

# XNA Game Studio ゲームプログラミング

## 3D編 - 第4回 カメラ

### カメラ

- ・モデルを3D空間に配置しただけでは画面に描画できない
- ・3D空間の「どこから」「どの方向を」「どのように」見るのか、視点と視界(=カメラ)を設定する
- ・カメラの位置、姿勢を設定するのがビュー変換、カメラからの見え方を設定するのが射影変換

### カメラ

3Dグラフィクスでは、仮想世界(ワールド座標)にオブジェクトを配置して仮想世界を作成します。次に行うのは、仮想世界の「どこから」「どの方向」を「どのように」見るのか、という「視点」と「視界」の設定です。

視点と視界は、ワールド座標の任意の位置から任意の方向をとらえることができます。XNAでは、これらのことをカメラまたはビューと呼びます。カメラも頂点データを画面へ描画するための一連の座標変換の一部です。よって行列で設定を行います。カメラの座標と方向を表す行列と、どのように見えるかを表す行列を作成し、デバイス(XNAではエフェクト)に適用します。その後、プリミティブの描画を行うと、カメラがとらえた仮想世界の画像が出力されるというわけです。

### ビュー変換行列

カメラから見る画像は、「ワールド座標に配置された3Dオブジェクトの頂点」を「カメラを基準とした座標」に変換したものとイえます。このカメラを基準とした座標をビュー座標系と呼びます。ワールド座標系からビュー座標系への変換は、ビュー変換行列と呼ばれる行列を用います。

3Dオブジェクトの頂点にワールド変換行列が掛けられることにより、ワールド座標系への変換が行われますが、さらにビュー変換行列を掛けると、カメラから見た世界、つまりカメラを原点としたビュー座標系へと変換されます。

ビュー変換行列は、カメラのx軸を示すベクトルを $r$ 、カメラのy軸を示すベクトルを $u$ 、カメラのz軸を示すベクトルを $v$ 、カメラの位置を示すベクトルを $p$ としたとき、以下のような行列になります。

$$\begin{pmatrix} r_x & u_x & v_x & 0 \\ r_y & u_y & v_y & 0 \\ r_z & u_z & v_z & 0 \\ -(R \cdot p) & -(U \cdot p) & -(V \cdot p) & 1 \end{pmatrix}$$

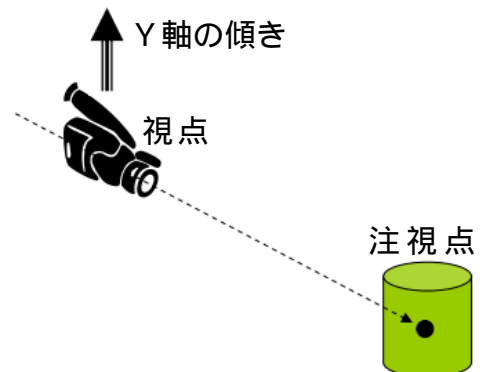
ビュー変換行列は、カメラの位置と方向を定義したものとイえます。

### LookAt方式

カメラを制御する場合、「視点の位置」「視点の方向」の2つの要素を用いますが、これは、ハンディカメラを扱うような感覚でカメラを制御することになります。この方式のほかに、テレビカメラのように、ある位置からある位置をとらえるLookAtという方式があります。

LookAt方式とは、直感的にカメラを扱うために考え出された方式で、XNA以外の3Dライブラリでもよく使われています。

LookAt方式では、カメラの制御に「視点の位置」「注視点の位置」「Y軸の傾きを制御するベクトル」の3つの要素を用います。



LookAt方式でビュー変換行列を作成するには、まず視点の位置から注視点の位置を引き、正規化します。これは、カメラの視点方向(z軸)を示すベクトル(v)となります。次に、このベクトルとY軸の傾きを制御するベクトルの外積を求めます。すると、カメラの右方向(x軸)を示すベクトル(r)となります。さらに、rとvの外積をとると、カメラの上方向(y軸)を示すベクトルをuが求められます。これで、カメラの(直交する)3つの軸が求められます。あとは、前述のビュー変換行列に代入し、位置と方向ベクトルの内積を計算するだけです。

## ビュー変換行列の作成と設定

XNAには、LookAt方式でビュー変換行列を作成するMatrix.CreateLookAtメソッドがあります。なお、この方式以外でビュー変換行列を作成するメソッドはありません。

ビュー変換行列が作成できたら、BasicEffectのViewメンバに設定します。以後、BasicEffect.Begin~Endメソッドに囲まれたすべてのプリミティブの描画時に、設定したビュー変換行列が適用されます。カメラの位置や姿勢などが変わった場合は、それに合わせてビュー変換行列を変更し、BasicEffect.Viewメンバに再設定します。

**書式** Matrix.CreateLookAt(Vector3, Vector3, Vector3)

Vector3	カメラの座標(ワールド座標)
Vector3	カメラの注視点座標(ワールド座標)
Vector3	カメラの上方向を示すベクトル。ここを変えるとカメラがz軸回転します

- 説明 -

LookAt方式で、右手座標系のビュー変換行列を作成します。

- 使用例 -

ワールド座標[0.0, 1.0, 5.0]から[0.0, 0.0, 0.0]を注視するカメラの設定

// basicEffectは生成済みのBasicEffect型変数

```
basicEffect.View = Matrix.CreateLookAt(new Vector3(0.0f, 1.0f, 5.0f),  
                                       new Vector3(0.0f, 0.0f, 0.0f), Vector3.Up);
```

## 射影変換(プロジェクション変換)

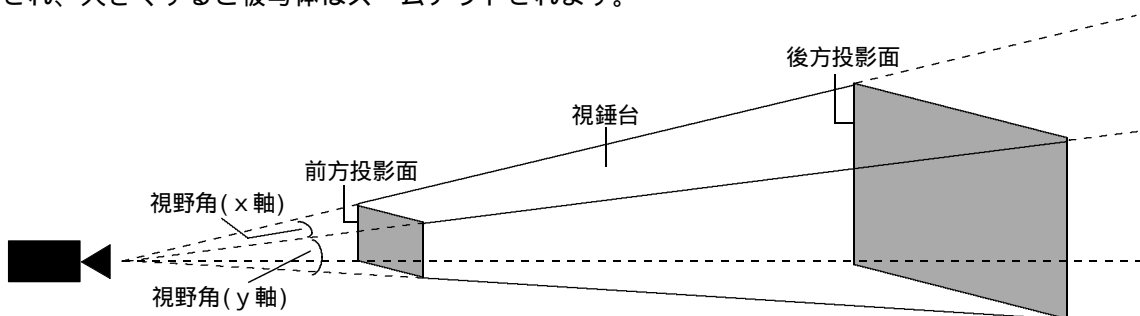
ビュー変換の次に行うのが、「カメラでどのようにとらえるか」という設定です。これを射影変換(プロジェクション変換または投影変換)と呼びます。これは、カメラのレンズを設定する作業といえます。射影変換を行うと、「カメラをとおして見た空間(カメラが捉えた映像)」に変換されます。

射影変換には、おもに透視投影(パースペクティブ射影: Perspective Projection)と正投影(正射影: Orthogonal Projection)が用いられます。

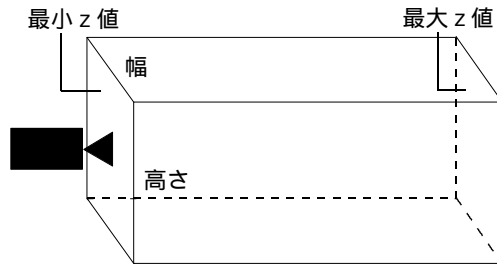
透視投影とは、「パースペクティブ: 遠近」という言葉どおり、遠くのもの小さく見えるという見え方のことです。

透視投影では、投影面という概念を導入します。投影面とは、スクリーンのように世界を写し取る平面のことです。この投影面を2つ導入します。カメラに近いところと遠いところ、それぞれ前方投影面と後方投影面と呼びます。前方投影面より近い範囲は見えない、後方投影面より遠い範囲も見えない、というようにすると、見える範囲がピラミッドの上部を切り取ったような形に区切られることとなります。この形を視錐台と呼びます。視錐台から完全にはずれているオブジェクトは描画されません。このようなオブジェクトのレンダリング処理を省くことにより、描画速度を向上させることができます。

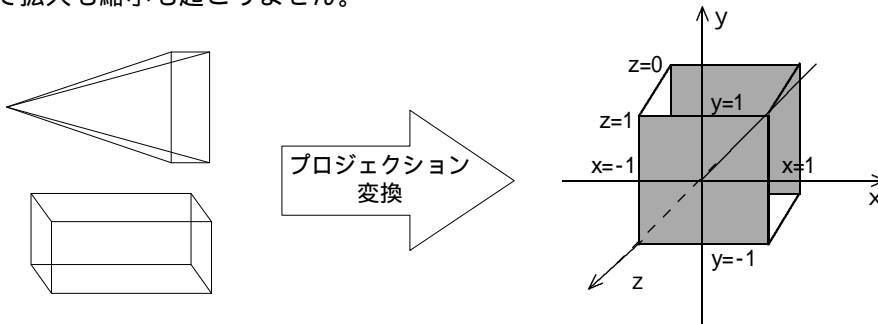
さらに、視野角(Field Of View)という概念を導入します。これは、見える範囲の角度のことで、視線の中心からどの角度までを描画するかを指定するものです。視野角を小さくすると被写体はズームインされ、大きくすると被写体はズームアウトされます。



正投影とは、オブジェクトを平面的に見るという方法です。遠近感がないので、遠くにあるものが小さく表示されたりしません。正投影は、CADのような図形を正確に表示しなければならない分野で使われています。



射影変換では、カメラがとらえた空間(ビューボリューム)を「 $X = -1 \sim +1$ 」「 $Y = -1 \sim +1$ 」「 $Z = 0 \sim +1$ 」の領域に収まるように変換します。この空間を射影空間と呼びます。  
透視投影では、ビューボリュームがカメラの原点を頂点とした錐になっているため、遠くのものほど縮小されます。そのため、遠いものは小さくなり、遠近感ができます。正投影では、ビューボリュームが立方体なので拡大も縮小も起こりません。



## 射影変換行列

ビュー座標系に変換された頂点の座標を射影空間の座標(射影座標系)に変換するには、射影変換行列(プロジェクション変換行列)と呼ばれる変換行列を用います。ビュー座標系の頂点に射影変換行列を掛けることにより、カメラを原点とした座標から、実際にカメラから見た座標へ変換されます。

射影変換行列は、投影方法により異なる行列となります。通常、パースペクティブ射影行列の場合は、以下のように視野をもとに作成します。

$$\begin{pmatrix} w & 0 & 0 & 0 \\ 0 & h & 0 & 0 \\ 0 & 0 & z_f/(z_n-z_f) & 1 \\ 0 & 0 & z_n*z_f/(z_n-z_f) & 0 \end{pmatrix} \quad \begin{array}{l} h = \cot(Y \text{軸の視野角} / 2) \\ w = h / \text{投影面の横} \div \text{縦} [= \text{アスペクト比}] \\ z_n = \text{視点から前方投影面までの距離} \\ z_f = \text{視点から後方投影面までの距離} \end{array}$$

正射影行列の場合は、以下のように作成します。

$$\begin{pmatrix} 2/w & 0 & 0 & 0 \\ 0 & 2/h & 0 & 0 \\ 0 & 0 & 1/(z_n-z_f) & 0 \\ 0 & 0 & z_n(z_n-z_f) & 1 \end{pmatrix} \quad \begin{array}{l} h = \text{ビューボリュームの高さ} \\ w = \text{ビューボリュームの幅} \\ z_n = \text{ビューボリュームの最小} z \text{値} \\ z_f = \text{ビューボリュームの最大} z \text{値} \end{array}$$

## 射影変換行列の作成と設定

XNAは、視野をもとにパースペクティブ射影行列を作成するMatrix.PerspectiveFieldOfViewメソッドや正射影行列を作成するMatrix.Orthographicメソッドなど、いくつかの射影行列作成のメソッドを提供しています。

射影変換行列が作成できたら、BasicEffectのProjectionメンバに設定します。以後、BasicEffect.Begin~Endメソッドに囲まれたすべてのオブジェクトの描画時に、設定した射影変換行列が適用されます。射影変換行列は基本的に変更しませんので、一度だけ設定しておけば十分です。

## 書式 Matrix.CreatePerspectiveFieldOfView(float, float, float, float)

float Y軸の視野角(ラジアン単位)  
float 投影面のアスペクト比(幅÷高さ)  
float 視点から前方投影面までの距離。これより手前にあるものは描画されません  
float 視点から後方投影面までの距離。これより後ろにあるものは描画されません

- 説明 -  
視野に基づいてパースペクティブ射影行列を作成します。

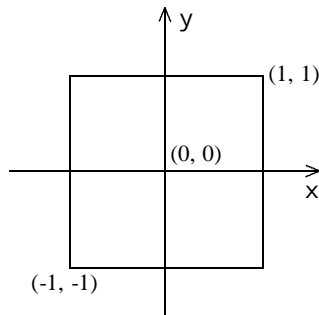
- 使用例 -  
Y軸視野角45度、前方投影面1.0、後方投影面100.0、アスペクト比をビューポートと同じ比率に設定する射影変換の作成

```
// basicEffectは生成済みのBasicEffect型変数  
basicEffect.Projection = Matrix.CreatePerspectiveFieldOfView(  
    MathHelper.ToRadians(45.0f),  
    (float)GraphicsDevice.Viewport.Width /  
    (float)GraphicsDevice.Viewport.Height,  
    1.0f, 100.0f);
```

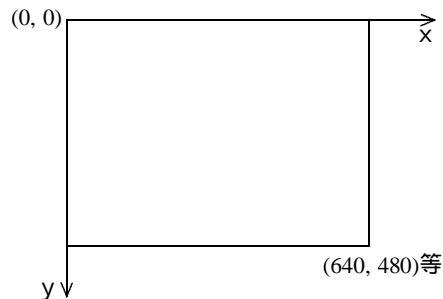
## スクリーン座標系

3Dグラフィックスを描画する際、最後に行われるのが射影座標系からスクリーン座標系への変換です。スクリーン座標系とは、画面のピクセルに1対1に対応する座標系のことです。射影座標系からスクリーン座標系への変換は、ビューポートスケール行列またはビューポート行列と呼ばれる変換行列が用いられます。

射影座標系とスクリーン座標系の違いは、中心の座標です。射影座標系では中央にあり、スクリーン座標系では左上にあるということです。また、y軸の向きが逆になっていることも大きな違いです。



射影座標系



スクリーン座標系

ビューポートスケール行列は、画面の幅をWidth、画面の高さをHeightとしたとき、以下のようになります。

$$\begin{pmatrix} \text{Width}/2 & 0 & 0 & 0 \\ 0 & -\text{Height}/2 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ \text{Width}/2 & \text{Height}/2 & 0 & 1 \end{pmatrix}$$

XNAでは、ビューポートの設定はGraphicsDeviceクラスのViewportメンバに適切な値を設定するだけで、自動的に変換行列として適用され、スクリーン座標系への変換が行われます。デフォルトではウィンドウと同じ幅、高さになっています。このメンバを工夫すると、左右分割して別の空間を映し出すといったことも可能です。

最後に、モデル座標系からスクリーン座標系への変換をまとめると、

スクリーン座標 = モデル座標 × ワールド行列 × ビュー行列 × 射影行列 × ビューポート行列  
となります。

## 課 題

ビュー変換行列と射影変換行列(パースペクティブ)を作成しましょう。

(1) GameMainクラスのフィールド変数に、以下の変数を追加しましょう。

```
// カメラ行列  
Matrix viewMatrix, projMatrix;
```

(2) ワールド座標(0.0, 0.0, 5.0)から(0.0, 0.0, 0.0)を注視するビュー変換行列をviewMatrixに設定するプログラムをGameMain.Initializeメソッドに作成しましょう。

(3) 「Y軸視野角45度、前方投影面1.0、後方投影面1000.0、アスペクト比をビューポートと同じ比率に設定する射影変換行列」をprojMatrixに設定するプログラムを適切な場所に作成しましょう。