インタラクティブ３Ｄアプリケーション開発ツールを用いた３Ｄゲーム教材の作成

Development of an Educational Material for Making 3D-Games using an Interactive 3D Authoring Tool

玉真 昭男*
Teruo TAMAMA

Abstract: Using an interactive 3D authoring tool, named "Virtools 4.1," a 3D-racing game has been developed, in which an amphibious vehicle speeds up and down over a mountainous region with an undulating surface. Since this includes various elements necessary for developing a high-level 3D racing game, it is a good educational material for making 3D games. With this, students can develop their own 3D games easily by revising it or adding new functions.

１．はじめに

過去７年間に、卒業研究や修士論文の課題として約30個の３Ｄコンピュータアクションゲームを開発してきた。種類もレーシングゲーム、シューティングゲーム、格闘ゲーム、ロールプレイゲーム、落ちモノゲームなどいろいろである。大学祭やオープンキャンパスなどのイベントで、見物客に十分に楽しんでもらえるレベルで、しかも安定して動作するゲームも10個以上存在する[1, 2]。

ゲーム作りは、出来栄えを自分で評価できる、何か１つ作るとアイデアが次々に湧いてもっと作りたくなる、更に高度な機能を作りこみたくなる、といった自己拡張性があり、完成したときの達成感も大きいので、アルゴリズム考案やプログラミングといった「知的もの造り」教育の課題として非常に優れている。

また、将来、プログラマーやＳＥ（システムエンジニア）を目指す学生には、プログラミング言語の文法を理解し、多くの演習問題を解くだけでは不十分で、卒業研究などで例えば3000行以上の大規模プログラミング開発の体験が必要である。その課題として、出来栄えを自分で評価でき、アイデアが次々に湧いてもっと作りたくなる「ゲーム」は格好の題材である。

大学でゲームを開発するもう一つの意義は、大学祭、オープンキャンパスなどのイベントで、展示の目玉として宣伝に使えることである。子供、小中高校生、大学生から社会人まで、長く居座ってゲームを楽しむ姿が見受けられ、非常に人気のある会場の一つとなる。特に高校生は、大学進学後自分でもゲームを作ってみたいと思うからであろうが、全てのゲームを試した上で、どうやって作るのか、作成にはどれ位時間が掛かるのかなど、熱心に質問する人が多い。

このような背景から、Windows用のC/C++コンパイラであるVisual C++.NETと３Ｄゲーム開発用ライブラリDirectXを駆使して、３Ｄゲーム作りを行ってきた。レーシングゲームに於いては、物理効果の導入、フォースフィードバックの掛かる操縦席とのドッキングにより、Ｆ１カーの加速性能を体験できるレーシングシミュレータを開発することが出来た[3]。また今年は、運転再現モードやコンピュータ対戦モードを追加して、運転の評価・分析が行えるシステムに拡張した。これを「全日本学生フォーミュラ大会」に向けたドライバーの運転練習に活用し、総合成績アップにつながる成果が得られた[4]。

しかし、このレーシングゲームに関しても、２人の院生が４年次から３年ずつ、計６年間掛け、我が研究室の傑作を作ってくれたが、平坦なコースを自由に動き回れるレベルでしかない、ということも言える。Visual C++とDirectXの組み合わせでは、依然として起伏のある山野を車が駆け巡るゲームは出来ていない。物理効果も、フェンスや道路上に置いたコーンと車との衝突による跳ね返り動作の表

2009年３月13日受理

* 総合情報学部　コンピュータシステム学科

現に留まっている。

最近の市販ゲームの特徴は、まず登場するキャラクタの表情がリアルで数が多いことである。実写の映画と見まがうほどの、或いはそれを超えたスーパーリアルな映像が現れる。また、周りの景観も壮大で、スケールの大きな仮想空間の中で圧倒的な迫力の活劇が繰り広げられる。これは経験豊富な数十人規模のプロフェッショナル集団がビジネスレベルで、しかも数年掛かりで創る作品だからであり、大学で、しかも数人規模では太刀打ちできる話ではない。

今回、規模は小さく、登場するキャラクタも少ないが、レーシングゲームの本質と考える要素は全て含んだサンプルゲームを作成し、学生用教材とすることを目的とした。ゲームスタート時に最初に現れる背景が壮大だとユーザーに期待を抱かせる効果がある。そこで、今回はゲーム画面背景の景観にもこだわった。

２．使用したソフトウェア

開発環境としてはDassault Systems社のインタラクティブ３Ｄアプリケーション開発ツール“Virtools 4.1”を用いた。Virtoolsは450個以上もあるビルディングブロック（ＢＢ）と呼ばれるモジュールを組み合わせることで３Ｄシーンを作っていく。ＢＢには、キャラクタ、車、背景、大道具、小道具などの３Ｄオブジェクトだけでなく、テクスチャ（表面材質）、キーやマウス操作の設定、アニメーションやサウンド設定、などの種類がある。これらをSchematicと呼ばれる画面上でつないで行くだけでインタラクティブな３Ｄアプリケーションの生成が行われる。

モデリングには３ＤＣＧモデラー“Metasequoia”[5]を使用した。これにはシェアウェア版とフリーウェア版があり、現在Ver. 2.4.9とR2.4がそれぞれの最新バージョンである。

３．背景と車のモデリング

３．１　背景のモデリング

シェアウェア版のMetasequoiaには凹凸地形の作成機能があるので、背景の作成に使った。地形全体のサイズと縦横の分割数を指定する。今回、サイズは200×200×100、分割数は縦50×横50を用いた。全体は草地の模様にしたが、高さで山岳部を分離し、赤茶けた土色の画像を貼った。山や谷の間を縫う道路を配置し、コンクリート色に塗った。完成図をFig. 1に示す。また、四方についたてを立て、ここに空の写真画像を貼り付けて遠方に青空が見える設定とした。Fig. 1では、そのうちの2方向、前方と左側のみ示してある。

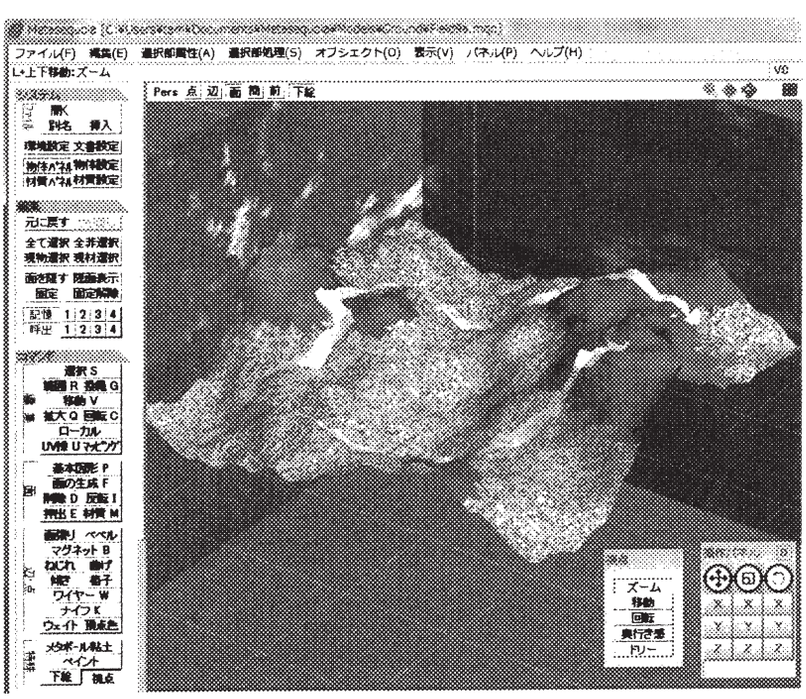

Fig. 1 背景の完成図（Metasequoia画面）

3．2　車のモデリング

Fig. 2 に、同じく Metasequoia で作った車（クラシックカー）の完成図を示す。

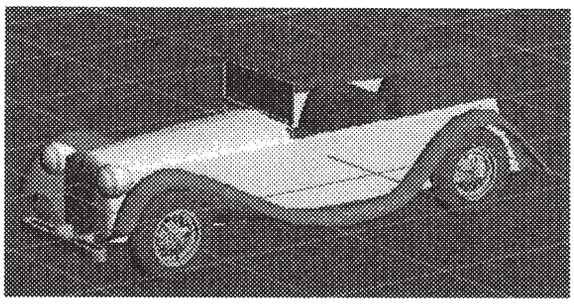

Fig. 2 クラシックカー（Metasequoia 画面）

Virtools に CG で作った「車」を取り込むには次の２点を守るのがポイントである。

（1）　車体とタイヤを個別に取り込み、しかもそれぞれのローカル座標を Fig. 3 のようにそれぞれの中心に設定しておく。

（2）　タイヤに_FR（前輪右側）、_FL（前輪左側）、_BR（後輪右側）、_BL（後輪左側）の名前を付ける。
例：Tire_FR、 Tire_FL、 Tire_BR、 Tire_BL

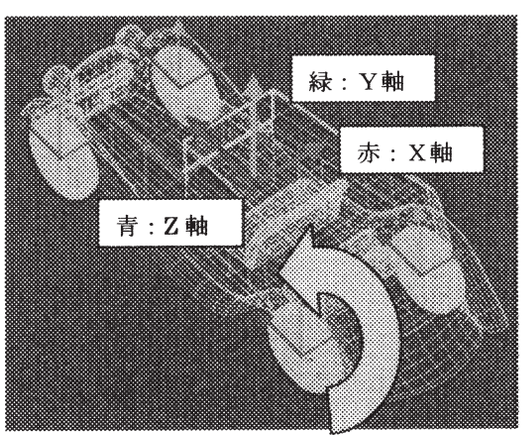

Fig. 3　ローカル座標系の設定(DirectX:左手座標系)

Virtools は DirectX のオブジェクトを使って動く。DirectX の場合、左手座標系になっていることにまず注意する必要がある。次に、各ローカル座標系を Fig. 3 のように個別に設定しておけば、タイヤはそれぞれの座標軸原点、この場合は車軸の回りで正常に回る。しかし、タイヤも含めて車全体を一体化して作り、そのまま Virtools に持ち込むと、タイヤの原点も車体の原点と同一になる。こうすると、タイヤ全体が車体の原点を中心にして回ってしまう。タイヤが車軸の回りを回るのでなく、車体の中心の回りを回ってはおかしな動きになるので、注意しなくてはならない。

4．Virtools への取り込み

4．1　車オブジェクトの作成

Metasequoia で作った車のモデルを Virtools に取り込むにはＸファイル形式を使う。車のモデルファイルをタイヤとそれ以外の本体の５個に分け、Ｘファイル形式で保存する。Fig. 4 はタイヤを除いた本体のオブジェクトである。

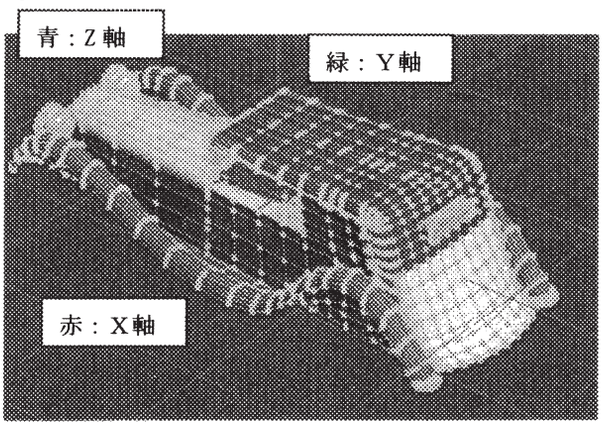

Fig. 4　車体部の Metasequoia 画面（車体の中心をローカル座標系の中心に合わせる）
～座標系は右手座標系になっていることに注意～

Fig. 5 は Metasequoia でＸファイル形式でモデルを保存する時の設定用ダイアログである。両者で座標系が異なる（右手座標系と左手座標系）ので、ファイル保存する時「Z軸反転」をクリックすることが重要である。

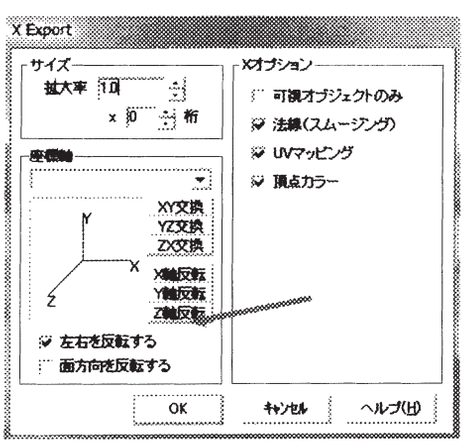

Fig. 5　Metasequoia でＸファイル形式(Direct3D Retained Mode)で保存する時の設定

ホイール付きのタイヤで左右の形が異なる場合は、左側と右側１個ずつのＸファイルを作る必要がある。本モデルではタイヤは左右対称にしたので１個だけで良い。

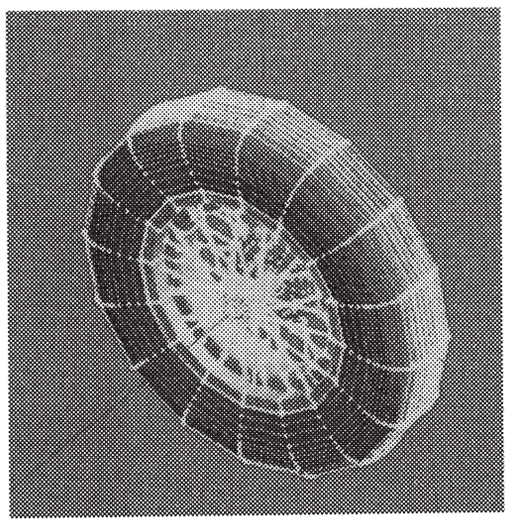

Fig. 6　タイヤを原点に置いてＸファイルを作成

４．２　Virtools での車オブジェクトの作成

まず、車本体のＸファイルを Virtools に読み込む。次にタイヤのＸファイルを４回読み込み、本来の位置に移動させる。このようにすると、各タイヤ・オブジェクトのローカル座標系はそれぞれの中心にあるまま車体に取り付けられるので Fig. 3 の条件が満たされる。このようにして Virtools 上で完成させたのが下図の車オブジェクトである。作成した車オブジェクトは 3dXML 形式で保存する。3dXML 形式とは３Ｄ・ＣＡＤ分野で標準となっている３Ｄモデルの保存形式である。

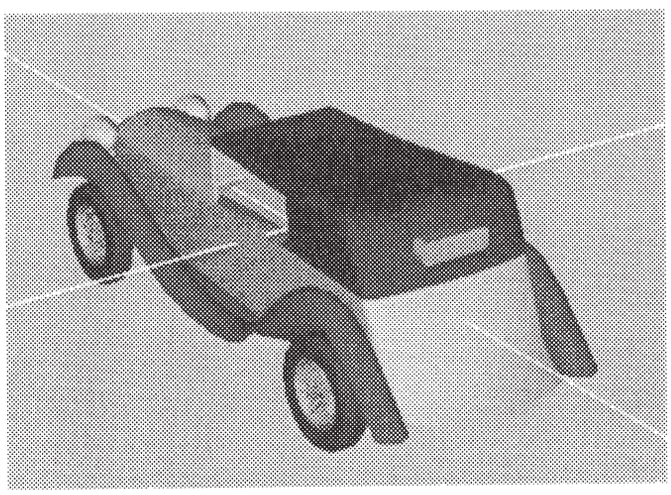

Fig. 7　Virtools 上で完成した車オブジェク

４．３　３Ｄシーンの作成

Virtools の新規画面に、やはりＸファイル形式で保存した Fig. 1 の背景モデルを取り込む。その上に 3dXML 形式で保存した車のオブジェクトを取り込むと全体のシーンが出来上がる。(後述 Fig. 15 参照)

５．車を走らせる

５．１　グラフィック言語によるプログラミング

Virtools では、オブジェクトの動作や振舞(Behavior)を表すプログラミングはグラフィック言語を用いて行う。Ｃ言語やＢＡＳＩＣなどは文章でプログラムを書くプログラミング言語であるが、グラフィック言語は、計測分野に於ける LabView のように、機能を表すアイコンを図面の上に並べ、線でつないで前後関係を表すことでプログラミングを行うものである。簡単で直感的なプログラミングが出来る。この、機能アイコンを並べるための図面を Schematic と呼んでいる。

機能を表すアイコンはビルディングブロック(ＢＢ)に対応する。Schematic 画面上で、ＢＢを線でつないだ図をスクリプトと呼ぶ。ＢＢは下図に示すように、基本的に長方形で表される。左右には振舞(Behavior)を制御する入出力端子 bIn、 bOut、 上下には設定するパラメータを指定する入出力端子 pIn、 pOut を表す三角形が付いている。

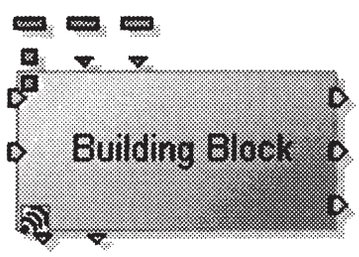

Fig. 8　ビルディングブロック(BB)の標準形

５．２　車の操作と振舞のプログラミング

“Switch ON Key”と“Physics Car”の２つのＢＢをFig. 9 のように結ぶことで車の操作と振舞を定義することが出来る。“Switch On Key”は車の前進や後退などの振舞にキーボードのどのキーを割り当てるかを指定するＢＢで、その設定は次のようにして行う。Schematic 上で“Switch On Key” をダブルクリックすると、Fig. 10 のダイアログが立ち上がる。Key0 横の箱をクリックして上向き矢印ボタンを押すと”Up”が入る。同様にして、各方向キーと、スペース、右側シフトキーを割り当てる。これでＰＣの方向キーを押した時、車が↑前進、↓後退、←左旋回、→右旋回などの動作をするように設定できる。スペースキーはブレーキ、右シフトキーは急加速に対応する。

次に、”Physics Car”の設定を行う。これは車に「物理法則に則って振る舞う」ように指定するＢＢである。Schematic 上で”Physics Car”を右クリックし、Edit Parameters を選ぶ。Fig. 11 のダイアログが現れるので、それぞれ右端にある下向き矢印をクリックしてパラメータを選び、図のように設定する。ここで、BODY Parameters、ENGINE - STERRING Parameters など３つのパラメータファイルは車体やエンジンに関する各種パラメータを表形式で定義したものである。車オブジェクトにこの“Physics Car BB”の設定をすると、起伏のある地面を感知し、起伏に沿って車が走るようになる。また、指定した各種物理パラメータに従って、エンジンのパワー、回転数、最高速度、タイヤの摩擦係数を持つ「物理カー」となる。

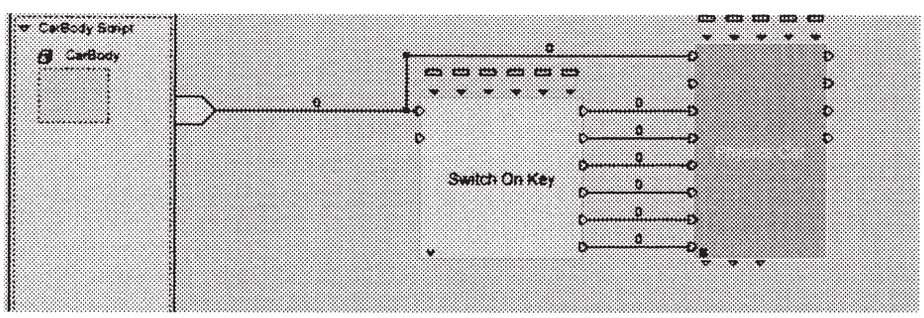

Fig. 9　車の操作と振舞を定義するスクリプト

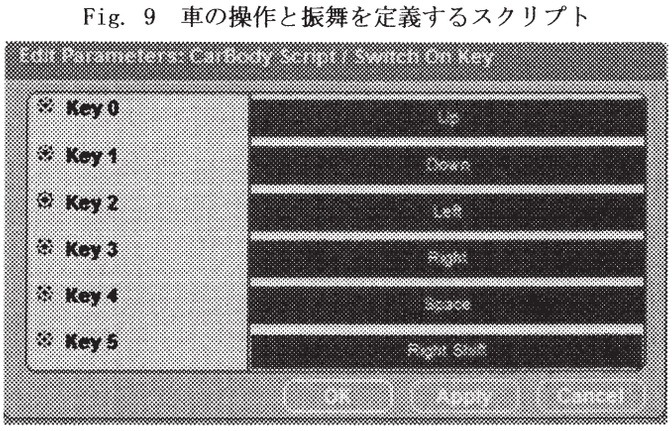

Fig. 10 “Switch On Key”のパラメータ設定用ダイアログ

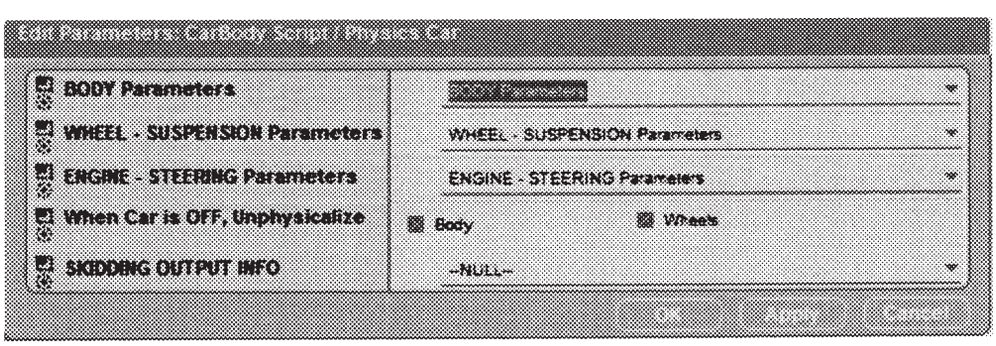

Fig. 11 “Physics Car”の設定用ダイアログ

５．３　物理効果の対象としての地面の設定

車の物理効果の対象となるのは、地面や障害物である。Fig. 12 のように “Physicalize BB” を使って地面に物理設定を与えると、その起伏を感知しながら車が動くようになる。起伏に沿って走るだけでなく、急勾配では車が倒れたりもする。

５．４　ブレーキライトの設定

シーンにインタラクティブなライトを設置する例を示すため車にブレーキライトを付けた。ブレーキを踏んだ時このライトが点くようにするには、５．２節でブレーキにはスペースキーを対応させている（Fig. 10）ので、スペースキーを押した時だけ光るようにすれば良い。そのスクリプトはFig. 13 のようになる。“Key Event BB” でスペースキーが押された時を定義し、“Set Light Range BB” でライト光の届く範囲を指定する。上側の “Set Light Range BB” でその値を 20、下側では 0 に設定すると、スペースキーが押された時だけライトが光るようになる。

５．５　池の配置

背景とした台地(Fig. 1)の谷の部分に一枚の板を水平にかぶせ、テクスチャとして水面の画像（jpeg ファイル）を貼り付ける。このテクスチャにFig. 14 のスクリプトを設定する。図の “Texture Sine BB” は水面(Water)の材質を sine 曲線に従って揺する効果を与える BB である。これにより、水面が風で波立つ様子を表現することが出来る。また、池には “Physicalize BB” を設定しない。すると、車はこの池の中を潜って進むことが出来、水陸両用車となる。

６．シーン全体の完成

完成したシーン全体図を Fig. 15 に示す。右奥に山脈、左側に低い山、中央に表面が風で揺れる池、山間部を貫く道路、その手前にクラシックカーが見える。この作品（CMO ファイル）を「実行」すると、起伏のある山間地帯を水陸両用車が駆け抜けるゲームとなる。

７．　まとめ

インタラクティブ３Ｄアプリケーション開発ツール” Virtools 4.1” を用いて「水陸両用車山岳地帯疾走ゲーム」を開発した。これは、（１）起伏のある地形モデルの生成法、（２）池などの特殊な環境の追加法、（３）自作した車を物理カーにする方法、（４）バックライトなど特殊装備の組み込み法など、高度なレーシングゲームの制作に必要な要素を多く含んだゲーム教材となっている。

学生が、この教材でレーシングゲームの作り方の基本をマスターした上で、これをベースにいろいろな改良を施し、また新しい機能を付け加えていくことで、更に高度で多様なオリジナルレーシングゲームを容易に開発することが出来るようになると期待している。

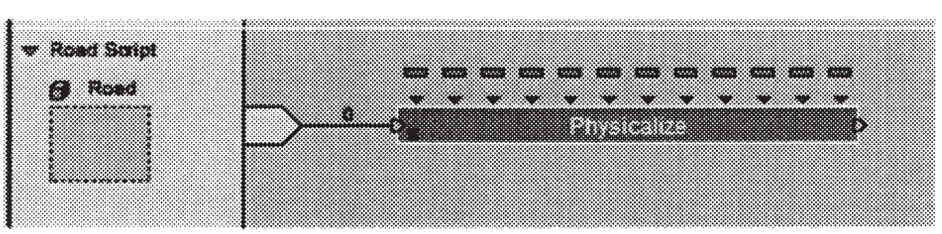

Fig. 12　地面のスクリプト

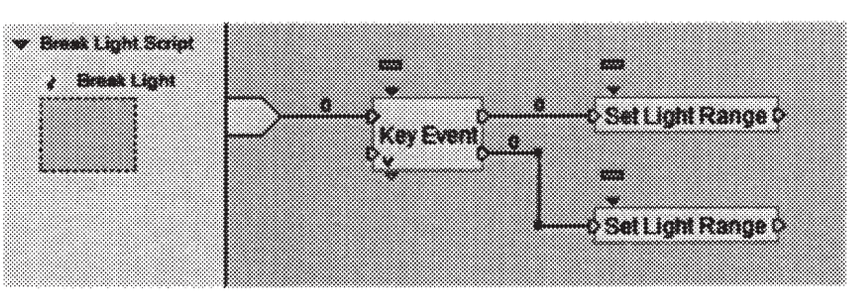

Fig. 13　ブレーキライト用スクリプト

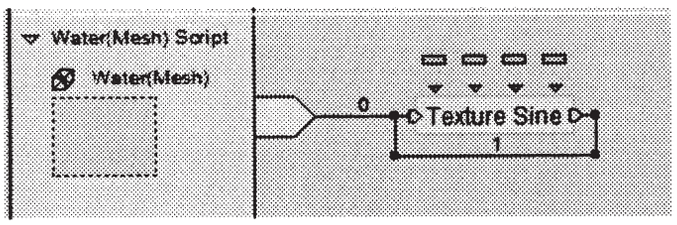

Fig. 14　池のスクリプト

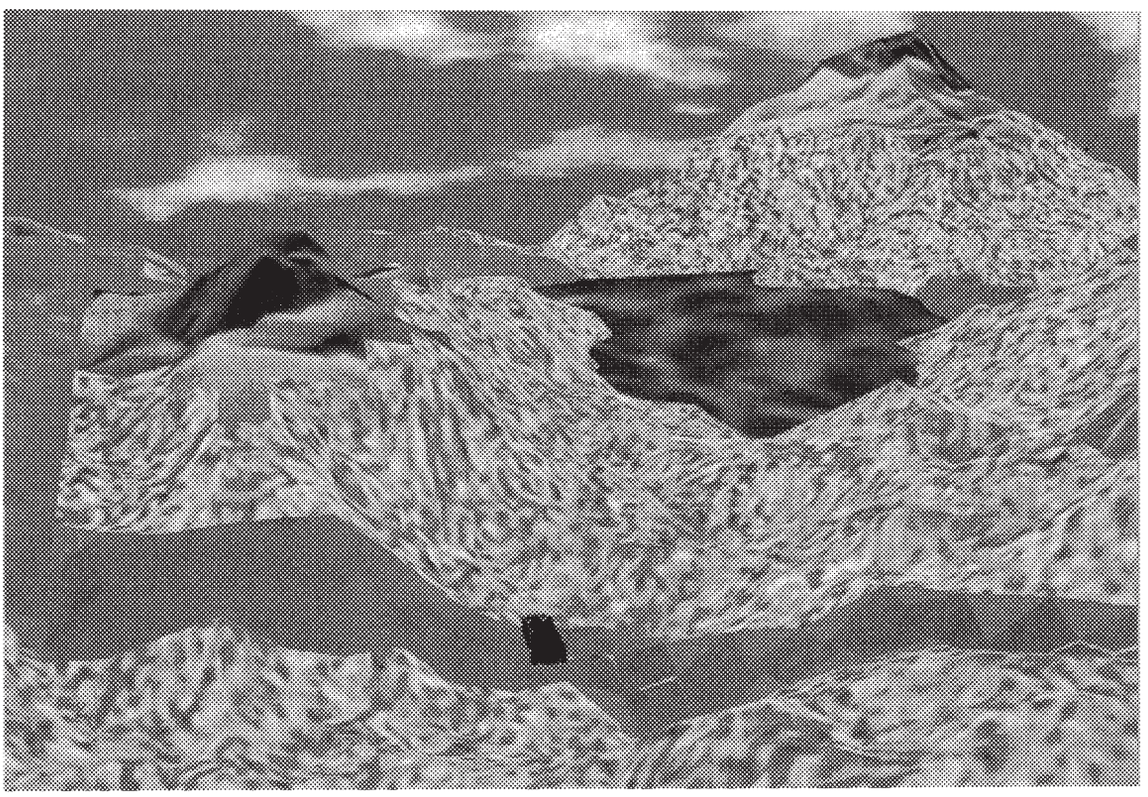

Fig. 15　完成したゲームの全体画面

参考文献

1) 玉真 昭男，小松 隆，青木 悠：プログラミング教育と３Ｄコンピュータゲーム開発，静岡理工科大学紀要，第15巻，2007，pp. 39-46.

2) 玉真昭男：３Ｄコンピュータゲーム開発を課題としたプログラミング教育，情報処理学会研究報告，2008-CE-97(5)，pp. 29-36，2008.

3) 小松隆，玉真昭男，宮田圭介(静岡文芸大)：DirectXを活用した3Dレーシングシミュレータの作成，情報処理北海道シンポジウム 2006，ポスターセッションE-8、2006.

4) 三浦義弘，鈴木絵美子，玉真昭男：物理モデルを使用したドライビングシミュレータ及び運転評価システムの開発，情報処理学会研究報告，2008-CG-133(10)，pp. 55-59，2008.

5) Metasequoia ホームページ：http://www.metaseq.net/index.html