Graphics with Processing

2008-04 幾何変換と同次座標系

http://vilab.org

塩澤秀和

4.1 幾何変換

幾何変換とアフィン変換

- 座標変換

$$\begin{pmatrix} x \\ y \end{pmatrix} \xrightarrow{f} \begin{pmatrix} x' \\ y' \end{pmatrix}$$

- 幾何変換
  - 平行移動
  - 拡大・縮小
  - 回転
- アフィン変換

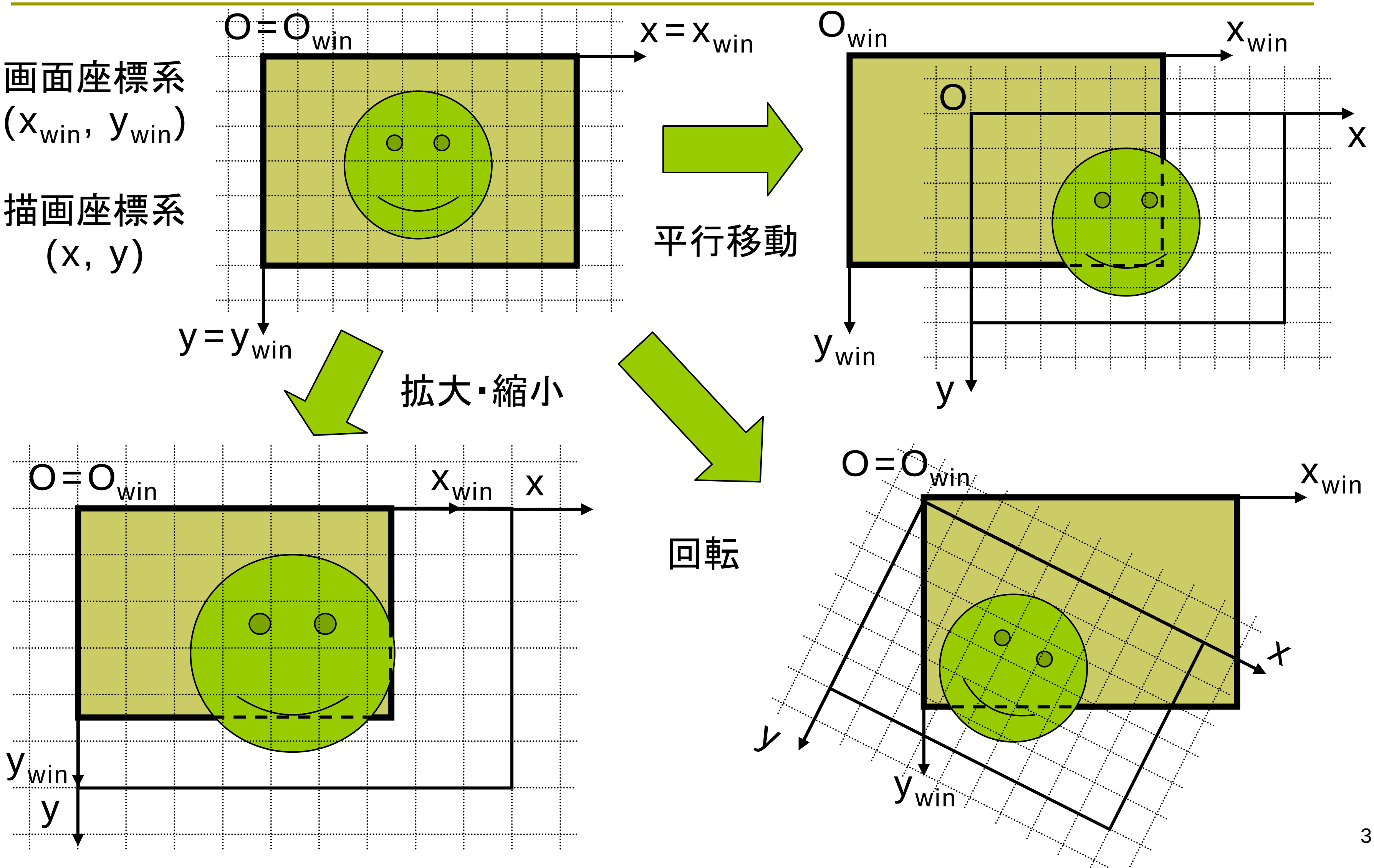
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} e \\ f \end{bmatrix}$$

  - 幾何変換の組合せになる

幾何変換関数

- translate(x_0, y_0)
  - 描画座標系を平行移動
  - x軸方向に x_0 移動
  - y軸方向に y_0 移動
  - y軸は**下向き**なことに注意
- scale(α , β)
  - 描画座標系を拡大・縮小
  - x軸方向（左右）に α 倍
  - y軸方向（上下）に β 倍
- rotate(θ)
  - 描画座標系を回転
  - 原点中心に θ 回転
  - プラスの方向は**時計回り**

4.2 幾何変換の効果



画面座標系
(x_{win}, y_{win})
描画座標系
(x, y)
$O=O_{win}$
$x=x_{win}$
$y=y_{win}$
平行移動
O_{win}
x_{win}
O
x
y_{win}
y
拡大・縮小
$O=O_{win}$
x_{win}
x
y_{win}
y
回転
$O=O_{win}$
x_{win}
x
y
y_{win}

4.3 幾何変換の数学表現

数式による表現

- 描画座標系から画面座標系へ

$$\begin{pmatrix} x \\ y \end{pmatrix} \rightarrow \begin{pmatrix} x' \\ y' \end{pmatrix} \rightarrow \ldots \rightarrow \begin{pmatrix} x_{win} \\ y_{win} \end{pmatrix}$$

- 平行移動と拡大・縮小

$$x' = x + x_0 \qquad x' = \alpha x$$

$$y' = y + y_0 \qquad y' = \beta y$$

- 回転

$$x' = x\cos\theta - y\sin\theta$$

$$y' = x\sin\theta + y\cos\theta$$

ベクトルと行列による表現

- 平行移動

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} x_0 \\ y_0 \end{bmatrix}$$

- 拡大・縮小

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \alpha & 0 \\ 0 & \beta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

- 回転

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

4.4 同次座標系

同次行列

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} e \\ f \end{bmatrix}$$

数学的に
より簡便な表記

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} a & b & e \\ c & d & f \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

行列1つですべての座標変換を表せる

同次座標系による表現

- 平行移動

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & x_0 \\ 0 & 1 & y_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

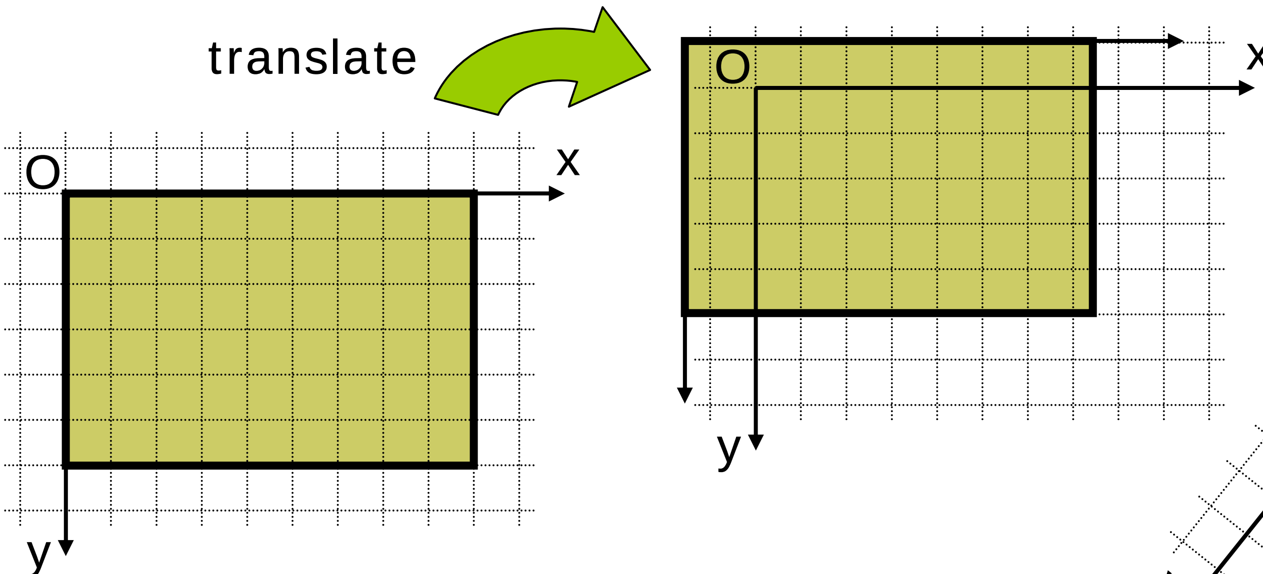
- 拡大縮小

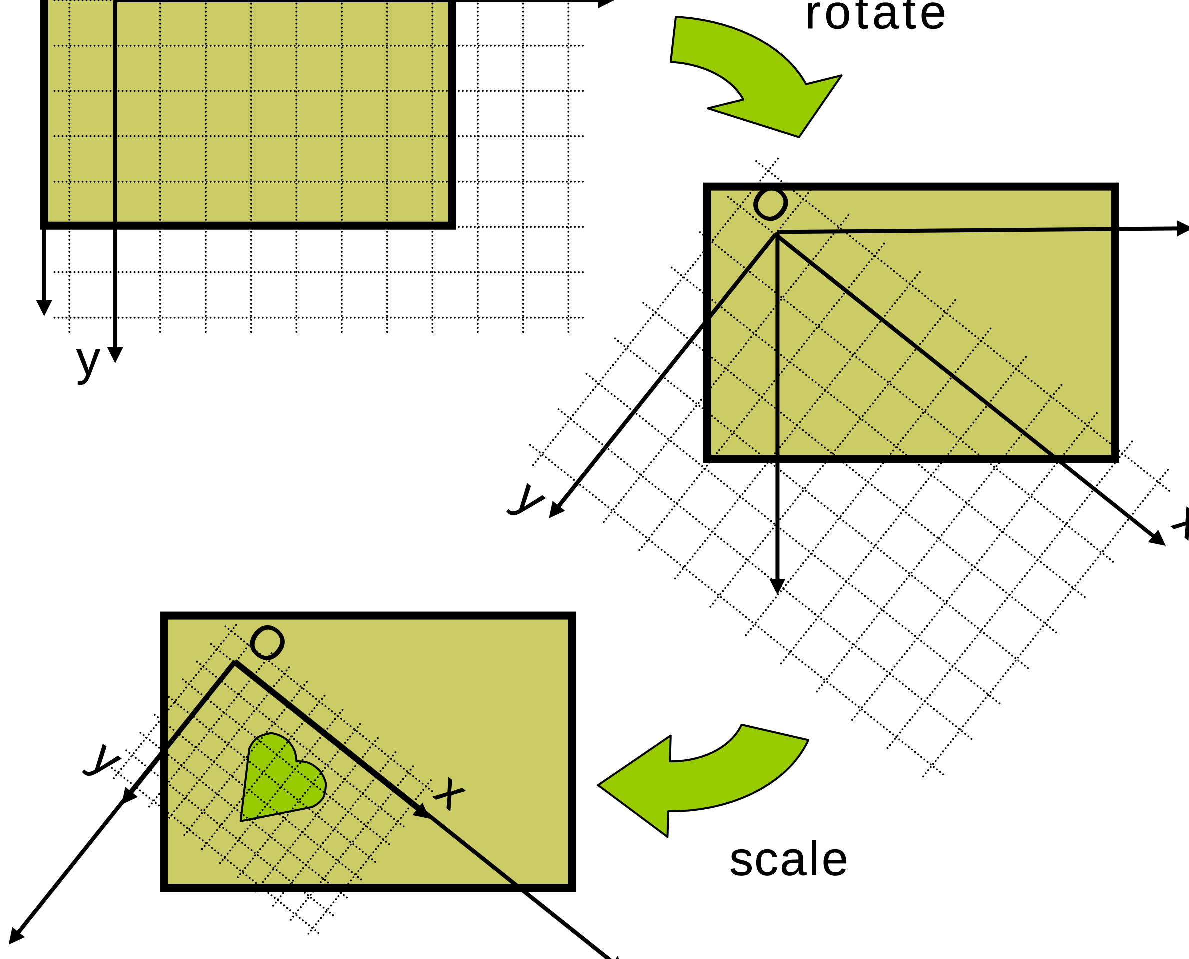
$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \alpha & 0 & 0 \\ 0 & \beta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

- 回転

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

4.5 幾何変換の合成





- Processingコード

```
translate(15, 10);
rotate(PI/4);
scale(0.5, 0.5);
// 図形描画...
```

4.5' 図形移動での考え方

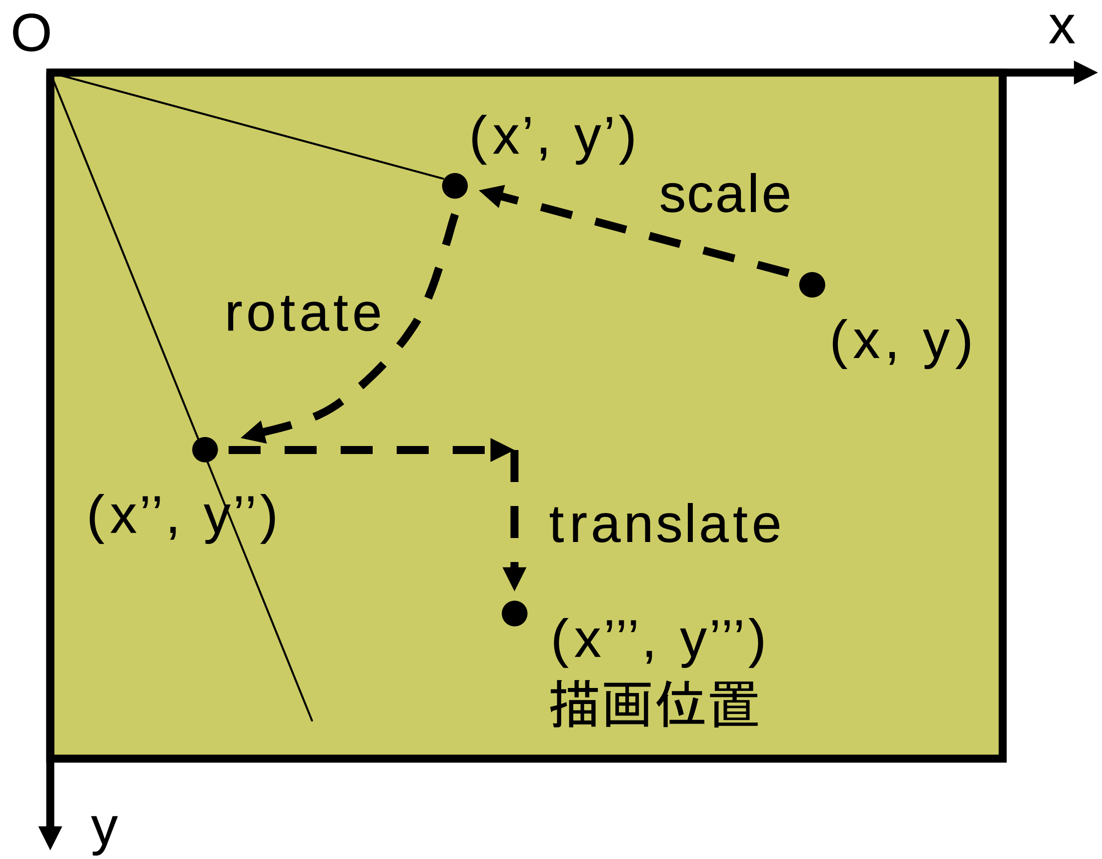
別の考え方

- 座標系の移動ではなく，同じ画面座標系上での図形ごとの移動としても考えられる
- 描画からさかのぼって，図形に命令の逆順で変換を作用させる
- 数学的に同じこと＝結果はどちらも同じ

- 右図の例

```
translate(15, 10);
rotate(PI/4);
scale(0.5, 0.5);
// 図形描画...
```

逆順



4.6 合成変換行列

合成変換の数学表現

- 同次変換行列の積になる

$$P_{win} = M_1 M_2 M_3 \cdots M_n P$$

$$M = M_1 M_2 M_3 \cdots M_n$$

- Processingコード（4.3の例）

```
translate(15, 10);   // 変換 M1
rotate(PI/4);        // 変換 M2
scale(0.5, 0.5);     // 変換 M3
// 図形描画...
```

- 右上の例の行列表現

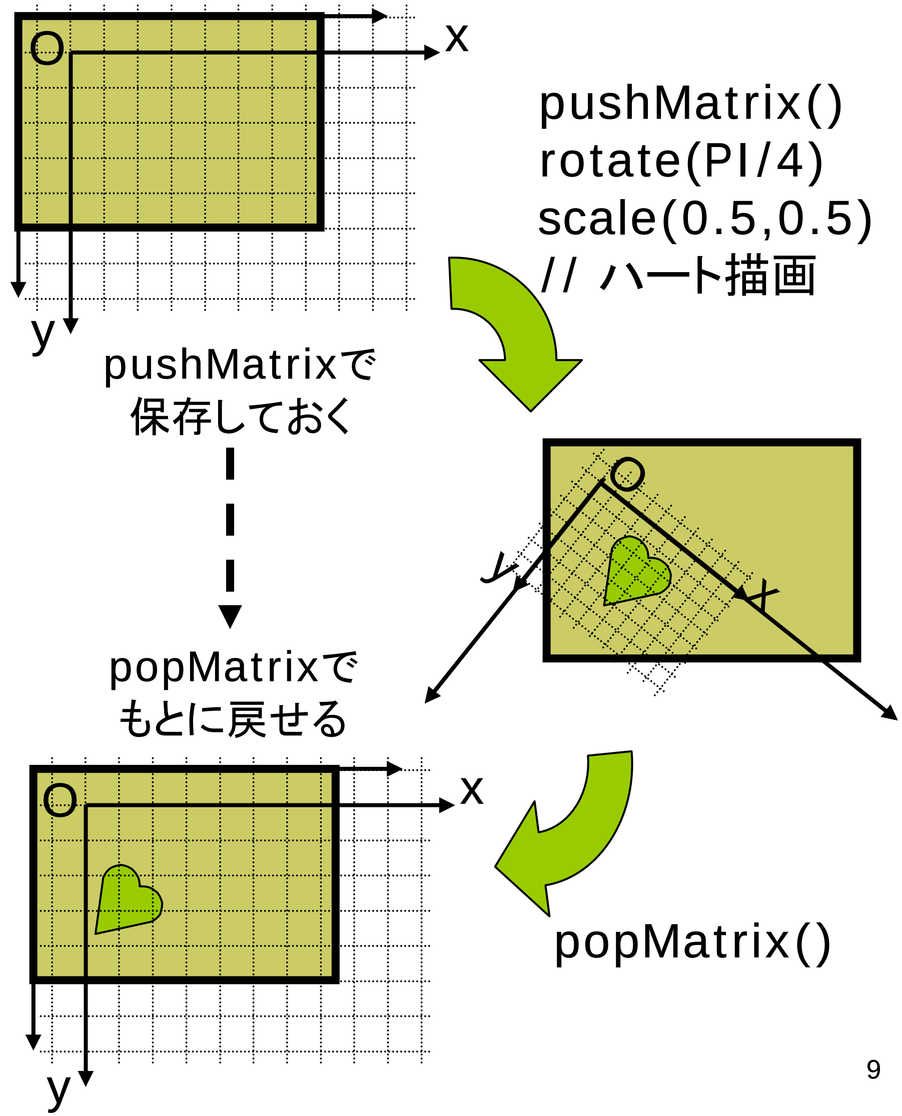
$$\begin{bmatrix} x_{win} \\ y_{win} \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 15 \\ 0 & 1 & 10 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos(\pi/4) & -\sin(\pi/4) & 0 \\ \sin(\pi/4) & \cos(\pi/4) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0.5 & 0 & 0 \\ 0 & 0.5 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} x_{win} \\ y_{win} \\ 1 \end{bmatrix} = \begin{bmatrix} \sqrt{2}/4 & -\sqrt{2}/4 & 15 \\ \sqrt{2}/4 & \sqrt{2}/4 & 10 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \qquad \therefore\ M = \begin{bmatrix} \sqrt{2}/4 & -\sqrt{2}/4 & 15 \\ \sqrt{2}/4 & \sqrt{2}/4 & 10 \\ 0 & 0 & 1 \end{bmatrix}$$

4.7 行列操作

変換行列の操作

- 変換行列
  - システム変換行列は幾何変換（translate, rotate, scale）の処理のたびに合成されていく
  - 変換行列＝描画座標系
- pushMatrix()
  - システム変換行列（描画座標系）を一時待避する
- popMatrix()
  - 最近保存した変換行列を戻す
  - pushMatrix()と必ず対にする
- resetMatrix()
  - 変換行列をリセットする
  - 描画座標系＝画面座標系



4.8 演習課題

演習課題

- 4.9のプログラムは，スマイリー（にこちゃんマーク）を2つ描画するものである

問1) 中心と外側の顔の描画位置を決めている**合成変換行列**（$M_{中心}$と$M_{外側}$）の**両方を**求めなさい

- $M_{中心}$は右のヒント参照
- 次回**A4レポート用紙**で提出

問2) 外側の顔の大きさを半分にし，その向きが回転しないようにプログラムを改造しなさい

- ただし，顔を描画する関数は，変更や追加をしないこと
- 2つ幾何変換を加えればよい
- プログラム（.pde）をWeb提出

問1の$M_{中心}$のヒント

- $M_{中心}$は次の2つの変換の合成
  - M_1 = translate(200, 200)
  - M_2 = rotate(-a)
- それぞれの行列表現は

$$M_1 = \begin{bmatrix} 1 & 0 & 200 \\ 0 & 1 & 200 \\ 0 & 0 & 1 \end{bmatrix}$$

$$M_2 = \begin{bmatrix} \cos(-a) & -\sin(-a) & 0 \\ \sin(-a) & \cos(-a) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

- $M_{中心}$はこの2つの合成なので

$$M = \begin{bmatrix} 1 & 0 & 200 \\ 0 & 1 & 200 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos(-a) & -\sin(-a) & 0 \\ \sin(-a) & \cos(-a) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

4.9 幾何変換の使用例

```
void setup()
{
  size(400, 400);
  frameRate(30);
}
void draw_smiley()
{
  ellipseMode(CENTER);
  strokeWeight(3);
  stroke(0); fill(#ffff00);
  ellipse(0, 0, 100, 100);
  noStroke(); fill(0);
  ellipse(-15, -15, 12, 12);
  ellipse(+15, -15, 12, 12);
  stroke(#ff0000);
  noFill();
  bezier(-25,20, -10,35,
            10,35,   25,20);
}
```

```
void draw()
{
  float a = radians(frameCount);
  background(255);
  translate(200, 200);

  pushMatrix();
    rotate(-a);
    draw_smiley();
  popMatrix();

  pushMatrix();
    rotate(-a);
    translate(130, 0);
    // ここに2つ幾何変換を追加する
    draw_smiley();
  popMatrix();
}
```