

# Geometric and Graphics Programming Laboratory: Lecture 8

Alberto Paoluzzi

October 20, 2017

## 1 2D Affine Transformations – (1)

- Translation
- Scaling
- Reflection

## 2 2D Affine transformations – (2)

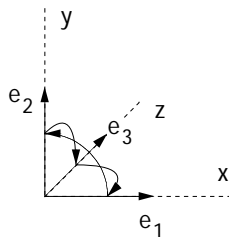
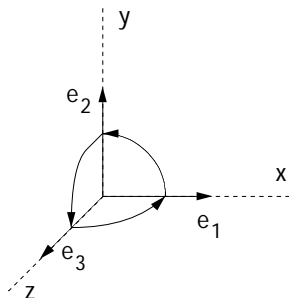
- Rotation
- Shearing
- General transformations
- Representation of tensors

# Introduction

- **Affine transformations** are used to map a figure or model into another of different size, position or orientation;
- they reduce to an invertible linear transformation by using **homogeneous coordinates**
- fixed a reference system, they are represented by squared invertible matrices, said **transformation matrices**
- we study the structure and properties of “**elementary**” **transformations** of 2D plane and 3D space.

# Assumptions

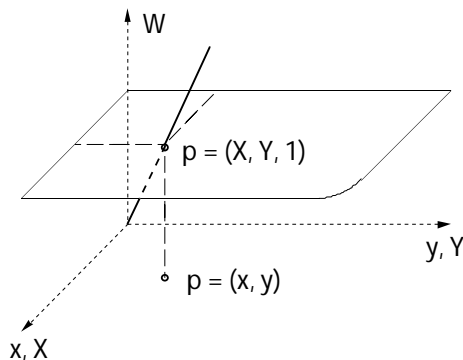
- vectors and points are represented as **column vectors**
- transformations are given by **left products** by a matrix
- the **reference frame** is assumed **left-handed**



**positive rotations:** (a) right-handed frame (b) left-handed frame

# Homogeneous coordinates

define a **bijective mapping** between the set of **points** of Cartesian plane and the set of **lines through the origin**  $\mathbf{o}$  of 3D space



Homogeneous coordinates of 2D plane

# Homogeneous coordinates

in such  $\mathbb{E}^2 \rightarrow \mathbb{E}^3$  mapping, every point  $(x, y)^T \in \mathbb{E}^2$  is represented as the set of points

$$\{(X, Y, W)^T \in \mathbb{E}^3 \mid x = X/W, y = Y/W, W \neq 0\}$$

to transform the homogeneous point  $\mathbf{p}' = (X, Y, W)$  into the Cartesian point  $\mathbf{p} = (x, y)$  two divisions by the homogeneous coordinate  $W$  are needed.

to avoid this computation we use the **homogeneous normalized representation**  $(X, Y, 1)^T$ , such that

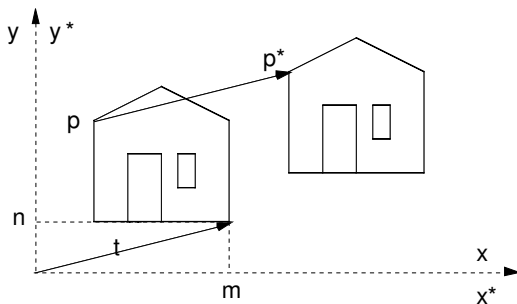
$$x = X, \quad y = Y$$

the point  $(x, y)^T$  of plane is represented by a vector  $\lambda(x, y, 1)^T$ , with  $\lambda \in \mathbb{R}$  e  $\lambda \neq 0$ .

# Translation

A **translation** of 2D plane is a function  $\mathbf{T} : \mathbb{E}^2 \rightarrow \mathbb{E}^2$ , where a fixed vector  $\mathbf{t} = (m, n)^T$  is summed to each point  $\mathbf{p} = (x, y)^T$ , so that

$$\mathbf{p}^* = \mathbf{T}(\mathbf{p}) = \mathbf{p} + \mathbf{t} = \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} m \\ n \end{pmatrix} = \begin{pmatrix} x + m \\ y + n \end{pmatrix}.$$



# Traslation

A movement of origin implies that the translation is not a linear transformation. Therefore, it cannot be represented in coordinates by a matrix

the translation is linear when using homogeneous coordinates. In fact, the translation that maps the  $\mathbf{p}$  point to

$$\mathbf{p}^* = \mathbf{p} + \mathbf{t},$$

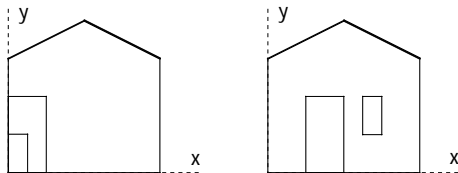
with  $\mathbf{t} = (m, n)^T$ , becomes, in homogeneous coordinates:

$$\mathbf{p}^* = \mathbf{T} \mathbf{p} = \begin{pmatrix} 1 & 0 & m \\ 0 & 1 & n \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} x + m \\ y + n \\ 1 \end{pmatrix}$$



# Translation

Higher-order functions need a double application over (a) integer specifiers and (b) real parameters, in order to generate the transformation tensor



write the code to do the above example

# Scaling

## definition

A **scaling**  $\mathbf{S}$  is a transformation tensor represented by a **diagonal matrix** with positive coefficients, so that:

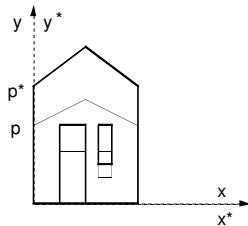
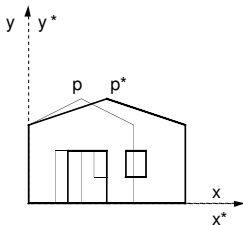
$$\mathbf{p}^* = \mathbf{S} \mathbf{p} = \begin{pmatrix} a & 0 \\ 0 & b \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} ax \\ by \end{pmatrix}, \quad a, b > 0$$

- if  $a, b > 1$ , then  $\mathbf{S}$  is a **dilatation** tensor
- if  $a = b = 1$ , then  $\mathbf{S}$  is the **identity** tensor
- if  $a, b < 1$ , then  $\mathbf{S}$  is a **compression** tensor

# Scaling

## elementary scalings

$$\mathbf{p}^* = \mathbf{S}_x \mathbf{p} = \begin{pmatrix} a & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} ax \\ y \end{pmatrix}$$



$$\mathbf{p}^* = \mathbf{S}_y \mathbf{p} = \begin{pmatrix} 1 & 0 \\ 0 & b \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x \\ by \end{pmatrix}$$

# Scaling

## homogeneous coordinates

the homogeneous normalized coordinate matrix  $\mathbf{S}' \in \mathbb{R}_3^3$  of a 2D scaling tensor may be easily derived from the non-homogeneous matrix  $\mathbf{S} \in \mathbb{R}_2^2$ , by adding a unit row and column:

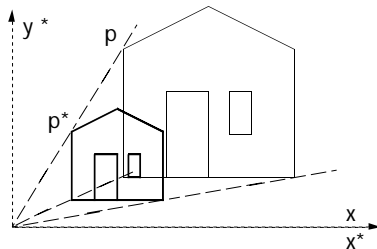
$$\mathbf{p}^* = \mathbf{S}'\mathbf{p} = \begin{pmatrix} \mathbf{S} & \mathbf{0} \\ \mathbf{0} & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} ax \\ by \\ 1 \end{pmatrix}.$$

# Scaling

## uniform scaling

When  $a = b$  the scaling is said **uniform** or **homothetic** transformation

- ① with  $a = b = 0.5$  the length of all segments is halved
- ② the image  $\mathbf{p}^*$  of each  $\mathbf{p}$  goes on the line through  $\mathbf{p}$  and the origin
- ③ the transformed figure is also **closer** to the origin



action of a tensor of uniform scaling

# Reflection

## definition

Linear transformation defined by a matrix that differs from the identity since one of diagonal coefficients is  $-1$

Two elementary reflections  $\mathbf{M}_x$  e  $\mathbf{M}_y$  may be defined in the plane  $\mathbb{E}^2$

$$\mathbf{M}_x = \begin{pmatrix} -1 & 0 \\ 0 & 1 \end{pmatrix}, \quad \mathbf{M}_y = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

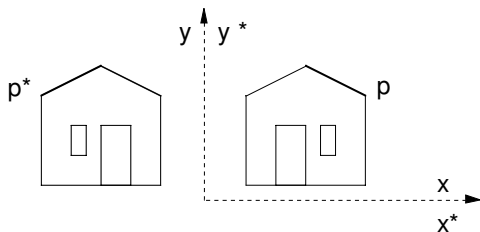
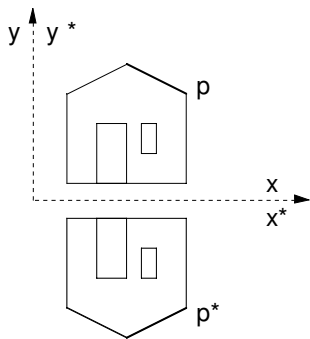
The action of a reflection tensor inverts the sign of one of coordinates of points

# Reflection

## homogeneous representation

As usual, the normalized homogeneous representation of such transformations is obtained by adding a unit row and column to their matrices

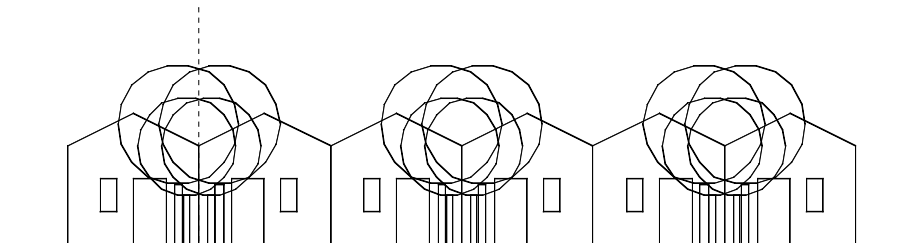
$$\mathbf{M}'_x = \begin{pmatrix} \mathbf{M}_x & \mathbf{0} \\ \mathbf{0} & 1 \end{pmatrix}, \quad \mathbf{M}'_y = \begin{pmatrix} \mathbf{M}_y & \mathbf{0} \\ \mathbf{0} & 1 \end{pmatrix}$$



# Reflection

## example

Let us continue the house example by adding **simmetry** to the scene





# Elementary rotation of plane

An **elementary rotation** of 2D plane is a linear function that maps every point  $\mathbf{p} \in \mathbb{E}^2$  to the second extreme  $\mathbf{p}^* = \mathbf{R}(\mathbf{p})$  of a circle arc with first extreme in  $\mathbf{p}$ , center in the origin and constant angle  $\alpha$

The matrix of a rotation tensor is easily computed by considering the images of basis vectors ( $\mathbf{e}_i$ )

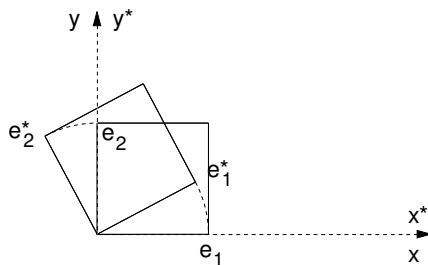
$$\begin{pmatrix} \mathbf{e}_1^* & \mathbf{e}_2^* \end{pmatrix} = \mathbf{R} \begin{pmatrix} \mathbf{e}_1 & \mathbf{e}_2 \end{pmatrix}.$$

where  $\mathbf{R}$  is the unknown rotation matrix

# Elementary rotation of plane

more explicitly:

$$\begin{pmatrix} \cos \alpha & -\sin \alpha \\ \sin \alpha & \cos \alpha \end{pmatrix} = \mathbf{R} \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix},$$



# Elementary rotation of plane

in homogeneous coordinates

The normalized homogeneous matrix  $\mathbf{R}' \in \text{lin } \mathbb{R}^3$  of a plane rotation is obtained from the non-homogeneous matrix  $\mathbf{R} \in \text{lin } \mathbb{R}^2$

$$\mathbf{p}^* = \mathbf{R}'\mathbf{p} = \begin{pmatrix} \mathbf{R} & \mathbf{0} \\ \mathbf{0} & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} x \cos \alpha + y \sin \alpha \\ -x \sin \alpha + y \cos \alpha \\ 1 \end{pmatrix}$$

in the usual way, by adding a unit row and column ...

# Examples

## Elementary transformations in pyplasm

```
from larlib import *

filename = "test/svg/inters/house.svg"
lines = svg2lines(filename)
lines

VIEW(STRUCT(AA(POLYLINE)(lines)))
AA(POLYLINE)(lines)
house = STRUCT(AA(POLYLINE)(lines))
house

VIEW(STRUCT([house, T([1,2])([2,0.5])(house)]))
VIEW(STRUCT([house, S([1,2])([2,2])(house)]))
VIEW(STRUCT([house, S(1)(2)(house)]))
VIEW(STRUCT([house, S(1)(-1)(house)]))
VIEW(STRUCT([house, R([1,2])(PI/6)(house)]))
```

# Shearing

## elementary

The plane is seen as a **bundle of lines** parallel to a coordinate axis

A 2D **elementary shearing** is a tensor which maps the points of a line in other points of the same line, in a way such that:

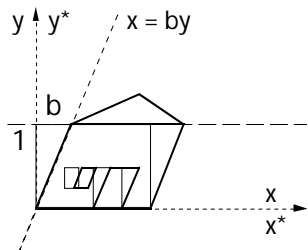
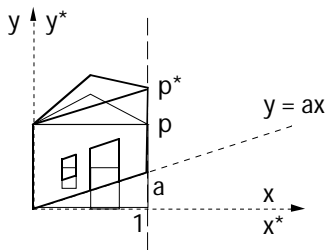
- 1 all points of a line translate by the same vector
- 2 only the coordinate axis parallel to the line bundle remains fixed
- 3 the translation of each line is proportional to its distance to the fixed line

# Shearing

An elementary shearing tensor **does not change** one coordinate, whereas the other changes linearly with the value of the fixed coordinate

$$\mathbf{p}^* = \mathbf{H}_x \mathbf{p} = \begin{pmatrix} 1 & 0 \\ a & 1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x \\ y + ax \end{pmatrix},$$

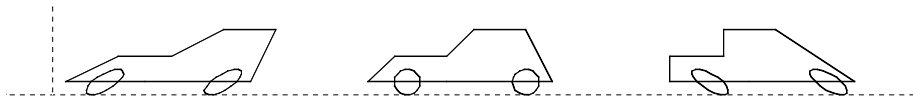
$$\mathbf{p}^* = \mathbf{H}_y \mathbf{p} = \begin{pmatrix} 1 & b \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x + by \\ y \end{pmatrix}.$$



Action of  $\mathbf{H}_x$ , normal to the  $x$  axis, and  $\mathbf{H}_y$ , normal to the  $y$  axis

# Shearing

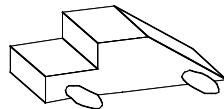
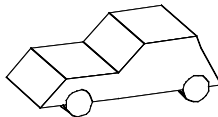
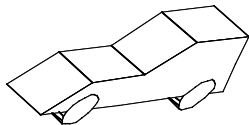
## example



# Shearing

## example

Three **keyframes** of the **storyboard** of 3D animation entitled: “My wife’s car”





# Arbitrary linear transformation

Let consider the action of a general **Q** tensor on the unit square built on the basis of the Cartesian frame  $(\mathbf{o}, \mathbf{e}_i)$ , with

$$\mathbf{Q} = \begin{pmatrix} a & c \\ b & d \end{pmatrix}.$$

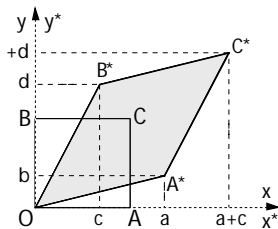
arbitrary, but **invertible matrix**

such arbitrary linear transformation:

- 1 does not move the origin;
- 2 maps parallel lines to parallel lines;
- 3 does'nt conserve, in general, the size of areas.

# General transformation

action of a general tensor on the unit standard square



$$\begin{pmatrix} \mathbf{o}^* & \mathbf{a}^* & \mathbf{b}^* & \mathbf{c}^* \end{pmatrix} = \mathbf{Q} \begin{pmatrix} \mathbf{o} & \mathbf{a} & \mathbf{b} & \mathbf{c} \end{pmatrix},$$

or, by using the corresponding coordinates:

$$\begin{pmatrix} 0 & a & c & a+c \\ 0 & b & d & b+d \end{pmatrix} = \begin{pmatrix} a & c \\ b & d \end{pmatrix} \begin{pmatrix} 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{pmatrix}.$$

# Transformation with fixed point

different from the origin

Every invertible linear transformation  $\mathbf{Q}$  has the origin  $\mathbf{o}$  of the Cartesian frame as its unique **fixed point**, i.e.  $\mathbf{Q}(\mathbf{o}) = \mathbf{o}$

To have a **fixed point  $\mathbf{q}$  different from origin** we must compose three transformations, such that:

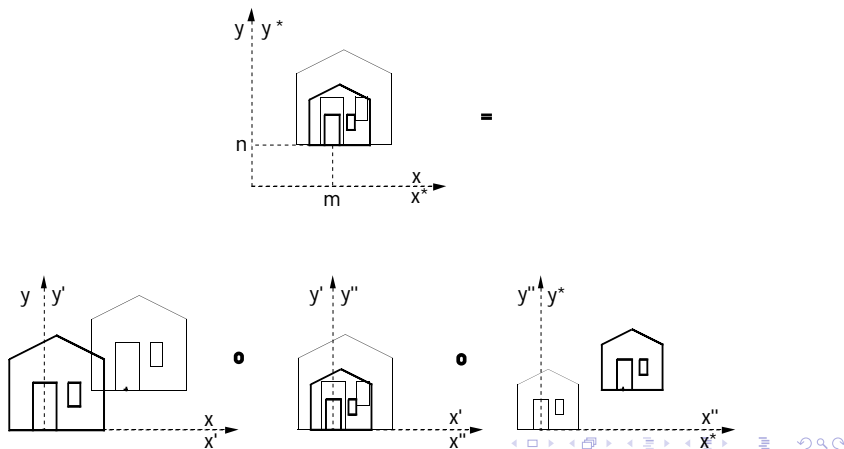
- 1 map  $\mathbf{q}$  to the origin  $\mathbf{o}$ ;
- 2 apply the required transformation;
- 3 map back  $\mathbf{o}$  to  $\mathbf{q}$ .

# Transformation with fixed point

## scaling

Let consider a **scaling tensor** with fixed point  $\mathbf{q} = (m, n)^T \neq \mathbf{o}$ :

$$\mathbf{S}_{\mathbf{q}}(m, n, a, b) = \mathbf{T}_{xy}(m, n) \circ \mathbf{S}_{xy}(a, b) \circ \mathbf{T}_{xy}(-m, -n).$$

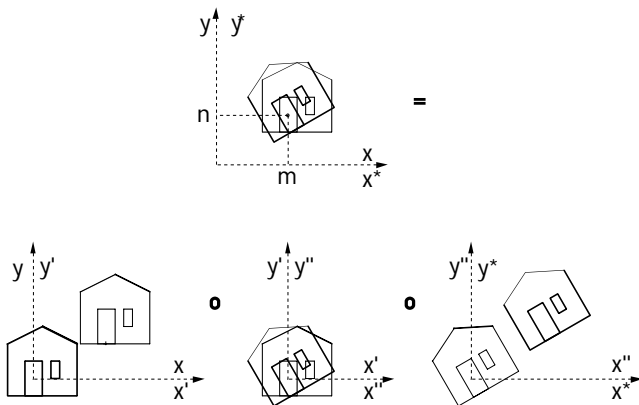


# Transformation with fixed point

## rotation

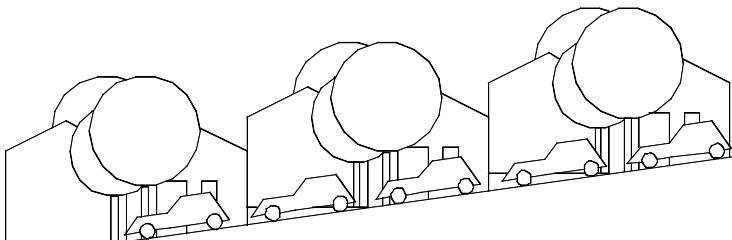
Let consider a **rotation tensor** with fixed point  $\mathbf{q} = (m, n)^T \neq \mathbf{o}$ :

$$\mathbf{R}_{\mathbf{q}}(m, n, \alpha) = \mathbf{T}_{xy}(m, n) \circ \mathbf{R}_{xy}(\alpha) \circ \mathbf{T}_{xy}(-m, -n).$$



# 2D Affine transformations

## example



### Remark (Assignment)

Convert the example on pages 230-231 (chapter 6) of book [GP4CAD](#) from classic PLaSM (FL style) to *pyplasm*

# Representation of tensors

Tensors are represented in PLaSM by applying the predefined function `MAT` to the tensor matrix (list of lists of coordinates)

$$\text{MAT} : \mathbb{R}_3^3 \rightarrow \text{lin } \mathbb{R}^3$$

Tensors, defined as linear endomorphisms of a vector space, have first-grade citizenship in PLaSM, and can be composed to generate new tensors. For example:

Tensors can be applied to polyhedral complexes of arbitrary dimensions  $(d, n)$

# Representation of tensors

## example

```
from pyplasm import *  
  
wall = MKPOL([ [[0,0],[4,0],[4,4],[2,6],[0,4]],  
               [[1,2,3,4,5]],None ])  
Q = MAT([[1,0,0],[0,1,0.5],[0,0,1]])  
  
VIEW(Q(wall))
```

Remember that:

- we use homogeneous coordinates (2D matrices are  $3 \times 3$ );
- in PlaSM the homogeneous coordinate is the first