部は、改行区切りで名称を列挙している。

リスト３－４４：ENRO_SEC.SET

```
roadsection1.geo
roadsection2.geo
roadsection3.geo
roadsection4.geo
1117doro.geo
```

（7）ＡＵＴＯＴＥＸ．ＳＥＴ

テクスチャ編集画面のメニューにある［自動貼り付け］を選択した時に表示するメニューを階層的に定義する。

リスト３－４５：AUTOTEX.SET

```
1 水面
        2 速い流れmizu-D1.sgi 2 2
        2 やや速い流れmizu-E1.sgi 2 2
        2 普通の流れmizu-C1.sgi 2 2
        2 遅い流れmizu-B1.sgi 2 2
        2 静止水面mizu-A1.sgi 2 2
1 法面
        2 コンクリートslop_1.sgi 3 3
        2 芝１slop_2.sgi 3 3
        2 芝２slop-E1.sgi 1 1
1 ブロック
        2 石１rock_1.sgi 1 1
        2 石２slop-G1.sgi 1 1
        2 石３slop-G2.sgi 1 1
        2 煉瓦renga_1.sgi 1 1
```

ファイルの内部は、改行で区切られた行単位のデータで構成される。一つのデータは、スペース区切りで、メニューの階層を示す数値、メニュー表示に用いる各言語の文字列、テクスチャファイル名、横倍率、縦倍率を記述する。Ver.2.09 においては、メニュー項目として表示する文字列の部分のみ各言語に翻訳し、ファイル名称は、autotex.set のままで、各言語ディレクトリ下に置く。

３－１１．その他の一時的ファイル

景観シミュレーションの操作を行う際に、一時的ファイルが動的に生成される。

（1）景観データベースとのインターフェース

基幹部分における配置ダイアログにおいて、配置するオブジェクトを選択する際に、検索のために景観データベース検索画面(別実行形式)を起動することができる。検索された結果は、一時的ファイル tmp001.txt により基幹部分に返される。このファイルは、スペース区切りで、データベース種類、ファイル種類(画像、モデル等)、およびファイル名を示す１行のデータから成るテキスト・ファイルであり、検索画面の起動時には、

０ ０ NULL

に初期化される。検索が行われると、

１ ８ M_3a.geo （景観構成要素、LSS-G 形式、ファイル名「M_3a.geo」の意味）

のように、具体的なファイルを示す内容が記入されて基幹部分に返される。

検索画面において、検索された物件の３次元データを確認表示する際に、ファイルを指定して、別プロセスとして基幹部分が起動される。この時、基幹部分が初期表示するファイルを指定するために、tmp002.txt が使用される。このファイルの内容は、tmp001.txt と同様である。

この他に同様の形式の tmp003.txt が、ファイル保存操作の記録に用いられている。

これらの一時的ファイルは、ksim/temp ディレクトリに作成される。入出力は、アプリケーション・ライブラリ関数（４－３）の WriteToFile 関数、及び ReadFromFile 関数により処理している。

（２）成熟都市シミュレータ等とのインターフェース

市街地自動生成を行う成熟都市シミュレータ（文献 16）は、景観シミュレータを三次元表示のための出力装置として使用する。データ送出は、設定ファイル timerpath.txt の１行目のフラグファイルと２行目のデータファイルを用いて行う。市街地生成の側では、フラグファイルの内容がゼロであることを確認した上で、データファイルに新しいデータを記録し、フラグファイルを１にセットする。景観シミュレータの側では、フラグが１であることを確認して、データファイルを読み込み、フラグをゼロにセットする。データファイルの内容は、ＬＳＳ－Ｇ形式である。

（３）外部関数ダイアログとのインターフェース

ユーザーが外部関数（原始図形およびユーザー定義）のパラメータ設定ダイアログを操作し、OK で終了した場合には、「関数名.geo」という名称の１行だけから成る、パラメータを記述したファイルが ksim/temp ディレクトリに作成され、次の形状生成過程で使用される。また、過去に外部関数を用いて生成したオブジェクトが選択され、パラメータが再設定される場合にも、同様のファイルが作成され、ダイアログ部の初期表示のパラメータの受け渡しに使用される（５－２参照)。

（４）外部関数による形状生成時

パラメータの設定が終了し、外部関数が実際に形状を生成する際には、外部関数により「外部関数名.g」というファイルが、ksim/temp ディレクトリに作成される。制御が戻ってきた段階で、基幹部分でこのファイルを開き、内容を解析して、地物の中に追加する（５－３参照)。