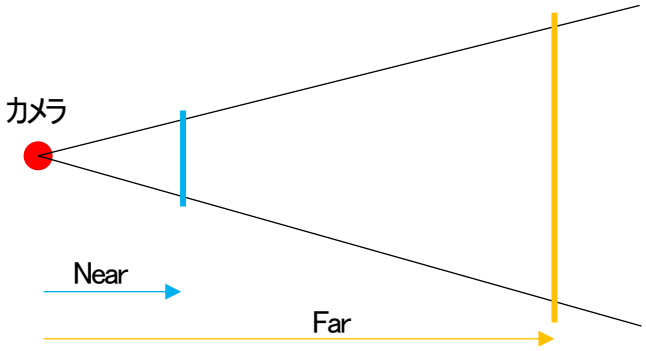
ゲームプログラミング応用Ⅰ／ゲームプログラミング実践Ⅰ

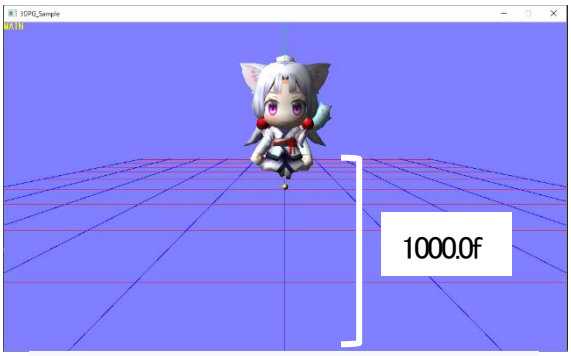
20 日で理解する3Dプログラミング 「その 5：小技」

■カメラの「手前クリップ距離」と「奥クリップ距離」

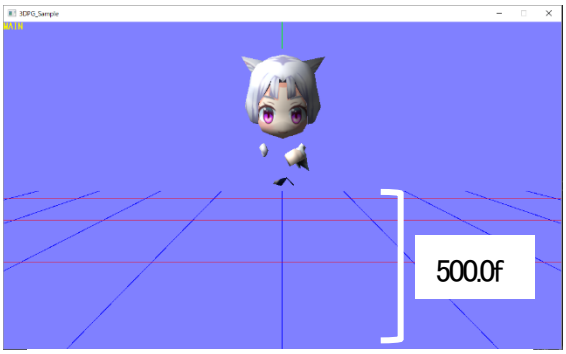
3D空間に何かを描画する際に、カメラからどれだけ離れたところ(Near)から、どこまで(Far)のものを描画するかを設定します。この関数の設定値は使用する3D空間の範囲に合わせて適切な値を設定する必要があります。

又、Near の値は「なるべくカメラに近くても描画したい」という考えから小さい値にしがちですが、不都合が無い範囲でなるべく大きな値(1.0f～10.0fぐらい)を、Far の値は描画したい最奥のモノのより少し大きな値を設定するようにしましょう。





SetCameraNearFar(10.0f, 1000.0f);



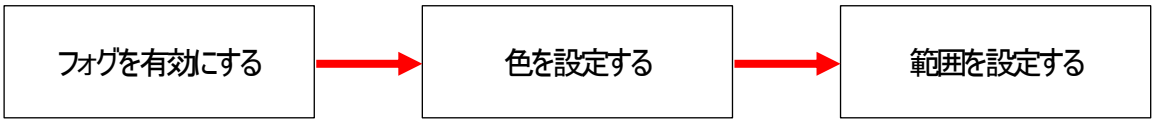
SetCameraNearFar(10.0f, 500.0f);

設定は、SetCameraNearFar()関数で行いますが、途中で変更を加えないのであれば初期化時に一度だけ実行すれば良いです。又、環境によってはカメラから 100.0f 以上離れると描画されなくなったりしますので、Near の値は不都合が無い範囲でなるべく大きな値を、Far の値は描画したい最奥のモノのより少し大きな値を設定するようにしてください。

■フォグ

フォグ機能とは言葉通り立体空間で霧を表現する手段の一つです。具体的には、カメラ(画面)から一定距離離れた物体に霧が掛かったように任意の色を合成することができます。これを使用することで空気が淀んでいる日に遠くのものを見ようとすると白く霧が掛かったようになってよく見えない、というような空気遠近法も再現することができます。

因みにこの機能はカメラ(画面)から一定距離以上離れたものを描画しなくても済むようにする目的で使用することもできます。(フォグ終了距離以降の部分はフォグ色のみとなるため)



①フォグの有効・無効

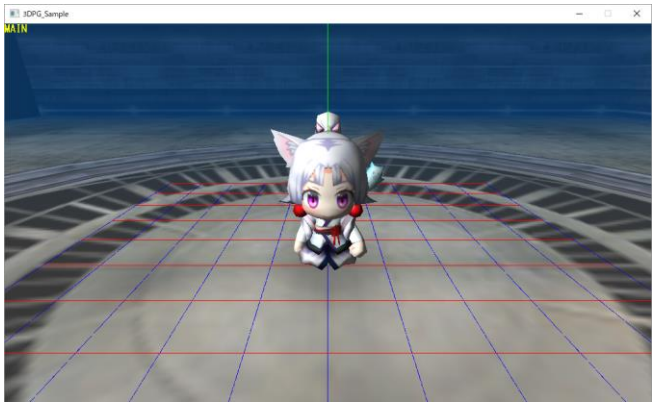
```
SetFogEnable( int Flag ) ;
```

②フォグの色指定

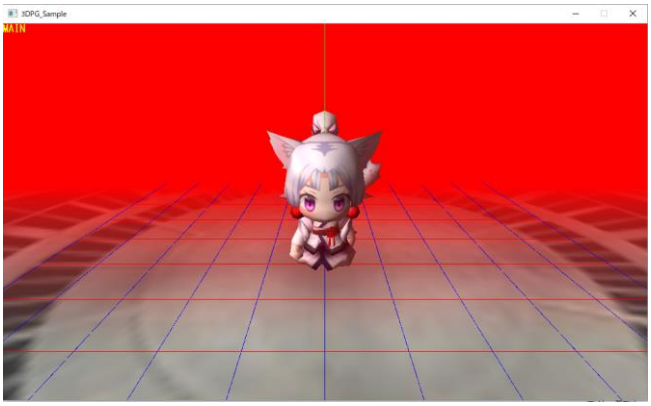
```
SetFogColor( int Red, int Green, int Blue ) ;
```

③フォグの範囲指定

```
SetFogStartEnd( float start, float end ) ;
```




```
// フォグを有効にする
SetFogEnable(true);

// フォグの色をRGBで設定する
SetFogColor(0, 50, 100);

// フォグの開始距離と終了距離を設定する
SetFogStartEnd(600.0f, 3500.0f);
```



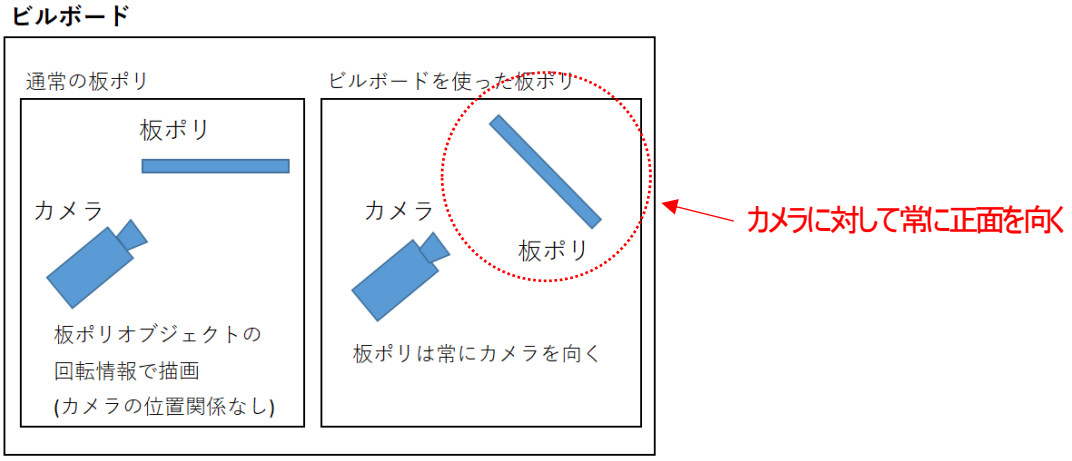

```
// フォグを有効にする
SetFogEnable(true);

// フォグの色をRGBで設定する
SetFogColor(255, 0, 0);

// フォグの開始距離と終了距離を設定する
SetFogStartEnd(400.0f, 1000.0f);
```

フォグの掛かり具合については背景オブジェクトを置くと分かりやすくなります。遠景の背景を上手くぼかす様に「カメラからの開始距離」と「カメラからの終了距離(色が100%になる所)」の調整を行い、効果的な表現を行って行きます。

■ビルボード

3D 空間に 2D 画像を描画する際に、常にカメラの方向を自動で向くようになっているものです。3D 空間上に表示する HP ゲージの UI やエフェクトでよく使用されています。

ビルボードを使用することによるメリットは描画コストを減らせることです。3D のオブジェクトを全てポリゴンとして作成していたら大量のポリゴンを描画することになるので、描画コストに多大な負荷がかかります。そこでオブジェクトの一部をビルボードにすることで描画を行うポリゴン数を減少させ、描画コストを軽減させていきます。



3D 空間上に、常にカメラを向く様に 2D を配置する
DrawBillboard3D(描画する 3D 座標, 画像の中心, サイズ, 回転, 画像ハンドル, 透明度);

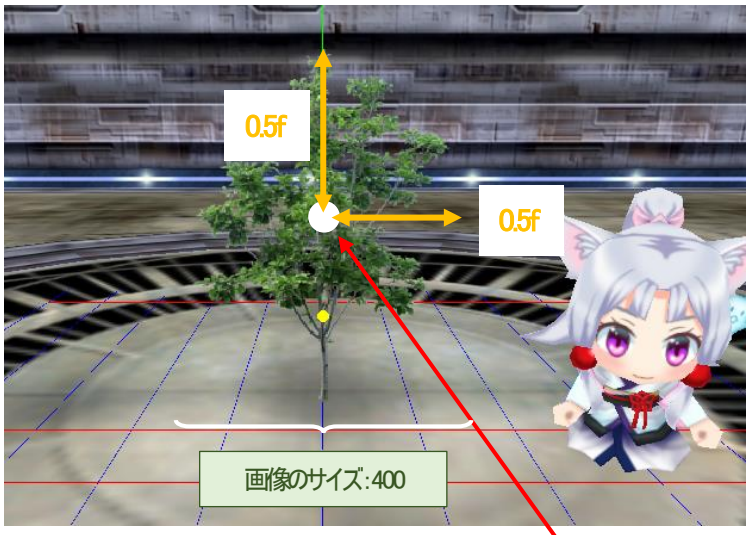
描画する座標・・・・・・・・VECTOR Pos
画像の中心場所・・・・・float cx, cy （0.0f ～ 1.0f）
画像のサイズ・・・・・・・・float Size
画像の回転角度・・・・・float Angle (ラジアン単位)
画像ハンドル・・・・・・・・int GrHandle
透明化・・・・・・・・・・・・・int TransFlag; するかどうか(true:有効 false:無効)

3D 空間上に、常にカメラを向く様に 2D を配置する ※変形機能付き
DrawModiBillboard3D(描画する 3D 座標, 描画する座標の頂点, 画像ハンドル, 透明度);

画像を描画する座標・・・・・VECTOR Pos;
描画する画像の座標・・・・・float x1, y1, ※左上
float x2, y2, ※右上
float x3, y3, ※左下
float x4, y4, ※右下
描画する画像・・・・・・・・・・・int GrHandle;
描画像の透明化・・・・・・・・int TransFlag; するかどうか(true:有効 false:無効)

■BrawBillboard3D 関数でのビルボードの描画

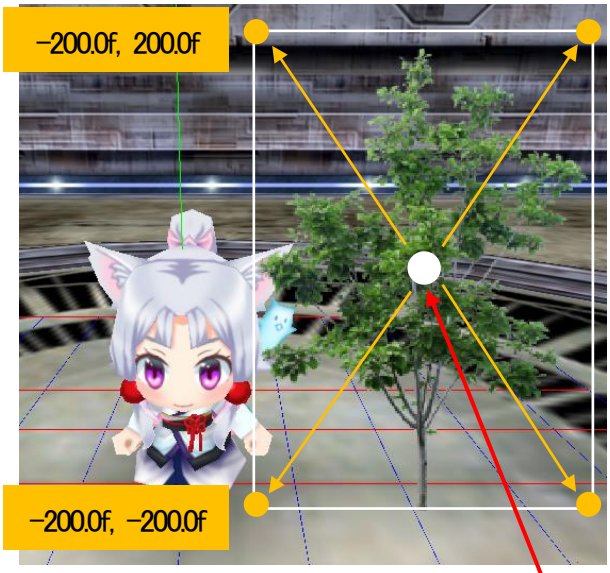
```
DrawBillboard3D(VGet(0.0f, 200.0f, 0.0f), 0.5f, 0.5f,
                400.0f, 0.0f, treeImage, true);
```



表示座標(0.0f, 200.0f, 0.0f);

■BrawModiBillboard3D 関数でのビルボードの描画

```
DrawModiBillboard3D(VGet(200.0f, 200.0f, 0.0f), // 中心
    -200.0f,  200.0f,        // 左上(中心からの距離)
     200.0f,  200.0f,        // 右上(中心からの距離)
     200.0f, -200.0f,        // 右下(中心からの距離)
    -200.0f, -200.0f,        // 左下(中心からの距離)
    treeImage, true);
```

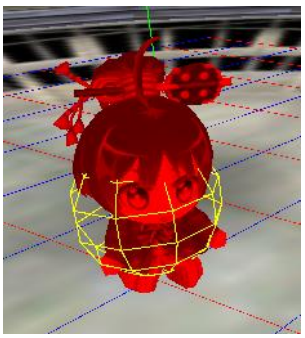


表示座標(200.0f, 200.0f, 0.0f);

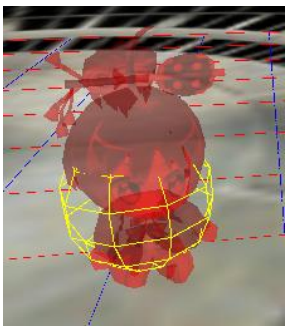
■3D モデルの描画色を変える

```
// 物体色を赤(1.0f, 0.0f, 0.0f)でモデルを描画する
MV1SetDifColorScale(modelID, GetColorF(1.0f, 0.0f, 0.0f, 1.0f));
MV1DrawModel(modelID);
```

色成分　　α値



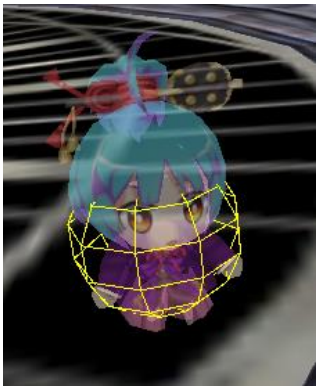
(1.0f, 0.0f, 0.0f, 1.0f)の場合



(1.0f, 0.0f, 0.0f, 0.5f)の場合

■3D モデルの不透明度を変える

```
// 透明度を0.5にして描画する
MV1SetOpacityRate(modelID, 0.5f);
MV1DrawModel(modelID);
```



■3D モデルをワイヤーフレームで描画する

```
// ワイヤーフレームで描画する
MV1SetWireFrameDrawFlag(modelID, true);
MV1DrawModel(modelID);
// 通常状態に戻す
MV1SetWireFrameDrawFlag(modelID, false);
```