Geometric Transformations

Computer Graphics

Introduction

- **Geometric transformation**

  Variation of an object size, shape, position or orientation within a scene

- 2D and 3D geometric affine transformations
  - Translation
  - Scaling
  - Rotation

- Other non-affine transformations: cutting, reflection…

2D Transformations

- **Translation**
  - Change of an object **position** along a straight line. A point P'(x',y') is obtained from moving point P(x,y):
    - dx parallel units to the x axis $x' = x + d_x$
    - dy parallel units to the y axis $y' = y + d_y$

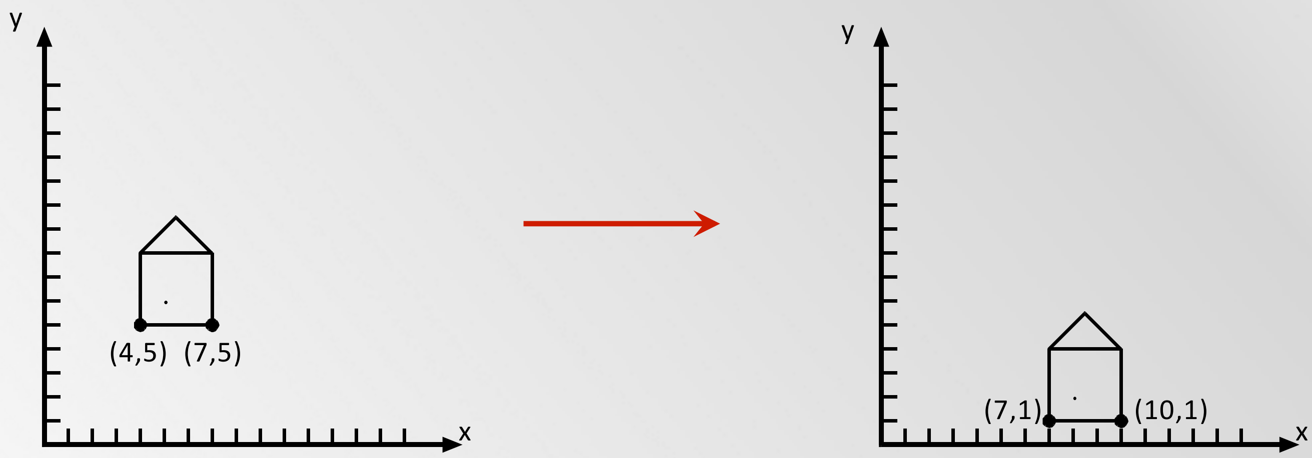
    being **(dx , dy) the translation vector**
  - Equation **formula** and **matrix representation**:

$$P' = P + T$$

$$P = \begin{bmatrix} x \\ y \end{bmatrix}, P' = \begin{bmatrix} x' \\ y' \end{bmatrix}, T = \begin{bmatrix} d_x \\ d_y \end{bmatrix},$$

2D Transformations

- **Translation**
  - Example of **translation** – every point moves 3 units along the x axis and -3 units along the y axis



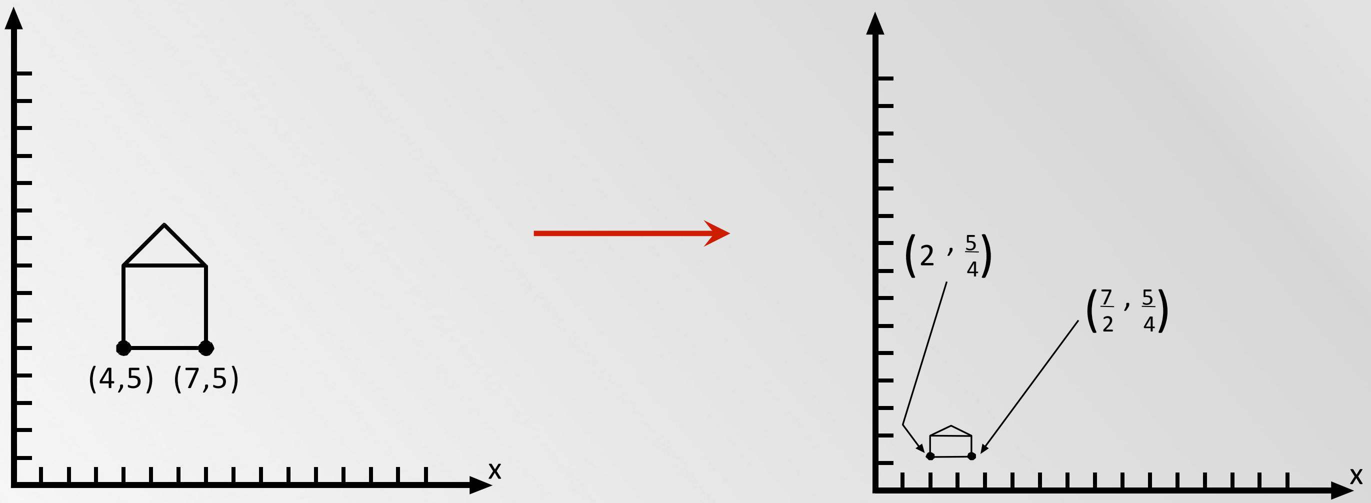
2D Transformations

- **Scaling**
  - Modify an object **dimension**
    - Segments parallel to x axis are multiplied by sx $x' = s_x \cdot x$
    - Segments parallel to y axis are multiplied by sy $y' = s_y \cdot y$
  - When sx = sy → **uniform scaling**
  - Equation **formula** and **matrix representation**

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix} \Leftrightarrow P' = S \cdot P$$

2D Transformations

- **Scaling**
  - Example of **non-uniform scaling** – 1/2 is scaled on the x axis and 1/4 on the y axis (scaled with respect to the origin)

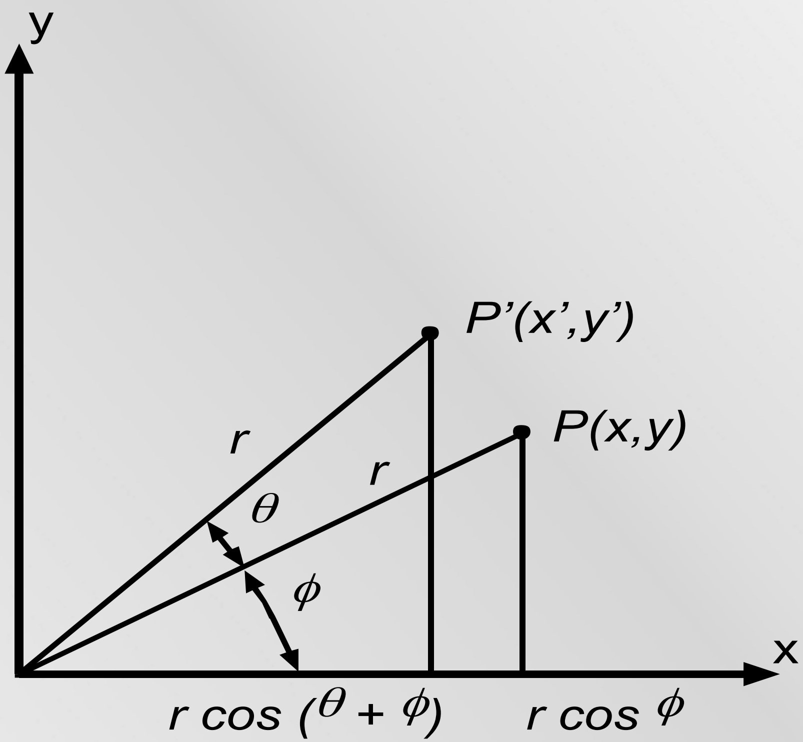


Before scaling

After scaling

2D Transformations

- **Rotation**
  - Change of an object **orientation**
  - Point P'(x' , y') is obtained from rotating point P(x , y) with angle θ with respect to the origin
  - ϕ is P(x , y) angular position with respect to the origin



2D Transformations

- **Rotation**
  - From the definition of **cos(a)** and **cos(a + b)** we know that:

$$x' = r \cdot \cos(\theta + \phi) = r \cdot \cos\phi \cdot \cos\theta - r \cdot sen\phi \cdot sen\theta \qquad x = r \cdot \cos\phi$$

$$y' = r \cdot sen(\theta + \phi) = r \cdot \cos\phi \cdot sen\theta + r \cdot sen\phi \cdot \cos\theta \qquad y = r \cdot \mathrm{sen}\,\phi$$

  - Equation formulas:

$$x' = x \cdot \cos\theta - y \cdot \mathrm{sen}\,\theta$$

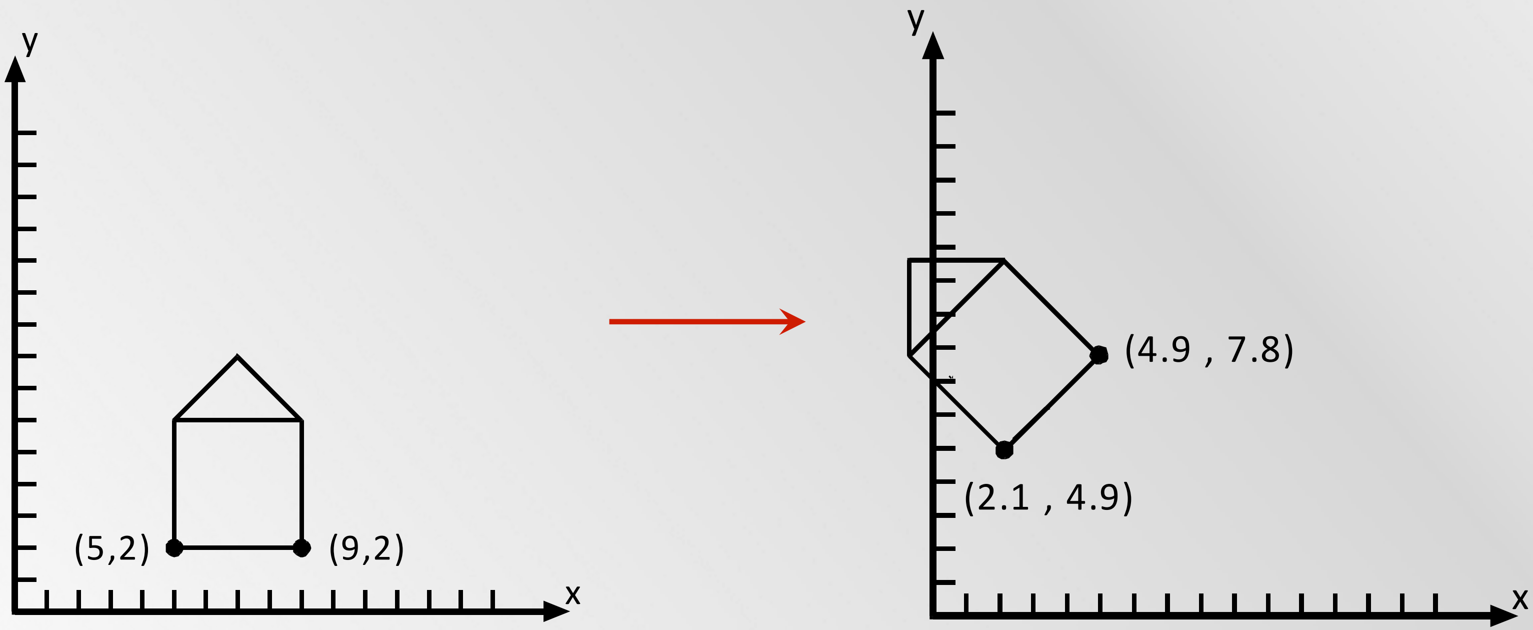
$$y' = x \cdot \mathrm{sen}\,\theta + y \cdot \cos\theta$$

  - Matrix representation:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\theta & -\mathrm{sen}\,\theta \\ \mathrm{sen}\,\theta & \cos\theta \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix}$$

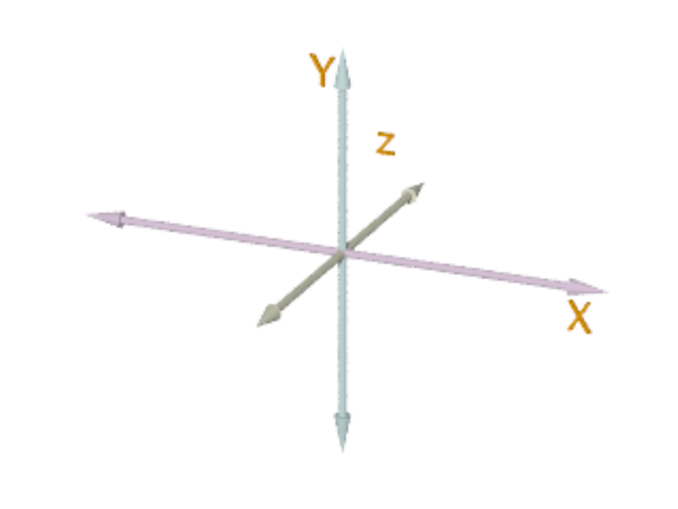
2D Transformations

- **Rotation**
  - Example – **rotate the object 45° with respect to the origin**



2D Transformations

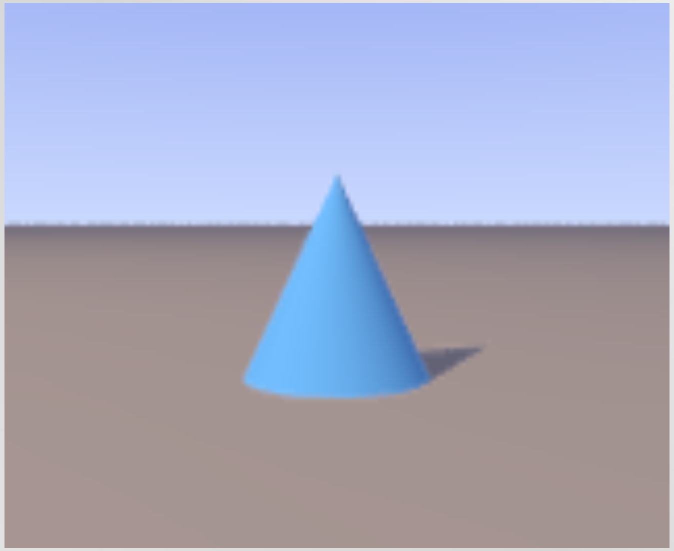
- **PovRay**
  - Coordinates system – **left hand rule**



2D Transformations

- PovRay
  - **Cone** with its base center at the origin and the top vertex at (0, 1, 0)

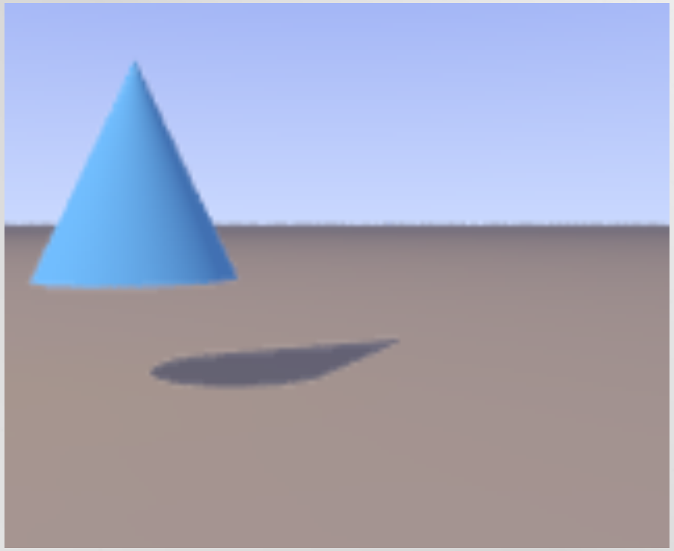
```
cone { <0, 0, 0> , 1.0 , <0, 1, 0> , 0.0
  pigment { color rgb < 0.2 , 0.6 , 0.9 > }
......................
}
```



2D Transformations

- PovRay
  - **Translation** of the cone along the vector <-2.0, 1.0, -0.5> direction

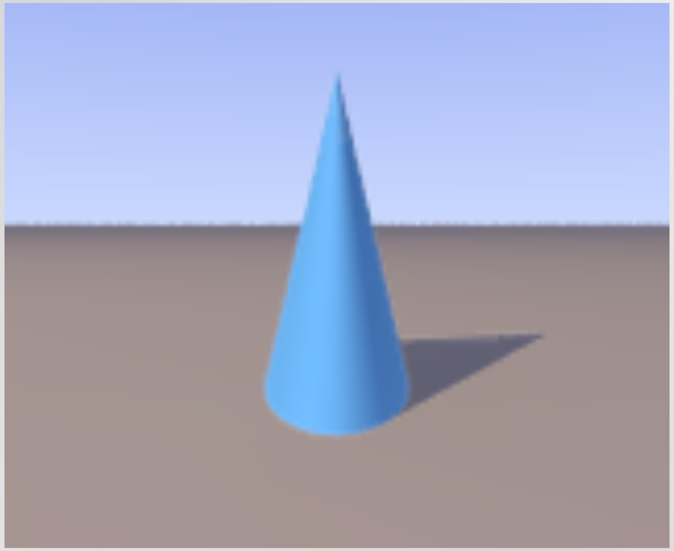
```
cone { <0, 0, 0> , 1.0 , <0, 1, 0> , 0.0
  pigment { color rgb < 0.2 , 0.6 , 0.9 > }
}
translate <-2.0, 1.0, -0.5>  //<dX,dY,dZ>
translate 1*y
```



2D Transformations

- PovRay
  - **Scaling** of the cone with a factor of **2.0*0.4** on the x axis, a factor of **2.0*1.0** on the y axis and a factor of **2.0*0.4** on the z axis

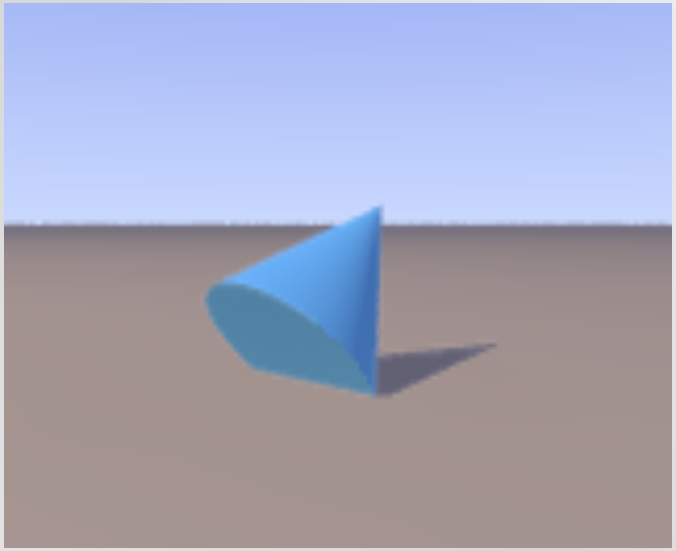
```
cone { <0, 0, 0> , 1.0 , <0, 1, 0> , 0.0
  pigment { color rgb < 0.2 , 0.6 , 0.9 > }
}
scale < 0.4, 1.0 ,  0.4 > // <dX, dY, dZ>
scale 2.0
```



2D Transformations

- PovRay
  - **Rotation of** the cone 2*30°, 2*0° and 2*(-45°) with respect to the x, y and z axis respectively

```
cone { <0, 0, 0> , 1.0 , <0, 1, 0> , 0.0
  pigment { color rgb < 0.2, 0.6 , 0.9> }
}
rotate <30, 0, -45> // <dX, dY, dZ>
rotate 2*x
```



2D Transformations

- Matrix representations reviewed so far:
  - **Translation** $P' = T + P$
  - **Scaling** $P' = S \cdot P$
  - **Rotation** $P' = R \cdot P$
- **Problem**
  - We want to more efficiently execute a sequence of transformations
  - We need an homogeneous reference system that is able to express every sequence of transformations as a matrix product

2D Homogeneous Transformations

- **Homogeneous coordinate system**
  - Each point **(x , y)** in the Cartesian system is represented by **(xW , yW , W)** in the homogeneous system, where **W ≠ 0** can be any real number
  - When **W ≠ 1**, in order to simplify calculations, all coordinates are divided by W to obtain **(x , y , 1)** (W is commonly used for the **z axis** since this is commonly used to represent **depth**)
  - When we set **W = 1**, we are actually performing transformations of the point (x , y) on the **z = 1 plane**
  - Considering that **W can be any real number**, (xW , yW , W) represents:
    - A point in the projective space
    - A projected line in the euclidean space

2D Homogeneous Transformations

- **Translation with homogeneous coordinates**
  - The **translation operator** is:

$$T(d_x, d_y) = \begin{bmatrix} 1 & 0 & d_x \\ 0 & 1 & d_y \\ 0 & 0 & 1 \end{bmatrix}$$

  - Equation **formula** and **matrix representation**

$$P' = T(d_x, d_y) \cdot P$$

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & d_x \\ 0 & 1 & d_y \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

2D Homogeneous Transformations

- **Translation compositions**
  - Lets suppose that a point P is translated by $T(d_{x1},d_{y1})$ to point P' and then translated by $T(d_{x2},d_{y2})$ to point P''

$$P' = T(d_{x1},d_{y1}) \cdot P, \qquad P'' = T(d_{x2},d_{y2}) \cdot P'$$

  - Composing them we obtain:

$$P'' = T(d_{x2},d_{y2}) \cdot \left(T(d_{x1},d_{y1}) \cdot P\right) = \left(T(d_{x2},d_{y2}) \cdot T(d_{x1},d_{y1})\right) \cdot P$$

2D Homogeneous Transformations

- **Translation compositions**
  - And the net translation operator is:

    $$T\,(\,d_{x1}+d_{x2}\,,\,d_{y1}+d_{y2}\,)$$

  - The **matrix representation** is:

$$\begin{bmatrix}1 & 0 & d_{x2}\\0 & 1 & d_{y2}\\0 & 0 & 1\end{bmatrix}\cdot\begin{bmatrix}1 & 0 & d_{x1}\\0 & 1 & d_{y1}\\0 & 0 & 1\end{bmatrix}=\begin{bmatrix}1 & 0 & d_{x1}+d_{x2}\\0 & 1 & d_{y1}+d_{y2}\\0 & 0 & 1\end{bmatrix}$$

2D Homogeneous Transformations

- **Scaling with homogeneous coordinates**
  - The **scaling operator** is:

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

  - The equation **formula** and **matrix representation**:

$$P' = S(s_x, s_y) \cdot P$$

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

2D Homogeneous Transformations

- **Scaling composition**
  - Lets suppose that a point P is scaled by $S(s_{x1},s_{y1})$ to the point P' and then it is scaled by $S(s_{x2},s_{y2})$ to the point P''

$$P' = S(s_{x1}, s_{y1}) \cdot P, \qquad P'' = S(s_{x2}, s_{y2}) \cdot P'$$

  - Substituting, we obtain

$$P'' = S(s_{x2}, s_{y2}) \cdot \left(S(s_{x1}, s_{y1}) \cdot P\right) = \left(S(s_{x2}, s_{y2}) \cdot S(s_{x1}, s_{y1})\right) \cdot P$$

2D Homogeneous Transformations

- **Scaling composition**
  - And the net scaling operator is:

    $$S\left(s_{x1}+s_{x2},\ s_{y1}+s_{y2}\right)$$

  - The **matrix representation** is:

$$\begin{bmatrix} s_{x2} & 0 & 0 \\ 0 & s_{y2} & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} s_{x1} & 0 & 0 \\ 0 & s_{y1} & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} s_{x1} \cdot s_{x2} & 0 & 0 \\ 0 & s_{y1} \cdot s_{y2} & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

2D Homogeneous Transformations

- **Rotation with homogeneous coordinates**
  - The **rotation operator** is:

$$R(\theta) = \begin{bmatrix} \cos\theta & -sen\theta & 0 \\ sen\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

  - The equation **formula** and **matrix representation** are:

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

$$P' = R(\theta) \cdot P$$

2D Homogeneous Transformations

- **Rotations composition**
  - The **rotation composition equation** is:

$$R(\theta_1)\cdot R(\theta_2) = R(\theta_1 + \theta_2).$$

2D Homogeneous Transformations

- **Inverse transformations**
  - A translation $\mathbf{T}(\mathbf{d_x}, \mathbf{d_y})$ inverse transformation is:

    $T^{-1}(d_x, d_y) = \mathbf{T}(-\mathbf{d_x}, -\mathbf{d_y})$

  - A scaling $\mathbf{S}(\mathbf{s_x}, \mathbf{s_y})$ inverse transformation is :

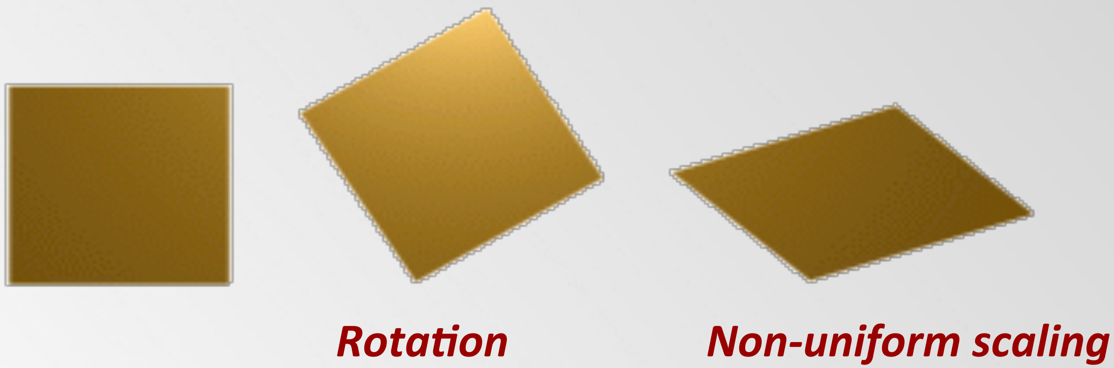
    $S^{-1}(s_x, s_y) = \mathbf{S}(1/\mathbf{s_x}, 1/\mathbf{s_y})$

  - A rotation $\mathbf{R}(\theta)$ inverse transformation is :

    $R^{-1}(\theta) = \mathbf{R}(-\theta)$

2D Homogeneous Transformations

- Affine transformations
  - Product of a sequence of rotation, translation and scaling matrices
  - Property of preserving parallelism (unlike length and angles). Subsequent rotations, scaling and translations could not make the parallelism go away



2D Homogeneous Transformations

- **“Stretching”**
  - It can be done with respect to both axis

Stretching with respect to y

Stretching with respect to x

  - Matrix for stretching with respect to both axis:

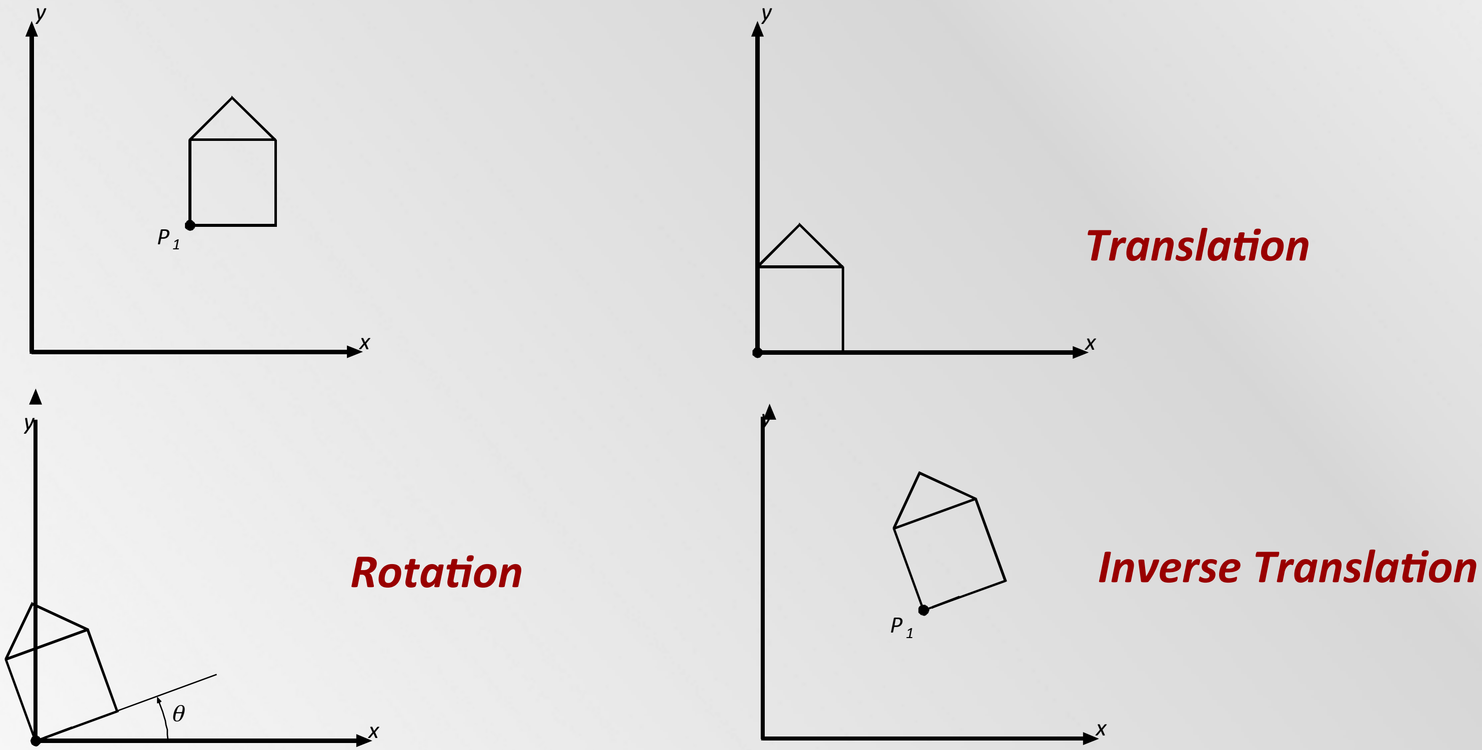
$$SH_x = \begin{bmatrix} 1 & a & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad SH_y = \begin{bmatrix} 1 & 0 & 0 \\ b & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

2D Homogeneous Transformations

- Transformations composition
  - **Objective** – gain efficiency through applying only one transformation to any point (instead of a sequence of those)
  - **Example – rotation of an object with respect to a point P_1**
    It can be divided in 3 steps
    1. Translate so that P_1 matches the origin
    2. Rotate with respect to the origin
    3. Translate so that the point in the origin comes back to P_1

2D Homogeneous Transformations

- **Transformations composition – rotation with respect to point $P_1(x_1,y_1)$**



2D Homogeneous Transformations

- **Rotation with respect to the point $P_1(x_1, y_1)$** – matrix expression
  - The net transformation equation is:

$$T(x_1, y_1)\cdot R(\theta)\cdot T(-x_1,-y_1) = \begin{bmatrix} 1 & 0 & x_1 \\ 0 & 1 & y_1 \\ 0 & 0 & 1 \end{bmatrix}\cdot\begin{bmatrix} \cos\theta & -\mathrm{sen}\,\theta & 0 \\ \mathrm{sen}\,\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}\cdot\begin{bmatrix} 1 & 0 & -x_1 \\ 0 & 1 & -y_1 \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} \cos\theta & -sen\theta & x_1(1-\cos\theta)+y_1 sen\theta \\ sen\theta & \cos\theta & y_1(1-\cos\theta)-x_1 sen\theta \\ 0 & 0 & 1 \end{bmatrix}$$

2D Homogeneous Transformations

- **Scaling and rotating an object with respect to point $P_1(x_1 , y_1)$:**
  - Translate P_1 to the origin
  - Scale and rotate
  - Translate from the origin to the new position (inverse translation)

- The **transformation equation** is:

$$T(x_2, y_2) \cdot R(\theta) \cdot S(s_x, s_y) \cdot T(-x_1, -y_1)$$

2D Homogeneous Transformations

- **Transformations composition – scaling with respect to the point P_1**
  - The **net transformation equation** is:

$$T(x_1, y_1)\cdot S(s_x, s_y)\cdot T(-x_1, -y_1) = \begin{bmatrix} 1 & 0 & x_1 \\ 0 & 1 & y_1 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & -x_1 \\ 0 & 1 & -y_1 \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} s_x & 0 & x_1(1-s_x) \\ 0 & s_y & y_1(1-s_y) \\ 0 & 0 & 1 \end{bmatrix}$$

2D Homogeneous Transformations

- **Transformation matrices product**
  - Matrices product is **usually non-commutative**
  - There are some **special cases** where matrices product is commutative, when applying geometric transformations:
    - Translation matrix multiplied by another translation matrix
    - Scaling matrix multiplied by another Scaling matrix
    - Rotation matrix multiplied by another Rotation matrix
    - Rotation matrix multiplied by a uniform scaling matrix ($s_x = s_y$)

3D Transformations

- **Homogeneous coordinates**
  - Each point **(x , y , z)** in the Cartesian system is represented by **(xW , yW , zW , W)** in the homogeneous system, where **W ≠ 0** can be any real number
  - When **W ≠ 1**, in order to simplify calculations, all coordinates are divided by W to obtain **(x , y , 1)** (W is commonly used for the **z axis** since this is commonly used to represent **depth**)
  - When we set **W = 1**, we are actually performing transformations of the point (x , y , z) on the **t = 1 3D-space** (t being the 4^{th} coordinate)
  - Considering that **W can be any real number**, **(xW , yW , zW , W)** represents:
    - A line in the projective space
    - A projected plane in the euclidean space

3D Transformations

- **Translation with homogeneous coordinates**

$$T(Dx, Dy, Dz) = \begin{vmatrix} 1 & 0 & 0 & Dx \\ 0 & 1 & 0 & Dy \\ 0 & 0 & 1 & Dz \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

  - Applying the translation to (x , y , z ,1), we obtain:

$$T(Dx, Dy, Dz) \cdot \begin{vmatrix} x \\ y \\ z \\ 1 \end{vmatrix} = \begin{vmatrix} x + Dx \\ y + Dy \\ z + Dz \\ 1 \end{vmatrix}$$

3D Transformations

- **Scaling with homogeneous coordinates**

$$S(Sx, Sy, Sz) = \begin{vmatrix} Sx & 0 & 0 & 0 \\ 0 & Sy & 0 & 0 \\ 0 & 0 & Sz & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

  - Applying the scaling to (x, y, z,1), we obtain:

$$S(Sx, Sy, Sz) \cdot \begin{vmatrix} x \\ y \\ z \\ 1 \end{vmatrix} = \begin{vmatrix} x \cdot Sx \\ y \cdot Sy \\ z \cdot Sz \\ 1 \end{vmatrix}$$

3D Transformations

- **Rotation with homogeneous coordinates**
  - With **respect to x**

$$Rx(\theta) = \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta & 0 \\ 0 & \sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

  - With **respect to y**

$$Ry(\theta) = \begin{vmatrix} \cos\theta & 0 & \sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

  - With **respect to z**

$$Rz(\theta) = \begin{vmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

3D Transformations

- **Rotation with homogeneous coordinates**
  - The **composition of an arbitrary number of rotations** with respect to axis x, y, z, results in

$$A = \begin{vmatrix} r11 & r12 & r13 & 0 \\ r21 & r22 & r23 & 0 \\ r31 & r32 & r33 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

  - The top left 3x3 sub matrix is an **orthogonal matrix**
    - Its columns are unitary orthogonal vectors. If they were not, we would observe distortion in the figure (it would not be an affine transformation)
    - Inverting and transposing an orthogonal matrix is the same thing

3D Transformations

- When an arbitrary number of rotation, scaling and translating matrices are multiplied, it always results in a matrix of the form

$$\begin{vmatrix} r11 \cdot s_x & r12 & r13 & tx \\ r21 & r22 \cdot s_y & r23 & ty \\ r31 & r32 & r33 \cdot s_z & tz \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

3D Transformations

- **PovRay – 3D transformation matrix**

```
cone { <0, 1, 0>, 0.0, <0, -1, 0>, 1.0
pigment { color rgb <0.2,0.6,0.9> }
matrix
<
0.886, -0.5, 0.5, //the first 3 lines form a rotation matrix
0, 1, 0,  // since it is not orthogonal, shearing occurs
0.5 , 0 , -0.886 ,
0, 0.8 , 0   // last 3 values contain the translation
 >
 }
```

3D Transformations

- **PovRay – 3D transformation matrix**