アフィン変換と画像処理

北陸先端科学技術大学院大学 先端科学技術研究科（青木研究室）

長谷川 央

2022-11-20

自己紹介

名前

長谷川 央（ハセガワ アキラ）

経歴

1997	愛知県豊田市で生まれる
2016	名古屋大学教育学部附属中・高卒
2016–2020	三重大学 総合情報処理センター主催 講習会「パソコン分解講習会」TA
2019	三重大学 総合情報処理センター主催 講習会「Linux 実践入門」講師
2020	北陸先端科学技術大学院大学 入学

はじめに

映画・ゲームなど
CG 無しでは成立しない時代となっている

ゲームの例 (Unity)

- ポケモン GO
- 原神
- ポケモン BDSP (ダイパリメイク)
- ウマ娘
- Among us
- Fate/Ground Order (FGO)

今回は CG の描画で行われる計算について紹介する

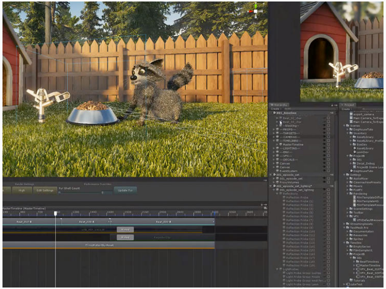
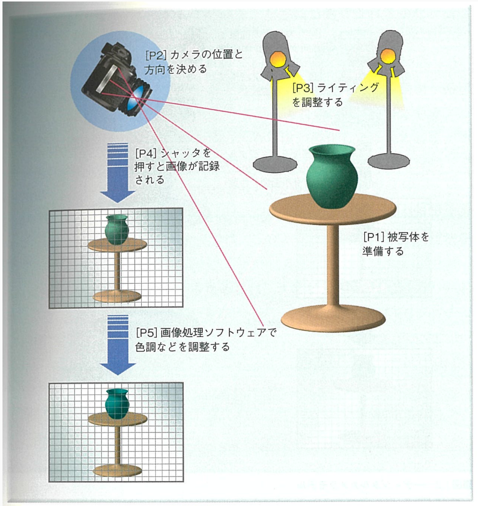
[1]

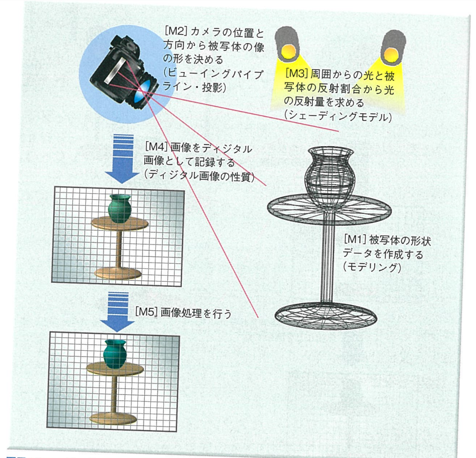
Figure 1: Unity での CG 作成風景

[1] (Unity–3D Animation Software for Film & Television, https://unity.com/sites/default/files/styles/cards_16_9/public/2019-06/unity-for-layout.jpg)

コンピュータグラフィックスの概要図



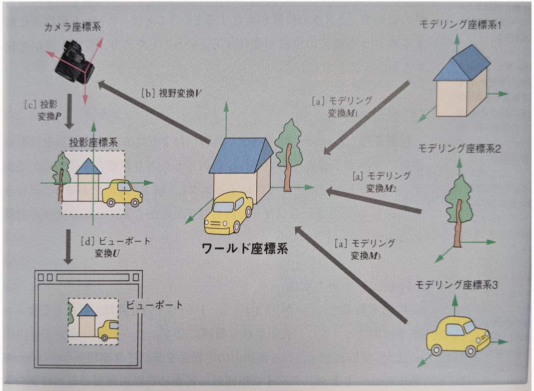
■図1.1——ディジタルカメラでの撮影過程



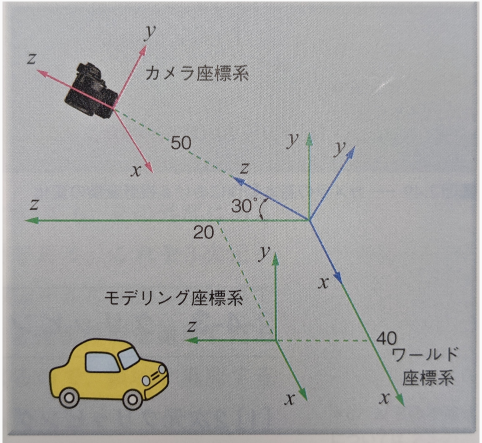
■図1.2——ディジタルカメラモデル [2]

[2] (コンピュータグラフィックス［改訂新版］. 公益財団法人 画像情報教育振興協会, 2017)

ビューイングパイプラインの概要図



■図2.46——モデルから表示までの変換(ビューイングパイプライン)



■図2.48——モデリング変換と視野変換の例[3]

[3] (コンピュータグラフィックス [改訂新版]. 公益財団法人 画像情報教育振興協会, 2017)

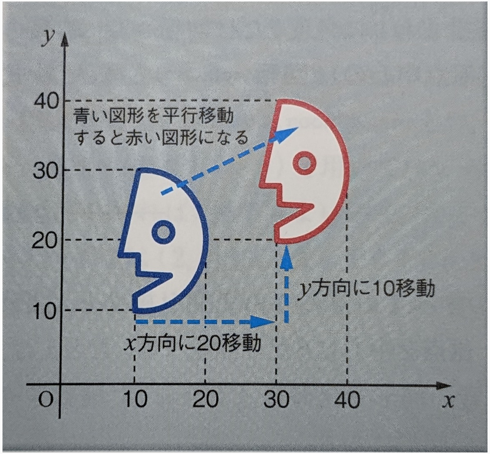
座標の幾何学的変換

線分やポリゴンなどの 2 次元図形に対して以下のような幾何学的変換を行う
（この幾何学的変換をアフィン変換という）

効率よく行う方法はあるか？

- 平行移動
- 拡大・縮小
- 回転
- 鏡映
- スキュー

平行移動



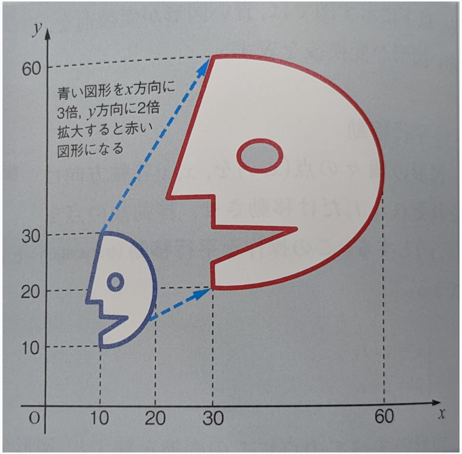
[4]

図形の各点 (x, y) をそれぞれ x 軸方向に t_x
y 軸方向に t_y 移動させる

$$\begin{pmatrix} x \\ y \end{pmatrix} \mapsto \begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} x + t_x \\ y + t_y \end{pmatrix}$$

[4] (コンピュータグラフィックス［改訂新版］. 公益財団法人 画像情報教育振興協会, 2017)

拡大・縮小



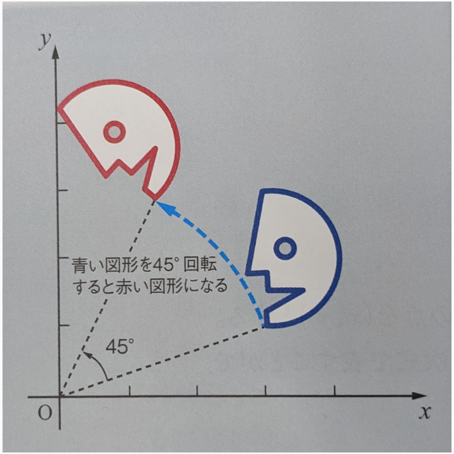
[5]

図形の各点 (x, y) をそれぞれ x 軸方向に s_x 倍 y 軸方向に s_y 倍する

$$\begin{pmatrix} x \\ y \end{pmatrix} \mapsto \begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} s_x \cdot x \\ s_y \cdot y \end{pmatrix}$$

[5] (コンピュータグラフィックス［改訂新版］. 公益財団法人 画像情報教育振興協会, 2017)

回転



[6]

図形の各点 (x, y) をそれぞれ反時計回りに角度 θ 回転する

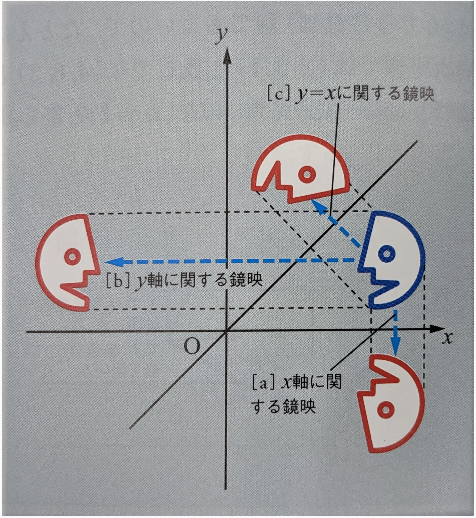
$$\begin{pmatrix} x \\ y \end{pmatrix} \mapsto \begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} x\cos\theta - y\sin\theta \\ x\sin\theta + y\cos\theta \end{pmatrix}$$

回転の中心から (x, y) へのベクトルを $\mathbf{v}$ とし偏角を α とすると

$$\begin{aligned} x' &= \|\mathbf{v}\|\cos(\alpha+\theta) \\ &= \|\mathbf{v}\|\cos\alpha\cos\theta - \|\mathbf{v}\|\sin\alpha\sin\theta \\ &= x\cos\theta - y\sin\theta \\ y' &= \|\mathbf{v}\|\sin(\alpha+\theta) \\ &= \|\mathbf{v}\|\sin\alpha\cos\theta + \|\mathbf{v}\|\cos\alpha\sin\theta \\ &= y\cos\theta + x\sin\theta \end{aligned}$$

[6] (コンピュータグラフィックス［改訂新版］. 公益財団法人 画像情報教育振興協会, 2017)

鏡映



[7]

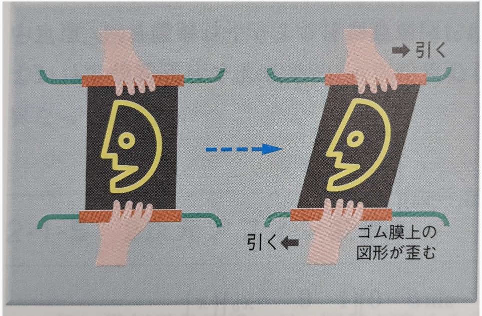
$$[a]\ \begin{pmatrix} x \\ y \end{pmatrix} \mapsto \begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} x \\ -y \end{pmatrix}$$

$$[b]\ \begin{pmatrix} x \\ y \end{pmatrix} \mapsto \begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} -x \\ y \end{pmatrix}$$

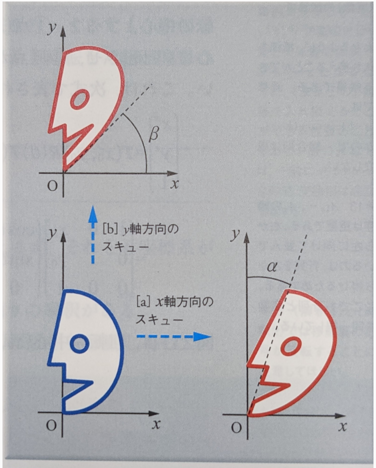
$$[c]\ \begin{pmatrix} x \\ y \end{pmatrix} \mapsto \begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} y \\ x \end{pmatrix}$$

[7] (コンピュータグラフィックス［改訂新版］. 公益財団法人 画像情報教育振興協会, 2017)

せん断（スキュー）



■図2.9——ゴム膜でのスキューの原理



■図2.10——スキューの例

8

8 (コンピュータグラフィックス［改訂新版］. 公益財団法人 画像情報教育振興協会, 2017)

複数の変換の組み合わせ

図形を下端最左の点はズラさずに x 軸方向に s_x 倍，y 軸方向に s_y 倍し
さらに θ 回転させたい

そのまま拡大・縮小や回転をすると下端最左の座標がズレるので次の手順で変換を行う

1. 原点に平行移動する
2. 拡大・縮小を行う
3. 回転を行う
4. 元の位置に戻す

この一連の変換は下端最左の点を (a, b) とすると次式で表せる

$$\begin{pmatrix} x \\ y \end{pmatrix} \mapsto \begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} s_x & 0 \\ 0 & s_y \end{pmatrix} \left(\begin{pmatrix} x \\ y \end{pmatrix} - \begin{pmatrix} a \\ b \end{pmatrix} \right) + \begin{pmatrix} a \\ b \end{pmatrix}$$

平行移動も加算を使わずに乗算として統一的に記述できないか

同次座標系 (Homogeneous Coordinates)

任意の 0 でない実数 w を用いて (x, y) を (wx, wy, w) と書く

これを**同次座標**と呼ぶ

同次座標を用いると前述の変換を乗算として統一的に記述できる

平行移動

$$\mathbf{T}(t_x, t_y) = \begin{pmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{pmatrix}$$

拡大・縮小

$$\mathbf{S}(s_x, s_y) = \begin{pmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

回転

$$\mathbf{R}(\theta) = \begin{pmatrix} \cos\theta & \sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

複数の変換の組み合わせ (同次座標系)

図形を下端最左の点はズラさずに x 軸方向に s_x 倍，y 軸方向に s_y 倍し
さらに θ 回転させたい

そのまま拡大・縮小や回転をすると下端最左の座標がズレるので次の手順で変換を行う

1. 原点に平行移動する
2. 拡大・縮小を行う
3. 回転を行う
4. 元の位置に戻す

この一連の変換は下端最左の点を (a, b) とすると次式で表せる

$$\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \mapsto \begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \mathbf{T}^{-1}(-a, -b)\mathbf{R}(\theta)\mathbf{S}(s_x, s_y)\mathbf{T}(-a, -b)\begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

行列演算は結合法則が成り立つので $\mathbf{TRST}$ を先に計算しても良い

おまけ：四元数 (1/2)

3 次元座標の計算ではアフィン変換の他にも四元数が用いられることがある

四元数 $\mathbb{H}$ は $\mathbb{R}^4$ と同型であり $\mathbb{H} \ni a + bi + cj + dk \ (a, b, c, d \in \mathbb{R})$ と書かれる

虚数単位を i, j, k とするとき次式が成り立つ

$$i^2 = j^2 = k^2 = ijk = -1$$

各虚数単位の積の関係は次表の通り

x	1	i	j	k
1	1	i	j	k
i	i	-1	k	-j
j	j	-k	-1	i
k	k	j	-i	-1

おまけ：四元数 (2/2)

点 $(x, y, z) \in \mathbb{R}^3$ を，単位方向ベクトルを (a, b, c) とする直線を軸に角度 θ だけ回転させたい

回転後の点を (x', y', z') とするとき次式が成り立つ

ただし，$\bar{q}$ は q と共役な四元数である

$$\mathbb{H} \ni q = \cos\frac{\theta}{2} + (ai + bj + ck)\sin\frac{\theta}{2} \text{として}$$

$$0 + x'i + y'j + z'k = q(0 + xi + yj + zk)\bar{q}$$

$\vec{v} = 0 + xi + yj + zk$ と書くと，$\vec{v'} = q\vec{v}\bar{q}$ と書ける

この性質を使えば**任意のベクトルを軸にした回転が簡単に計算できる**

まとめ

今回は CG の座標変換で使用される計算について紹介した

CG 映画の撮影やゲーム画面の描写の裏では 知らないうちに行列や四元数の性質をうまく利用して大量の処理が行われている