４．プログラム構成

本章では、景観シミュレーション・システム全体のプログラムの基本的構造を解説する。これは、システム全体管理、内部のデータ構造とデータ管理、ユーザー・インターフェース、ファイル入出力、ＯＳとのインターフェース等の要素を含む、一般的な三次元データを扱うシステムに必要な基本的構成を有している。この上に様々な景観検討のための機能を実現した。基本的な部分の理解は、特定の目的のための機能を今後開発していくために欠くことができない基礎となっている。

例えば、シーンを平面図として表示し、そこにデータベースからオブジェクトを選んで、配置操作を行う画面を取り上げてみよう。この画面には、以下のような機能が関係している：

①この編集画面構成のデザインに従って、ウィンドウを初期化し表示する機能
②シーンを構成する地物全体のデータを取得し、その内の指定された表示範囲を、指定された寸法の画面に配置図としてグラフィックに表示する機能
③配置するオブジェクトをデータベースから検索・選択する機能
④配置する場所、路線区間、エリアなどを画面で設定する機能
⑤配置する密度や間隔、向きや位置やサイズの揺らぎ等を設定する機能
⑥各配置地点の地面の標高を計算する機能
⑦配置の処理を実行し、その結果に基づいて地物構成を変更する機能
⑧結果が期待通りでなかった場合に、取り消して元の状態に戻す機能
⑨処理の途中でエラーが発生した場合に、メッセージを表示する機能
⑩ユーザーの理解を助けるためのヘルプの表示

これらを根底で支えているのが、ライブラリ関数群であり、システムを運用する土台となる基本的なデータ構造を管理している。

一方、ユーザーから最も近いところで、画面に表示されたボタン等のコントロールに対する操作や、画面上でのマウス操作を受け付け、対応する機能をライブラリから起動し、処理結果を画面表示に反映させるハンドラ・ルーチンがある。

更に、これらの間をつなぐ、中間的なアプリケーション・ライブラリ関数がいくつか介在している。

４－１．概要

（1）景観シミュレータと景観データベース

景観シミュレータは、単独で起動し動作するアプリケーションとして使用することもできるが、データベースと組み合わせて、データベースから起動し三次元モデルを表示したり、ウェブ・ブラウザと組み合わせて、インターネット上に公開されている三次元データを表示したりするような使い方にも対応している。

景観データベースには、国土交通省がデータを蓄積する事例データベース、景観検討データを構築する上で頻繁に利用される一般的な地物の三次元モデルを集めた、構成要素データベース、及びマテリアルを集めた景観材料データベースから成っている。

90 年代の初期のバージョンにおいては、これら全てをユーザーが操作する PC の中に置き、連携しながら作業を進める環境を前提としていた。2001 年に開発したまちづくり・コミュニケーション・システムにおいては、データを WEB 上に置き、必要に応じてダウンロードし、作業を行うような環境も実現した。

これら全体構成の内、サーバー側で動作するデータ配信機能や GIS 機能のソフトウェアは、別稿に譲ることとし、本書では、クライアント側（PC）で動作する景観シミュレータを中心とするシステムの構成について解説している。

アプリケーションとしては、景観シミュレータ（sim.exe）、３種類の景観データベースブラウザ（yuu.exe, kou.exe, zai.exe）が主なものである。それ以外に、ファイル・コンバータや、データベース入力機能などの支援プログラムが存在している。

更に、景観シミュレータと一体で動作する多くの外部関数(第 5 章)、及びプラグインＤＬＬ(第 6 章)が、それぞれ独立したビルドとなっている。

表４－１：景観シミュレーション・システムを構成する実行形式

Sim.exe　景観シミュレータ基幹部分
Yuu.exe　景観事例データベース検索
Kou.exe　景観構成要素データベース検索
Zai.exe　景観材料データベース検索
Editor.exe データベース入力編集機能
貿易.exe および関連するコンバータ別実行形式
外部関数
プラグイン DLL

（２）ソースコードの全体構成

景観シミュレーションのシステムは、大きく見て、８のライブラリと、ダイアログ毎の８０程度のハンドラ・ルーチンがその大半を占めている。前者は、開発初期の頃から一貫したデータ構造を処理するために維持・改良されてきたものである。可搬性が高いＡＮＳＩ－Ｃ言語をベースとして記述されている。初期の頃と比較すると、データ規模も大きくなってきており、またマシンのメモリ容量、ハードディスク容量、処理速度などは比較にならない程進歩したが、基本的な構造は何も変わっていない。

一方、後者は、現場のニーズや、新たなＯＳや開発環境に適応しつつ、柔軟に改良・追加を加えた結果、現状に至っている。全体の構成ダイアグラムは、図４－１に示したような構成となっている。

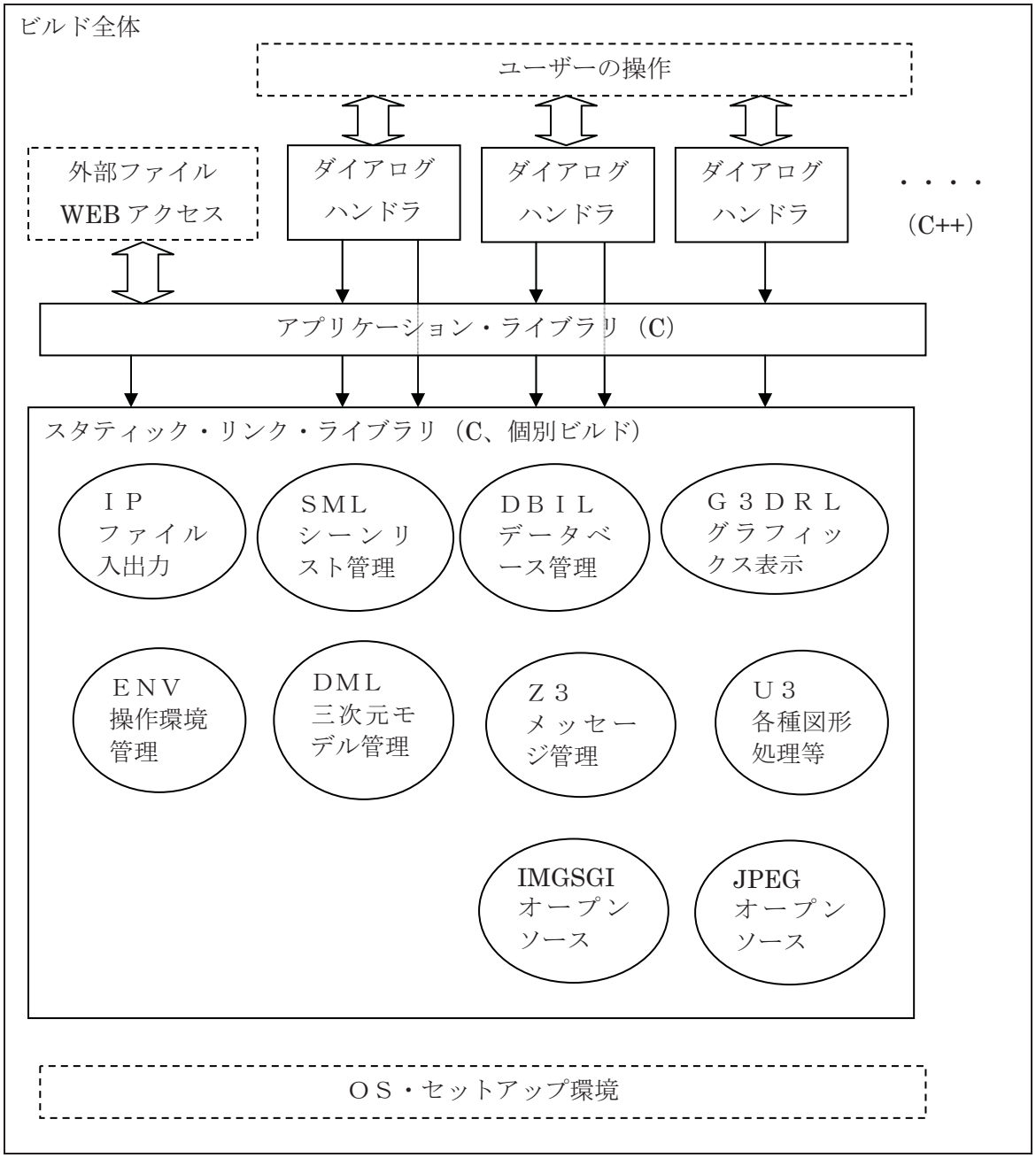

図４－１：景観シミュレータのビルド構成（参照元→参照先）

開発環境のビルドの編成は、まず C 言語による共通のライブラリが、基本的なデータ構造の管理機能を提供しており、スタティック・リンク・ライブラリ(.lib)としてライブラリ単位のビルドとなっている。景観シミュレータ、３種類の景観データベース検索機能などの実行形式(.exe)のビルドは、ユーザー・インターフェースを構成する各ダイアログのハンドラ（C++のクラス）のソースコード（Ｃ＋＋）と、アプリケーション・ライブラリのソースコード（Ｃ）から構成されているビルドに、予めビルドされた上記のライブラリ関数群をスタティック・リンク・ライブラリ(.lib)として組み込む流れとなっている。

①ライブラリ関数（C)

１章で示した開発当初のマルチ・プラットフォームの環境から、ライブラリはOSに依存しない機能としてC言語で記述されている（本章第2節)。共通ライブラリとしては、

Sml.lib：シーンリストを管理する（付録B-1　シーン管理ライブラリ）

Dml.lib：三次元モデルを管理する（付録B-2　データ管理ライブラリ）

Dbil.lib：景観データベースを管理する（付録B-3　景観データベースライブラリ）

G3drl.lib：OpenGL画面の描画処理を管理する（付録B-4　レンダリングライブラリ）

Ip.lib：外部ファイル入出力を管理する（付録B-5　インタプリタライブラリ）

Env.lib：動作環境を管理する（付録B-6　環境設定ライブラリ）

U3.lib：各種図形演算処理などのパッケージ（付録B-7　ユーティリティライブラリ）

Z3err.lib：エラーメッセージを処理する（付録B-8　メッセージライブラリ）

がある。この他に、公開オープンソースを利用しているものとして、

JPEG.lib：JPEG形式の画像ファイルの入出力

IMGSGI.lib：SGI形式の画像ファイルの入出力

をリンクしている。

ライブラリ関数からアプリケーション・ライブラリ関数やダイアログのハンドラを起動することはない。しかし、ライブラリのうちのいくつかは依存関係にある。これについては、（６）で解説する。

②アプリケーション・ライブラリ関数（C）

この他、C言語のライブラリと、C++言語のダイアログをつなぐ役割を果たす、C言語の共通関数群を用意している。主要なものは、主としてシステム状態を記憶するスタティック変数を管理するcommon.cと、データ変換を行っているdataope.cに記述された関数群があり、これらから起動する下請け的な関数を提供するいくつかのC言語のソースコードがある（本章第３節)。

この内、言語別のバージョン枝分かれの問題を解決するためにVer.2.09で新たに追加した、多言語対応のMULTILANG機能については、第１１章で解説している。

アプリケーション・ライブラリ関数からダイアログ・ハンドラを呼び出すことはない。

③ダイアログ・ハンドラ（C++)

景観シミュレータの外見を構成する大小８０程度のダイアログに関しては、C++のクラスとして記述された、それぞれを処理するハンドラが存在している（本章第4節)。これらからは、ライブラリ関数を直接起動するか、アプリケーション・ライブラリ関数を起動し、そこから間接的にライブラリ関数を起動する。

更に、景観シミュレータには、基幹部分(sim.exe)から派生的に起動する外部関数、及び連携動作するプラグインDLLを用いることができ、すでに多くの実装がある。このような外部関数やプラグインDLLを今後追加的に開発し連携動作するためのひな形やライブラリを用意している。これらの詳細については、第５章、第６章で解説している。上記のスタティック・リンク・ライブラリは、プラグインDLLのビルドにスタティックにリンクした

場合、グローバル変数等が基幹部分とは別のものとなるため、動作の一体性が失われる。そこで、基幹部分にリンクしたライブラリを直接利用するために、景観シミュレータで使用しているライブラリ関数は、DLL エクスポートするように設定してある。このため、各プラグイン DLL の開発においては、SIM.LIB だけをリンクすることで、基幹部分の実行形式に組み込まれたライブラリ関数を DLL インポートして利用することができる。

（3）動作環境

景観シミュレータは、単独で三次元モデル（LSS-G 形式のファイル）を見るだけのブラウザとして使用することもできる一方、大きなデータベースと共に本格的な三次元データ制作環境としてセットアップすることもできる。このため、景観シミュレータで操作する各種のファイル（三次元モデル、画像、テキストファイル等）は、目的に応じて特定のディレクトリに仕分けて置くことができる。とりわけ、景観データベースを構成する多くの共用ファイルと、現在設計検討中の建設プロジェクトに関係した固有のデータを区別することは整理のうえで有効である。更に、使用言語に依存したテキストファイルも、整然と整理する必要がある。

このディレクトリ編成は、環境設定ファイル kdbms.set の設定により行う。様々の目的のためのディレクトリを定義している。簡単な用途のためには、多くのデータを一つのディレクトリに集約するのが便利であろう。一方、大量のデータを扱う場合には、モデルやイメージなど、データの種類による仕分けや、プロジェクト単位の仕分けを行う方が便利である。このような必要とする操作環境に応じて柔軟に設定することができる。

システム起動直後にまず、env ライブラリがこの環境設定ファイルを最初に読み込み、様々なデータにアクセスできるように準備する。

（4）景観シミュレータの起動条件

単独のアプリケーションとしてユーザーが起動する場合には、OS から渡される引数はない。この場合には独立した操作環境として起動する。一方、景観データベースやウェブ・ブラウザから起動する場合には、引数として、表示すべきファイルや、データベースの種類に関する情報が渡される。

起動条件に応じた動作の場合分けは、CSimApp::InitInstance()関数の中で行っている。

単独で起動した場合には、CsplashWnd で初期画像を一定時間表示し、その間に初期化処理を行う。

データベースからの CreateProcess 関数による起動の場合には、「-f DB」という引数が渡される。この場合、初期化において CalledDB フラグ 1 を立て、以後の処理を行う。この場合、データベースによりテンポラリ・ディレクトリにある一時的ファイル「tmp000.txt」にデータベースの種類、表示すべきファイル等が格納されているので、そこから情報を取得し、ファイルの表示を行う。その際、部品として引用される三次元オブジェクト、画像

などは、データベースの種類に応じたディレクトリから取得する。

上記の一時的ファイルで指定された起動元のデータベースの種類は、dbms ライブラリのグローバル変数

```
int     DBno;
```

に登録される（０：事例、１：構成要素、２：材料）。この設定により、フルパスで指定されていない各種の参照ファイルの取得先ディレクトリを環境設定情報に基づいて定める。

Web ブラウザからの起動の場合には、フラグ２を立てる。初期化が終了すると、渡されたファイルをロードし、表示を行う。WEB から取得したシーン・ファイルの中の三次元モデルのファイルの所在は、http://から始まるアドレスで記述されているため、このサイトにアクセスして、関連するデータをダウンロードし表示する。ダウンロード先は、規定の作業用ディレクトリとするため、二回目からの表示にはダウンロードの必要がない。

（５）景観シミュレータが操作するデータのタイプ

景観シミュレータで編集し、表示するファイルには、三次元データをモデリングする LSS-G データ（ファイル拡張子.geo）と、モデリング結果に基づいて、光源や時間や視点設定を組み合わせたシーン群を構築する LSS-S データ（ファイル拡張子.scn）がある。扱うデータのタイプ区別は、［ファイル］以下のドロップダウンメニューから、新規作成またはファイルを開く選択を行う際に決定する。

現在編集しているデータのタイプは、common.c が記憶している。common.c は各種条件設定をスタティック変数に格納しており、各種ソースコードから共通に参照される。現在設定されているモードは、GetLssFileType()　関数で調べることができ、戻り値が K_LSS_S であればシーン、K_LSS_G であれば、ジオメトリと識別される。

LSS-S（シーン）タイプのデータは、SML ライブラリによって管理される。一つのシーンの構成は、s3Scene 構造体によって定義され、編集の結果は、この構造体の配列として、SML ライブラリの中で定義されたスタティック変数 scntab[]により管理されている。現在ロードされているシーンは、メイン画面の左下の操作ボタンにより逐次的に表示することができる。

LSS-G（ジオメトリ）タイプのデータは、

グループの一覧表 d3db を DML ライブラリの中で管理している。グループの作成(d3CreateGroup 関数による)や削除(d3DeleteGroup 関数による)の処理の中では、メモリブロックの取得や解放と同時に、この台帳への登録・削除を行っている。全てのグループからの検索は、このテーブルを利用して行っている。

一つのシーンは、表示する三次元モデルを持つことができる。更に、異なるシーンは異なるモデルを持つことができる。一つのシーンに定義されるモデルは、そのモデルを描画する起点となる最上位のグループへのポインタとして、シーン構造体に登録する。表示するシーンの切換に際して、表示すべきモデルが変わる場合は、モデルがない場合も含め、

表示しようとするシーンに登録されているルート・グループのアドレスを、da[ ]構造体にコピーすることにより実行される。

一方、DML ライブラリにおいては、このような複数の互いに独立したグループの集合体も、一つの共通テーブルで管理している。従って、表示において互いに無関係の、別シーンに属するモデル（それ以下の全てのグループ）も共通のテーブル上に登録される。登録に際して、FILE コマンドにより定義されるグループの内、外部関数ではなく LSS-G ファイルを直接参照している場合には、ロードの過程の中で起動される I3 ライブラリの i3LoadLssg 関数の中でこのテーブルの検索を行っている。もし、同一のファイルであれば、内容は固定的であり、再び同じ内容をロードする意味はないので、新たなグループは生成せずに、このグループを取り回して使用する。

リスト４－１：LSS-S 編集時におけるモデルのロード過程

LSS-S タイプの編集中で、シーンで定義するモデルのロードにおける呼び出し順序 s3CreateModel 関数→dbLoadGeometry2 関数→i3LoadLssg 関数

リスト４－２：LSS-G 編集時におけるモデルのロード過程

LSS-G タイプの編集中、LSS-G ファイルをロードする場合の呼び出し順序 common.c の LoadGeometry 関数→dbLoadGeometry2 関数→i3LoadLssg 関数

（6）リンクにおけるライブラリの依存関係

上記のように、景観シミュレータのビルドにおいては、８のライブラリをリンクしている。これらは互いに独立しているのではなく、ライブラリ間での相互依存関係が存在する。このため、例えば外部関数を作成するためにあるライブラリを利用すると、その前提条件として別のライブラリもリンクする必要が生じる場合がある。ライブラリの依存関係を図４－２に示した。

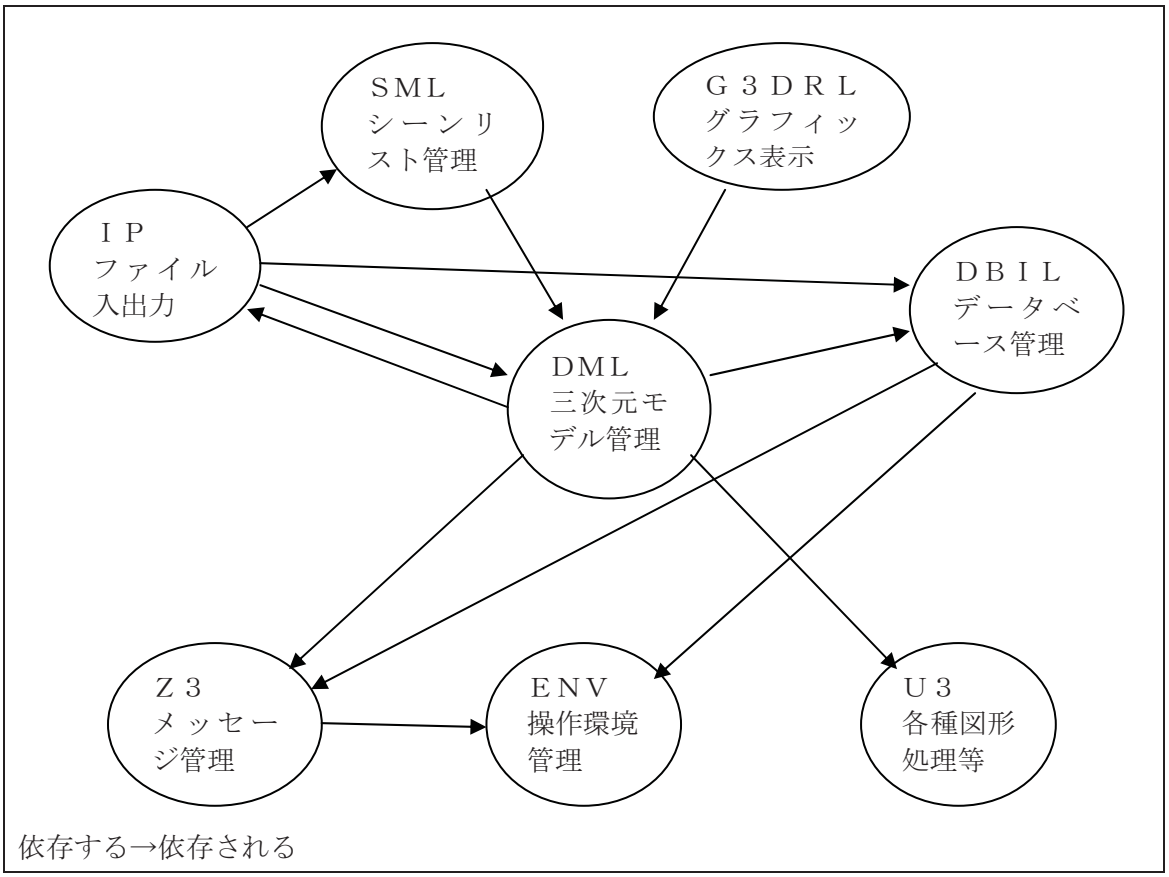

図４－２：ライブラリの依存関係

①┌　シーン管理ライブラリ（ＳＭＬ）

画像やモデルのロードに、ＤＭＬライブラリの関数を使用している。

②　データ管理ライブラリ（ＤＭＬ）

エラーメッセージにＺ３ライブラリの関数を使用している。

マテリアル、テクスチャのロードにＤＢＩＬライブラリの関数を使用している。

d3Gwaktu 関数,d3FilingFiles 関数が、ＩＰライブラリを使用している。

③景観データベースライブラリ（ＤＢＩＬ）

エラーメッセージにＺ３ライブラリの関数を使用している。

データの格納先ディレクトリを知るためにＥＮＶライブラリを使用している。

④グラフィックス表示ライブラリ（Ｇ３ＤＲＬ）

モデルのデータ構造を辿るために、ＤＭＬライブラリを使用している。

図形処理に、u3 ライブラリの関数を使用している。

エラーメッセージに z3err ライブラリの関数を使用している。

⑤インタプリタ（ＩＰ）

ファイル構築の途上で、SML、DML ライブラリの関数を使用している。

画像等の取得に際して、DBIL ライブラリの関数を使用している。

⑥環境設定ライブラリ（ＥＮＶ）

１箇所だけ z3err の関数を使用している

⑦ユーティリティライブラリ（Ｕ３）

u3grid.c が、DML、z3err の関数を使用している。

u3same~1.c１が、DML の関数を使用している。

u3slant.c が DML の関数を使用している。

u3trans.c が DML の関数を使用している。

⑧メッセージライブラリ（Ｚ３ＥＲＲ）

ENV でメッセージ・ファイルの所在ディレクトリを取得している。

（7）メモリ空間の相互関係

各ライブラリは、データ管理のためのスタティックなポインタまたは配列を有しており、これを用いて動的に取得・解放される各種メモリブロックを登録・管理するメモリ空間を有している。例えば、ENV ライブラリにおける環境変数テーブルや、z3err におけるエラーメッセージテーブルのように、独自に構築され、独自に開放される完結したデータは、ライブラリ間、あるいダイアログのハンドラとの間で相互に干渉することなく各ライブラリにおいて一貫して管理するこができる。

一方、例えばインタープリタが外部ファイル読み込みに際して動的にメモリブロックを取得し生成するデータの一部は、処理後に SML や DML ライブラリに移管され、シーン配列やグループ配列などの中に組み込まれる。これらについては、除却・解放に関する役割分担が明確に定められ協調的な処理が行われないと、使用中のメモリブロックの解放や、解放済みのメモリブロックを再度解放しようとすることによるシステム障害や、いかなるライブラリの管理からも遊離したメモリブロックの発生（メモリー・リーク）の原因となる。

Ver.2.09 においては、これを、以下のように整理した。

① IP と SML

インタープリタが、SCENE コマンドを処理した後に、一つのシーンに対応するメモリ・ブロック(s3Scene*)を、SML ライブラリが管理するテーブル(scntab)に登録する。

一つのシーンに関連づけられた光源グループ、効果グループ等の各メモリブロックは、移管されたシーンから参照されているものについては、インタープリタが終了した時点で解放しない。

別の処理において、ユーザーの編集操作によりシーンが削除された時点、新規作成が選択された時点、またはシステム終了時点で、シーンと、それに関連付けられたメモリブロックを解放する。シーンにおいては、一つの光源グループや効果グループを複数のシーンから共通で使用することができる。従って、一つのシーンの解放に際し

ては、それが参照している光源グループ、効果グループ等が、他のシーンから参照されているかどうかを検査し、該当するものがなければ、シーンと同時に解放する。

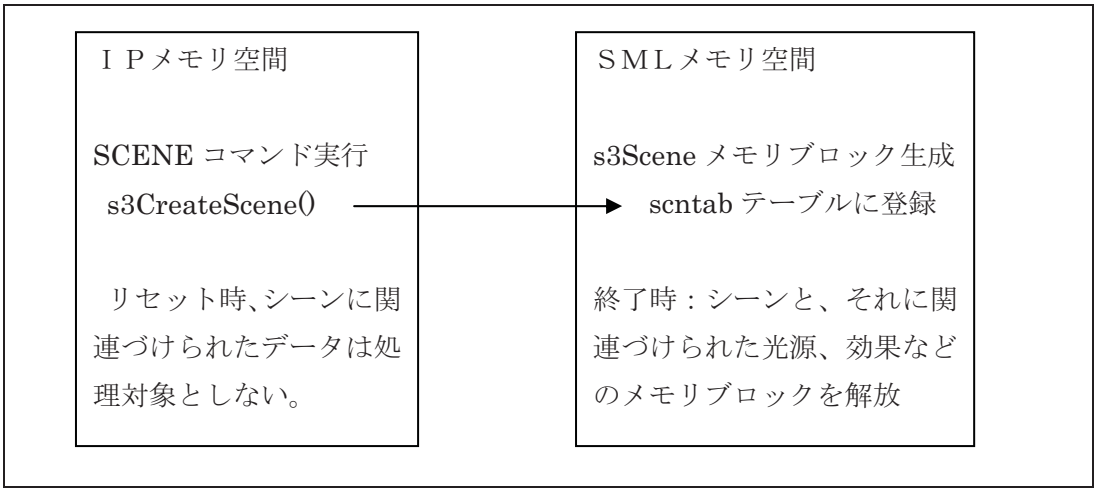

図４－３：IP と SML のメモリ管理

② IP と DML

DML は、グループを登録する d3db、マテリアル ID を登録する mtltab、テクスチャ ID を登録する textab をテーブルとして管理している。

インタープリタが、GROUP コマンド、または FILE コマンドを実行した後に、新たに作成されたメモリ・ブロック(d3Group)を DML が管理するテーブル d3db に登録する。

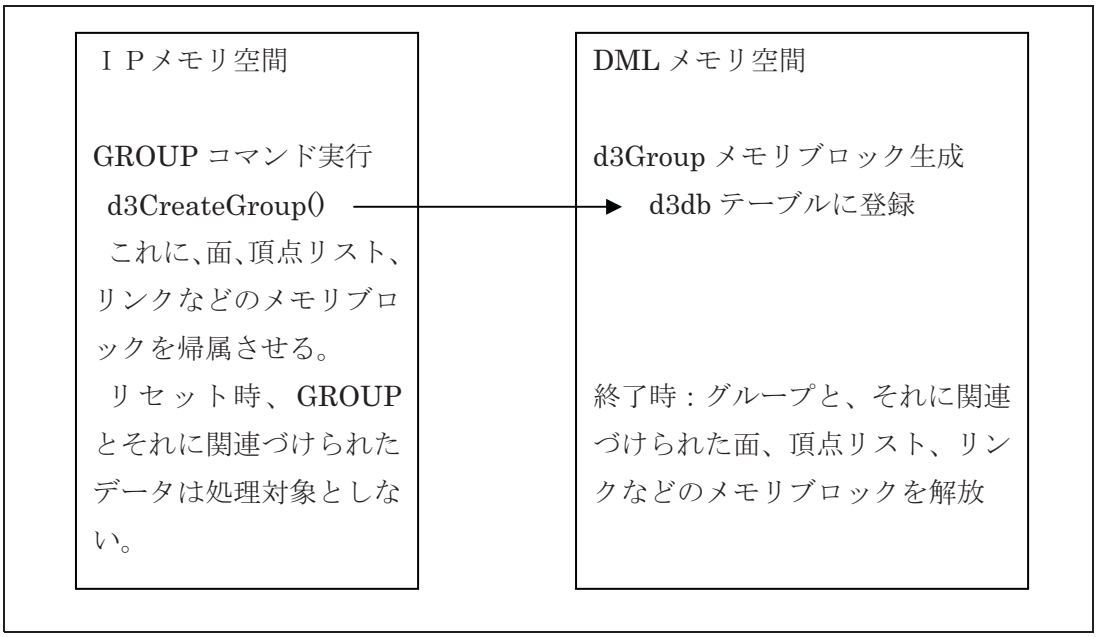

図４－４：IP と DML が管理するメモリ空間の関係

そのグループに属する面は､GROUP_FACE コマンドが実行された時点でコピーを作成し、グループの要素として登録する。それに際して面に帰属する名称、頂点リストなどのメモリブロックも全てコピーを作成する。一つの面（メモリー・ブロック）

は、別のグループから共有されることはなく、常に唯一のグループに帰属する。

LINK コマンドが実行された時点で作成されるメモリ・ブロック(d3Link, d3LinkUP)は、それぞれ関連する親グループ、子グループが保有するリスト（next ポインタでつなぎ）に登録する。

インタープリタが終了した時点で、グループ、リンク以外のメモリブロックを解放する。

マテリアル､及びテクスチャに関しては､インタープリタがMATERIALコマンド、TEXTUREコマンドを処理した後にDMLが管理するテーブルにレコードだけを新規作成する(d3RegisterMaterial 関数等)。この時点では画像等データのロードは実行せずメモリブロックは作成されない。GROUP_MATERIAL、FACE_MATERIAL などのコマンドが実行された時点で､適用対象となる面､グループに ID 番号を記入する。

実際のテクスチャ画像データ等の取得と画像データを格納したメモリブロックの作成は、インタープリタによるファイル読み込みが終了した後、表示段階で G3DRL ライブラリ(g3LoaMaterial, g3LoadTexture 関数)から参照される DBIL ライブラリが行っており、状況に応じた適切な参照先のディレクトリから画像データのロードを行う。その際にキャッシングを行っており、異なる ID で同じテクスチャ・ファイルが複数回参照されても、一つしかメモリブロックを作成しない。また、表示する時間が変更された場合にも、経年劣化するマテリアル等のデータの再ロードを行っている。

以上の機構により、テクスチャを多用した多くのモデルを参照するシーン・ファイルのロード時間やメモリ消費が非常に重くなることも防いでいる。

ユーザーの編集操作、新規作成または終了時点でグループが削除されると、グループ及びそれに帰属するメモリブロックが解放される。その際に、グループにリンクで関連づけられた子グループに関しては、別グループからの参照の有無を検査し、該当するものがなければ、リンクと同時に削除する。マテリアルおよびテクスチャのテーブルは、個々のグループの削除操作に際しては変更せず、新規作成または終了の処理の中で再初期化している。

③DML と DBIL

第３章で解説したように、グループは、リンクによって親子関係が設定され、一つのシーン中に存在するすべてのグループが、最上位のルート・グループを起点とする親子関係の連鎖として構造化されている。表示に際しては、ルート・グループからリンクを辿り、リンクで結び付けられた子グループの全てを表示する。

OpenGL による表示を管理する G3DRL ライブラリは、OpenGL 表示を行う全てのダイアログ画面を配列化したテーブル da[ ]を管理している。一つの表示画面には、様々な表示条件と共に、グループとリンクの全体を辿る起点となるルート・グループ（最上位のグループ）が登録される。描画は、このグループから、リンクを辿って行われる。LSS-G タイ

プのデータを編集する場合には、ルート・グループは通常は不変で、全ての表示画面に共通している。

しかしながら、地物を構成する一つのグループは、複数の親グループを持つことができ、従って、このリンクを辿る方法では、一つのグループが複数回ヒットする可能性がある。実際、リニア配置された街灯や、エリア配置された樹木などは、メモリの上では一つのグループとして格納されているデータが何回も位置を変えて再描画される。したがって、リンクを辿る方法では、特定の属性を有するグループを検索するような処理には無駄が多い。そこで、表示に関連したリンク構造とは別に、DML において、別途テーブルとして全てのグループを管理している（図４－５）。

d3CreateGroup()関数を使用せずに、例えば

d3Group *g = (d3Group*) d3Malloc(sizeof(d3Group));

のような方法で、グループを作成することが可能であるが、このようにして作成されたグループは、DML が管理するテーブルには登録されていないことに注意する必要がある。例えばデータ変換の途上でこのような方法で一時的に作成したグループのメモリブロックを、地物を構成するグループとリンクの中に組み込むような処理が行った場合には、終了時点での一貫性は保証されない。

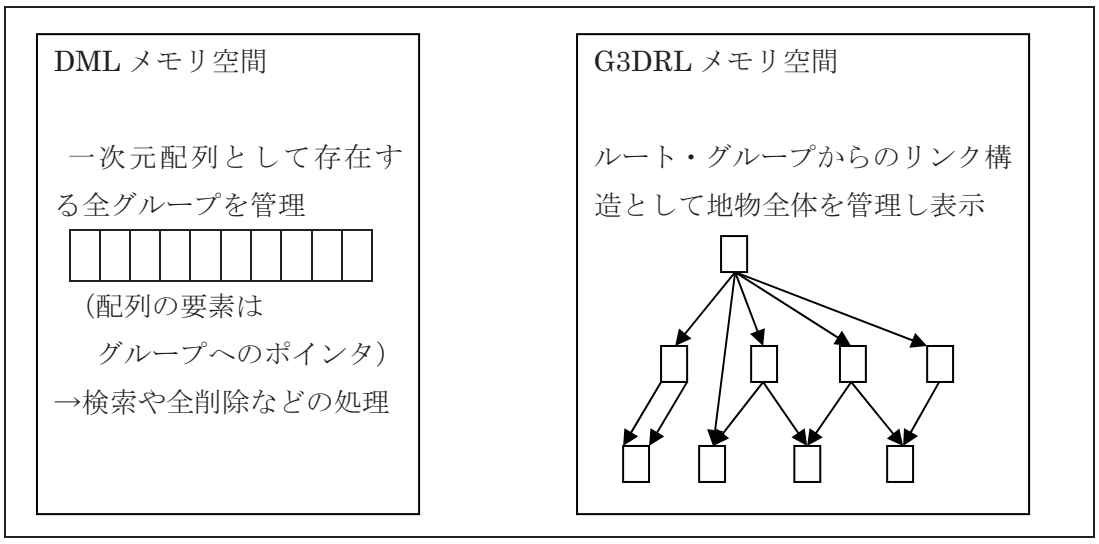

図４－５：DML と G3DRL が管理するメモリ空間の関係

４－２．ライブラリ関数

ライブラリ関数は、マルチ・プラットフォーム（UNIX 系と Windows 系を含む）の環境において、OS に依存しない景観シミュレータのデータ構造の管理と処理を行う諸関数のパッケージである。大きく、概要で述べたように８のライブラリが存在する。

それぞれのライブラリを構成する関数群は、全体のビルドにおいては、共通のヘッダーファイルに登録されているが、ソースコードは複数に分かれている。ソースコードはライブラリ毎に一つのディレクトリに置く一方、ヘッダーは、INCLUDE ディレクトリに一括してある。それぞれのライブラリを構成するソースコードは、コンパイル・リンクによって、それぞれのスタティック・リンク・ライブラリを生成する。

各ライブラリを構成する関数は、所属するライブラリがわかるように、共通のプレフィックスを持つ名称を有している。

以下に、各ライブラリの機能を解説する。　個別のライブラリ関数の解説を付録 B に掲載した。

（１）シーン管理ライブラリ（SML）

①概要

一つの景観画像を生成するために必要な背景・前景画像、三次元モデル、視点、光源、効果、時間を定義したシーンを構築する。更に、多くのシーンを作成し、一連の評価用画像として次々と提示できるように、シーン・リスト（配列）として管理する。

②シーン構造体

最も基本となるシーン構造体は、リスト４－３のように定義されている。

リスト４－３：シーン構造体の定義(sml.h)

```
typedef struct _s3Scene {
        char            *name;
        int             type;
        s3Image         *img_b, *img_f;
        s3Model         *mdl;
        s3Camera        *cam;
        s3LightGroup    *lg;
        s3EffectGroup   *eg;
        s3Time          *time;
} s3Scene;
```

名称（name）：シーン毎にユニークな名称を定義する。

タイプ(type)：　図法（透視図、平面図・・・）、表示モード（テクスチャ、シェーディング、ワイヤーフレーム）等を示すコード。

背景(img_b)・前景(img_f)：モデルの前後に表示する画像。

モデル(mdl)：地物を表す三次元モデル。
視点情報(cam)：視点、注視点、天頂ベクトルなどを記述する。
光源グループ(lg)：最大８の光源から成る光源グループを一つ指定する。
効果グループ(eg)：無制限個の効果を束ねた効果グループを一つ指定する。
時間(s3Time)：時間を記述する構造体を指定する。

名前とタイプ以外は、ファイル名やメモリブロックのアドレスを直接指定するのではなく、定義を記述した構造体（メモリ・ブロック）のアドレスを指すポインタとなっている。一つの定義を、複数のシーンで共通に使用することができる。

実際の運用においては、共通の背景を用いて様々のモデルを比較検討したり、共通のモデルで、様々な視点場からの検討を行ったり、あるいは光源条件（季節・時刻・天候）や築後年数のみを変更したヴァリエーションを一連のシーンとして用意し、評価セッションに臨む。このような様々のヴァリエーションの作成に際して、変更のないデータや設定内容に関しては、複数のシーンで共用することが、このような間接的な参照により可能となっている。

シーンの編集操作、あるいはＬＳＳ－Ｓ形式のファイルをロードすることにより、一連のシーンから成る表示用のデータ（メモリ・ブロック上の配列）を構築する。このブロックのアドレスを示すスタティック変数をＳＭＬライブラリが管理しており、通常はこのメモリブロックに対する様々な処理をライブラリ関数で実行することにより、シーンに対する様々な処理を行う。

③ ソースコード(s3xxx.c の名称)

ＳＭＬライブラリは、リスト４－４に示したソースコードから成る。

リスト４－４：SML ライブラリのソースコード一覧

S3sml.c	(構築)
S3set.c	(データのセット)
S3get.c	(データの参照)
S3delete.c	(データの削除)

（２）データ管理ライブラリ（DML）

①概要

三次元の地物（形状、表面の光学特性、属性）をデータとして構築する。更に、複数の地物の構成要素を、リンクで親子関係に結び、地物全体を構造的にデータ化し、管理することにより、様々な編集操作を下支えする。

モデルの記述

②グループ構造体

グループは、名称、ID やマテリアル、テクスチャ、親の数、子の数等の属性と、面

のリストを持っている

リスト４－５：グループ構造体の定義(dml.h)

```
struct _d3Group {
/*private:*/
        int grpid;
        int dbno;
        int num_u;/* numbers of parents */
        int num_d;/* numbers of children */
        int material;
        int texture;
        double xmin, xmax, ymin, ymax, zmin, zmax;/* bounding box */
        d3Link *link;
        d3LinkUp *lup;
        /*INT CONSTANT OOCURS HERE*/
        d3Face   *f;
        /**/
        void *display;
        void *user[D3_USERDATA_MAX];
/*public:*/
        char *name;
        char *kind;
        int type;
};
```

③面の構造体

面は、名称を持たず、頂点リスト、マテリアル、テクスチャ、法線ベクトル、カラー等の情報を持つ。next のポインタでつながれたリストとして、グループに所属する。その先頭に当たる面を、d3Group の f メンバが指し示している。

リスト４－６：面の構造体の定義(dml.h)

```
struct _d3Face {
/*private:*/
        d3Vertex *vl;
        d3Face *next;
/*public:*/
        unsigned long va_f;
        unsigned long va_v;
        int vnum;
        int material;
        int texture;
        float n[3];
        float c[4];
        int display;
        int shape; /* 1996.1.30 D3_SHP_xxxx */
};
```

③頂点リストの構造体

頂点は、座標、法線ベクトル、テクスチャ座標、カラーを有する点をリスト化した配列（メモリ・ブロック）として定義され、面に帰属する。

リスト４－７：頂点の構造体の定義(dml.h)

```
struct _d3Vertex {
/*private:*/
/*public:*/
        unsigned long va;
        double v[3];
        float n[3];
        float t[2];
        float c[4];
};
```

④ソースコード（d3xxxx.c の名称）

リスト４－８：DML ライブラリを構成するソースコード

d3dml.c	(データ構築)
d3malloc.c	(メモリ管理)
d3mat.c	(リンク・マトリックス計算)
d3pick.c	(オブジェクトの選択)

（３）データベース管理ライブラリ（DBIL）

①概要

多くは、景観データベースの検索機能において用いられるが、景観シミュレータの基幹部分(sim.exe)のビルドにおいても、画像ファイルの処理など、部分的な機能を利用している。

②ソースコード（dbxxx.c の名称）

リスト４－９：DBIL ライブラリを構成するソースコード

dbdata.c	(データベース検索)
dbms.c	(構築)
dbread.c	(データベース検索)
image.c	(SGI 形式の画像の読み込み)
isave.c	(SGI 形式の画像の保存)

（４）動作環境管理ライブラリ（ENV）

①概要

システムの動作環境を表す様々の環境変数を管理する。システム起動時に、環境設定ファイル kdbms.set を読み込むことによって初期化される。

②ソースコード(e3xxx.c の名称)

リスト４－１０：ENV ライブラリを構成するソースコード

e3env.c	(環境変数の管理)
e3kerja.c	(ユーザー作業環境)

（５）画像表示処理ライブラリ（G3DRL）

①概要

主に、三次元モデルを、各種の OpenGL 画面に描画する機能を提供している。表示の

ほかに、地面の高さを用いて処理を行う場合には、デプス・バッファを用いたOpenGLの描画機能を用いて地面の高さを抽出している。OpenGLによる描画は、メイン画面における透視図の表示や、配置・コピー、平面生成、可視範囲解析など、地物の平面図表示（オルソ画面）を主要部分とするダイアログにおいて用いられる。この他に、テクスチャ選択画面など、いくつかの編集ダイアログでは、OpenGLの小さな画面を用いて、ユーザーが選択しようとしているカラーやテクスチャ等を表示している。

②ソースコード(g3xxx.c、wg3xxx.c等の名称)

リスト４－１１：Ｇ３ＤＲＬライブラリを構成するソースコード

g3drl.c	(メイン画面、各編集画面への透視図・平面図の描画)
g3font.c	(文字の表示)
g3load.c	(マテリアル、テクスチャのロード)
g3road.c	(デプス・バッファによる道路法面の形状生成)
g3sdl.c	(マテリアル、光源のダイアログの色球の描画)
g3steel.c	(型鋼のダイアログの見本の描画)
g3stenci.c	(ステンシル・バッファの利用)
g3utl.c	(テクスチャ自動貼付等)
StereoSight.c	(ステレオ表示のパラメータ管理)
wg3.c	(メイン画面、オルソ画面の初期化
wg3swal.c	(ダイアログの小さなOpenGL画面の初期化)

（6）ファイル入出力ライブラリ（IP）

①概要

外部ファイルの入出力を行う。

②ソースコード(i3xxx.c等の名称)

リスト４－１２：ＩＰライブラリを構成するソースコード

i3ip.c	(構築)
i3ipbase.c	(コマンドの解析)
i3ipcall.c	(外部ファイル参照)
i3iperr.c	(エラー処理)
i3ipex.c	(外部関数処理)
i3iplssg.c	(LSS-Gのデータ構築)
i3iplsss.c	(LSS-Sのデータ構築)
i3ipout.c	(ファイル出力)
fopen_.c	(URLアクセス)

（7）図形演算処理ライブラリ（U3）

①概要

様々なダイアログにおいて必要となる図形演算処理のうち、共通性の高いものをまとめたライブラリである。

②ソースコード(u3xxx.c等の名称)

リスト４－１３：Ｕ３ライブラリを構成するソースコード

u3cross.c	(二次元図形演算)

u3cut.c	(二次元図形演算)
u3dml.c	(バックアップ・リストア)
u3grid.c	(メッシュデータ)
u3hsvrgb.c	(表色系)
u3ofs.c	(断面)
u3rand.c	(乱数)
u3same~1.c	(点の一致)
u3slant.c	(法面)
u3spline.c	(スプライン曲線)
u3trans.c	(メッシュの座標変換)

（8）メッセージ管理ライブラリ（Z3)

①概要

エラー、警告、情報提供、選択肢の提示などを行う。

②ソースコード(z3xxx.c 等の名称)

リスト４－１４：Ｚ３ライブラリを構成するソースコード

z3err.c	(環境変数の管理)
z3nt.c	(メッセージボックス表示)
z3unix.c	(メッセージウィンドウ表示)

４－３．アプリケーション・ライブラリ関数

アプリケーション・ライブラリは、各種ライブラリと、次に述べるダイアログ・ハンドラの中間に位置し、システム全体を管理している。これには、以下のような機能・役割がある。個別のライブラリ関数に関する解説は、付録 B に掲載した。

（1）システム管理機能

①システム全体の初期化処理を行う

②システムの状態をモニタリングし、問い合わせに回答する。

③システム全体の終了処理を行う

④エラー対策を行い、Z3 ライブラリにメッセージを送出する。

ソースコードは、

Common.c

Dataope.c

の二つの長いファイルにほぼ集約されている。

（2）ライブラリ支援機能

各ライブラリの機能を動かす上で必要となる、Windows 固有の条件に対応する機能を細くする。例えば、ファイルやディレクトリの記述法が、UNIX 系と Windows 系では異なっている。また、グラフィックス表示画面の初期化の方法なども、OS により異なっている。

①OpenGL 画面の初期化

pixel.c

wg3.c

wg3swal.c

②画面の印刷処理

b3bitmap.c

copy.c

print.c

（3）ダイアログ・ハンドラ支援機能

複数のダイアログ・ハンドラ（C++のクラス）に共通する機能を提供する。操作・処理の前後での画像のバックアップとリカバリーを行う。また、外部プロセスの起動を行う。

Multilang.cpp（多言語処理）

Pembantu.c （ヘルプ表示）

Genshi.c （原始図形計算）

Fopen_.c （URL アクセス）

（4）個別単発的な特殊演算処理機能

数値計算処理やデータ形式変換処理などで、一つのライブラリを形成する程の規模ではない専門性の高い計算処理は、アプリケーション・ライブラリとして分類している。画像の視点抽出計算や、異なる色彩の表色法の間での数値換算処理などがこの中に分類されている。

①異なる形式でのファイル出力

dxfout.c

vrmlout.c

fire.c

②カラー表色系の相互変換

cnvcol.c

③ビットマップによる動画作成等

bmp2dib.cpp

AviComposeWnd.cpp

MyImage.cpp

MyAvi.cpp

dib31.cpp

④画像視点抽出の解析計算

kamera2.c

４－４．ダイアログ・ハンドラ

Windows 固有のダイアログ画面に対するハンドラ・ルーチン群は、ユーザーによる画面操作を通じての様々なリクエストに対応する処理を実行する。これらの処理プログラムは、ＯＳに大きく依存しており、ステップ数の多くは、エラー対応処理やメッセージ、ヘルプなどを含め、ユーザーによる想定外の操作に対する対応に充当されている。

ダイアログの画面構成（デザイン）の大半は、リソースによって定義され、開発環境の中ではＧＵＩ環境でデザインされている。多言語環境において、画面レイアウトには、漢字系（日中韓）のダイアログと、アルファベット系のダイアログの二種類のデザインを用意してあり、選択された言語がいずれに属するかによって選択的に表示を行っている。ユーザーの操作は、マウスクリックおよびキーボード操作によってインプットされる。一つの画面に関して、初期化処理、様々な操作に対する応答と、終了時の処理から成るハンドラを、Ｃ＋＋のクラスとして構成している。

本節においては、各操作画面ごとに、主要な機能を解説した上で、デザインを定義しているリソース、これを処理する関数群をパッケージ化したクラスと、それを定義しているソースコードファイル名を示す。更に、必要に応じて、それぞれのダイアログの呼び出し元や、それぞれから起動するダイアログとの関係、要求された機能を実行するために使用されるライブラリ関数との関係を解説する。

（1）メイン画面

様々な図法（透視図、平面図、立面図その他）で地物を表示すると共に、多くの編集機能をメニューから起動する。下部にシンプルな視点移動ボタン、シーン切換ボタンがある他、OpenGL の表示画面におけるマウス操作で操作対象となるオブジェクトを選択し、情報を見たり編集を加える。

また、Ver.209 においては、マウスの右ボタンを押したままドラッグすることにより、高速でスムーズな視点移動を行う。枝分かれバージョンの統合により、ステレオ表示、影の表示を一つのバージョンで使い分けることができる。更に、多言語機能により、走りながら、メニュー、ダイアログ、ヘルプ、エラーメッセージの言語を切り換えることができる。

図４－６：メイン画面

構成要素

①メニュー：IDR_MAINFRAME　ハンドラ：CMainFrame (mainfrm.cpp)

メニュー構成
［ファイル］
＋［新規作成］
＋［LSS-S］　　LSS-S（シーン）データを新規作成
＋［LSS-G］　　LSS-G（ジオメトリ）データを新規作成
＋［開く LSS-S］　　LSS-S（シーン）ファイルを開く
＋［開く LSS-G］　　LSS-G（ジオメトリ）ファイルを開く
＋［読み込み LSS-G］　LSS-S（シーン）の編集でモデルを読み込む

＋［JPEG 形式で出力］表示されている画面を JPEG 形式で保存する
＋［上書き保存］　　読み込んだファイルに LSS-S または LSS-G を上書き保存する
＋［名前を付けて保存］編集中のデータ(LSS-S または LSS-G)に名前を付けて保存する
＋［最適化保存］　　LSS-G（ジオメトリ）データを最適化処理付きで保存する
＋［作業環境設定］　ファイルを取得／保存する作業用フォルダを指定する
＋［報告書執筆］　　現在編集中のデータに関するサマリーをテキスト出力する
＋［ファイル整理］　編集中のデータに関連するファイルを指定場所にコピー集約する
＋［スキャナー入力］　スキャナー入力のためのアプリケーションを起動する
＋［画面印刷］　　表示されている画面イメージをプリンターに出力する
＋［高解像度印刷］　プリンターの解像度で印刷する
＋［印刷プレビュー］　高解像度印刷をプレビューする
＋［プリンタの設定］　プリンターの選択と印刷条件の設定を行う
＋［使用言語の選択］　メニュー、ダイアログ、メッセージ、ヘルプの言語を選択する
＋［アプリケーションの終了］　終了する

［編集］
＋［他選択］
＋＋［次候補］　　クリックした点に係る、もう一つ裏側のオブジェクトに選択を変える
＋＋［親グループ］　選択したオブジェクトの親グループを選択する
＋＋［子グループ］　選択したオブジェクトの子グループを選択する
＋＋［情報を見る］　選択したグループの諸元素を表示・編集する
＋［選択取り消し］　選択・強調表示を取り消す
＋［選択モード］
＋＋［グループ］　　グループを選択する（通常のモード）
＋＋［面］　　面を選択する（マテリアルの編集対象として）
＋＋［同色面］　　同色面を選択する（同上）
＋［移動/回転/スケール］選択したオブジェクトの上方リンクのマトリクスを編集
＋［配置/コピー］　位置、線形、領域を指定して、オブジェクトを添加する
＋［削除］　　選択されているオブジェクトを削除する
　　オブジェクトが選択されていない場合には、面が無く、下方リンク
　　ンクもないグループ（「幽霊」と表記）を全て削除する
＋［視点設定］
＋＋［視点座標］　　視点座標・注視点座標等の視点情報を数値で表示・編集する
＋＋［可視範囲解析］選択したオブジェクトが見える範囲を解析する
＋＋［視点設定］　　平面図上の指定した位置に視点を設定する
＋＋［移動経路設定］移動経路と注視方向を指定し、動画を表示・記録する

＋［マテリアル／テクスチャ］選択したオブジェクトの表面仕上（光学特性）の編集
＋［エフェクト］シーンの効果（霧、ステレオ表示、影等）の設定
＋［画像視点抽出］背景画像の視点パラメータを求め、立体を位置合わせする
［表示］
＋［視点］
＋＋［全体視界］全てのオブジェクトが見える位置に移動する
＋＋［初期表示］最初の視点（シーンに定義された視点）に戻る
＋［透視図］　透視図として表示する
＋［平面図］　平面図（配置図）として表示する（真上から見る）
＋［南立面］　Y 軸の負から正の方向に見た立面（正面図）
＋［東立面］　X 軸の正から負の方向に見た立面（側面図）
＋［北立面］　Y 軸の正から負の方向に見た立面
＋［西立面］　X 軸の負から正の方向に見た立面
＋［経年変化］築後日数を設定する
＋［表示モード］
＋＋［テクスチャ表示］テクスチャ付で面を表示する
＋＋［シェーディング表示］テクスチャを無視して面を表示する
＋＋［ワイヤーフレーム表示］辺・稜を表示する
＋＋［オプション設定］
＋＋＋［地面のみ表示］地面の属性をもつオブジェクトだけを表示する
＋＋＋［地面テクスチャ表示］地面の属性をもつオブジェクトだけテクスチャ表示する
　　（大規模な市街地の編集などにおいて、通常の表示が遅い場合などに使用）
＋＋［オプション解除］オプションの設定を解除する
＋［グリッド］グリッドの表示を ON／OFF する
＋［アンチエリアシング］画面に対して斜の境界線を滑らかに表示する設定を行う
＋［影］
＋＋［影なし］影を表示しない
＋＋［地面以外（通常）］地面以外の物体の影を表示する
＋＋［全地物］全ての地物の影を表示する
＋＋［選択物］選択物の影を表示する
＋＋［影設定］影表示のパラメータ、詳細設定を行う
＋［ステレオ］ステレオ表示の有無とパラメータを設定する
＋［高速表示］表示の高速化のオプションを設定する
＋［シーン選択］シーンを一覧表示し、選択する
［形状生成］
＋［原始図形］

＋＋［直方体］直方体の位置と大きさを指定し、形状生成する
＋＋［球］中心位置と半径を与え、形状生成する
＋＋［円柱］上円と下円の中心位置と半径から形状生成する
＋＋［円錐・円錐台］上円と下円の中心位置とそれぞれの半径から形状生成する
＋＋［角柱］上底と下底の中心位置と半径・角数から形状生成する
＋＋［角錐・角錐台］上底と下底の中心位置とそれぞれの半径、角数から形状生成する
＋＋［掃引体（１面)］断面と、移動経路から立体を形状生成する
＋＋［掃引体（２面)］二つの面（上底と下底）から形状生成する
＋＋［平面］平面、及び押し出しによる立体、面の穴あけ等の編集操作を行う
＋＋［線］　頂点座標の配列から線分、折れ線を形状生成・編集する
＋［基本構成要素］
＋＋［形鋼］典型的な断面の鋼材を形状生成する
＋＋［橋］５種類の橋を配置する
＋＋［草］ランダムな形状として草を生成（未完成）
＋＋［水面］（廃止）テクスチャ編集画面に統合
＋＋［ブロック］（廃止）テクスチャ編集画面に統合
＋＋［道路］断面と中心点軌跡から立体を形状生成する
＋＋［河川］断面と中心点軌跡から立体を形状生成する
＋＋［法面］テクスチャを貼る
＋［道路法面生成］地形と道路断面、法面パラメータから道路と法面を形状生成する
＋［オプション］ユーザー定義によるパラメトリックな部品を選択し、形状生成する
　屋根、階段、正多面体、正面テクスチャ付きビル、各種コンバータ等
＋［プラグイン］プラグイン DLL を選択し、形状生成・編集する
　トンネル、園路、地形編集機能など
＋［シャッター］表示されてい状況を一つのシーンとして記録する
＋［ヘルプ］メイン画面のヘルプを表示する

②ポップアップメニュー：IDR_MENU_POPUP　ハンドラ：CDrawFrm (drawfrm.cpp)
　メイン画面でオブジェクトを選択した状態で、画面を右クリックすると表示される。

メニュー構成
［情報を見る］
＋［情報を見る］ダイアログ(9)を開く
＋［幾何属性］
＋＋［立体情報］ダイアログ(52)を開く
＋＋［面情報］　ダイアログ(51)を開く
＋＋［三次元ソート］オブジェクトの全ての頂点座標をソートし一覧表示する
＋［力学的属性］

＋＋［質量］ダイアログ(64)を開く
＋［化学的属性］
＋＋［炭素含有量］ダイアログ（65）を開く
＋［データベース参照］ダイアログ（66）を開く
＋［住宅情報］ダイアログ(54)を開く
＋＋［詳細を見る］単体の内部俯瞰等の LSS-G ファイルを別プロセスで表示
＋＋［眺望を見る］選択した物件からの眺望を表示
＋＋［物件情報］ダイアログを開く
［選択変更］
＋［次候補］同じクリック地点にある一つ裏側のオブジェクトを選択する
＋［親グループ］選択したオブジェクトの親グループを選択
＋［子グループ］選択したオブジェクトの最初の子グループを選択
＋［選択解除］
［編集する］
＋［移動/回転/スケール］ダイアログ(10)を開く
＋［削除］選択したオブジェクトを削除
＋［このグループを保存］選択したオブジェクトを LSS-G 形式で保存
＋［マテリアル］ダイアログ(21)を開く
＋［色々なマテリアル編集］
＋＋［オリジナル版］ダイアログ(21)を開く
＋＋［グラフィックなマテリアル編集］ダイアログ(25)を開く
＋＋［グラフィックなテクスチャ編集］ダイアログ(26)を開く
＋＋［様々なカラー編集］ダイアログ(28)を開く
［視点場］
＋［これを注視］選択されたオブジェクトの中心に注視点を設定する
＋［ここから見る］選択されたオブジェクトの中心に視点を設定する
＋［ここから見返す］選択されたオブジェクトの中心から現在の視点を注視する
［BOOL 演算］
＋［これを刃物として選択］図形演算の適用オブジェクトを選択する
＋［選択された刃物でこれを切る］図形演算の対象オブジェクトを選択し、掘削する
＋［　取り出す］図形演算の対象オブジェクトを選択し、掘削される部分だけを取り出す
＋［　へこませる］対象オブジェクトを掘削した後に、新たな表面を付け加えて閉じる
＋［　もぎ取る］掘削される部分に新たな表面を付け加えて立体として閉じる
＋［独立する］選択したオブジェクトを親グループから分離し、ルートに置く
＋［統一する］配下の全ての子グループの面を、選択したグループに直接帰属させる。
［オプション］既存のパラメトリックな部品のパラメータを再編集する

メニュー項目の多くに関して、現在のシステム状態により選択可／不可を切り替えている。その条件は、編集対象が LSS-S（シーン）であるか、LSS-G（モデル）であるかで大きく分かれる。表示状態が透視図であるか、平面図・立面図系であるかでも分かれる。更に、編集ダイアログが開いた時点で、同時に開かれると不具合が生じる別の編集ダイアログを選択不可としている。

このようなメニューの選択可／不可を系統的に切り替えるために、CMainFrame クラス(mainfrm.cpp)の関数として、SetSensitiveXXX(int mode)という一群の関数を用意してある。ダイアログの開始や終了に際して、引数として TRUE、FALSE 等を指定してこれらの関数を呼び出すことにより、上記の目的を果たしている。

③下部ダイアログ リソース : IDD_DLG_KBD ハンドラ : COpeDlg::m_opeDlg(opedlg.cpp)
④OpenGL 画面 : CDrawFrm m_pDraw (drawfrm.cpp)
⑤ヘルプ : mainfrm.txt

画面を左クリックすることにより、表示の中からオブジェクトを選択し、次に各種の編集ダイアログを開いて、選択したオブジェクトを編集する、という作業手順が、本システムの操作においてきわめて一般的である。画面左クリックが行われた時点で、上記④の OpenGL 画面をコントロールしている CDrawFrm クラスの、OnLButtonUp 関数を起点として、Ｇ３ＤＲＬライブラリの機能を用いて、クリックした画面上のポイントに表示されている面を探索し、結果を、g3SelPath 構造体のデータとして格納する。この中には、手前に表示されている面の裏側に隠れている面、それらが帰属するグループ、それらの親グループの全てが含まれる。そして、最も手前に表示されている面を含むグループを、強調表示する。この状態でメニューの、[編集][他選択][次候補]が選択されると、その面の裏側に隠れている別の面が帰属するグループに選択対象が変更される。

各編集ダイアログの側では、g3GetSelPath 関数を用いて、現在の選択状態を示す g3SelPath 構造体のデータにアクセスし、編集対象とすべきオブジェクトを特定する。

（2）バージョン情報

メイン画面の右上のヘルプのサブメニューであるバージョン情報から起動し、現在のバージョンを表示する。メイン画面のタイトルに示されたバージョンは、多言語機能により外部化されているため、セットアップが不適切で多言語対応の XML が古いままとなっている場合には、古いバージョンが表記されるのに対して、このバージョン情報で表記する、例えば 2.09 といった数字は、ソースコード中のリソースで表記されたバージョン番号をキーワードとして言語に応じた張替を行うため、旧バージョン番号に張り替えられることはない（意図的に偽の XML ファイルを作成するのでない限り、キーワードが不一致となり、ソースコード中の数字がそのまま表示される）。

ログボタンで、ヘルプと同様の機構により、メモ帳で履歴を表示する。

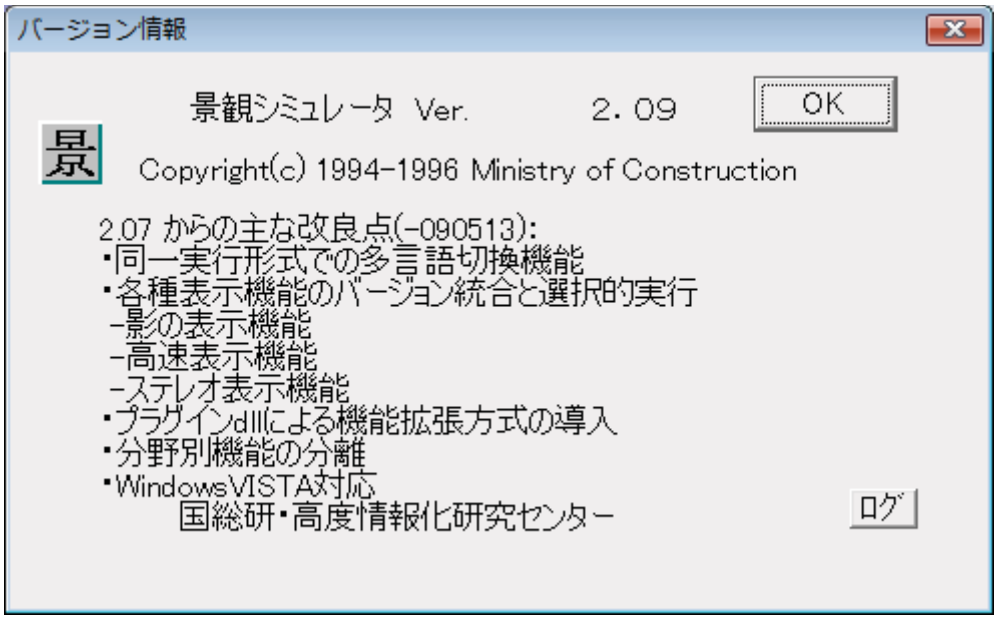

図４－７：バージョン情報表示画面

リソース：IDD_ABOUTBOX

ハンドラ：CAboutDlg(sim.cpp)

ヘルプ：log.txt（ログ　ボタンで開く）

（３）確認その１

重大な処理を実行する前に、ユーザーの再確認を求めるために使用する。二択の選択を簡単に求める場合にも使用可能である。下の例は、NT3.51 以前の WINDOWS 系 OS では使用できないダイアログを表示するに先立って選択を行っている。

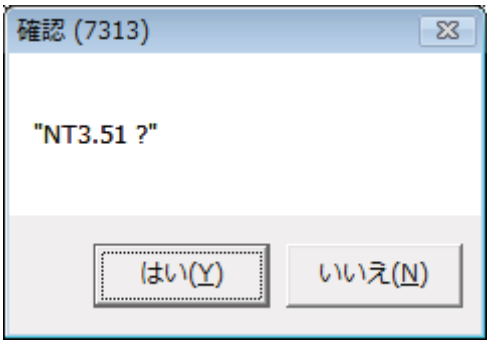

図４－８：確認画面１

z3Confirmation(メッセージ番号)　メッセージ番号は、ERR_MSG.txt による

ERR_MSG.txt に登録する各メッセージは、

コード、番号、テキストの構成となっている。

例「C 7001 データ（%s）を保存しますか？」

コードには、E（エラー）、W（警告）、I（情報提供）、C（確認）は４種類があり、C では、YES か NO の二つの選択ボタンを表示する。結果に基づいて、処理が分岐する。

（4）ファイル選択ダイアログ

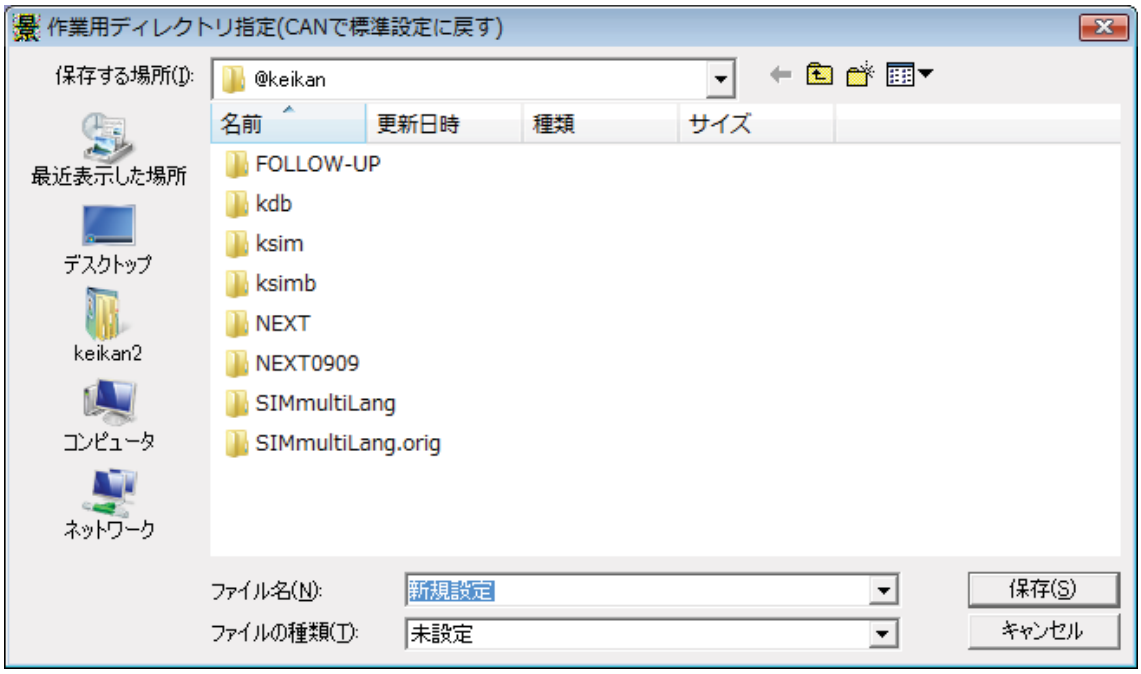

図４－９：ファイル選択画面

CFileDialog　(MFC の組み込みクラス)

ファイルを開く場合、名前を付けてファイルに保存する場合に、ファイル名をユーザーに選択させるために用いる。（5）が使用できない古い OS の場合には、ディレクトリを選択する場合にも用いる。

（5）ディレクトリ選択ダイアログ

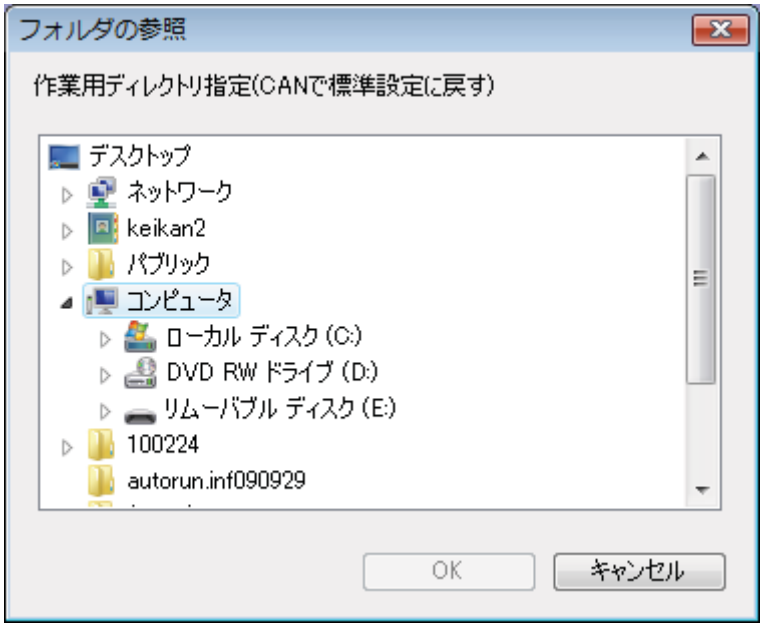

図４－１０：フォルダ選択画面

システム　::SHBrowseForFoler() (OS 組み込み関数)

ディレクトリを選択する場合に用いる。NT3.51 以前の OS では使用できない。

（６）エラー・警告・インフォメーション

あらゆる局面で、ユーザーに情報提供する必要が生じた場合に、現在選択されている言語でメッセージを出力する。原則として、「エラー」は、システムの側に起因する障害、「警告」はユーザーの誤操作に起因する障害、「インフォメーション」は障害ではなく、ユーザーに、現在の状況や操作の結果に関する理解を促すための情報を提示する。

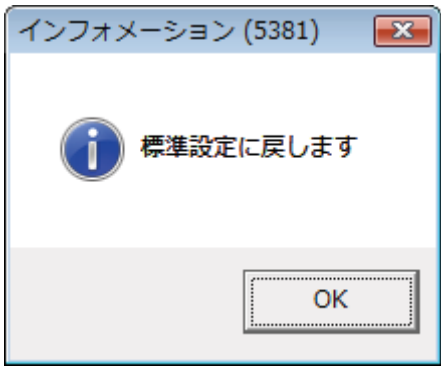

図４－１１：メッセージ表示画面

z3Message(番号)　　番号は、ERR_MSG.[言語コード].txt による（上記の例では、ERR_MSG.ja.txt)。

（７）確認その２

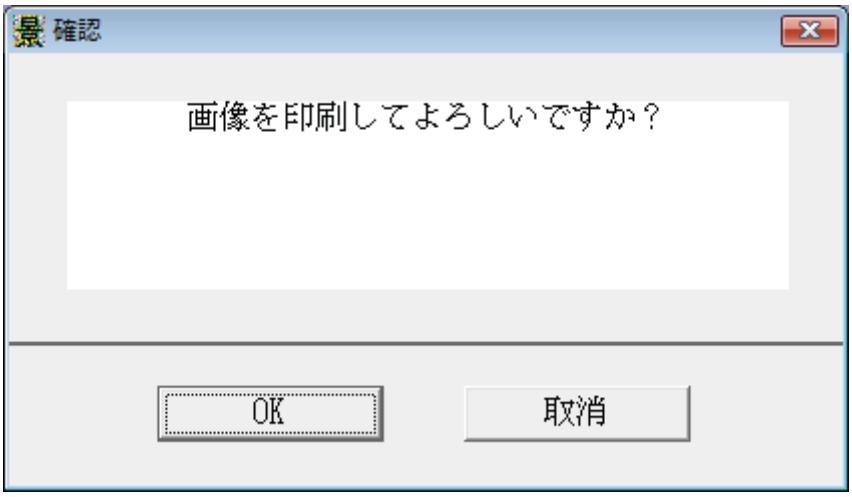

図４－１２：確認画面２

リソース：IDD_KAKUNIN　ハンドラ：CKakunin (kakunin.cpp)

上記（３）の z3Confirmation とほぼ同じ機能であるが、より長いテキストを表示することができる。

（８）言語選択

ダイアログのメニュー、コントロール（ボタンなど）の表示、メッセージ、ヘルプ等に

使用する言語を選択する。選択可能な言語は、Language ディレクトリの下にあるサブ・ディレクトリ名を以て認識する。ディレクトリ名は、ISO639 の言語コード（アルファベット２文字、第 11 章参照）を用いる。

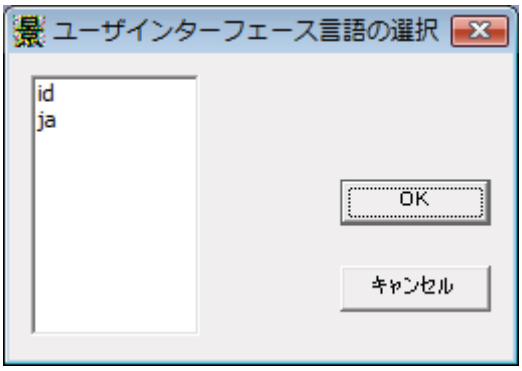

図４－１３：言語選択画面

リソース：IDD_DIALOG_SEL_LANG ハンドラ：CDlgSelLang(dlgsellang.cpp)

（9）選択したオブジェクトの情報を見る

メイン画面で選択し強調表示されているグループ（より正確には一つのリンクの子グループ）に関する情報を表示する。

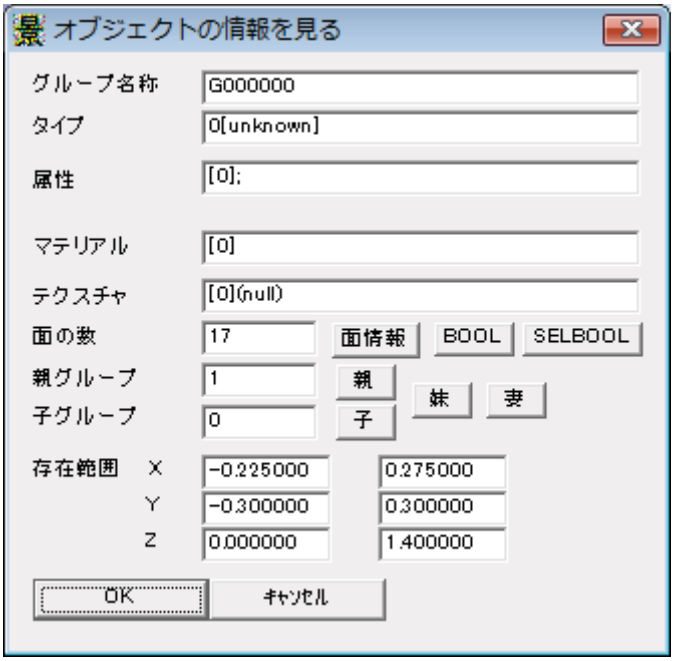

図４－１４：オブジェクト情報表示画面

リソース：IDD_DLG_INFO　ハンドラ：CInfo (info.cpp)

（１０）移動・回転・スケール

メイン画面で選択し強調表示されているグループ（より正確には一つのリンクの子グル

ープ）の上方リンクに定義されたマトリックスを変更することにより、オブジェクトの位置・スケール・角度を変更する。オブジェクトがサミット直下のグループであった場合には、ルートとのリンク・マトリクスを単位マトリクス以外のものに設定することは不適切であるため、中間に新たなグループを一つ挿入して、これにマトリクスを設定する。その場合には、メッセージを発して、ユーザーにその旨を伝える。

図４－１５：移動・回転・スケール編集画面

リソース：IDD_DLG_MOVE　ハンドラ：CEditMove (editmove.cpp)

ヘルプ：editmove.txt

（１１）配置・コピー

LSS-G ファイルを選択して、位置を指定し、モデルに追加する。

配置方法は、単体、リニア、エリアの３種類から選択する。単体の場合には、配置する位置を平面図表示画面上のマウス・クリックで次々と指定する。リニア配置では、平面図上で経路（例えば街路樹であれば、道路脇の歩道上）を指定し、配置間隔を指定した上で一括配置する。エリア配置では、平面図上に領域（例えば公園）を指定し、配置密度を指定した上で一括配置する。

配置する対象物は、５種類まで同時に選択した上で、指定した比率で乱数により混合しながら配置することができる。

配置する対象物の選択方法は、配置オブジェクト選択先選択ダイアログ(14)により、５通りの方法の内のいずれかを用いる。

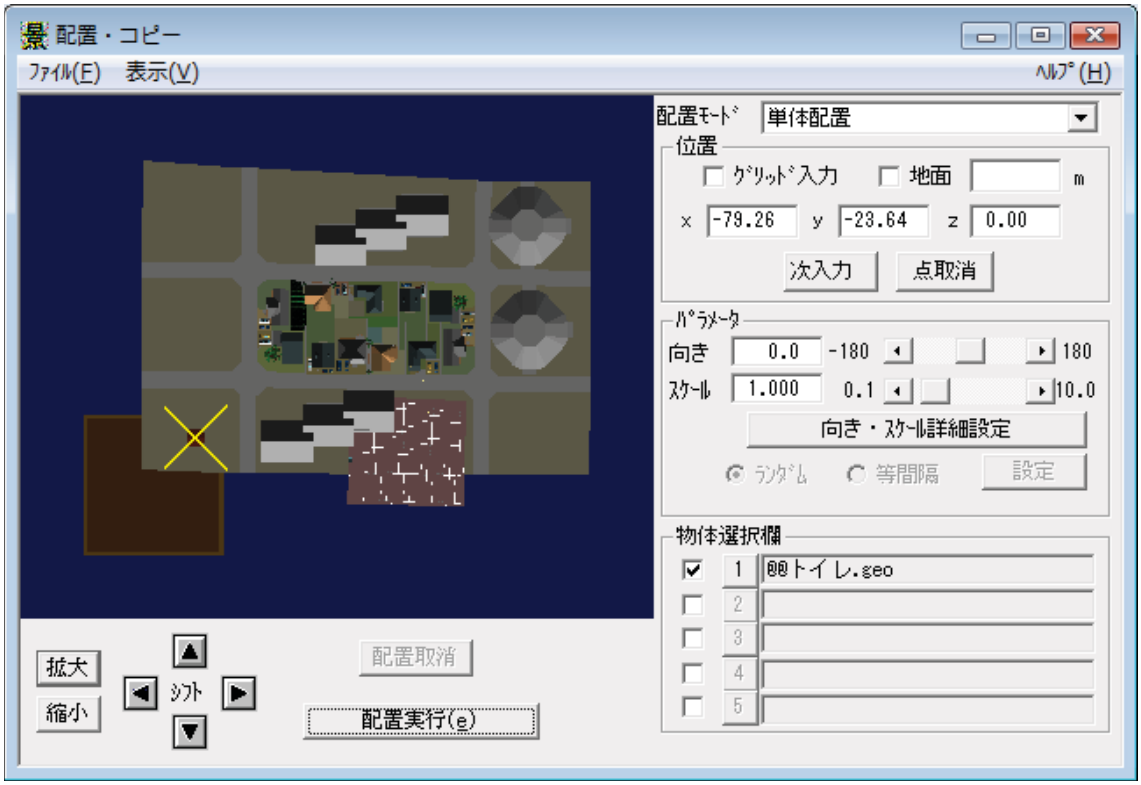

図４－１６：配置・コピー画面

① メニュー：IDR_MENU_HAICHI

メニュー構成：
［ファイル］
＋［終了］終了する
［表示］
＋［グリッド］１m 固定のグリッドの表示の ON／OFF
＋［メジャー］スケール自動設定による縮尺の表示の ON／OFF
＋［全体視界］存在するオブジェクト全てを表示する平面図表示範囲とする
＋［上下範囲］平面図表示する高さ範囲を設定する
＋［縦横範囲］平面表示する水平範囲を設定する
＋［表示モード］
＋＋［テクスチャ表示］平面図をテクスチャ表示する
＋＋［シェーディング表示］平面図を面で表示する
＋＋［ワイヤーフレーム表示］平面図を辺・稜で表示する
＋＋［オプション設定］
＋＋＋［地面のみ表示］地面の属性があるオブジェクトだけを表示する
＋＋＋［地面テクスチャ表示］地面の属性があるオブジェクトだけテクスチャ表示する
＋＋［オプション解除］オプションの設定を解除する

② GL 画面：COrthoVW m_orthoView (orthovw.cpp)
物体を配置する位置や、リニア配置の線形、エリア配置の領域等を指定する
③ 右側ダイアログ：IDD_DLG_HAICHI_R
ハンドラ：CHaichiWnd (haichiwn.cpp)、メンバ変数：CDialogBar::m_Prm_Bar
物体選択欄のボタン（１～５）でポップアップ・メニューを表示する
「向き・スケール詳細設定」ボタンで、詳細設定ダイアログを開く
「設定」ボタンで、配置パラメータ設定ダイアログを開く
④ 下側ダイアログ：IDD_DLG_HAICHI_D
ハンドラ：CHaichiWnd(haichiwn.cpp)、メンバ変数：CDialogBar::m_Ctl_Bar
⑤ヘルプ：haichi.txt

（１２）配置詳細設定パラメータ

配置ダイアログ（１１）の配置の向き・スケール詳細設定ボタンから起動する。単体は位置、リニア配置、エリア配置における、オブジェクトの向きとスケールを設定する。

図４－１７：配置詳細設定パラメータ編集画面

リソース：IDD_DLG_HAICHI_DTL　ハンドラ：CHaichiDtlDlg(haichidt.cpp)
ヘルプ：haichdt.txt

（１３）配置パラメータ

配置ダイアログ（１１）の配置の設定ボタンから起動する。リニア配置、エリア配置における向きやスケールのゆらぎ、ばらつきを設定する。

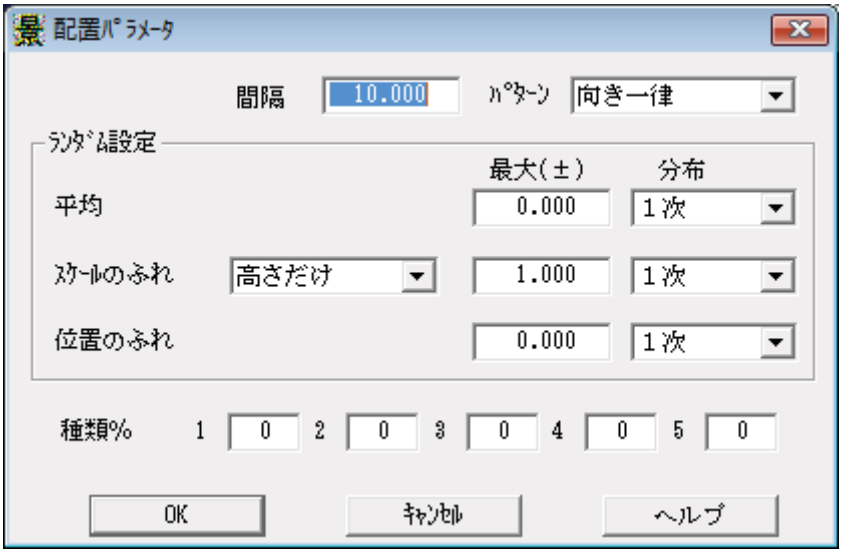

図４－１８：配置パラメータ編集画面

リソース：IDD_DLG_HAI_PRM　ハンドラ：CHaichiPrmDlg (haichipr.cpp)

ヘルプ：haichipr.txt

（１４）配置するオブジェクトの取得先選択

配置するオブジェクトの選択方法を選択する。既にモデル中に存在するグループをコピーして配置する「画面セレクション」、直接ファイル名を指定する「ＬＳＳ－Ｇ」、データベースから検索する「景観構成要素」「景観材料」および「外部関数」(パラメトリックな部品：原始図形及びユーザー定義による部品)が選択可能である。

図４－１９：配置オブジェクト取得先選択画面

リソース：IDD_DLG_HAI_SEL　ハンドラ：ChaiselDlg (haiseldl.cpp)

（１５）視点座標

視点情報(CAM)を構成する視点、注視点、天頂ベクトル、焦点距離、アスペクト比（縦横比）、前後範囲 zNear-zFar を設定する。

a（自動）にチェックを入れておくと、メイン画面下の接近・後退で自動的に zNear-zFar

を調整するため、近づき過ぎて表示から消えることはない。

このダイアログは開いた時点の視点を表示したり、ダイアログで数値を変更するだけではなく、ダイアログを開いたままにしておくと、別の視点設定方法によるＣＡＭ情報の変化を、逐次表示する機能も有している。

図４－２０：視点座標編集画面

リソース：IDD_DLG_SHITEN ハンドラ：CShiten (editshit.cpp)

ヘルプ：editshit.txt

（１６）可視範囲解析

選択したオブジェクトが見える割合を、設定したエリアについて解析し、可視率を色分けで表示する。処理アルゴリスムは、設定したエリアにおいて、解析の精度でメッシュ点を設定し、それぞれの点に関して、地面が存在すれば地面の高さを設定したうえで、各点からオブジェクトだけを表示た場合の表示画面に占めるオブジェクトの面積（ドット数）に対する、全地物を表示した場合のオブジェクトの画面上の面積（ドット数）の割合を計算する。

計算結果は、色分けの画像として表現する。ＬＳＳ－Ｓ編集モードにおいて解析を実行すると、解析結果を画像として保存し、メイン画面を平面図とした場合にも表示する。さらに、このシーンをシャッターで記録した上でファイル保存すると、画像名称と表示範囲を EFFECT コマンドとして保存する。その際に画像には、例えば「VISUAL003.sgi」といった名称を自動的に付し、kdb/image ディレクトリに保存する。これに対応する効果コマンドは、例えば

[例]: E3=EFFECT(VISUAL,62.18, 182, 54.81, 854.81, VISUAL003.sgi);

（効果コマンド名 VISUAL、画像の左下隅の座標と横縦サイズ、画像ファイル名）

という形式で自動的に生成し、LSS-S ファイルに保存する。

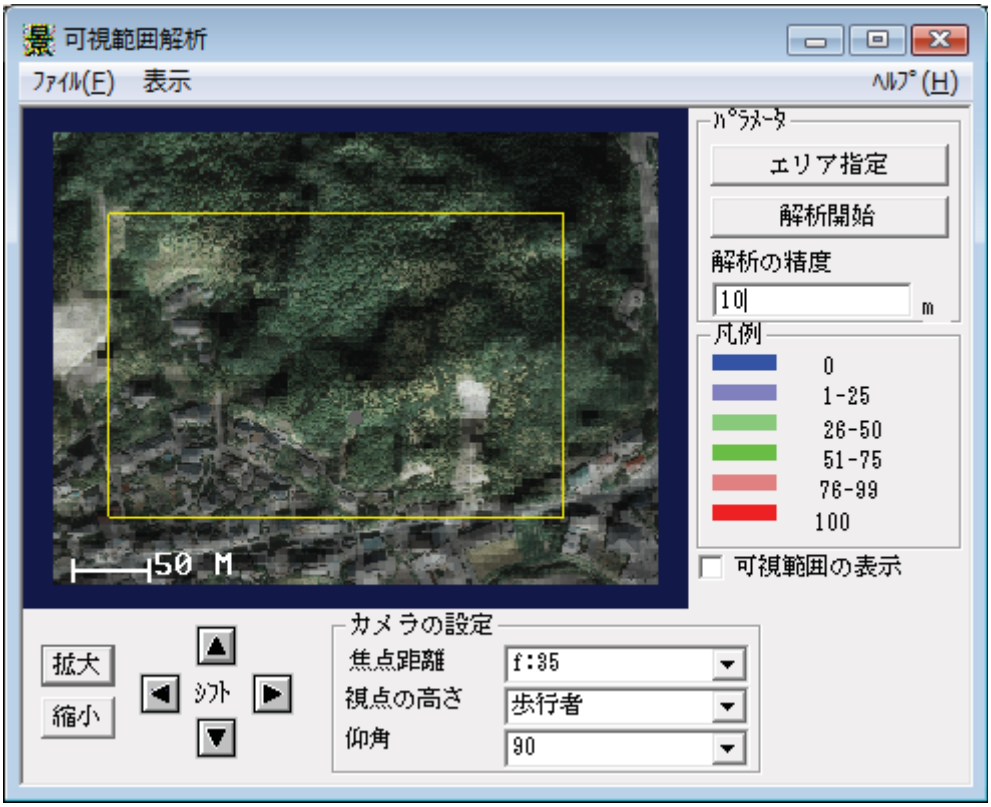

図４－２１：可視範囲解析画面

①　メニュー：IDR_MENU_KEIRO

メニュー構成：
［ファイル］
＋［終了］終了する
［表示］
＋［全体視界］存在するオブジェクト全てを表示する平面図表示範囲とする
＋［上下範囲］平面図表示する高さ範囲を設定する
＋［縦横範囲］平面表示する水平範囲を設定する
＋［表示モード］
＋＋［テクスチャ表示］平面図をテクスチャ表示する
＋＋［シェーディング表示］平面図を面で表示する
＋＋［ワイヤーフレーム表示］平面図を辺・稜で表示する
＋＋［オプション設定］
＋＋＋［地面のみ表示］地面の属性があるオブジェクトだけを表示する
＋＋＋［地面テクスチャ表示］地面の属性があるオブジェクトだけテクスチャ表示する
＋＋［オプション解除］オプションの設定を解除する

②　ＧＬ画面：COrthoVW m_orthoView (orthovw.cpp)

③　右側ダイアログ：リソース：IDD_DLG_PRM_KASHI

ハンドラ：CKashiWnd (kashiwnd.cpp)
メンバ変数：CDialogBar::m_Prm_Bar
④　下側ダイアログ：リソース：IDD_DLG_CTL
ハンドラ：CKashiWnd(　.cpp)
メンバ変数：CDialogBar::m_Ctl_Bar
⑤　ヘルプ：kashiwnd.txt

（１７）平面図表示画面における表示の上下範囲の設定

平面図表示（オルソ画面）を中心とするダイアログ（配置/コピー、平面生成など）のメニューから起動し、平面図表示における上下の範囲を設定する。建築物のデータ等の場合には、ある階の床と天井の間の高さを上限とすると、その階の平面図を表示することができる。地面よりも高い高さを下限とすると、地面の表示を消すことができる。

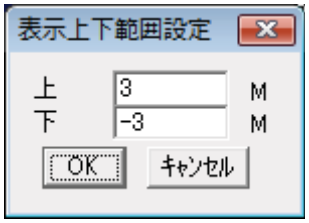

図４－２２：表示上下範囲設定画面

リソース：IDD_DIALOG_KELIHATAN ハンドラ：CKelihatan (kelihatan.cpp)

（１８）平面図表示画面におけるシフトと拡大・縮小の移動レートの設定

平面図表示（オルソ表示）を用いた編集ダイアログにおいて、表示範囲の選択を詳細に行いたい場合は、ステップレートを小さく設定する。また、広域的に探索する場合にはステップレートを大きく設定する。シフトの値は画面サイズに対する比率であるため、１を超えた値に設定すると、離れた領域をサンプルすることとなり、中間が抜けるため、見落とす場合がありうる。

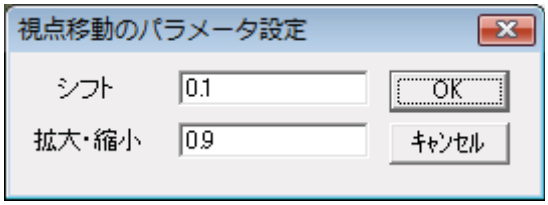

図４－２３：視点移動パラメータ設定画面

リソース：IDD_VIEWPARAM ハンドラ：CVwParam (vwparam.cpp)

（１９）視点設定

平面図上で位置をクリック指定することにより、その地点から眺めた景観をメイン画面で表示する。注視点はオブジェクトの選択または画面クリックで設定する。

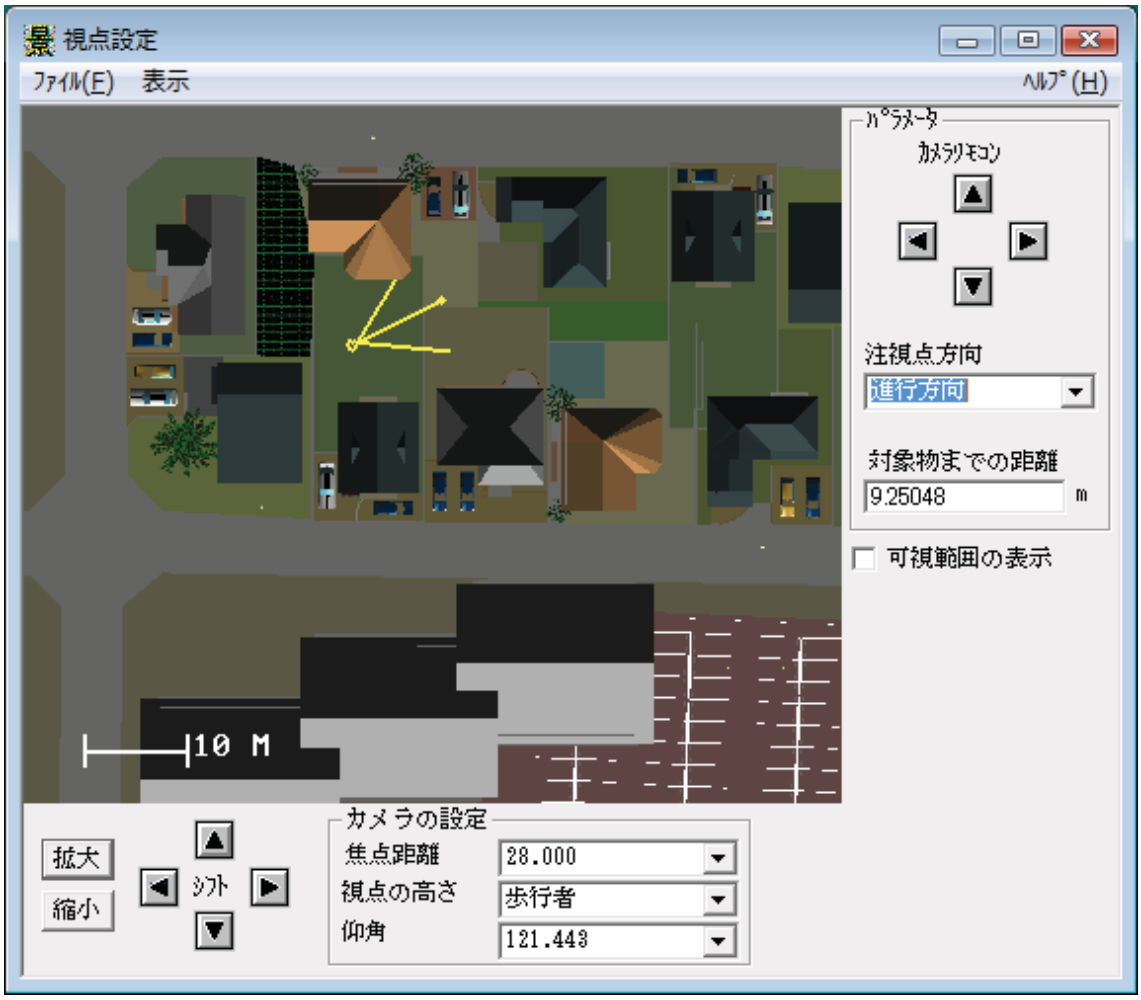

図４－２４：視点設定画面

① メニュー：IDR_MENU_KEIRO　（１６．①と同じ）

② ＧＬ画面：COrthoVW m_orthoView (orthovw.cpp)

③ 右側ダイアログ：リソース：IDD_DLG_PRM_SHITEN

ハンドラ：CShitenWnd (shitenwn.cpp)

メンバ変数：CDialogBar::m_Prm_Bar

③ 下側ダイアログ：リソース：IDD_DLG_CTL

ハンドラ：CShitenWnd(shitenwn.cpp)

メンバ変数：CDialogBar::m_Ctl_Bar

⑤ ヘルプ：shitenwn.txt

（２０）移動経路設定画面

画面上で移動経路（軌道）を設定し、メイン画面に移動途中の風景の変化をアニメーションで表示する。移動経路は、点列で指定するが、スプライン曲線で滑らかにした上で、一定の間隔で移動することにより、スムーズなカメラワークとすることができる。設定し

た経路は、LSS-G 形式の線としてファイル保存し、後日再利用することができる。

なお、アニメーションとして保存される AVI 形式のファイルは、通常の場合、三次元データよりもはるかに大きくなる。

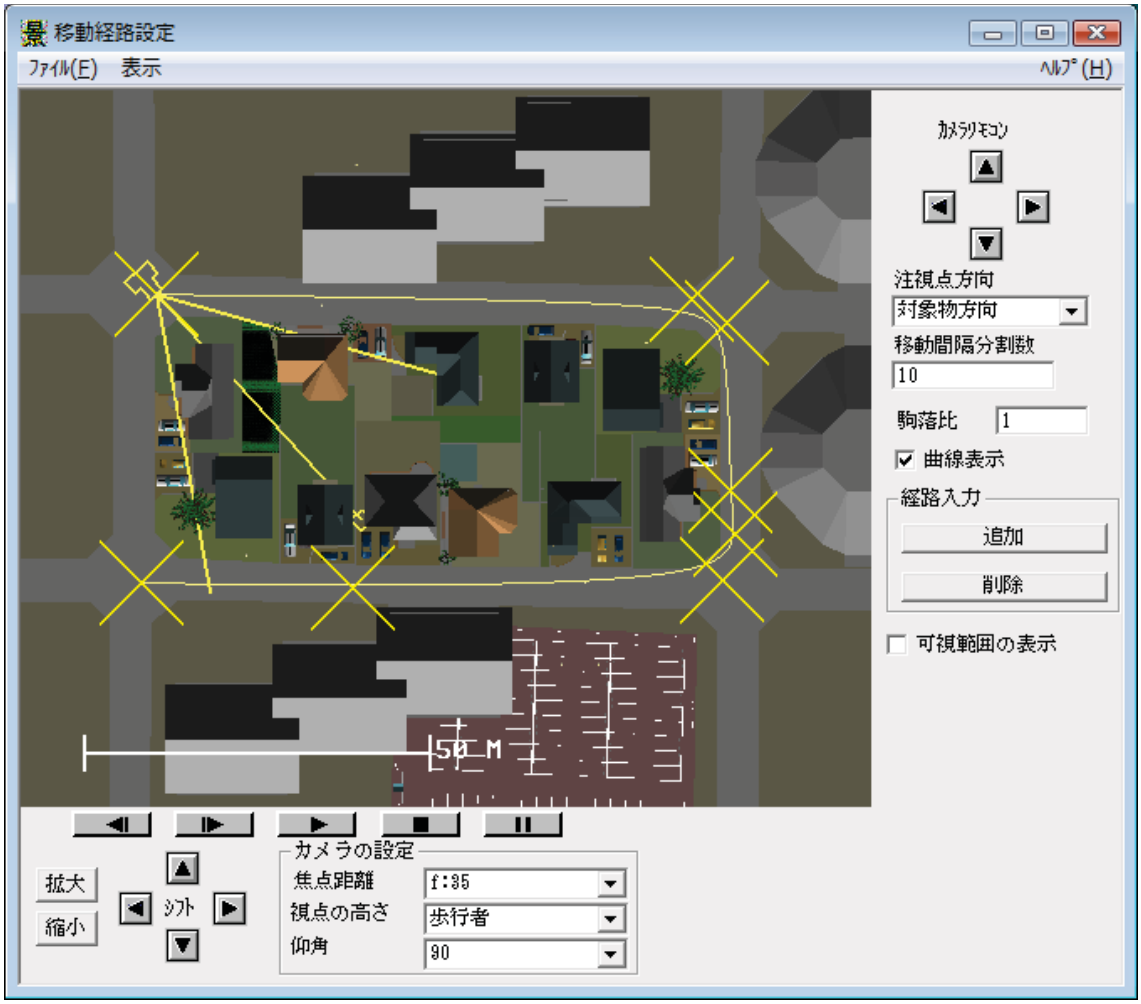

図４－２５：移動経路設定画面

① メニュー：IDM_MENU_KEIRO

メニュー構成：
［ファイル］
＋［経路読込］ファイル名を選択して、保存してある経路を読み込む
＋［経路保存］ファイル名を指定して、経路を保存する
＋［上書保存］読み込んだ名前、または以前保存した名前で保存する
＋［動画保存］上演と同時にアニメーションを avi ファイルとして保存する
＋［終了］終了する
［表示］
＋［全体視界］存在するオブジェクト全てを表示する平面図表示範囲とする
＋［上下範囲］平面図表示する高さ範囲を設定する

＋［縦横範囲］平面表示する水平範囲を設定する
＋［表示モード］
＋＋［テクスチャ表示］平面図をテクスチャ表示する
＋＋［シェーディング表示］平面図を面で表示する
＋＋［ワイヤーフレーム表示］平面図を辺・稜で表示する
＋＋［オプション設定］
＋＋＋［地面のみ表示］地面の属性があるオブジェクトだけを表示する
＋＋＋［地面テクスチャ表示］地面の属性があるオブジェクトだけテクスチャ表示する
＋＋［オプション解除］オプションの設定を解除する

② OpenGL 画面：COrthoVW m_orthoView(orthovw.cpp)
③ 右側ダイアログ：
リソース：IDD_DLG_PRM_IDO　ハンドラ：CKeiroWnd (keirownd.cpp)
CDialogBar:: m_Prm_Bar
④ 下側中段ダイアログ：
IDD_DLG_PLAY　ハンドラ：CKeiroWnd(keirownd.cpp)
⑤ 下側下段ダイアログ
IDD_DLG_CTL　ハンドラ：CKeiroWnd (keirownd.cpp)
CdialogBar::m_Ctl_Bar
⑥ ヘルプ：keirownd.txt

（２１）マテリアル、カラー、テクスチャの編集（古典的スタイル）

メイン画面で選択したオブジェクトに関して、カラーやマテリアルの編集を行う。

カラー編集は、三原色とアルファ値をスライドバーで指定する他、左側の数値表示欄の数値をキーボード入力し、フォーカスを別の数値表示欄などに移す方法でも指定することができ、スライドバーの位置も自動的に追随する。

上方の見出し用の OpenGL 画面に球面（左）と平面（右）があり、設定したカラーを表示する。

このダイアログを開いたまま、メイン画面で次々と選択対象を変更し、能率的に作業することができる。メイン画面のメニュー[編集][選択モード]のサブメニューから、選択方法を、グループ、面、同色面のいずれかに設定する。初期値はグループとしており、その場合、選択したグループの全ての面に、同じカラーが設定される。

特殊な機能として、[地面の属性]のチェックボックスにより、選択したオブジェクトに対して地面の属性を設定／解除することができる。

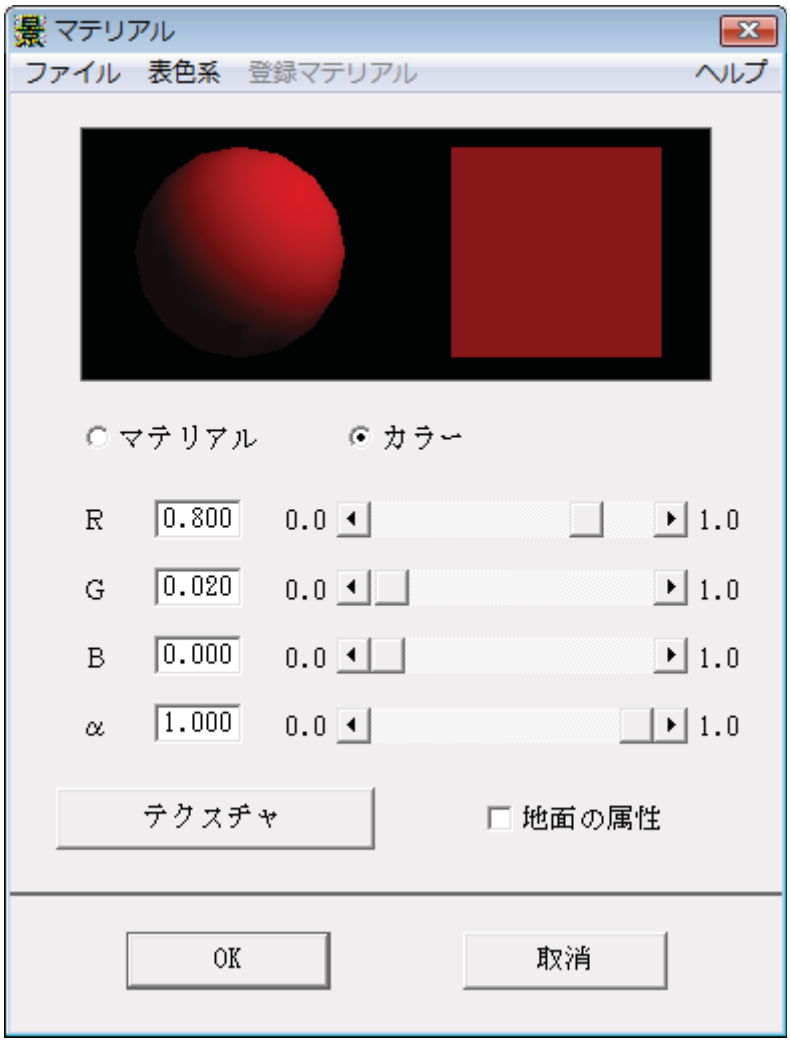

図４－２６：カラー・マテリアル編集画面

① メニュー：IDR_MENU_MATERIAL

メニュー構成
［ファイル］
＋［閉じる］ダイアログを終了する
＋［カラーのファイル保存］カラーを LSS-G 形式のファイルに保存する
＋［カラーの上書保存］読み込んだファイル名、以前保存したファイル名にて保存する
＋［カラーのファイル読込］保存してあるカラーを読み込む
［表色系］３種類の表色系で相互変換するダイアログ（２８）を開く
［登録マテリアル］
＋［マテリアル選択画面］グラフィックなマテリアルの選択画面(25)を開く
＋［マテリアルファイル選択］マテリアルファイル選択画面(24)を開く
（この画面でマテリアルファイルが選択された場合、同じダイアログを用いて
マテリアル選択に進む）

＋［選択済マテリアルファイル］直ちにファイル中のマテリアルを選択する(24)
＋［指定マテリアル内容表示］選択対象物に設定されているマテリアルを表示

② OpenGL 画面：CMate2Dlg m_dlg (mate2.cpp)
③ ダイアログ　リソース:IDD_DLG_MATERIAL　ハンドラ:CEditMate (editmate.cpp)
④　ヘルプ：edimate.txt

（２２）テクスチャ編集（クラシック）

テクスチャをファイル名で選択し、選択したオブジェクトに貼り付ける。

図４－２７：テクスチャ編集画面

① メニュー：IDR_MENU_TEXTURE
② ダイアログ　リソース：IDD_DLG_TEXTURE ハンドラ：CEditText (edittext.cpp)
③　ヘルプ：edittext.txt

（２３）テクスチャ・リスト（古典的スタイル）

・概要：テクスチャ編集ダイアログのテクスチャ選択ボタンで起動する。SGI 形式のテクスチャの一覧を表示する。何も選択せずに OK ボタンを押すと、一般的なファイル選択ダイアログを開き、JPG など、様々な形式の画像をテクスチャとして選択する。

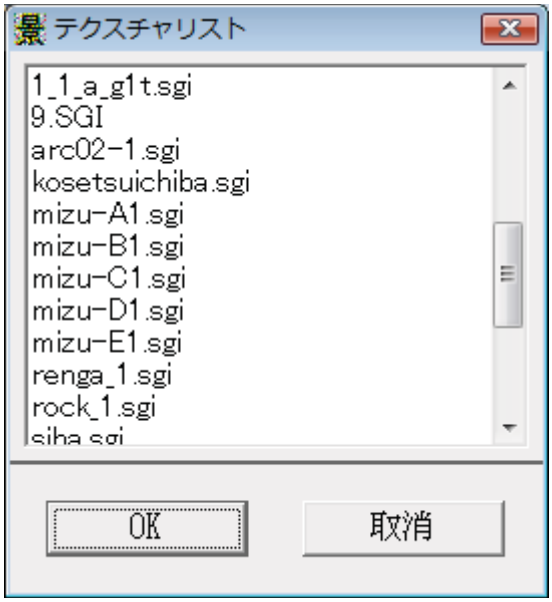

図４－２８：テクスチャファイル選択画面

リソース：IDD_DLG_TEXLIST ハンドラ：CEditTexl (edittexl.cpp)

（２４）マテリアル選択（古典的スタイル）

マテリアル編集画面で、ラジオボタンをマテリアルに変更し、メニューのマテリアルを選択すると開く。同じダイアログを二段構成で使用し、第一段階では、マテリアルファイルの選択を、また第二段階ではマテリアルの選択を行う。

第一段階：

選択して OK ボタンを押すと、そのマテリアルファイルの中で定義されたマテリアルの一覧を表示して第 2 段階に進む。また、何も選択されない状態で OK ボタンが押された場合には、現在編集中の地物のいずれかの部分に適用されているマテリアルの一覧を表示して第２段階に進む。

第二段階：

一覧表示されているマテリアルからユーザーが選択した時点で、マテリアル設定画面(21)の上方の色玉に、そのマテリアルを反映させる。

OK ボタンが押された時点で、メイン画面で選択されている編集対象物にマテリアルを設定する。この時、画面で何も選択されておらず、選択したマテリアルを適用する対象がない場合には、選択したマテリアルファイルの内容をメモ帳で表示する。

また、一覧表示されているマテリアルからユーザーが何も選択していない状態で、OK ボタンが押された場合には、設定されているマテリアルを解除する。このとき、メイン画面で何も選択されていない場合には、確認の上でマテリアル解消のための特殊な処理を実行する。この処理においては、全ての地物に適用されているマテリアルを解除し、近い色のカラーに置き換える。この時、マテリアルに含まれていたカラー以外の、鏡面反射率や輝

度などに関する情報は失われる。

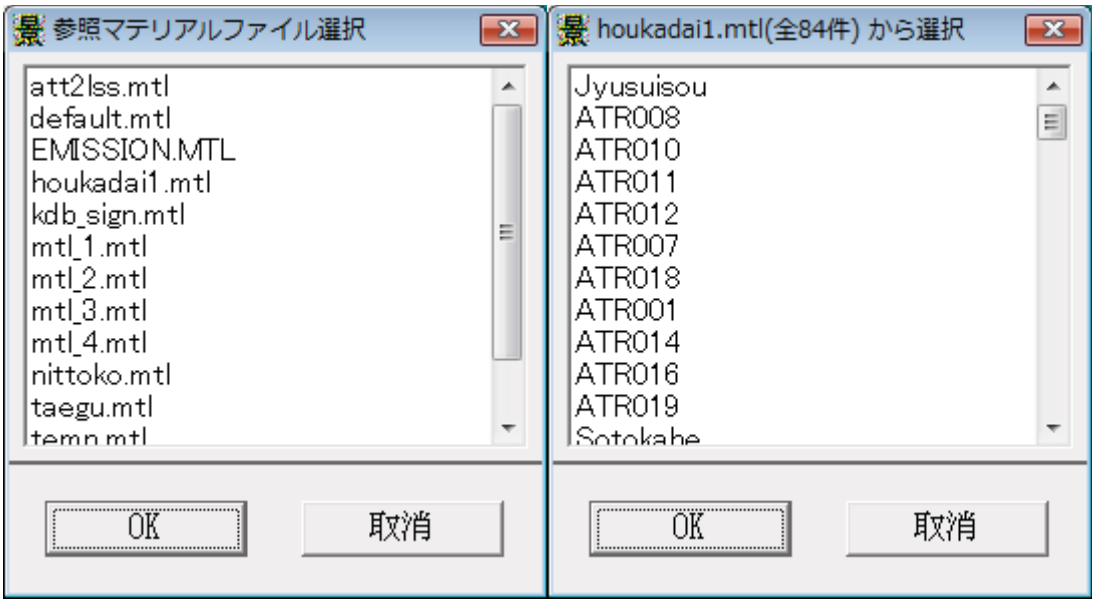

図４－２９：マテリアル選択画面（古典的）

リソース：IDD_DLG_MAT_LIST ハンドラ：CMatList (matlist.cpp)

（２５）グラフィックなマテリアルの選択

選択しているマテリアル・ファイルの中で定義されたマテリアルを、参照表示しながら選択できる編集画面である。

中央にマテリアル名称のリストを表示する。左側に縦に並んだ７の小さなカラーボックスは、各々OpenGL の画面である。右側のスクロールバーは一つの OpenGL 画面である。右側の OpenGL 画面は、マテリアル・ファイルに含まれる全てのマテリアルを総覧しており、スクロールバーのつまみは、マテリアルが７を超える場合に表示され、その内７個分の縦幅となっている。スクロールバーに対応する位置にある 7 種類のマテリアルが、左側の７個のカラーボックスに表示される。

上にある色球の OpenGL 画面は左右二つあり、設計検討の途上で、右側に一時保存し、比較しながら決定することができる。

現在一覧表示しているマテリアル・ファイル名称を、タイトルに補足表示している。

図４－３０：グラフィックなマテリアル選択画面

① メニュー：IDR_MENU_MATERIAL2

［ファイル］
＋［閉じる］ダイアログを終了する。
［色見本］所定のディレクトリにある全てマテリアルファイル名をサブメニューに表示

② 上左の OpenGL 画面 CSphereDlg m_KyuDlg (spheredlg.cpp)
③ 上右の OpenGL 画面 CSphereDlg m_saveKyuDlg (spheredlg.cpp)
④ 左側の OpenGL 画面 CMatFrm m_colFrm[7] (matfrm.cpp)
⑤ 右側の OpenGL 画面 CColorSashDlg m_obiDlg (colsashdlg.cpp)
⑥ ダイアログ
　リソース：IDD_DIALOG_MATERIAL　ハンドラ：CEditMaterial (editmat2.cpp)
⑦ ヘルプ：editmate2.txt

（２６）グラフィックなテクスチャ編集

ディレクトリにあるテクスチャを画像で一覧表示する。

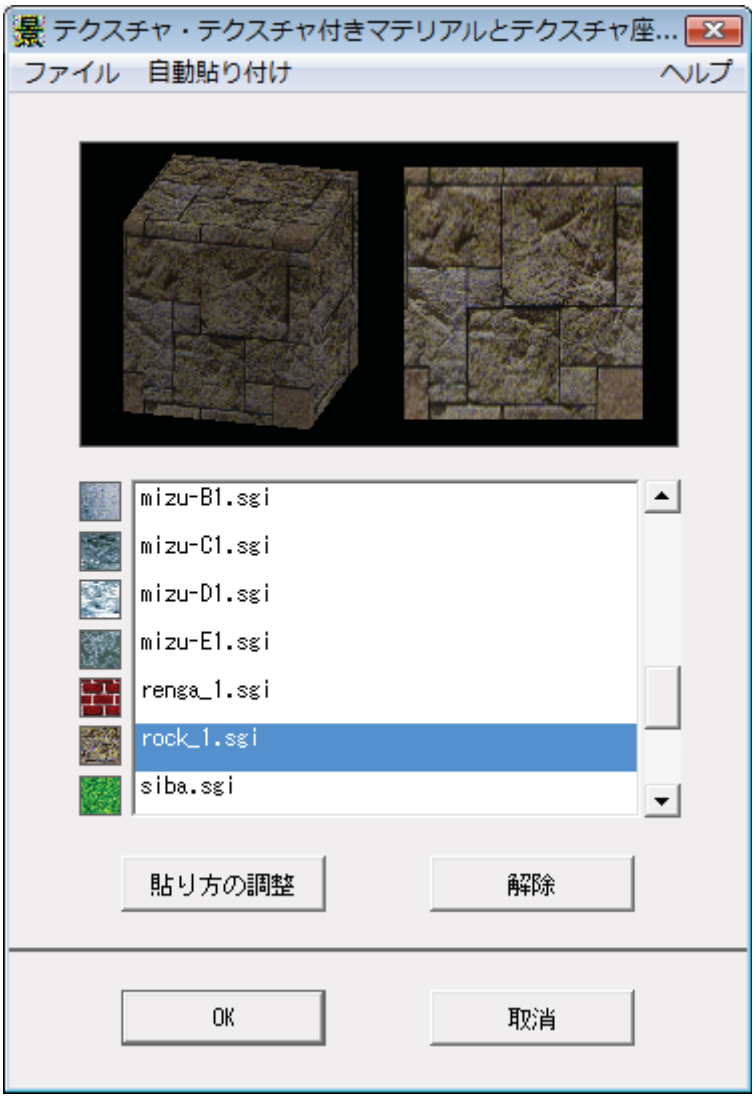

図４－３１：グラフィックなテクスチャ選択画面

① メニュー：IDR_MENU_TEXTURE2

［ファイル］ ＋［閉じる］ダイアログを終了する。 ［自動貼り付け］autotex.set ファイルによる定義内容をサブメニューとして構成する （autotex.set は、選択されている言語のディレクトリにあるものを使用する）

② 上の OpenGL 画面 CCubeDlg m_boxDlg (cubedlg.cpp)

③ 左の 7 個の OpenGL 画面 CTexFrm m_texFrm[7] (texfrm.cpp)

④ ダイアログ

リソース：IDD_DIALOG_TEXTURE ハンドラ：CEditTexture(edittex2.cpp)

⑤ ヘルプ：edittex2.txt

（２７）テクスチャの貼り方の調整

多くの面から成る近似的曲面に対して、平行投影で、連続的に見えるようにテクスチャを貼るために、選択したグループの全ての面テクスチャ座標（２－９参照）を一括設定する。このための個々の面のテクスチャ座標を計算のために必要となる、集合的な座標軸とスケールを設定する。u 軸と v 軸をベクトルとして指定すると、uv 原点として指定したポイントから、u 軸 v 軸に直角な向きに、u スケール・v スケールで倍率を掛けたしたテクスチャを投影され、この時選択したグループを構成する各面の頂点の uv 座標を計算し、これをテクスチャ座標として各頂点に設定する。

図４－３２：テクスチャの貼り方の調整画面

リソース：IDD_DIALOG_TEXTURE_MAP ハンドラ：CTextureMap(textmap.cpp)

（２８）様々な表色方式によるカラー編集

選択した表色方式のスライドバーを移動すると、他の表色方式による換算値に対応して、スライドバーを自動的に動かす。

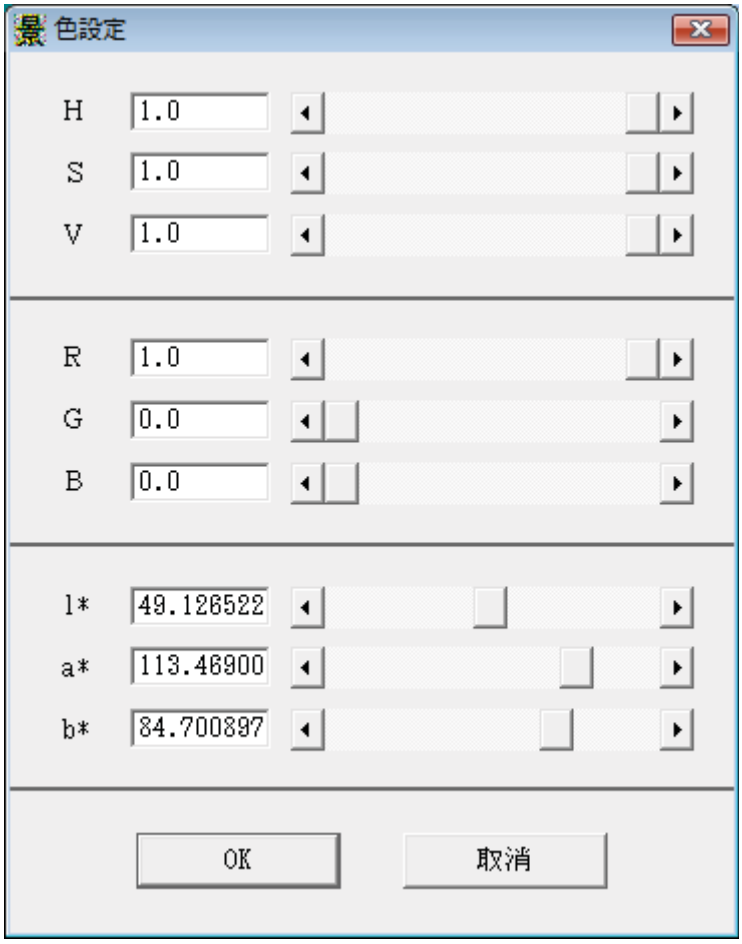

図４－３３：様々な表色系による色設定画面

リソース：IDD_DIALOG_COLOR_SET ハンドラ：CColorSet (colorset.cpp)

（２９）光源の自動設定

月日・時分と、検討地域の緯度経度から、太陽方位を自動的に計算する。同時に、選択した天候に応じて、副光源を設定する。更に、効果（EFFECT）を用いて、霧も設定することができる。

詳細設定ボタンで、クラシックな光源設定ダイアログを開く（一つづつ色と位置を設定する）。

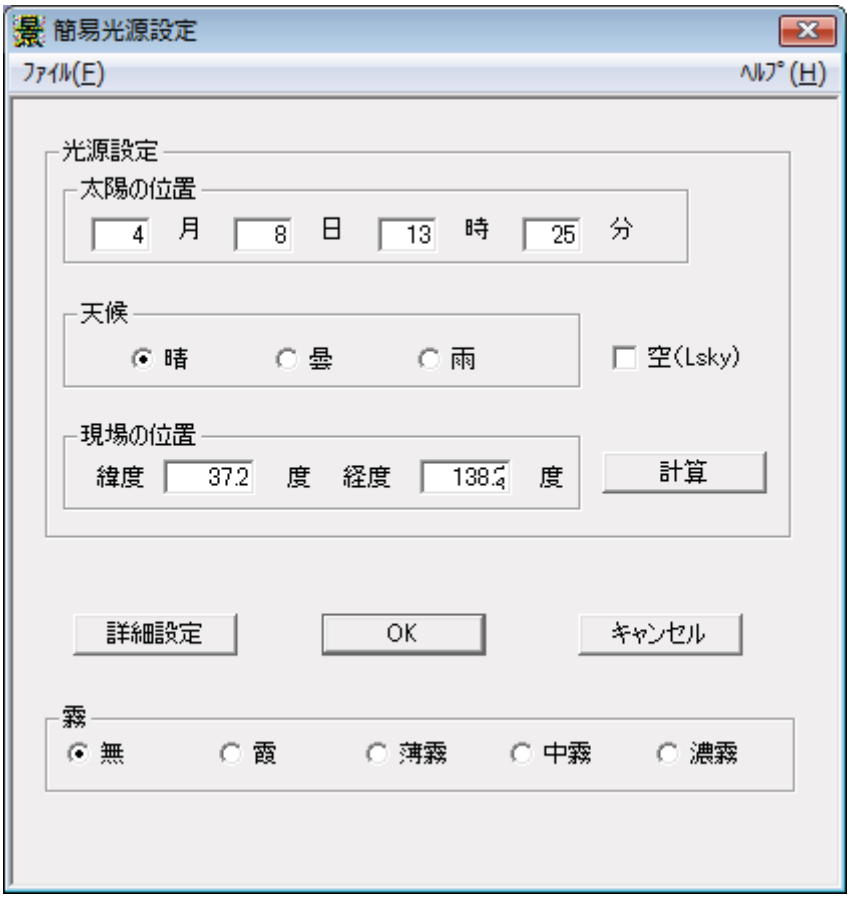

図４－３４：簡易光源設定画面

① メニュー：IDR_MENU_GY

［光源読込］LSS-S 形式のサブセットを用いたファイル（拡張子.s）を読み込む ［光源保存］光源をファイルに名前を付けて保存する ［光源上書］読み込んだファイル名または以前保存したファイル名で保存する ［終了］ダイアログを閉じる

② リソース：IDD_DLG_NEW_LIGHT ハンドラ：Cgydlg (gydlg.cpp)

③ ヘルプ：gydlg.txt

（３０）光源グループ設定

光源グループを構成する光源を編集する。シーン・ファイルを編集する中で、同じ光源条件を複数のシーンに対して共通で使用するような編集を行うことができる。

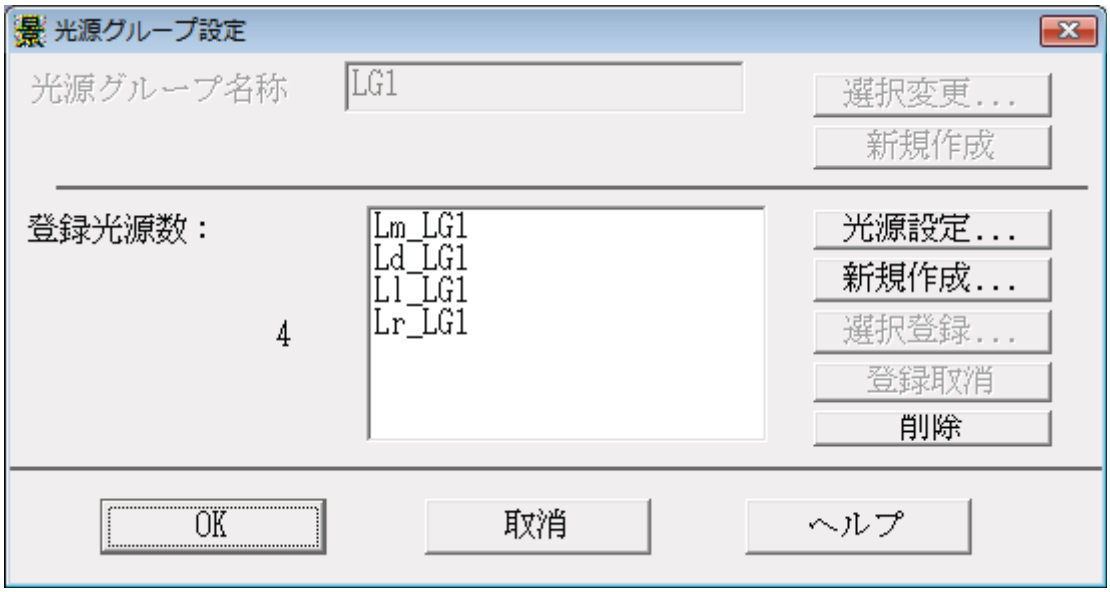

図４－３５：光源グループ設定画面

①リソース：IDD_DLG_LIGHT　ハンドラ：CEditLigh (editligh.cpp)

②ヘルプ：editligh.txt

（３１）古典的な光源編集

光源グループ編集ダイアログで選択した一つの光源のカラーと位置を編集する。OpenGLの表示画面で、球と正方形に試し適用する。

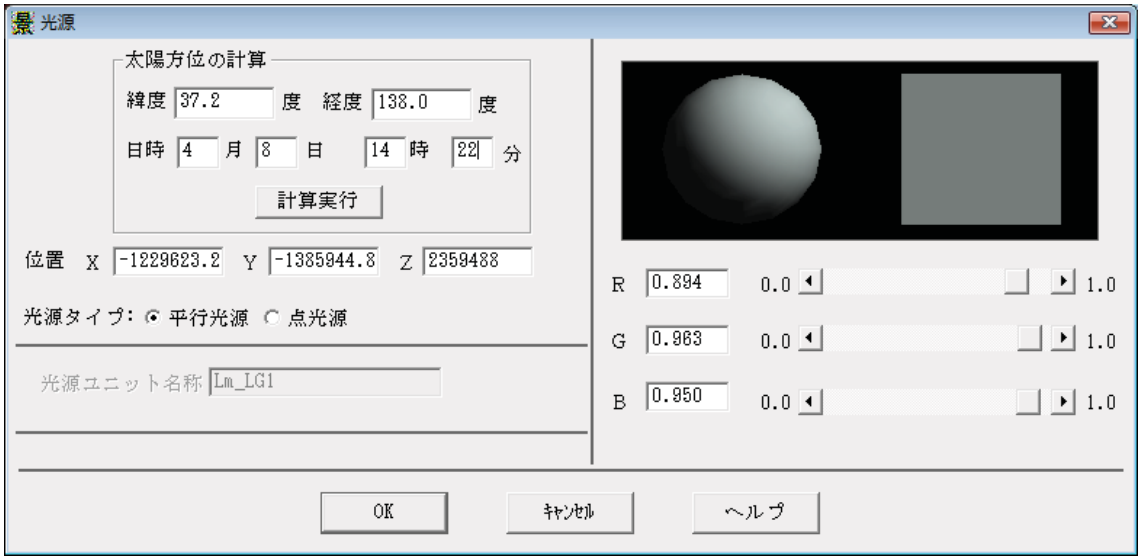

図４－３６：光源ユニット設定画面

① OpenGL 画面：CeditLight2Dlg　m_dlg (editlig2.cpp)

② ダイアログ　リソース：IDD_DLG_KOUGEN　ハンドラ：Ckougen (kougen.cpp)

③ ヘルプ：kougen.txt

（３２）経年変化

システムの時間を変更する。時間依存するマテリアルを貼り替えると共に、時間依存するモデルも更新する。

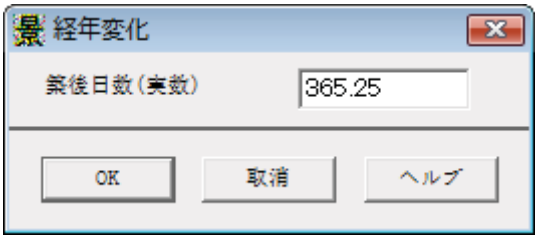

図４－３７：経年変化設定画面

①ダイアログ

リソース：IDD_DLG_KEINEN_HENKA　ハンドラ：CDispKeinen (dispkein.cpp)

②ヘルプ：dispkein.txt

（３３）グリッド

オルソ系画面（平面図、立面図）表示モードにおいて、指定した格子間隔のグリッドを表示する。このほかに、縮尺、可視範囲解析結果の表示の ON-OFF をこのダイアログで指定する。

図４－３８：グリッド設定画面

①ダイアログ　　リソース：IDD_ DLG_GRID ハンドラ：CGrid (grid.cpp)

②ヘルプ　　grid.txt

（３４）アンチエリアシング(Anti Aliasing)

鋸状の境界線の表示を解消する。このために、表示画面よりも細かい解像度の画面に対する表示処理を行い、表示画面の解像度に下げる際に、集約するメッシュ格子点の色彩を平均することにより、中間色を挿入し、ギザギザ感を減じる。

OK ボタンを待たずに、設定が変更された時点で、直ちに表示に反映する。

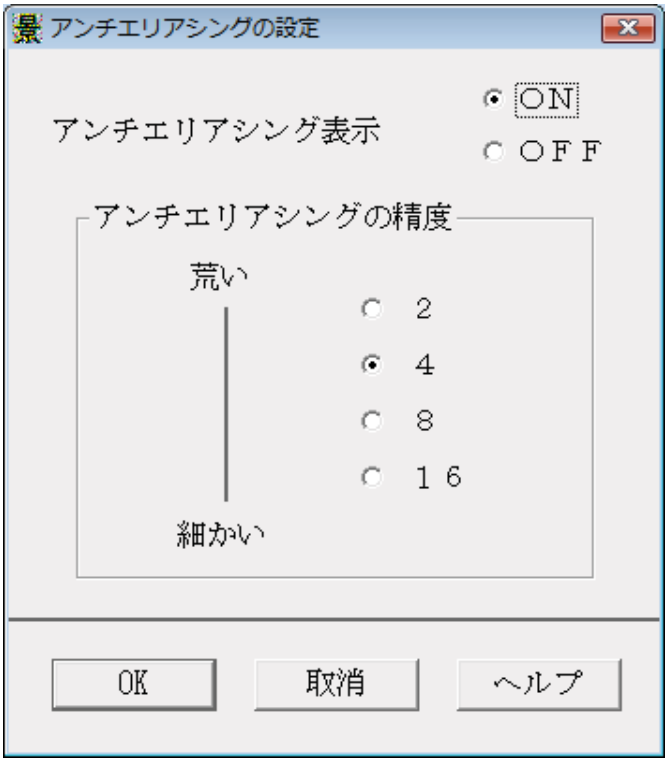

図４－３９：アンチエリアシング設定画面

①ダイアログ　リソース：IDD_DLG_ANTI_AREA ハンドラ：CantiAlias (antialia.cpp)

②ヘルプ　antiarea.txt

（３５）影のパラメータ設定

影のタイプ、副次的光源を影の部分に適用するか否かを設定するチェック・ボックスを有する。

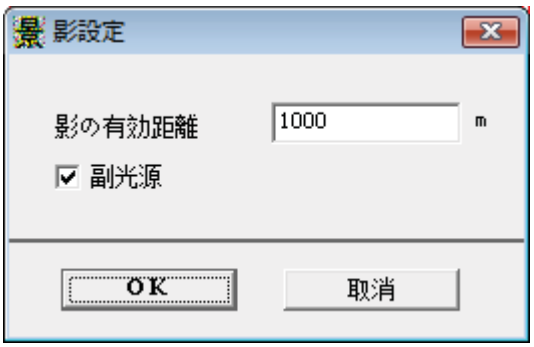

図４－４０：影設定画面

リソース：IDD_DLG_SHADOW　ハンドラ：CShadowDlg(shadowdlg.cpp)

（３６）ステレオ表示設定

ステレオ・モード、影の長さ、ステレオ・スワップを指定する。

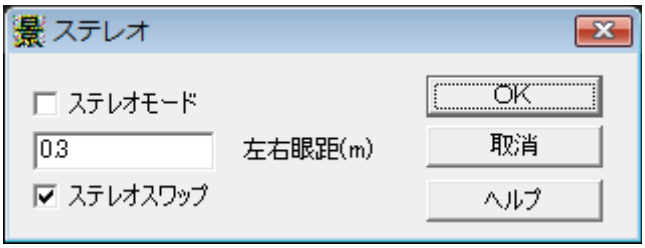

図４－４１：ステレオ表示設定画面

リソース： IDD_DIALOG_STEREO　ハンドラ：Cstereo (stereo.cpp)

（３７）平面の編集

メイン画面でセレクト行為が無い場合、頂点列から新しい平面を生成する。

メイン画面でセレクトが行われていた場合には、既存平面の再編集を行う。

このダイアログが開いてからメイン画面で面が選択された場合には、既存平面に対する穴あけ加工を行う。

その他、面を定義するのと同様な手順で指定した頂点列を用いて道路や河川等の掃引体のための断面を定義したり、病的な面を治療するなど、様々な機能がある。

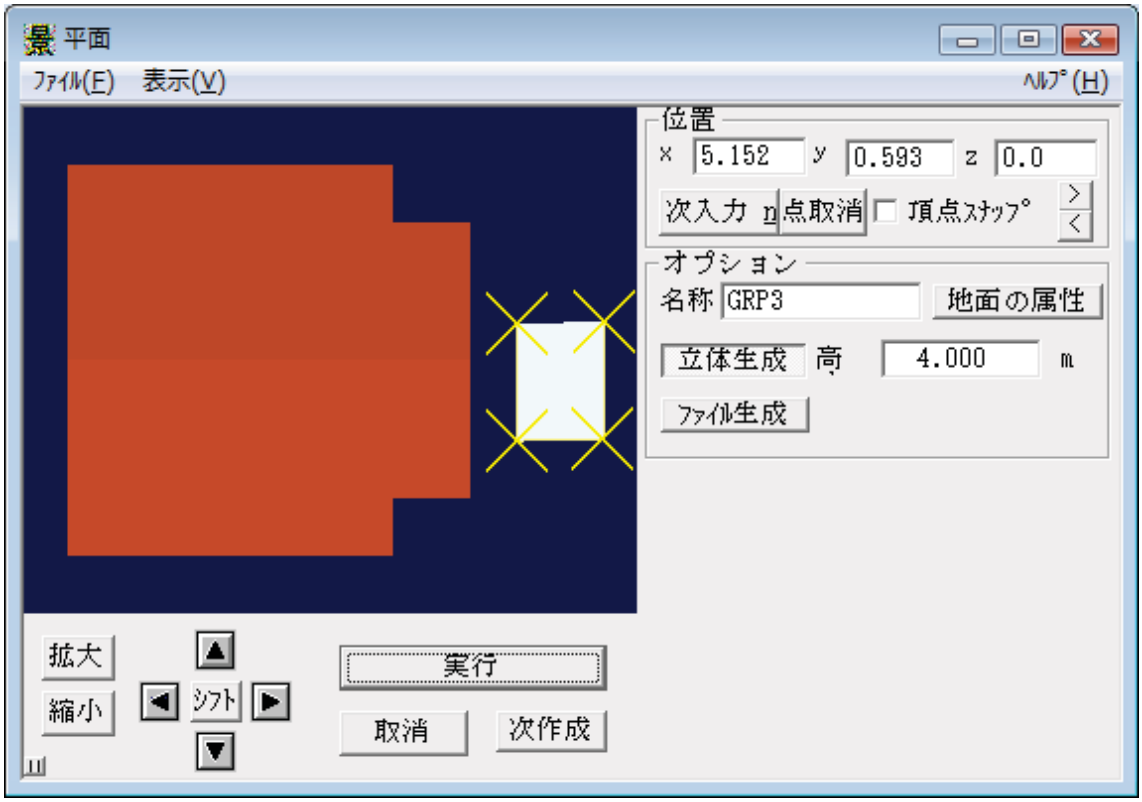

図４－４２：平面入力画面

① メニュー：IDR_MENU_PLANE

メニュー構成：
［ファイル］
＋［選択された面の編集］
＋＋［面の削除］選択した面を削除する
＋＋［面を別グループにする］新たなグループを作成し、選択した面を移管する
＋＋［面をファイル保存する］ファイル名を指定して、選択した面を保存する
＋＋［面を裏返す］面を構成する頂点列を逆回りに変換する
＋＋［面の情報を見る］頂点数、法線、カラー等の情報を表示する
＋＋［面の病気を治療する］病的な面を正常な面に変換する
＋［点列を線として保存］
＋＋［道路断面として保存］図形を XZ 平面上の線として保存、道路断面リストに追加
＋＋［河川断面として保存］図形を XZ 平面上の線として保存、河川断面リストに追加
＋＋［線として保存］頂点列を線として保存する
＋［終了］終了する
［表示］
＋［グリッド］1 m 固定のグリッドの表示の ON／OFF
＋［メジャー］スケール自動設定による縮尺の表示の ON／OFF
＋［全体視界］平面図表示範囲を調整し、存在するオブジェクト全てを表示する
＋［上下範囲］平面図表示する高さ範囲を設定する
＋［縦横範囲］平面図表示する水平範囲の移動の刻みを設定する
＋［表示モード］
＋＋［テクスチャ表示］平面図をテクスチャ表示する
＋＋［シェーディング表示］平面図を面で表示する
＋＋［ワイヤーフレーム表示］平面図を辺・稜で表示する
＋＋［オプション設定］
＋＋＋［地面のみ表示］地面の属性があるオブジェクトだけを表示する
＋＋＋［地面テクスチャ表示］地面の属性があるオブジェクトだけテクスチャ表示する
＋＋［オプション解除］オプションの設定を解除する

② OpenGL 画面：COrthoVW　m_orthoView (orthovw.cpp)

③ 右側ダイアログ
リソース：IDD_DLG_PLANE_PRM　ハンドラ：CPlaneWnd (planewnd.cpp)
メンバ変数：CDialogBar　m_Prm_Bar

④ 下ダイアログ
リソース：IDD_DLG_CTL　ハンドラ：CPlaneWnd (planewnd.cpp)
メンバ変数：CDialogBar m_Ctl_Bar

（３８）クラシックな線の編集

以下のような操作を行う。

① 頂点数を指定した上で、各頂点の座標値を入力し、新たな線を生成する。
このとき、線分名称、適用するマテリアルを指定することができる。

② 探索ボタン：線の選択ダイアログを開き、編集対象とする既存の線を選択する

③ 接続ボタン：線の選択ダイアログで対象とする線分を選択した上で、この線分と接続している線分を検索し、存在する場合には一つの折れ線に統合する。

④ 軌跡保存：現在選択している線を、道路生成・河川生成で使用する中心線軌跡データとして保存する。

⑤ 軌跡読み込み：中心線軌跡データを読み込み、線として編集する

⑥ >X>Y>Z：線の XYZ 座標を、巡回的に交換する

⑦ 逆順：線の頂点列を逆準に変換する

⑧ 一括変換：ダイアログ(62)を開き、平行移動、倍率の一括変換を行う

図４－４３：線の編集画面

ダイアログ：IDD_DLG_PRIM_LINE　ハンドラ：CPrimLineDlg (primline.cpp)

ヘルプ：primline.txt

（３９）線分の一括変換

選択した線に対する一括処理内容とパラメータを設定する。

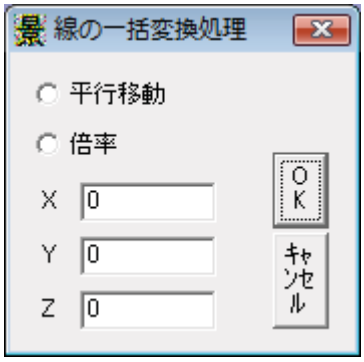

図４－４４：線分の一括変換画面

リソース：IDD_LINE_PROCESS　ハンドラ：CLineProcess (lineproces.cpp)

実際のデータ変換は、ダイアログ呼び出し元の CPrimLineDlg の中で実行している。

（４０）橋

開発初期の原始的機能。選択した種類の橋（固定的外部ファイルデータ）を読み込んで、シーンの中に表示する。現在では配置コマンドによりデータベースから検索したり、直接ファイルとして読み込むことが可能。博物館的な意味合いで残している。

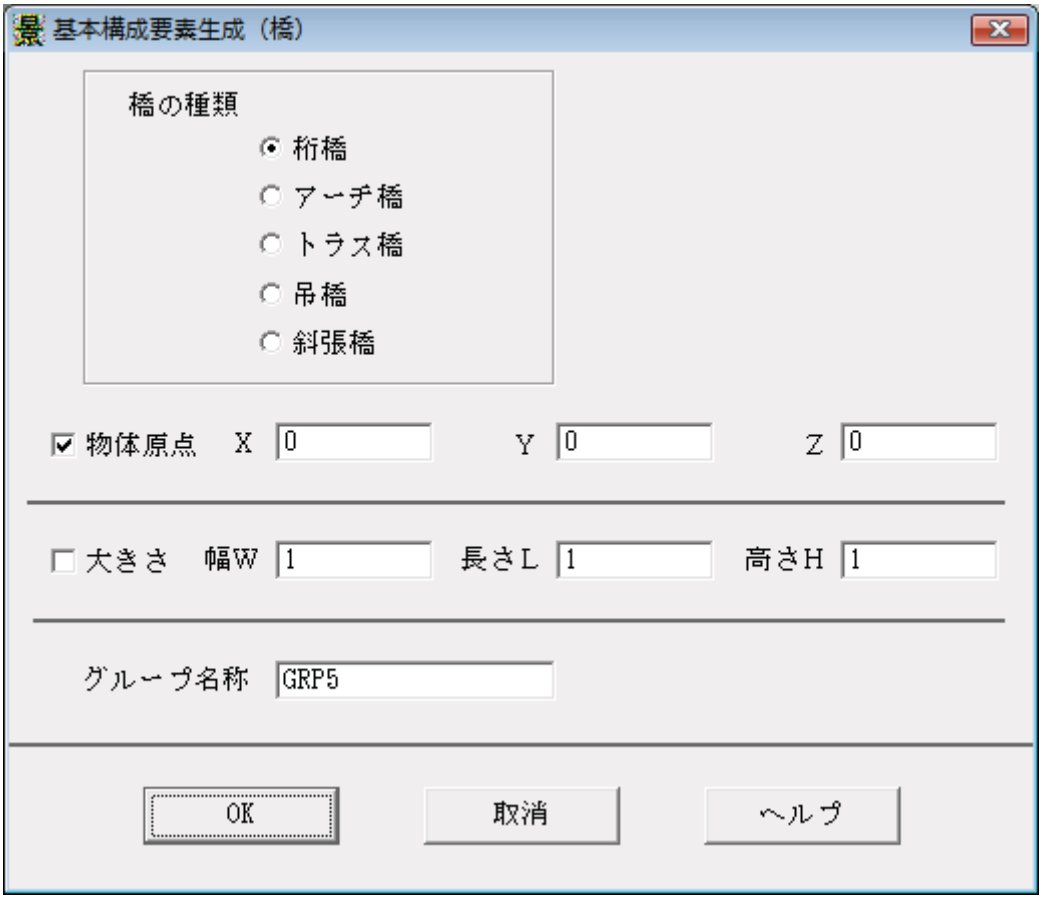

図４－４５：橋の画面

リソース：IDD_DLG_ELEM_BRIDGE ハンドラ：CElemBridgeDlg(elembrid.cpp)
ヘルプ：elembrid.txt

（４１）基本構成要素（草）

初期の機能である。その後、配置機能のエリア配置で代替可能。博物館的な意味しかない。

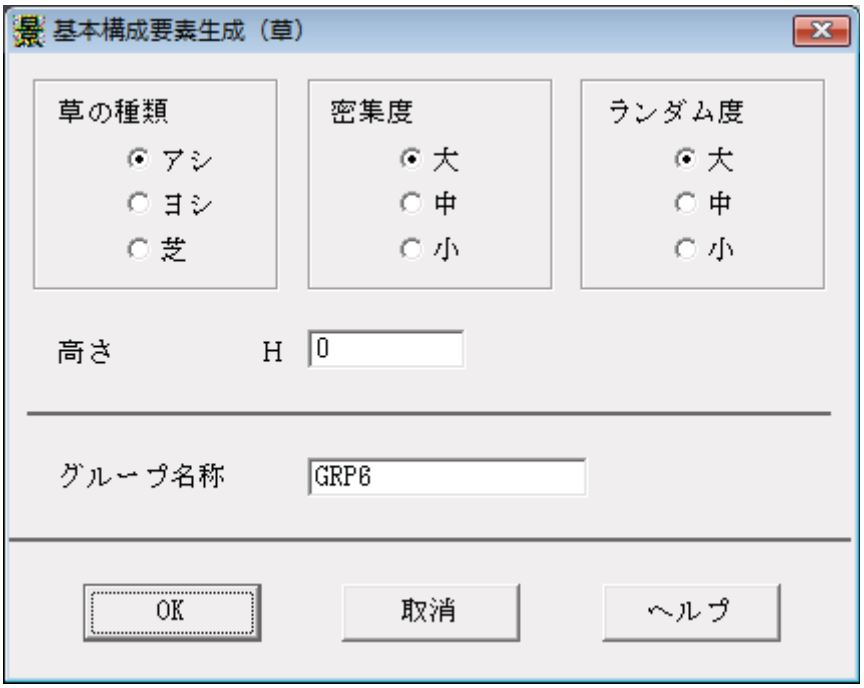

図４－４６：草の画面

リソース：IDD_DLG_ELEM_GRASS ハンドラ：CElemGrassDlg(elemgras.cpp)
ヘルプ：elemgras.txt

（４２）基本構成要素：道路

選択した断面を、平面図表示の OpenGL 画面でユーザーが指定した中心線軌跡に従って掃引して、立体形状を生成する。

中心線の軌跡は、LSS-G 形式としてファイル保存・読み込みができる。スプライン曲線により、入力された折れ線を滑らかに補間する機能がある。

オプションとして地面を識別し、地面から相対的な高さとして、軌跡上の点のｚ座標を指定することができる。

形状生成のロジックは、掃引体１面と同様である。既存の地面や地形に対する加工機能が無いため、生成した道路が地面よりも低い場合には、下に隠れてしまう。

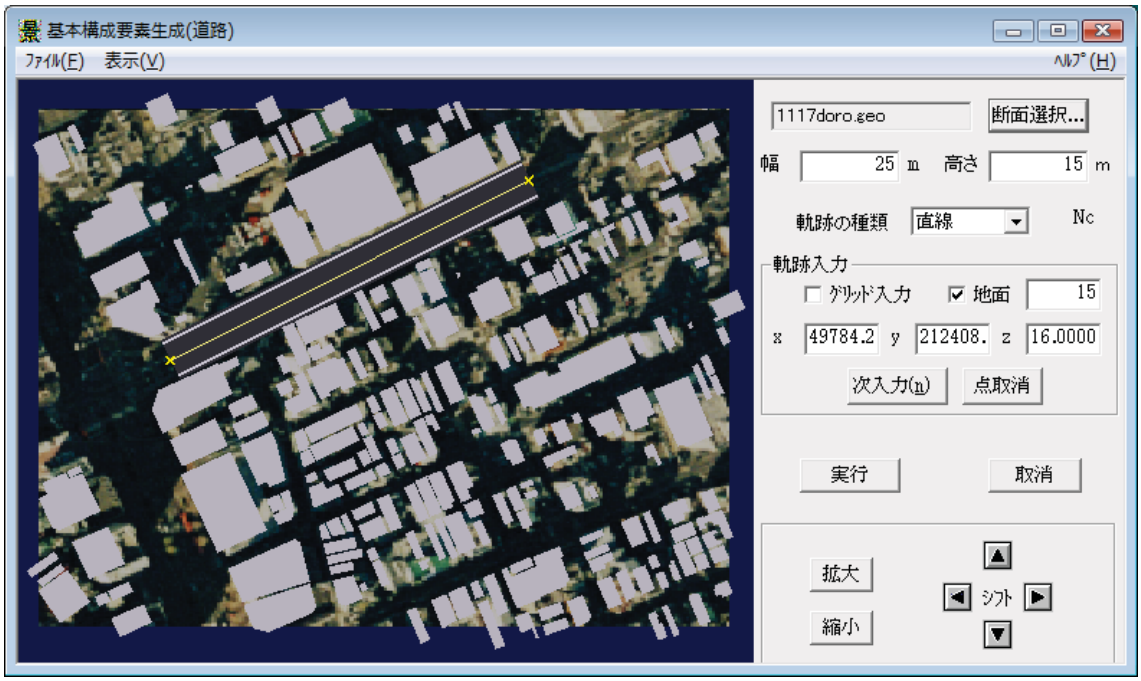

図４－４７：道路生成画面

①┌　　メニュー：IDR_MENU_HAICHI

メニュー構成：
［ファイル］
＋［経路読込］ファイル名を選択して、保存してある経路を読み込む
＋［経路保存］ファイル名を指定して、経路を保存する
＋［上書保存］読み込んだ名前、または以前保存した名前で保存する
＋［終了］終了する
［表示］
＋［グリッド］１m 固定のグリッドの表示の ON／OFF
＋［メジャー］スケール自動設定による縮尺の表示の ON／OFF
＋［全体視界］平面図表示範囲を調整し、存在するオブジェクト全てを表示する
＋［上下範囲］平面図表示する高さ範囲を設定する
＋［縦横範囲］平面図表示する水平範囲を設定する
＋［表示モード］
＋＋［テクスチャ表示］平面図をテクスチャ表示する
＋＋［シェーディング表示］平面図を面で表示する
＋＋［ワイヤーフレーム表示］平面図を辺・稜で表示する
＋＋［オプション設定］
＋＋＋［地面のみ表示］地面の属性があるオブジェクトだけを表示する
＋＋＋［地面テクスチャ表示］地面の属性があるオブジェクトだけテクスチャ表示する
＋＋［オプション解除］オプションの設定を解除する

② OpenGL 画面：COrthoVW m_orthoView (orthovw.cpp)
③ 右側ダイアログ
リソース：IDD_DLG_ROAD　ハンドラ：CRoadWnd (roadwnd.cpp)
メンバ変数：CDialogBar::m_Prm_Bar
④ ヘルプファイル roadwnd.txt

その他：リソース：IDD_DLG_ELEM_ROAD と関連ハンドラ
C:ElemRoadwayDlg(elemroad.cpp)は Ver.2.09 では使用されていない。

（４３）断面ファイルリスト

コントロール･ファイル　ROADSEC.SET に登録されている断面ファイル（LSS-G 形式）の一覧を表示し、選択・OK で、ファイルを開き、呼び出し元の道路生成ダイアログの断面データに登録を行う。

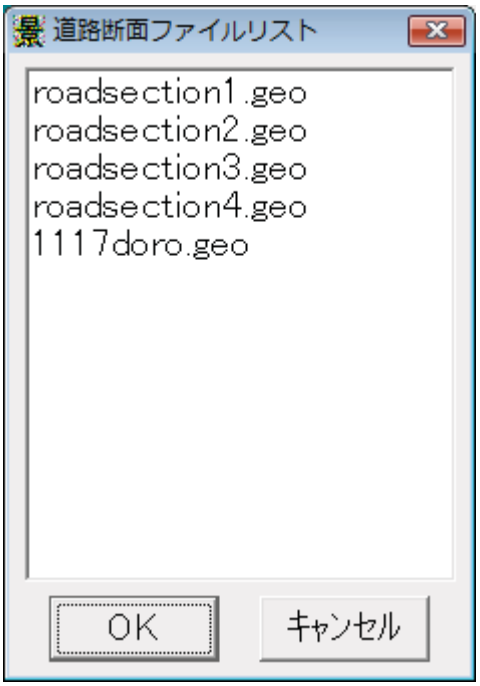

図４－４８：道路断面ファイル選択画面

リソース：IDD_DLG_DANMEN　ハンドラ：CRoadDanmen (roaddan.cpp)

（４４）基本構成要素：河川

前項と殆ど同じロジックにより、河川（堤防＋川原）を生成する。

図４－４９：河川生成画面

① メニュー：IDR_MENU_HAICHI

メニュー構成：
［ファイル］
＋［経路読込］ファイル名を選択して、保存してある経路を読み込む
＋［経路保存］ファイル名を指定して、経路を保存する
＋［上書保存］読み込んだ名前、または以前保存した名前で保存する
＋［終了］終了する
［表示］
＋［グリッド］１m 固定のグリッドの表示の ON／OFF
＋［メジャー］スケール自動設定による縮尺の表示の ON／OFF
＋［全体視界］平面図表示範囲を調整し、存在するオブジェクト全てを表示する
＋［上下範囲］平面図表示する高さ範囲を設定する
＋［縦横範囲］平面図表示する水平範囲を設定する
＋［表示モード］
＋＋［テクスチャ表示］平面図をテクスチャ表示する
＋＋［シェーディング表示］平面図を面で表示する
＋＋［ワイヤーフレーム表示］平面図を辺・稜で表示する
＋＋［オプション設定］
＋＋＋［地面のみ表示］地面の属性があるオブジェクトだけを表示する
＋＋＋［地面テクスチャ表示］地面の属性があるオブジェクトだけテクスチャ表示する
＋＋［オプション解除］オプションの設定を解除する

② OpenGL の編集画面：COrthoVW　m_orthoView (orthovw.cpp)

③ 右側のダイアログ

リソース：IDD_DLG_ROAD　ハンドラ：CRoadWnd (roadwnd.cpp)

メンバ変数：CDialogBar::m_Prm_Bar

ヘルプ：roadwnd.txt

（４５）河川断面の選択ダイアログ

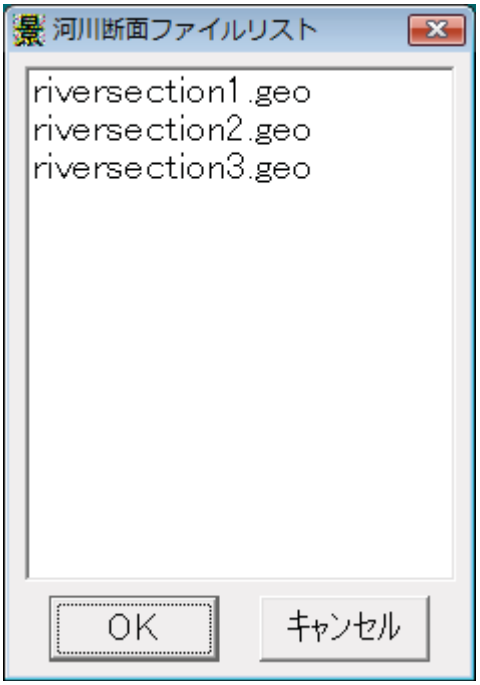

図４－５０：河川断面ファイル選択画面

リソース：IDD_DLG_ROAD　ハンドラ：CRoadDanmen (roaddan.cpp)

（４６）シャッター

シーン編集中に、ユーザーが操作すると、その時点の視点情報を記録する新たなシーンを生成し、末尾に追加する。

初期は、視点情報を記録するのみであったが、Ver.2.09 においては、機能を大幅に増補し、その他の全ての編集操作（モデルの変更、時刻の変更、背景・前景の変更）を反映させた新たなシーンを生成する仕様とした。また、既存シーンの修正も可能とするために、現在のシーンを更新するボタンを追加した。更に、ステレオ表示（７－４）、影の表示（７－５）、高速表示処理（７－６）など、表示方法が多様化したことに対応して、このようなオプショナルな表示環境に関する情報も、従来活用されていなかった EFFECT コマンドの形でシーンデータ(s3Scene 構造体及び LSS-S ファイル)に記録できるようにした。これらにより、

外部ファイル形式（３－２）を変更することなく、異なる実行環境におけるプレゼンテーションに際しての表示の再現性を高めている。

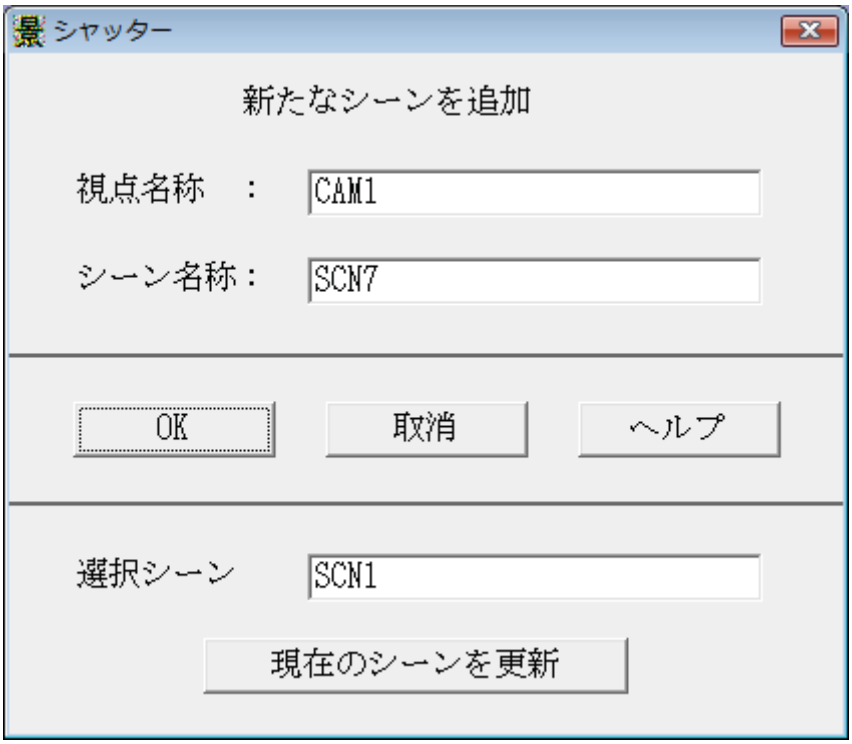

図４－５１：シャッター画面

リソース：IDD_DLG_SUTTER　ハンドラ：CShutterDlg(shutterd.cpp)

ヘルプ：shutterd.txt

（４７）LSS-G ファイルの最適化保存

地形データなど、規則的なパターンを繰り返すデータは、コンパクトに保存することが可能であり、コンバータの中で様々な工夫が可能であるが、そのようなファイルを読み込んだ上で加工を行うと、データの規則性は失われてしまう。しかし、加工されなかった周辺の大部分の形状は、元の地形の性質を保っている。そこで最適化保存により、可能な限りコンパクトな形でデータを保存する。

共通のカラー、テクスチャ、マテリアルに関しては、それぞれのグループ、面の出力に際して新たな記述を起こすのではなく、ファイル書き出しに先立って、保存しようとする全体に関してリストアップ、ソーティングを行い、共通の定義を冒頭で行った上で、個々のグループや面の書き出しでこれらを参照する形式とすることで、重複を避け、コンパクトに記述することができる。

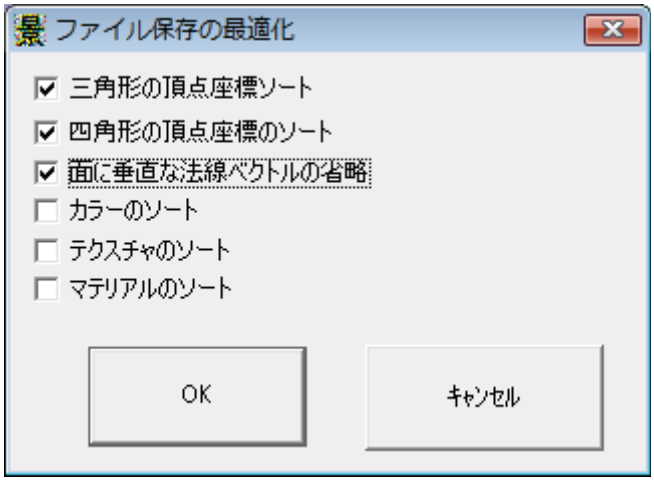

図４－５２：最適化保存設定画面

リソース：IDD_DLG_SAVE_OPTIM　ハンドラ：COptim (optim.cpp)

（４８）画像視点抽出

初期のバージョンから提供している、背景写真と三次元モデルの位置合わせを支援する機能である。対象区域に座標系を定義した上で、映っている対象物から座標のわかるポイントを指定した上で、それぞれのポイントを画面クリック選択し、対応する三次元座標を入力する。作成された画像ピクセル座標と実空間の三次元座標のセットに関して、誤差を最小とする視点位置を反復計算し、画像を撮影したカメラ情報（視点・注視点・焦点距離・傾きなど）を計算する。

計算の結果得られたカメラ情報を、モデルの表示に適用することにより、正しい位置に表示が行われる。

図４－５３：画像視点抽出画面

① メニュー：IDR_MENU_EXTRACT

メニュー構成： ［ファイル］ ＋［閉じる］終了する ［ヘルプ(H)］ヘルプを表示する

② OpenGL 画面：COrthoVW m_orthoView (orthovw.cpp)

③ 右側ダイアログ

リソース：IDD_DLG_PRM_EXTRACT ハンドラ：CShitenWnd (shitenwn.cpp)

④ ヘルプ：shitenwn.txt

（４９）シーン選択

シーン編集中に、現在存在するシーンの一覧を表示し、選択したシーンに移動する。

なお、画面下の矢印ボタンで、一つ前のシーン、次のシーンに移動することができるため、主に離れたシーンにジャンプする場合に使用する他、編集のために現在までに作成されているシーンを確認するために有用である。

図４－５４：シーン選択画面

リソース：IDD_DLD_SCENELIST　ハンドラ：CSceneList (scenelst.cpp)

（５０）ユーザー定義によるパラメトリック部品の選択

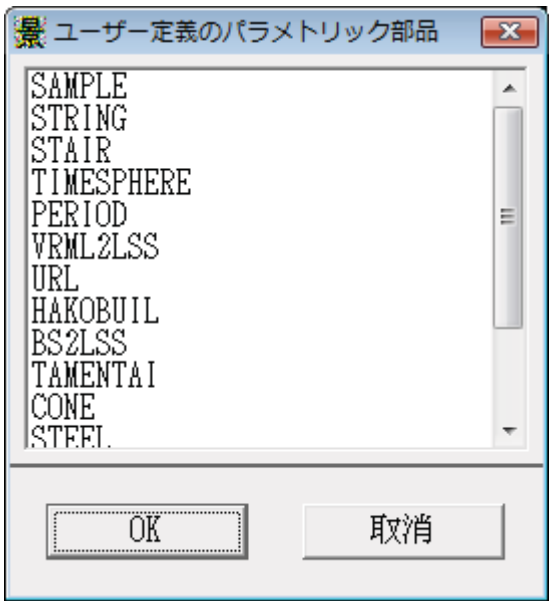

図４－５５：パラメトリック部品選択画面

リソース：IDD_DLD_SCENELIST　ハンドラ：CSceneList (scenelst.cpp)

（５１）面情報

選択したグループの配下にある面に関する情報を表示する。

４－５６：面情報表示画面

リソース：IDD_DLG_FACE　ハンドラ：CFaceinfo (faceinfo.cpp)

（５２）ソリッド分析

一つのグループに属する面群が、閉多面体を形成しているかどうかを検査し、該当する場合には、それによって規定されるソリッドの諸元を計算して表示する。

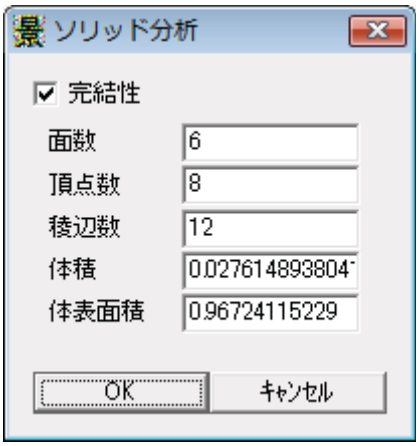

図４－５７：ソリッド分析画面

リソース：IDD_DLG_SOLIDANAL　ハンドラ：CSolidanal (solidanal.cpp)

（５３）単面分析

一つの面を対象として、正常な面であるかどうかを判定する。穴あきポリゴンとなっている場合や、ブリッジで離れた島（飛び地）が連結している場合、全ての頂点が（誤差の範囲で）１平面上にあるかどうか、自己交差しながら、多重ループを形成していないか等についての検査を行う。

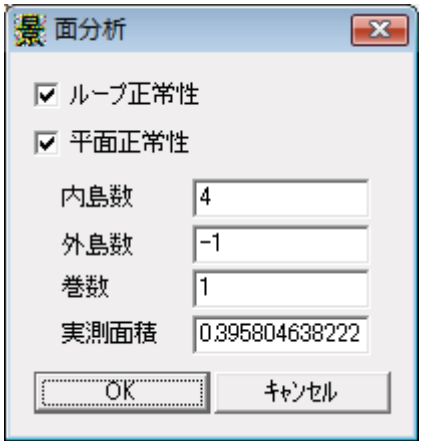

図４－５８：単面分析画面

リソース：IDD_FACEANAL　ハンドラ：CFaceanal (faceanal.cpp)

（５４）頂点検査

一つの面を構成する頂点について逐次的に表示・編集する。頂点の座標のみならず、VERTEX として定義されているテクスチャ座標、法線ベクトル、カラーについて表示し、必要であれば変更する。

図４－５９：頂点検査画面

リソース：IDD_DLG_VANAL　ハンドラ：CVanal (vanal.cpp)

（５５）住宅情報

オブジェクトの属性として、住宅情報が登録されている場合に、表示を行う。通常は、住戸･住宅や敷地の形状を記述する外部ファイルとして配置された FILE コマンドにより記述されたグループに対して属性を定義し、情報そのものは、各戸を記述する外部ファイルではなく、これらを集合的に束ねる上位のグループ（例えば街区）の側に記録する。

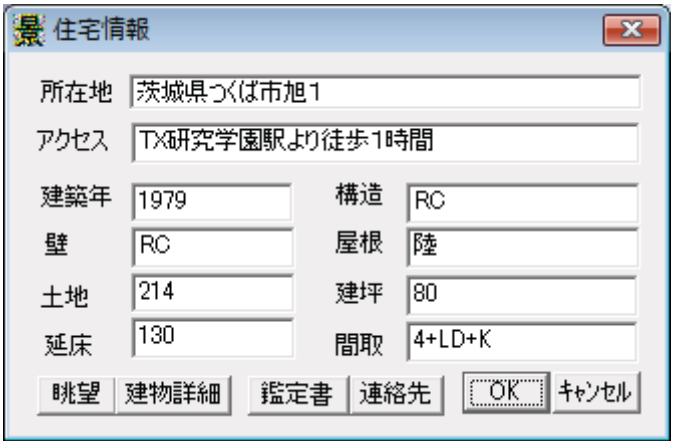

図４－６０：住宅情報画面

リソース：IDD_DIALOG1　ハンドラ：CHouse (house.cpp)

（５６）効果グループ

各シーンに対して一つ定義される効果グループを表示・編集する。効果グループは、複数の効果ユニットの組み合わせとして定義する。

このダイアログにおいては、編集対象のシーンに適用する、効果グループの「新規作成」や、別のシーンのために既に定義されている別の効果グループの再利用を指定するための「選択変更」を行う。更に、現在指定されている効果グループを構成する効果ユニットに関して、その内部構成を編集するダイアログを起動したり、定義済みの効果ユニットを効果グループに追加したり、効果グループを構成する効果ユニットを登録解除・削除する操作を行うことができる。

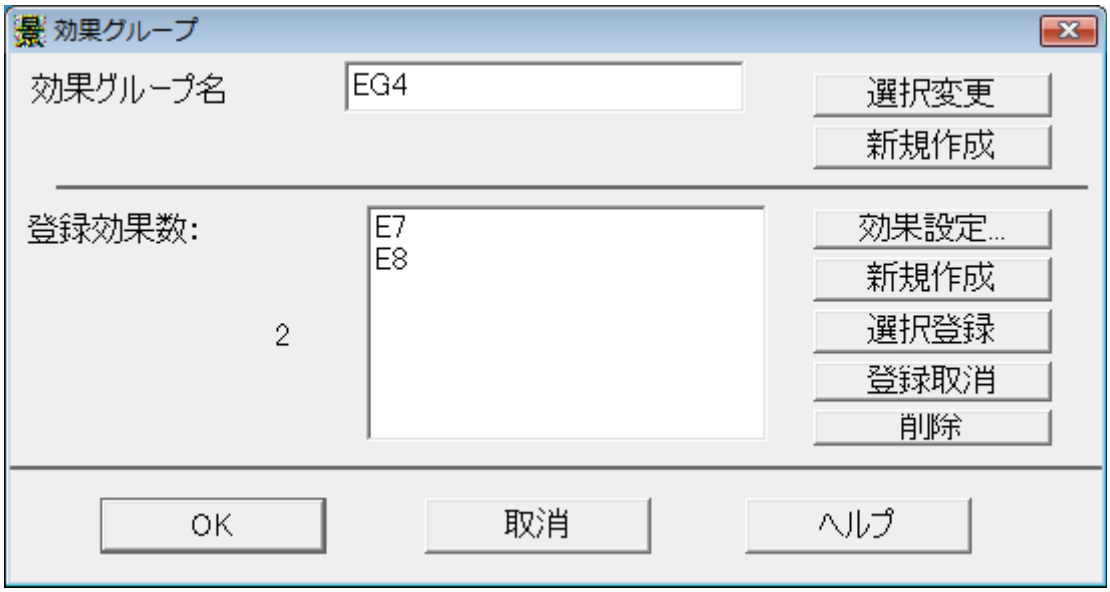

図４－６１：効果グループ編集画面

リソース：IDD_DLG_EFFECT　ハンドラ：CEditEffect (effect.cpp)

（５７）効果ユニット

効果グループを構成する効果ユニットを編集する。多くの場合、効果は表示に関する各種の設定を記録・再現するために用いており、各ダイアログにおける設定を行うことで自動的に設定されるため、ユーザーはその存在を意識する必要はない。このダイアログで表示することにより、具体的にどのようなキーワードとパラメータにより効果ユニットが定義されているかを確認することができる。

システム基幹部分で定義・実装されていない新たなキーワードを定義した場合、表示には何も影響しないが、LSS-S 形式のファイルには保存される。プラグイン DLL などの開発にあたり、シーン表示機能の拡張などを行う際に活用することが可能である。

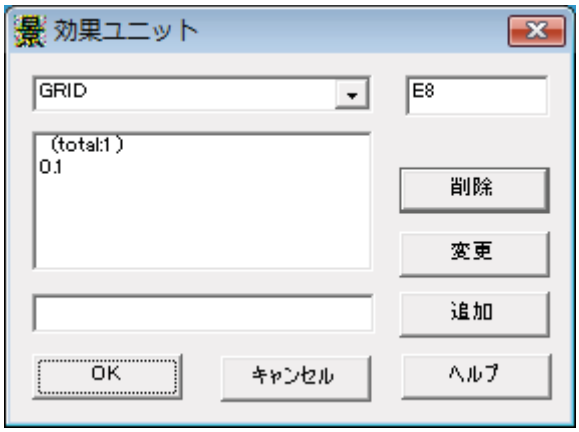

図４－６２：効果ユニット編集画面

リソース：IDD_DLG_EFFECTUNIT ハンドラ：CEffectUnit (effectunit.cpp)

（５８）環境設定

システム起動時に、環境編集 KSIM_ENV で指定した環境設定ファイル（デフォルト名：kdbms.set）で定義された各種の環境設定項目を表示・編集する。編集結果を保存することにより、環境設定ファイルの内容を更新する。このダイアログでファイル保存操作を実行しなかった場合には、元の環境設定ファイルはそのまま維持されるため、次に起動した時点では編集結果は反映されない。

環境設定ファイルの読み込みに際しては、Ver.2.05 以前のバージョンでの定義方法を読み替える処理が行われているため、何も編集せずに上書き保存すると、現在のバージョンにおける定義方法により出力される。従って、本ダイアログから保存した環境設定ファイルを、2.05 以前のバージョンの景観シミュレータ、景観データベースで使用する場合（殆ど無いケースと思われる）、動作は保障されない。

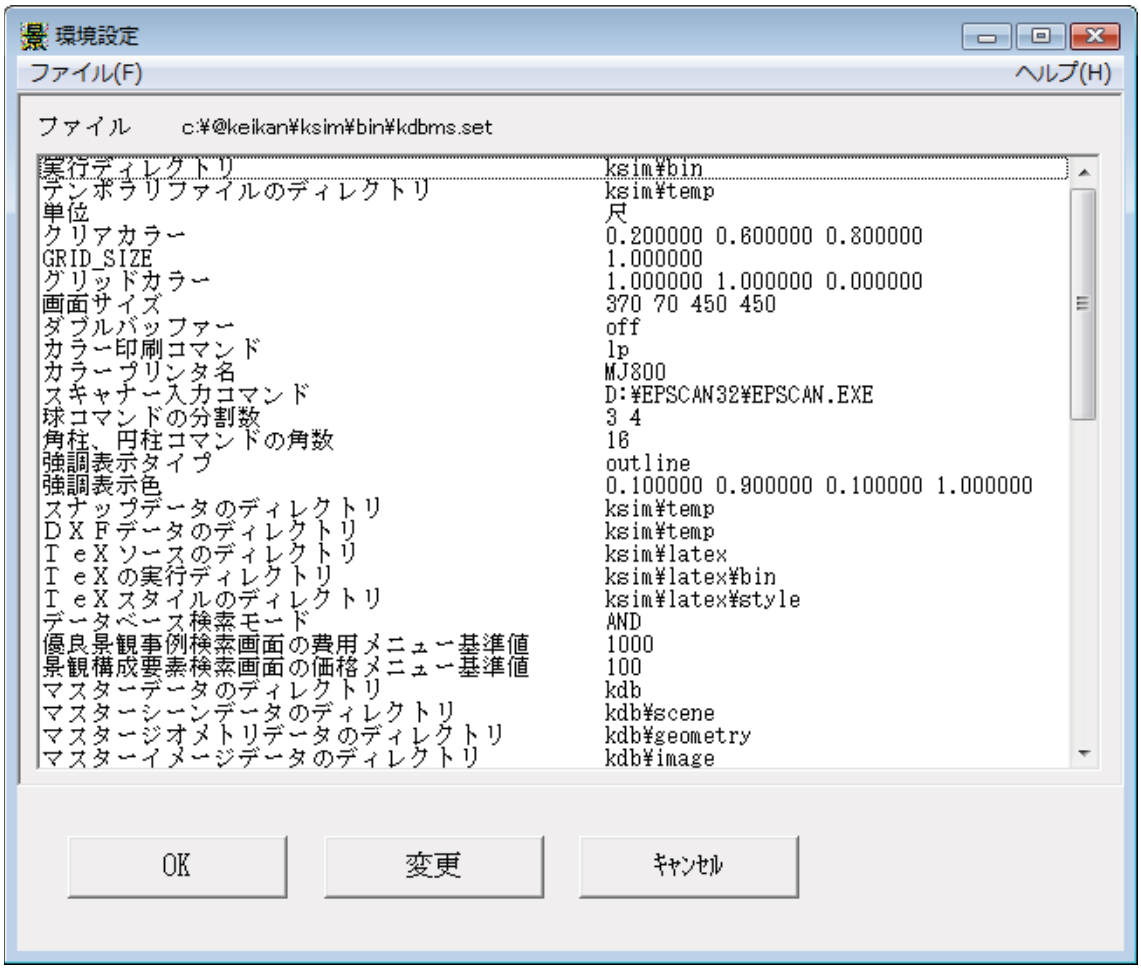

図４－６３：環境設定画面

①メニュー

メニュー構成：IDR_MENU_ENV

［ファイル］

＋［環境設定ファイルを開く］初期化に用いた現在のファイルとは別の環境設定ファイルをロードする。

＋［環境設定ファイルに名前を付けて保存］ファイル名を指定して、保存する

＋［環境設定ファイルの上書保存］読み込んだ名前、または以前保存した名前で保存する

＋［環境編集終了］このダイアログを終了する

②ダイアログ

上部(リスト)　リソース：IDD_MAIN_FORM ハンドラ：CEnvView (envview.cpp)

下部(３ボタン) リソース：IDD_BTN_FORM ハンドラ：CEnvEdit(envedit.cpp)

（５９）環境設定：文字列型

環境設定ダイアログのリストボックスから、一つの設定項目を選択して変更ボタンを押すと、設定内容が文字列型であった場合に起動する。

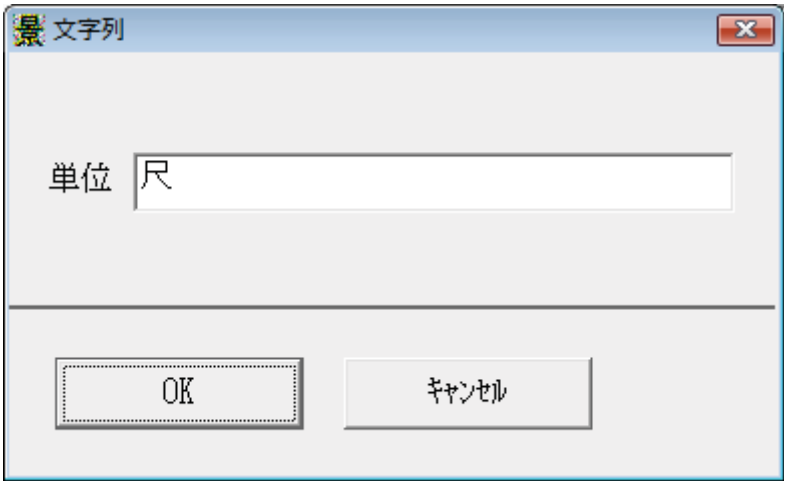

図４－６４：文字列型環境設定項目入力画面

リソース：IDD_DLG_STRING ハンドラ：CStrDlg (strdlg.cpp)

（６０）環境設定：選択型

環境設定ダイアログのリストボックスから、一つの設定項目を選択して変更ボタンを押すと、設定内容が選択型であった場合に起動する。選択肢は、コンボボックスのプルダウンとして表示する。

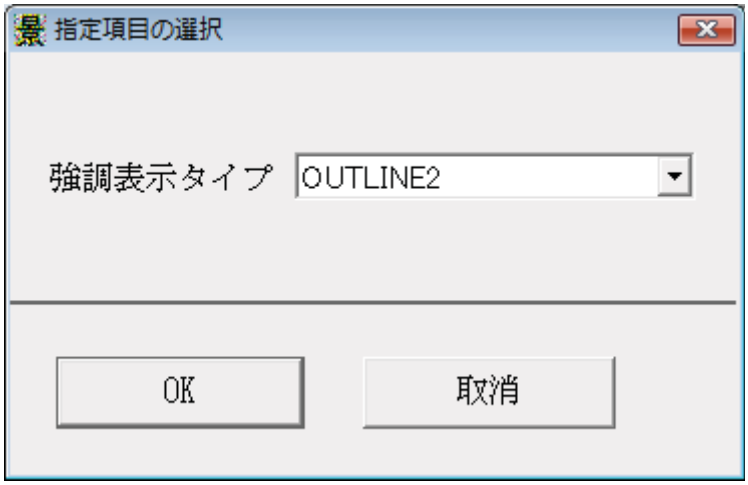

図４－６５：選択型環境設定項目入力画面

リソース：IDD_DLG_SELECT ハンドラ：CSelDlg (seldlg.cpp)

（６１）環境設定：数値・範囲型

環境設定ダイアログのリストボックスから、一つの設定項目を選択して変更ボタンを押すと、設定内容が数値・範囲型であった場合に起動する。数値や範囲の指定方法（項目数）が環境変数によって異なるため、その差異に対応して、必要な数だけ、数値入力のためのエディットボックスを表示する。

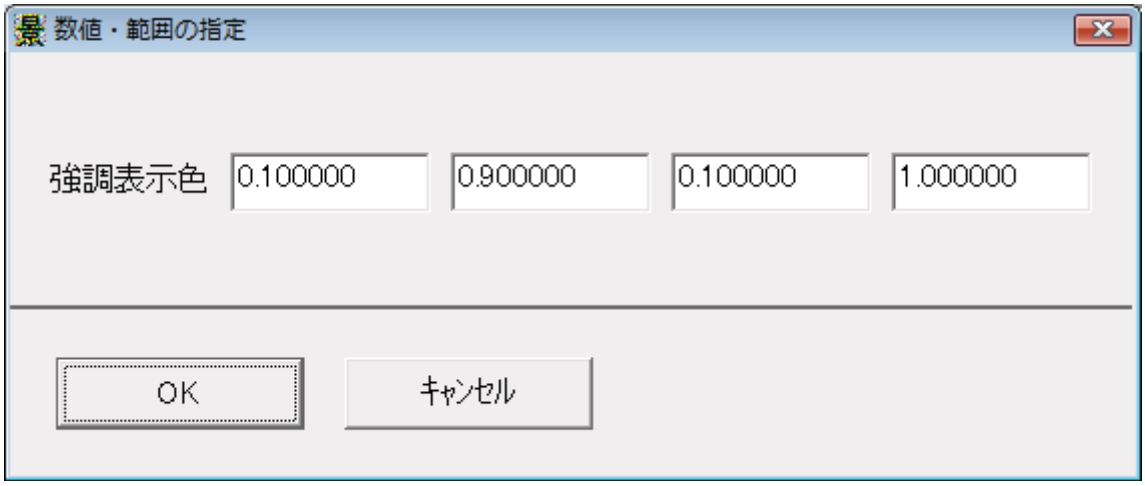

図４－６６：数値・範囲型環境設定項目入力画面

リソース：IDD_DLG_RANG ハンドラ：CRangDlg (rangdlg.cpp)

（６２）線分の選択

現在編集中の地物の中に存在する全ての線データを一覧表示し、その中から特定の線を選択する。また、このダイアログの中で線を削除したり、選択した線と同色の線を全て削除する。線の編集ダイアログ(38)から起動する。

図４－６７：線分の選択画面

リソース：IDD_DLG_LINELIST　ハンドラ：CLineList (linelist.cpp)

（６２）プリンターの選択

開発環境が提供している CView クラスに標準で装備されているメニュー項目を、そのまま利用している。

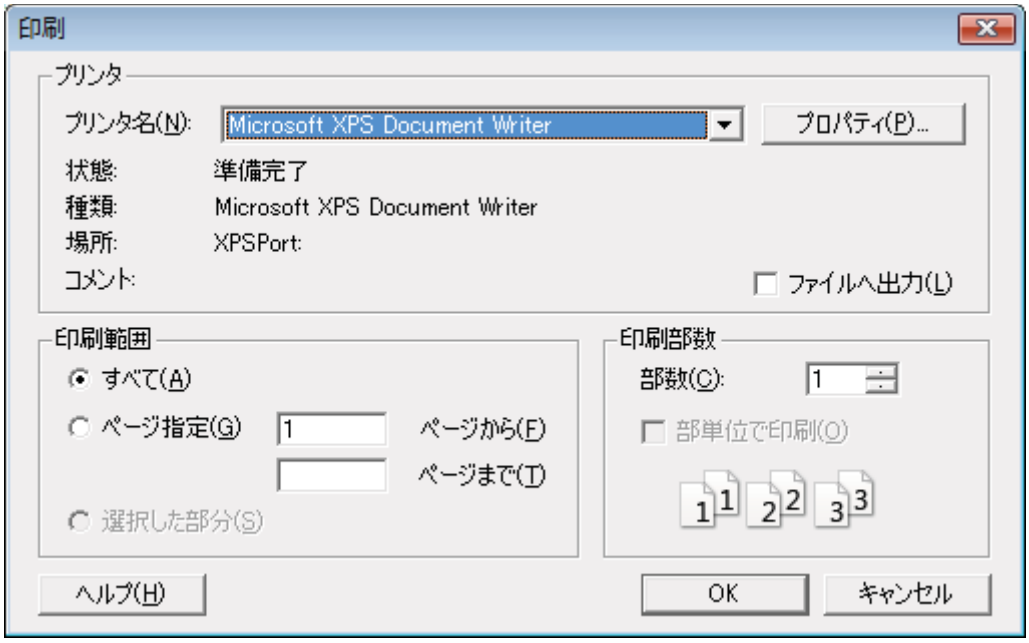

図４－６８：プリンターの選択画面

ハンドラ：CPrintDialog　(MFC の組み込みクラス, dlgprnt.cpp)

（６４）質量

オブジェクトの物理属性の一実装例として用意している。オブジェクトを選択した上で、ポップアップメニューからこのダイアログを開くと、現在選択しているグループに属性として定義されている質量、及び子グループとしてリンクしている配下のグループの合計質量を表示する。下のコンボボックスで、選択階層をルートから最末端の最初に選択したグループまで変更することができる。ルートを選択すると、世界全体の質量となる。

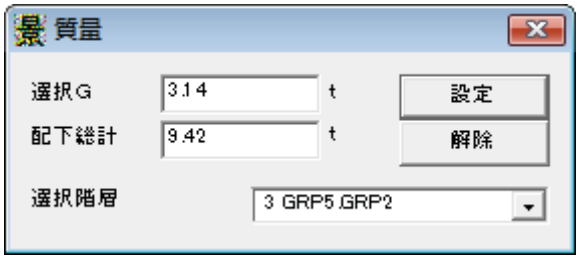

図４－６９：質量画面

リソース：IDD_PHISI　ハンドラ：CPhisi (phisi.cpp)

（６５）炭素含量

オブジェクトの化学属性の一実装例として用意している。オブジェクトを選択した上で、ポップアップメニューからこのダイアログを開くと、現在選択しているグループに属性として定義されている炭素含量、及び子グループとしてリンクしている配下のグループの合計炭素含量を表示する。下のコンボボックスで、選択階層をルートから最末端の最初に選

択したグループまで変更することができる。ルートを選択すると、世界全体の炭素含量となる。

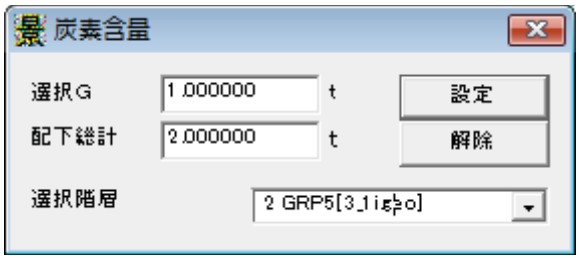

図４－７０：炭素含量画面

リソース：IDD_CHEMI　ハンドラ：CChemi(chemi.cpp)

（６６）データベース情報

景観データベースから選択され、配置されたオブジェクトに関して、データベースに関する情報にアクセスする。

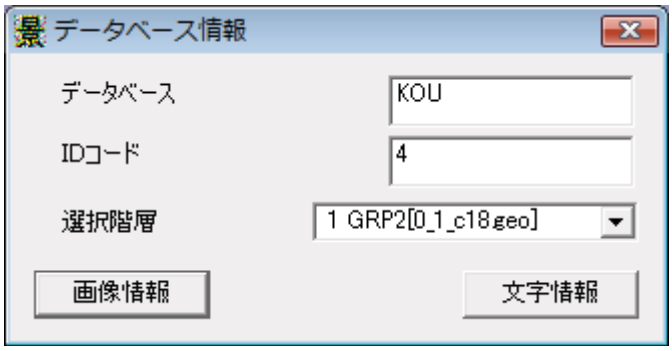

図４－７１：データベース情報画面

リソース：IDD_DBINFO　ハンドラ：CInfoDB (infodb.cpp)

（６７）高速表示設定

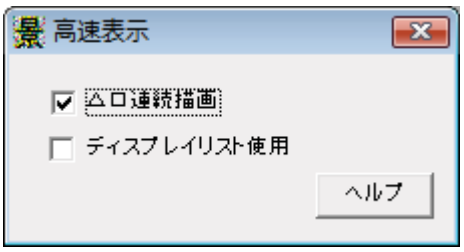

図４－７２：高速表示設定

リソース：IDD_DLG_HIGHSPEED ハンドラ：CHiSpe(HiSpe.cpp)