C++によるカーレースゲームの製作

生島 章伍 高野 祐樹
高橋 義季

１．研究概要

C++言語により記述されたサンプルコードを改良して,よりカーレースゲームらしく仕上げる。

２．研究の具体的内容

グラフィックス作成を生島,BGM 作成を高野,プログラミングを高橋が,それぞれ担当した。

(１)グラフィックス制作

ア.概要

Blender の基本的な操作を用いて作成した。

イ.Blender について

Blender は,オープンソースの３ＤＣＧソフトウェア。３Ｄモデルの作成やアニメーション,レンダリングなど,３ＤＣＧにおける必要な機能を備えている。

ウ.カーモデル作成手順

Blender の最初の画面は中央に立方体がある(図１)。

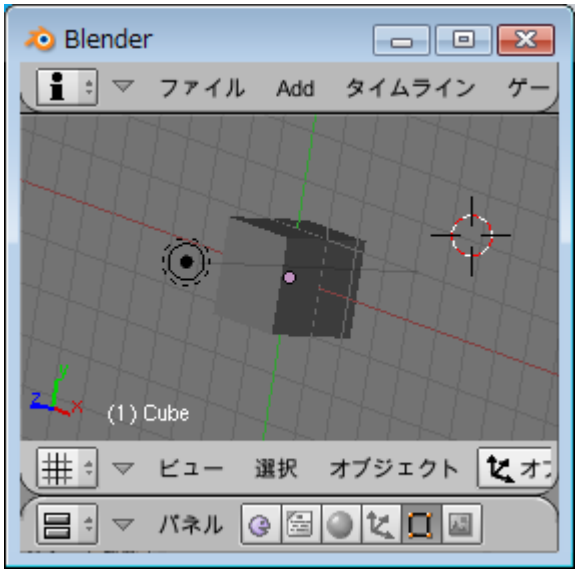

図 1.初期状態

Step１:拡大･縮小を用いて車体の大体の大きさを決める(図２)。

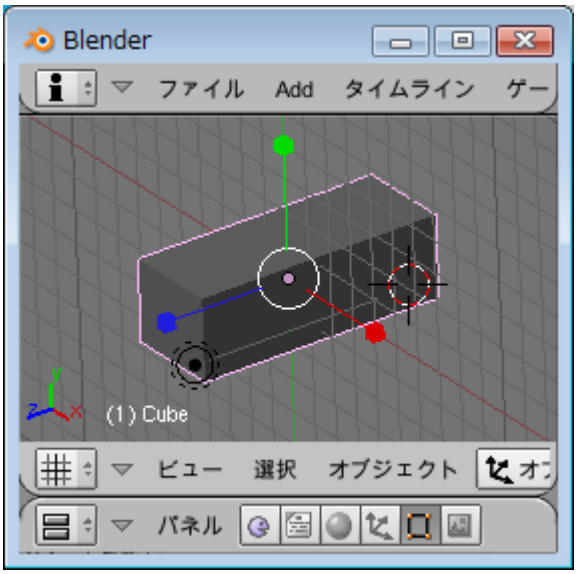

図２.拡大・縮小

Step２:オブジェクトを細分化する(図３)。

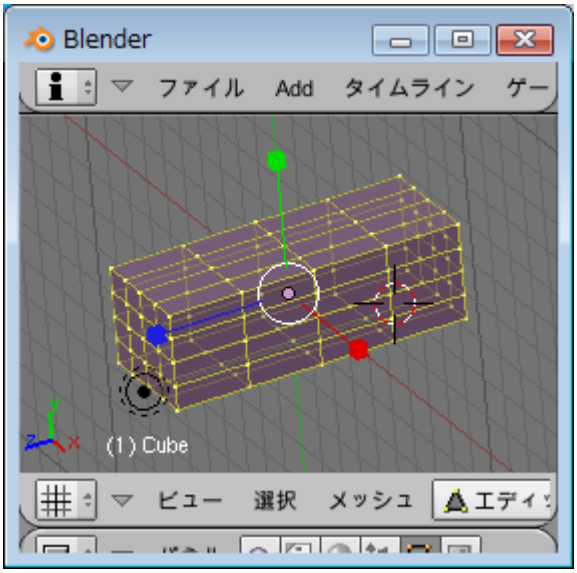

図３.細分化

Step３:パーツの範囲を削除する。この時ある範囲での細分化を用いることもある。

例：タイヤの配置部分(図４)

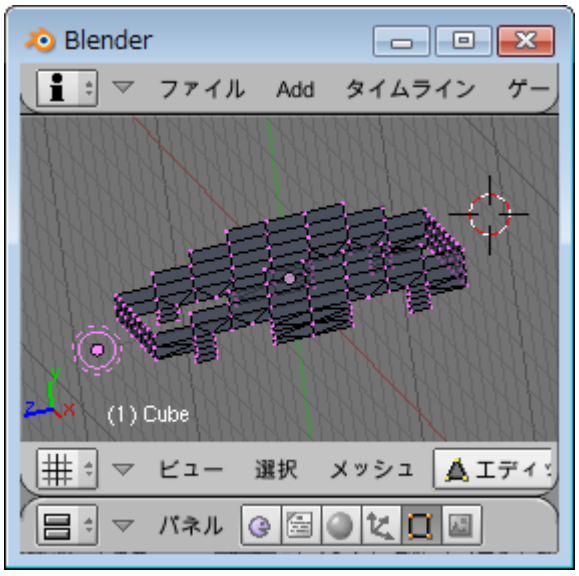

図４.削除

Step４:面と辺の作成をする。面を作る時には頂点だけでなく辺もいる時がある(図５)。

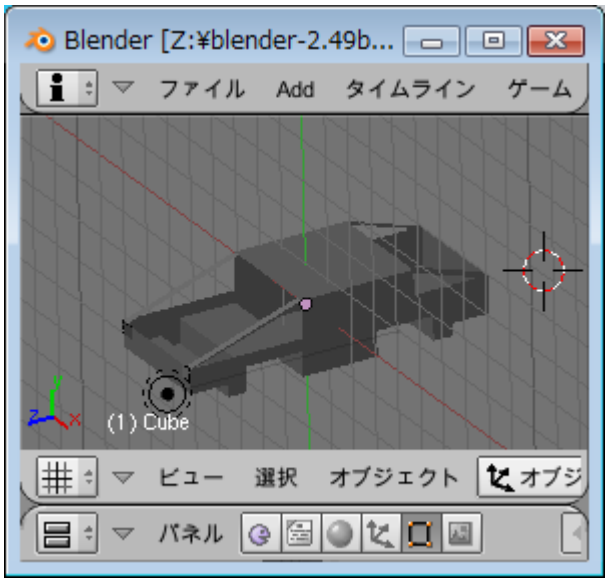

図５.面・辺の生成

Step５:窓やタイヤなどの他のパーツを別のオブジェクトとして作成,色付けする。この時に各オブジェクトの位置も揃える(図６)。

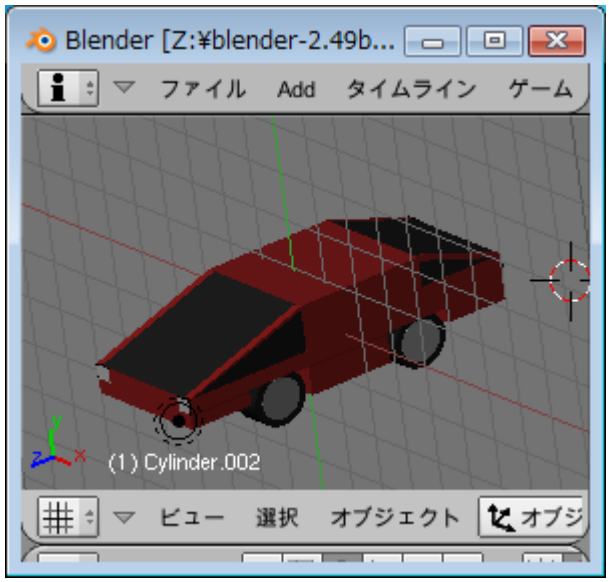

図６.他のオブジェクトの生成

Step６:すべてのオブジェクトを結合し,実際に走らせて細かい修正を加える(図７)。

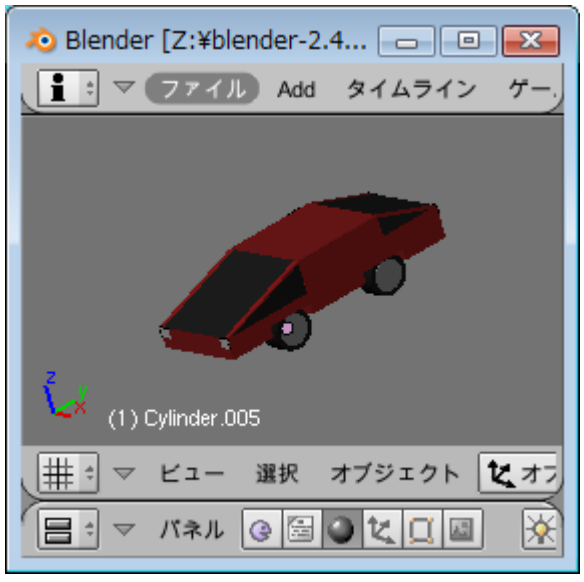

図７.オブジェクトの結合

(２)BGM 制作

ア. 概要

フリーソフトの Domino を使用して、ゲーム中に流れる BGM の制作をした。

Domino はピアノロールという編集方法を使った MIDI 専用の音楽編集ソフト。無料ですべての機能が使え,使い方もシンプルで使いやすい。

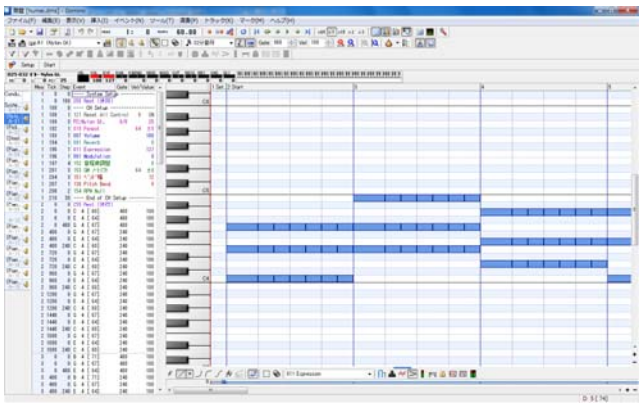

図８.作曲中

イ．具体的内容

参考サイト（管理人の作曲講座 http://supuhuri.sub.jp/sakyoku.htm）から作曲の仕方や方法を調べ,簡単な曲から少しずつ作った。音楽に関しては,初心者だったので,音階を覚えるところから始め､そこを覚えると,次は和音の構成などを覚えていった。

音楽の基礎知識を抑えたところで,次は作曲の仕方を勉強した。コード進行や,メロディの作りを勉強し,私たちが普段聞いている音楽はこういう構成によって出来ているのだと理解しながら進めた。

この二点を理解したところで,実際にＢＧＭの作成を行った。曲を作る手順は,

（ア）コード(元になる音)を作る。

（イ）メロディー(主体になる音)を作る。

（ウ）ベースライン(リズム体となる音)を作る。

（エ）アレンジを加える。

以上の手順で曲を作成した。

ウ．Domino の機能

Domino で使える機能をいくつか記載する。

･最大 16 ポート(256 チャンネル)を処理可能。

･ピアノロールで選択範囲の移動中に音階プレビューが可能。

･音楽定義ファイルを用意することで多くの音源に対応可能。

･ピアノロールに他のトラックの内容を同時に表示するオニオンスキン機能。

･演奏位置と音源の内部状態を合わせる機能（手動または自動)。

･リアルタイム入力が可能。

今回載せている機能以外にも多数の機能があるので興味があれば調べてみてほしい。

(３)プログラミング

ア.サンプルについて

もとにしたカーレースゲームのサンプルコードは、DirectX の初期化や車の挙動処理など基本機能が実装されている。

DirectX の設定には専門的な知識とそれを理解する時間が必要になるため、それを省けたことで助かった。

イ.サンプルの問題点

基本的な動作はできていたが,カーレースゲームとして至らない点も多く残っていた。

・スピード感の演出

・BGM,SE 再生機能

・車種変更機能

・操作性の向上

・周回機能,等々……

ウ．改善点

・スピード感を出すために単位時間当たりの移動量を変えてゲームスピードを上げた。すると,想定外にも敵車のスピードが大幅に上がってレベルが高くなった。

・DirectX の PlaySound 関数をゲーム開始時と左クリックされたときに呼び出して BGM を流すようにした。

･車種選択画面は実装に時間がかかるので、プレイ中に「Shift」キーを押すことで車の性能を switch 文で切り替える(図９)。変数 aKey にはキーボードの入力状態が格納されている。If 文は,「Shift」キーが押されていて,かつ,その他の入力が無い場合に限り,実行される。int 型グローバル変数の MyCarNo は自車の車種識別番号を持つ。インクリメントして次の車種番号にうつり,switch 文で分岐し,それ

ぞれの車体モデルや,性能値を代入して車種変更としている。

```
//自車種変更
if(( aKey & kKEY_SHIFT )&&(theKeyLock == FALSE))
{
    MyCarNo++;
    switch(MyCarNo){
        case 1:
            g_Object[kOBJ_B_CAR01].MdlType      = kMDL_CAR_02;

            g_Object[kOBJ_B_CAR01].CarWeight    = kCAR_WEIGHT02;
            g_Object[kOBJ_B_CAR01].CarWidth     = kCAR_WIDTH02;
            g_Object[kOBJ_B_CAR01].CarLength    = kCAR_LENGTH02;
            g_Object[kOBJ_B_CAR01].WheelUnit    = kWHEEL_UNIT02;
            g_Object[kOBJ_B_CAR01].WheelMax     = kWHELL_MAX02;
            g_Object[kOBJ_B_CAR01].SpeedMax     = kSPEED_MAX02;
            g_Object[kOBJ_B_CAR01].BrakeUnit    = kBRAKE_UNIT02;
            g_Object[kOBJ_B_CAR01].BreakMax     = kBRAKE_MAX02;
            g_Object[kOBJ_B_CAR01].RevMax       = kREV_MAX02;
            g_Object[kOBJ_B_CAR01].RevIdling    = kREV_MAX02 * 0.1f;
            g_Object[kOBJ_B_CAR01].RevUnit      = kREV_MAX02 / kDIV_UNIT;
            g_Object[kOBJ_B_CAR01].AccelUnit    = kSPEED_MAX02 / kDIV_UNIT;
            break;
```

図９.車種変更

エ. 画面のレンダリング(描画)について

ゲーム内の車や木,ガードレールといったモデルは個々に構造体オブジェクトを持ち,メンバ変数を所有している（図 10)。常時動作する車のモデルには、上方のメンバに計算された値を格納し,真ん中のメンバには性能値が設定される。

```
struct theObject{
    D3DXVECTOR3 objPos;     // オブジェクトの座標。
    D3DXVECTOR3 objTarget;  // ターゲット位置。
    float       Yaw;        // ヨー
    float       YawVel;     // ヨー速度
    float       Roll;       // ロール
    float       RollVel;    // ロール速度
    float       Pitch;      // ピッチ
    float       PitchVel;   // ピッチ速度
    float       Vel;        // 速さ
    float       VelTheta;   // 速さの方向
    float       wheelAngle; // ステアリングの角度
    float       Brake;      // ブレーキの踏み込み
    float       Rev;        // エンジン回転数
    /*車の性能*/
    float       CarWeight;  // 車の重さ
    float       CarWidth;   // 車の幅
    float       CarLength;  // 車の長さ
    float       WheelUnit;  // 単位旋回角度
    float       WheelMax;   // 最大旋回角度
    float       SpeedMax;   // 最高速度
    float       BrakeUnit;  // ブレーキ係数
    float       BreakMax;   // 最大ブレーキ力
    float       RevMax;     // 最高回転速度
    float       RevIdling;  // アイドリング
    float       RevUnit;    //
    float       AccelUnit;  //
    /*車の性能  ここまで*/

    int         MdlType;    // オブジェクトのモデルの種類
    int         LDflag;     // オブジェクトの発生状態
    int         VISflag;    // レンダリングするか否か
    int         objCounter; // 汎用カウンタ
    int         courceCeck; // コース曲がったかチェック
    int         thinkMode;  // オブジェクトの思考状態
    int         SLIPflag;   // スリップしてますか？
};
```

図 10.構造体オブジェクトの宣言

オ.フローチャート

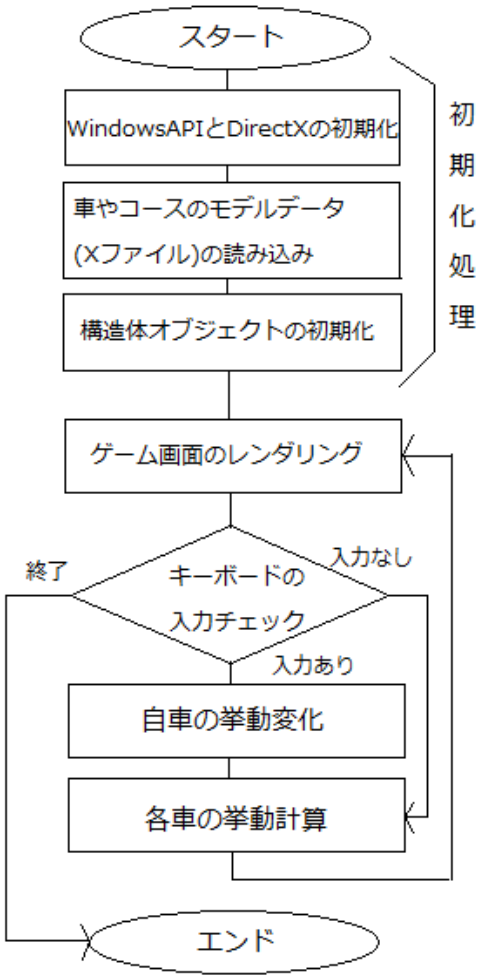

図 11.フローチャート

カ.開発環境

プログラミング環境は VisualStudio という統合開発環境の VC++を利用した。Intellisense という自動補完システムを搭載しており,変数名や関数名を素早く,間違えのないように支援する機能もある。

家庭などで使用する場合はオプションで自分の好みに合わせた設定ができるのだが,学校で行う場合は,ログインシステムなどのせいか,設定が保存されなかった。そのため,起動させるたびに Include するファイルや DirectX ライブラリの設定をしなければならなかった。

キ.用語補足

WindowsAPI （ Aplication Programming Interfaces）とは,Windows の機能を呼び出すための関数セットのこと。例えば,ウィンドウの作成や音楽の再生を行うために使用する。

DirectX とは,ソフトウェアからハードウェアを直接制御するための API。VRAM の性能を検知してレンダリングを最適化するほか,キーボードの入力処理,ディスプレイデバイスの初期化などゲームプログラミングをする上で必要な処理を担う。

３．研究のまとめ

（ア）操作説明

図 12.ゲーム画面

上の画像は実際のゲーム画面のもの。画面の中央で走っているのが自分の操作する車。ほかの黒い車はコンピュータが操作する。右下のメーターは左からタコメーター,速度計となっている。

操作コマンド

・↑キー or Z キー　　：アクセル
・↓キー or X キー　　：ブレーキ
・←キー,→キー　　：左右旋回
・Tab キー or C キー　：視点変更
・Shift キー　　：自車の外見変更
・画面左クリック　　：BGM 変更
・画面右クリック　　：BGM 停止
・ESC キー　　：ゲーム終了

（イ）感想

短い時間の中で,どれだけまともな出来の車などのモデルを作る事が出来るのかが一番の課題だった。使用する操作を基本的な操作だけにしてもそれなりのものができると思っていた。しかし、実際にしてみると車の形がうまく作る事が出来なかった。その後もなんども試行錯誤を加えていくたびに自分なりの操作が出来るようになり,それなりの出来のモデルを作成することが出来た。残された時間の中で少しでも多くのモデルの作成をしていきたい。

（生島 章伍）

１年間,カーレースゲームの製作ということで､課題研究に取り組んできたが,最初は簡単に作ることができるだろうと軽い気持ちで課題研究を始めたが,実際は 1 年かかっても自分達が予想していた形にはすることでできないほどに,ゲームを作るというのは難しい作業だった。

私は BGM の制作を担当して,作曲の難しさ,音楽の編成を学んだ。ゲームをしている時,必ずと言っていいほど流れている BGM。その BGM の曲調によって,テンションが高くなったり,低くなったり,BGM が人に与える影響はすごく大きい。私たちはカーレースゲームの制作をしていたので,やはりゲームを盛り上げるような BGM にしようと考えながら作っていった。残された時間で,より良い BGM が作れるようにさらに頑張りたいと思う。

（高野 祐樹）

この課題研究は,シューティングやアドベンチャーといったジャンルの参考書が立ち並ぶ中,タイトルにあげられることのない,レースゲームの参考書を探すところから始まった。結局,見つかったのは一冊の本に付属された一つのサンプルだけだった。それだけ,3D レースゲームというものの希少さが表れているのではないかと思う。それについて一年間試行錯誤できたのは貴重な体験だった。まだ SE

再生機能や周回機能などカーレースゲームとして至らない点が多く残っている。また,C++と銘打っておきながら,実際のコードは C 言語的なものだ。オブジェクトのクラス化等で,まだまだ C++言語の力を発揮することができるだろう。卒業まで尽力しようと思う。

（高橋 義季）

４.参考文献

書籍

・ゲームのアルゴリズム思考ルーチンと物理シミュレーション改訂版(橋口 ゆうすけ 著)

・Windows ゲームプログラミング（赤坂 玲音 著）

サイト

・ゲームのアルゴリズム思考ルーチンと物理シミュレーションサンプルダウンロードサイト
http://www.sbcr.jp/products/4797359251.html?sku=4797359251

・MSDN ライブラリ
http://msdn.microsoft.com/ja-jp/library/ms123401.aspx

・blender.jp
http://blender.jp/

・TAKABO SOFT
http://takabosoft.com/domino

・シーケンスソフト Domino の使い方メモ
http://miku.motion.ne.jp/beginner/DominoMemo.html

・ピアノコード Clip
http://www.piano-c.com/pianoClip.html

・管理人の作曲講座
http://supuhuri.sub.jp/sakyoku.htm