ESライブラリ＆＆
ゲームプログラミング

○　３Ｄ編 - 第１１回　アルファブレンド　○

アルファブレンド

・アルファブレンドは、２つの画像をα値により合成すること
・半透明のオブジェクトを表現するためにたびたび使用される
・描画順を間違えると正しく合成されない場合がある
・不透明オブジェクトを描画→半透明オブジェクトを奥から描画が基本
・ｚバッファにより、半透明であっても遮蔽されるポリゴンは描画されない

概要

アルファブレンドは、２つの画像をα値によって合成して表示するものです。半透明のオブジェクトを表現するために、たびたび使用されます。

半透明のオブジェクトを描画するときに問題となるのが描画順です。描画順によっては、半透明で描画されているにもかかわらず、奥のものが透けて見えない場合があります。半透明オブジェクトはｚソートにより並べ替え、奥から描画するのが基本となっています。不透明と半透明が混在する場合でも、不透明オブジェクトを描画した後に、半透明オブジェクトを奥から描画、という流れになります。

2Dグラフィクスでは、基本的に描画されるオブジェクトが平面のため、上記の方法で問題ありませんが、3Dになるとさまざまな問題が発生します。

3Dグラフィクスにおける半透明描画の問題点

立体的なモデルを描画する場合、前後関係はｚバッファによって判断され、奥にあるピクセルは手前のピクセルによって上書きされます。半透明モデルであっても、以下のように「ｚバッファで前後関係が判断された最終ピクセル」と転送先画像との半透明処理が行われてしまいます。


▲不透明モデル


▲そのまま半透明合成


▲ｚバッファ無効で半透明合成

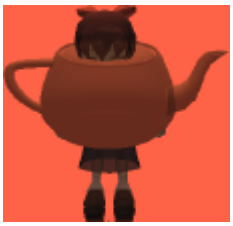
すべて背面カリングなしで描画しています。ｚバッファによって前にあるものだけ処理しようとするので、半透明の場合は中央のように、奥のポリゴンが削除されて描画されます。半透明描画では、表裏、前後、関係なくすべてのポリゴンを削除せず半透明で描画しなければなりません。

ｚバッファが問題なので、レンダリングステートを操作し、ｚバッファを無効にしてみます。すると右のようになります。前後判断をしないので、すべてのポリゴンが半透明で描画されています。このとき、ｚ値は使われないので、前後判断をせず、とにかく描画しようとします。2D画像と同じように「最後に描画したものが一番前」という結果になります。右もパーツの描画順により、どことなくおかしな映像になっています。また、ｚ値がないため、ほかのモデルと重なるとかなり怪しげな映像になります。
正しく描画するには、モデルだけではなくパーツごと、正確には「すべての半透明モデルのすべてポリゴン」をｚソートしなければなりません。そこまで考慮するとプログラムも複雑になり、速度的な問題が出てきます。

ｚ値を考慮しつつ、奥から描画されるよう、裏面を半透明描画した後、表面を半透明描画してみます。レンダリングステートのカリングを設定して片面ずつ描画すれば簡単に行えます。

結果はかなり改善されたように見えます。半透明合成では、完全に矛盾なく前後関係を考慮して半透明合成するのは大変難しいので、さまざまな工夫をしながら、ある程度の矛盾には目をつぶるか、目立たないようにして描画します。

ほかのモデルとの重なり



複数のモデルを半透明で描画する場合、カメラからの距離によってｚソートし、奥のモデルから描画します。このとき、モデルの体積を考慮しないと、右のように不透明モデルのように重なってしまいます。これから描画する半透明モデルより奥のモデルがあらかじめ描画されていないと、色の合成できないからです。

たくさんのモデルを半透明合成し、まったく矛盾なく重ねることは難しい問題ですが、ある程度体積を考慮することによって軽減することができます。

また、遠くのモデル同士を合成してもレンダリング上は見えない場合もあるので、一定の距離以上のものは合成しないという考え方もあります。

ESライブラリでの対応

ESライブラリでは、半透明の頂点バッファ、モデル、アニメーションモデルをなるべく問題が起こらないように描画できるよう、専用のモードが用意されています。

GraphicsDevice.BeginAlphaBlend関数を呼び出せば、アルファブレンド描画に適した状態にデバイスを設定できます。アルファブレンド描画の終了はGraphicsDevice.EndAlphaBlend関数です。この関数を呼び出すことにより、デバイスの状態をアルファブレンド描画開始前の状態に復元します。また、頂点バッファ、モデル、アニメーションモデルには、アルファブレンド専用の描画関数DrawAlphaがあります。この関数に透明度を指定するれば、半透明描画が行われます。

課　　題

アルファブレンドで複数のモデルを合成して描画しましょう。

(１)モデルやアニメーションモデル、影が描画されるシーンを準備しましょう。

(２)半透明を行って描画します。

１．半透明モデルは最後に描画した方がより自然に見えます。よって、描画順は

不透明モデル → 影 → 半透明モデル

となります。ただし、半透明モデルは奥から描画するようにｚソートする必要があります。

２．半透明描画前に、GraphicsDevice.BeginAlphaBlend関数で各種ステートの設定を行います。

３．半透明描画後に、GraphicsDevice.EndAlphaBlend関数で各種ステートを元に戻せます。

４．半透明描画自体は、頂点バッファ／モデル／アニメーションモデルのDrawAlpha関数で行います。
引数は、従来のDraw関数＋透明度(正確には不透明度)です。

５．以上をまとめると、以下のようなコードになります。

```
void CGameMain::Draw()
{
    GraphicsDevice.Clear(Color_Tomato);

    // TODO: Add your drawing code here
    GraphicsDevice.BeginScene();

    // 不透明モデル描画(略)

    // 影描画(略)

    // 半透明モデル描画
    GraphicsDevice.BeginAlphaBlend();

    if(pos.z <= model->GetPosition().z - 1.1f) {
        model->DrawAlpha(0.5f);                                  // ティーポット
        anime->DrawAlpha(GameTimer.GetElapsedSecond(), 0.75f);  // アニメーションモデル
    } else {
        anime->DrawAlpha(GameTimer.GetElapsedSecond(), 0.75f);  // アニメーションモデル
        model->DrawAlpha(0.5f);                                  // ティーポット
    }

    GraphicsDevice.EndAlphaBlend();

    GraphicsDevice.EndScene();
}
```

(３)合成モードを変更してみましょう。

合成の方法は、GraphicsDevice.BeginAlphaBlend関数の引数またはGraphicsDevice.SetBlendMode関数で指定できます。以下のモードがあります。

```
    BlendOperation_Modulate    乗算合成(デフォルト)
    BlendOperation_Add         加算合成
    BlendOperation_Subtract    減算合成
    BlendOperation_SrcColor    ？？合成
    BlendOperation_ZeroColor   転送元を黒として合成
```