Geometric Transformations

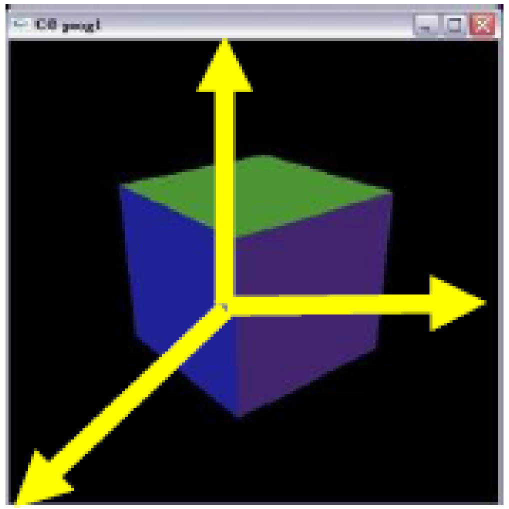
Hakan Bilen
University of Edinburgh

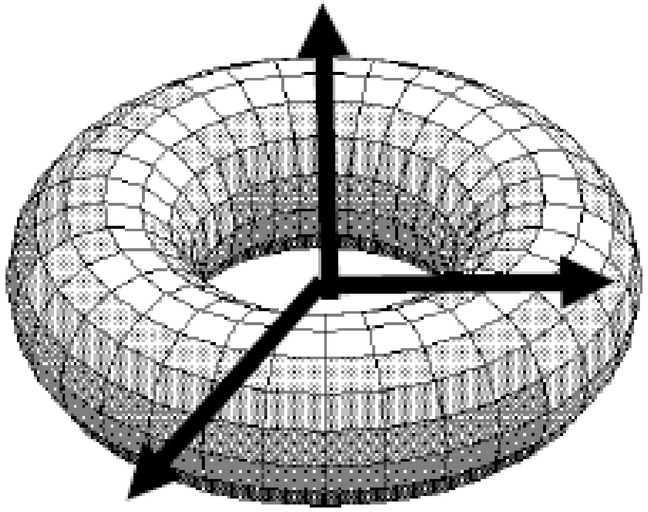
Computer Graphics
Fall 2017

PIAZZA!

Setting Objects in the Scene

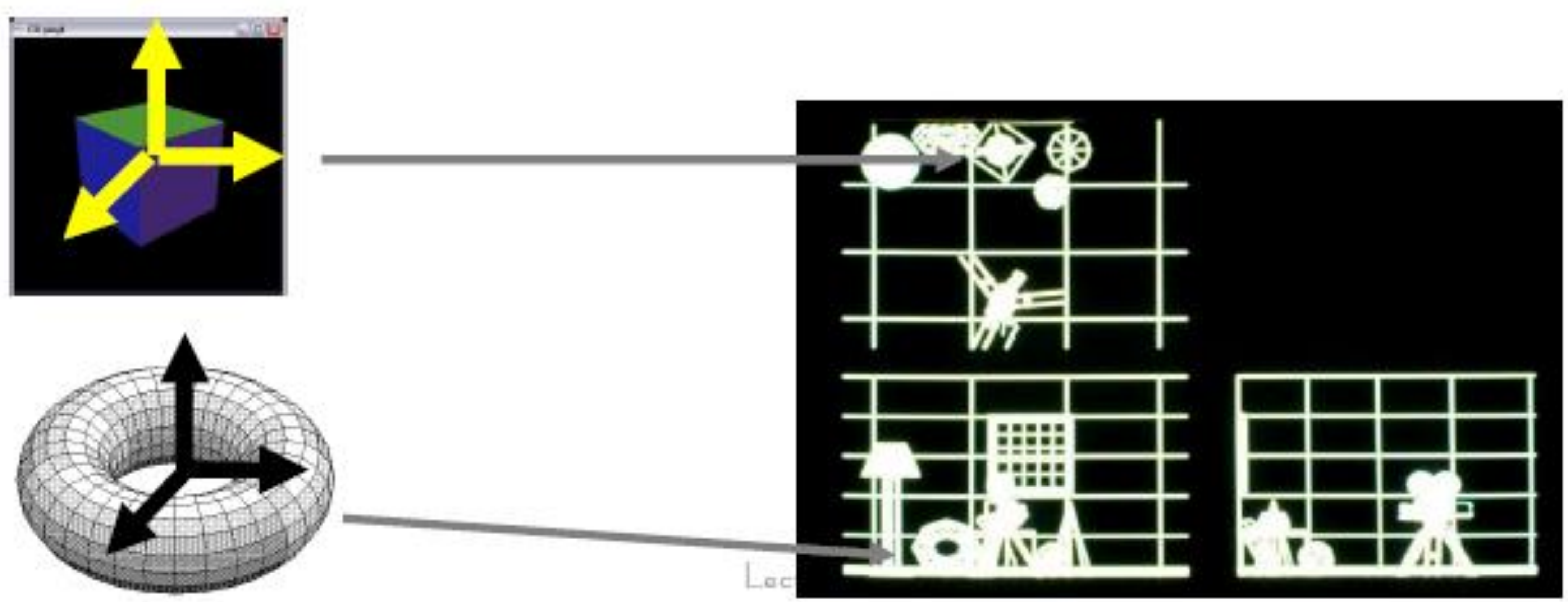
- Once the models are prepared, we need to place them in the environment
- We need to know the vertex locations of the objects in the world coordinate system
- But objects are only defined in their own local coordinate system





Transformations

We translate, rotate and scale the vertices in the world coordinate system



Today

- 2D geometric transformations
- Homogeneous coordinates
- Change of coordinates
- 3D cases

Some notation

- Scalars are in Greek letters α, β, γ etc.
- Vectors are in bold and lowercase $\boldsymbol{u} = \begin{pmatrix} u \\ v \end{pmatrix}$
- Matrices are in bold and UPPERCASE $\mathbf{M} = \begin{pmatrix} m_{xx} & m_{xy} \\ m_{yx} & m_{yy} \end{pmatrix}$

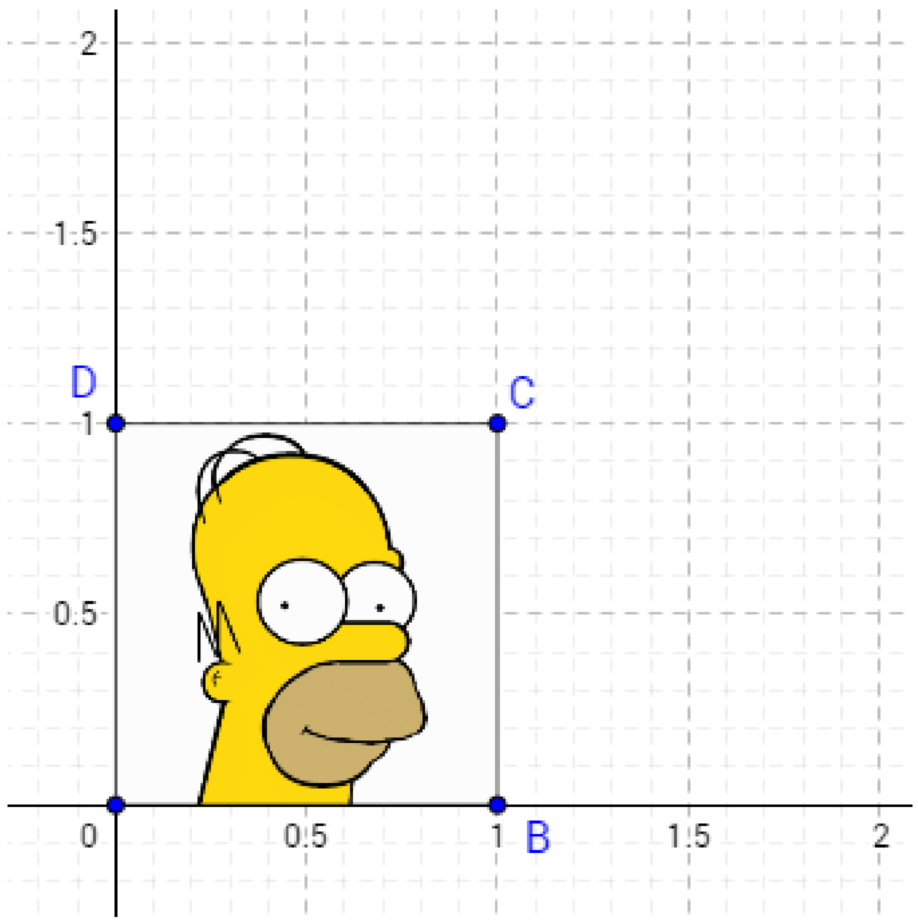
2D Linear Transformations

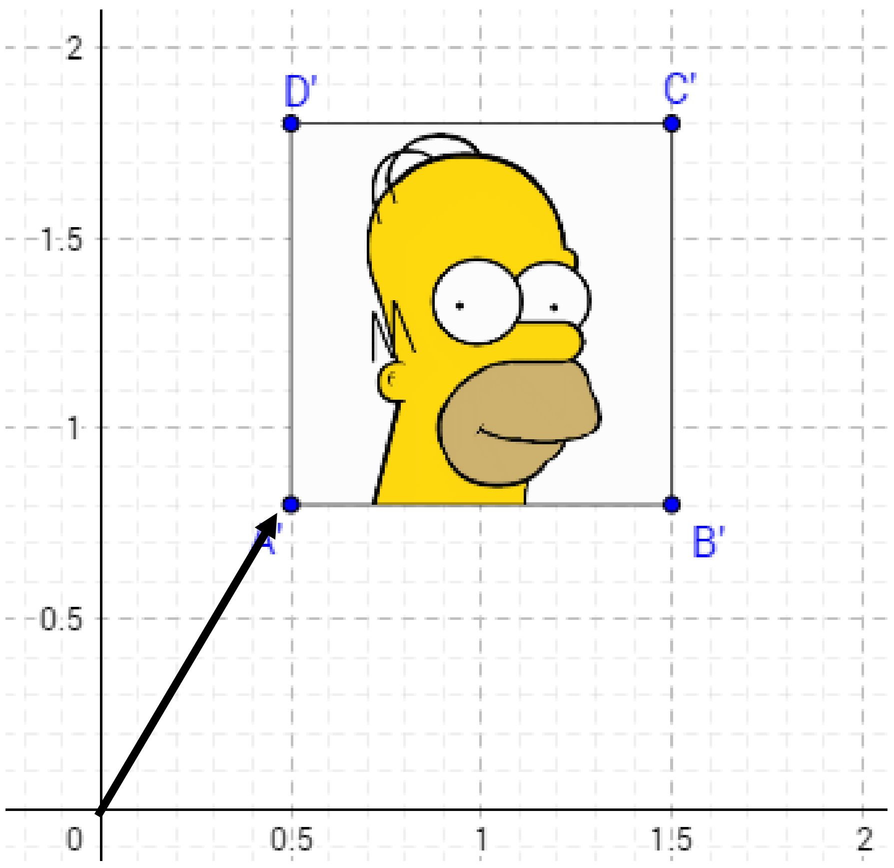
Translation

- Simplest transformation: $\mathbf{x}' = \mathbf{x} + \boldsymbol{t}$

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

- Inverse: $\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x' \\ y' \end{pmatrix} - \begin{pmatrix} t_x \\ t_y \end{pmatrix}$





Linear Transformations

- A more general family of geometric transformations

$$\boldsymbol{x}' = \boldsymbol{M}\boldsymbol{x}, \qquad \begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} m_{xx} & m_{xy} \\ m_{yx} & m_{yy} \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$$

- Why is it linear?

$$\boldsymbol{M}(\alpha\boldsymbol{x} + \boldsymbol{y}) = \alpha\boldsymbol{M}\boldsymbol{x} + \boldsymbol{M}\boldsymbol{y}$$

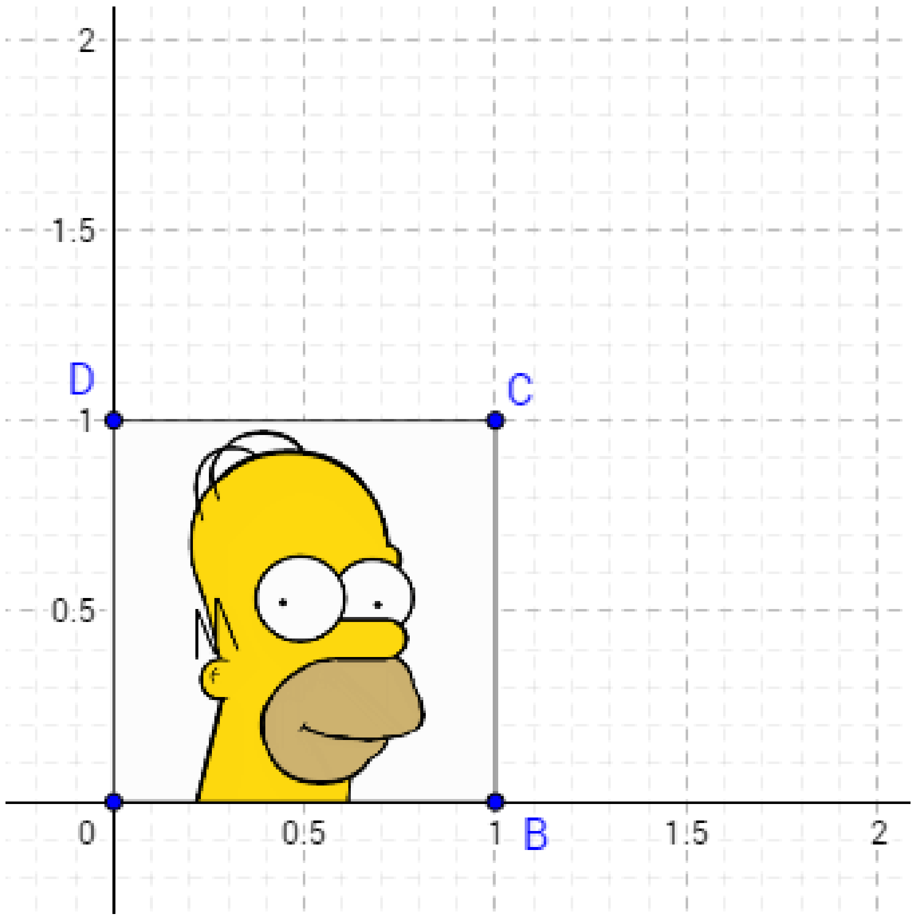
- What can we do with such transformations?
  - Uniform scale
  - Non-uniform scale
  - Rotation
  - Shear
  - Reflection

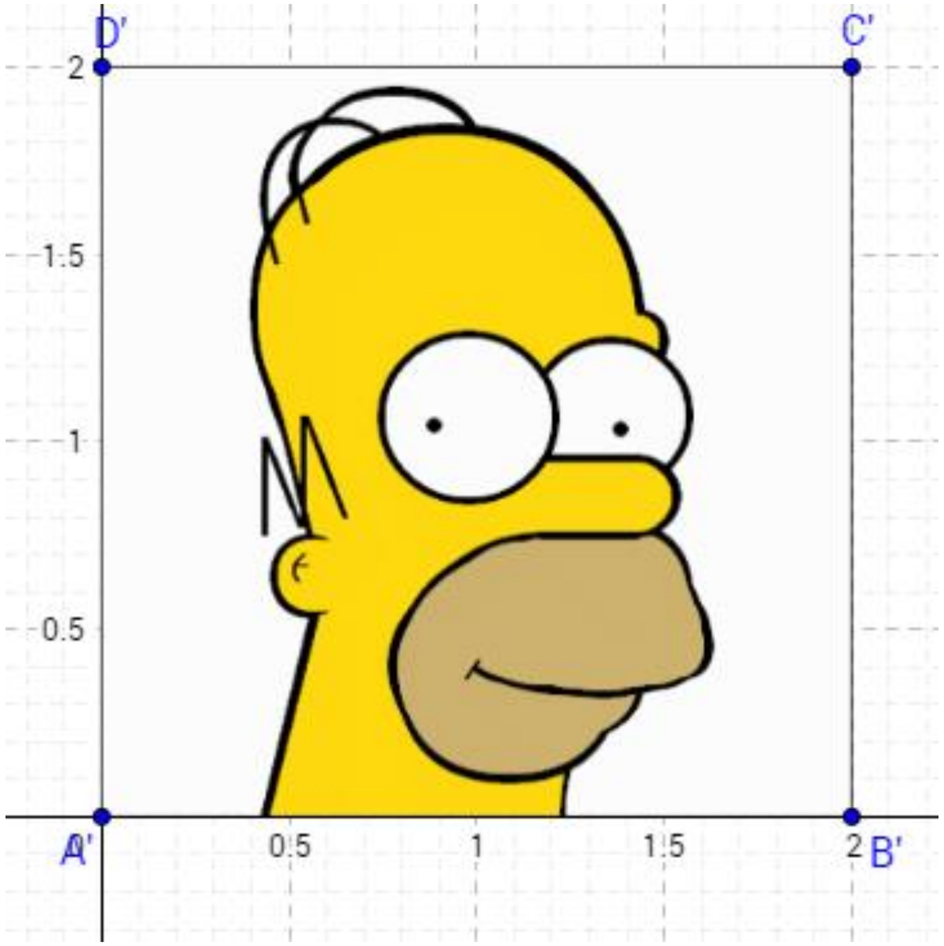
2D Linear Transformations

Uniform Scale

- $\begin{pmatrix} s & 0 \\ 0 & s \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} sx \\ sy \end{pmatrix}$

$$\begin{pmatrix} 2 & 0 \\ 0 & 2 \end{pmatrix}$$



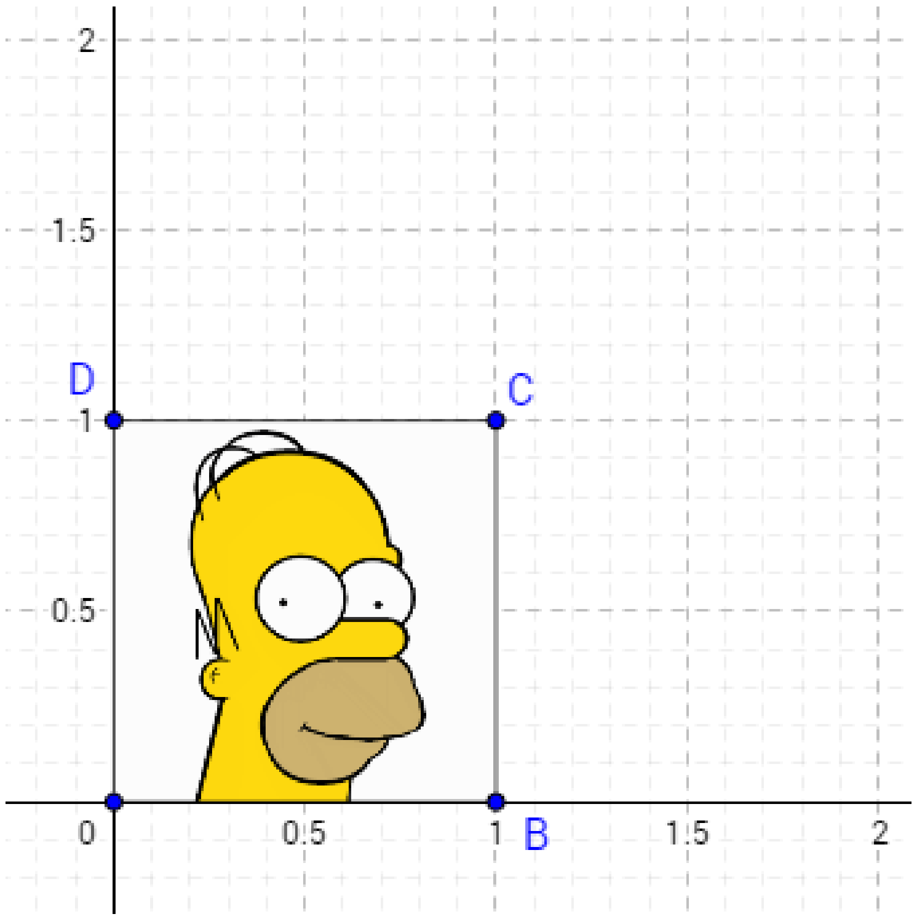


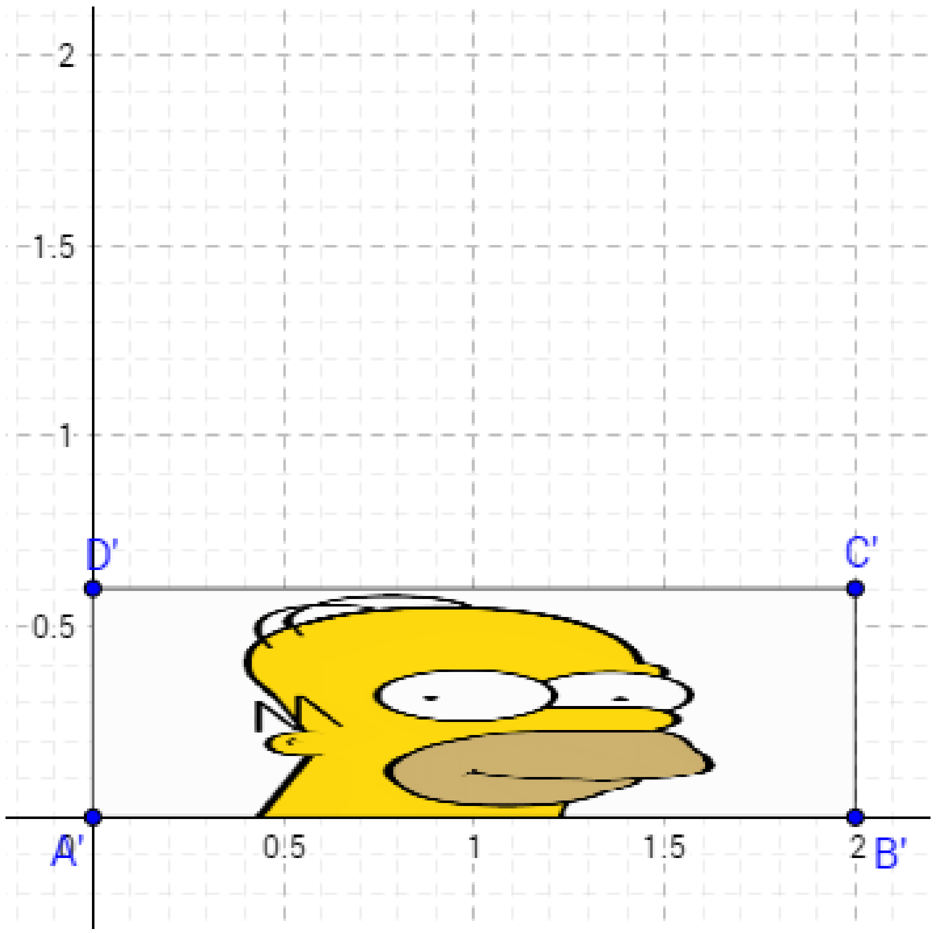
2D Linear Transformations

Non-uniform Scale

- $$\begin{pmatrix} s_x & 0 \\ 0 & s_y \end{pmatrix}\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} s_x x \\ s_y y \end{pmatrix}$$

$$\begin{pmatrix} 2 & 0 \\ 0 & 0.6 \end{pmatrix}$$



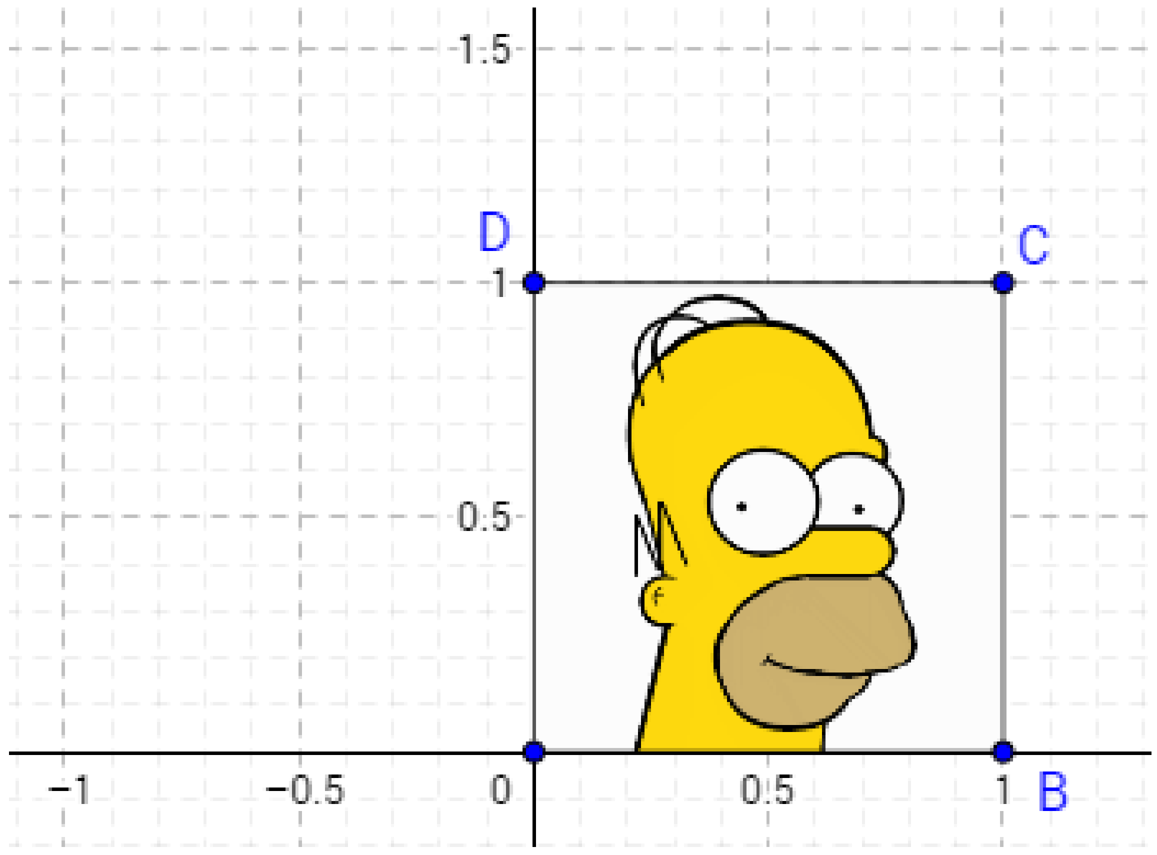


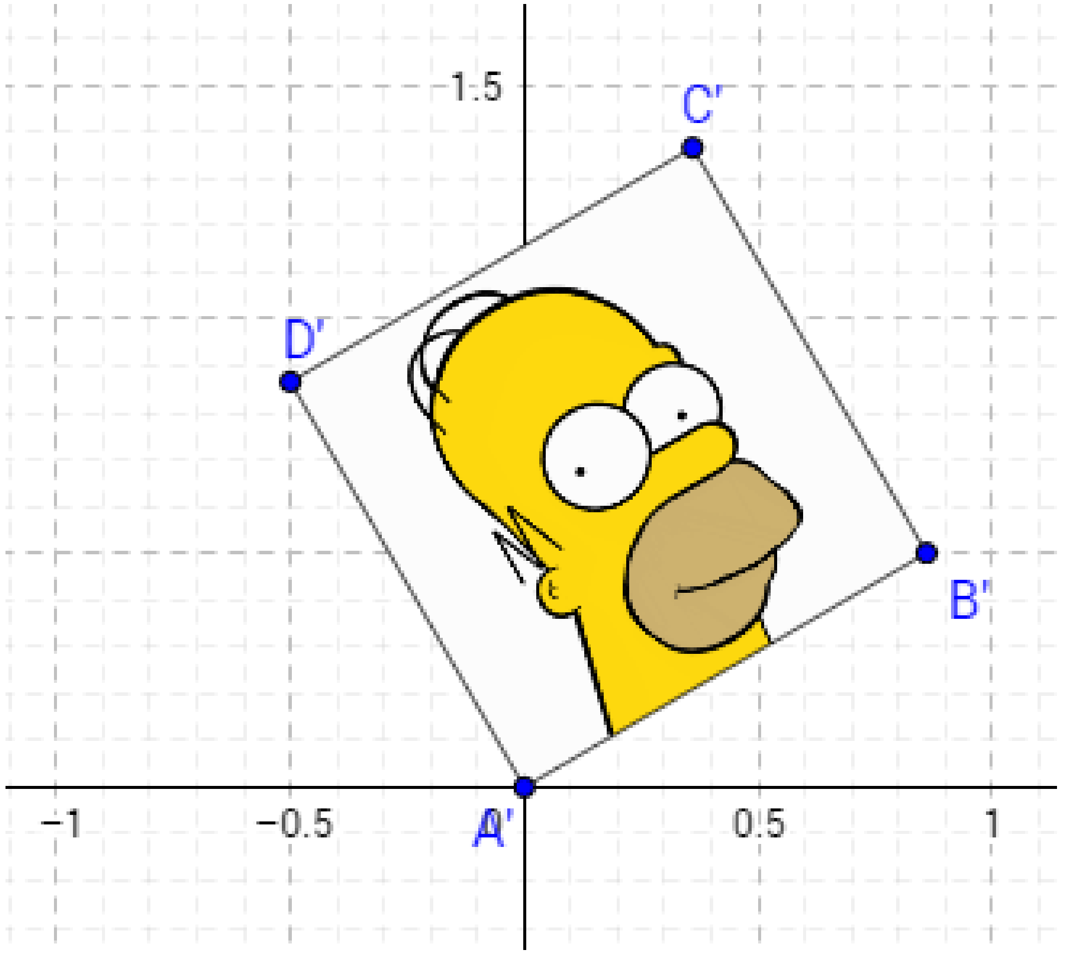
2D Linear Transformations

Rotation

- $\begin{pmatrix} cos\theta & -sin\theta \\ sin\theta & cos\theta \end{pmatrix}\begin{pmatrix} x \\ y \end{pmatrix}$

$$\begin{pmatrix} 0.87 & -0.5 \\ 0.5 & 0.87 \end{pmatrix}$$

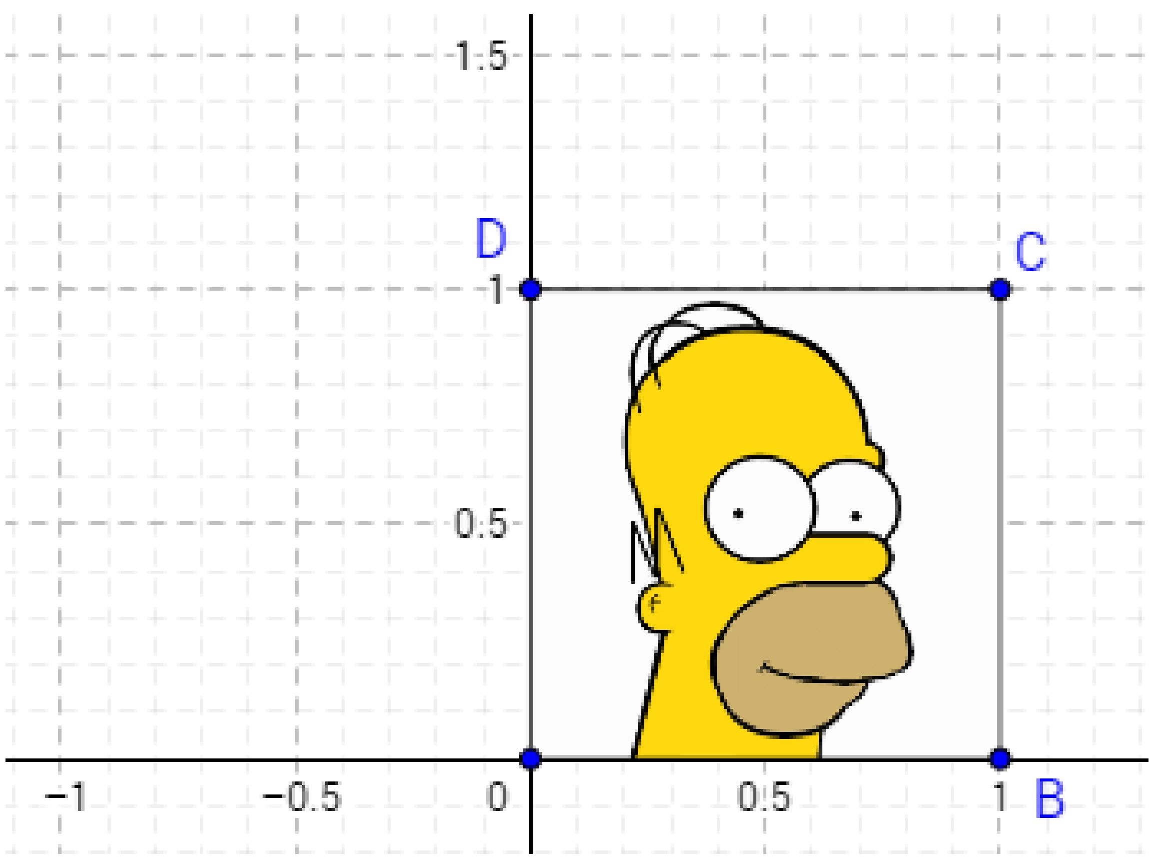


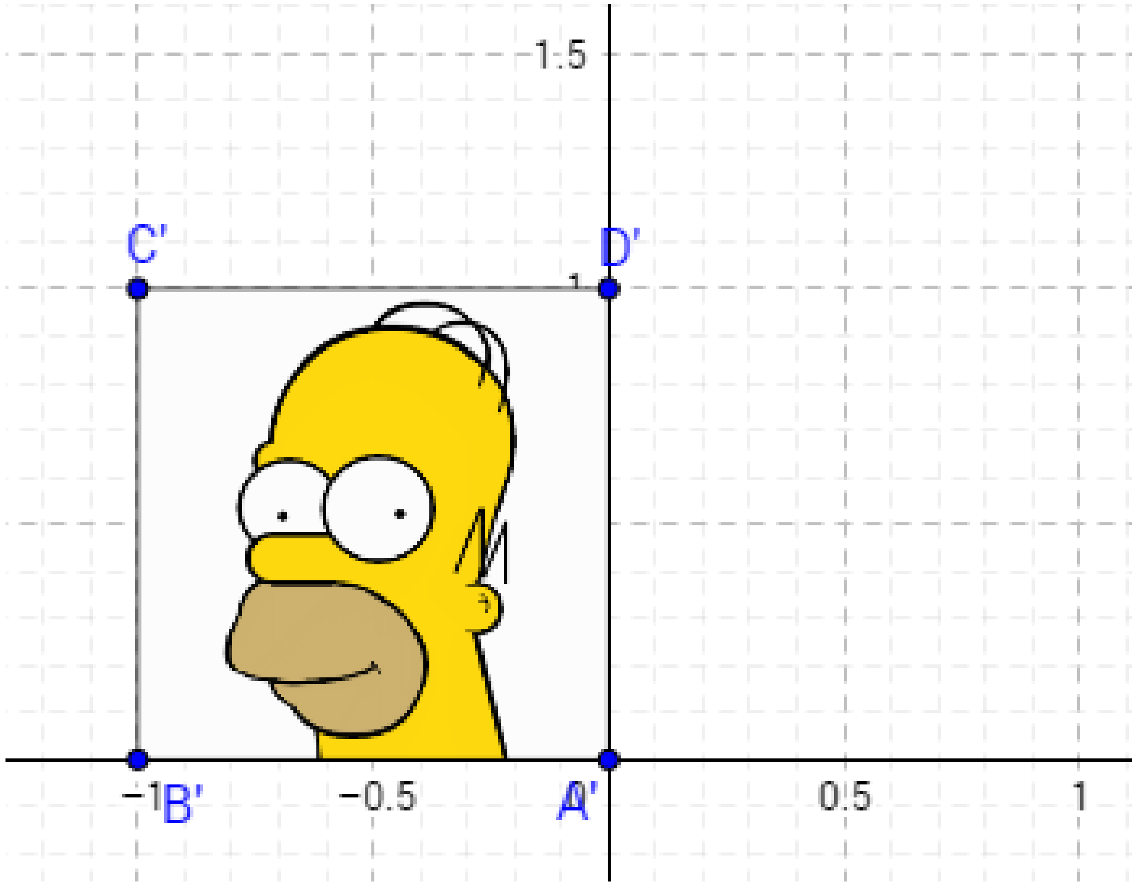


2D Linear Transformations

Reflection

- $\begin{pmatrix} -1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} -x \\ y \end{pmatrix}$



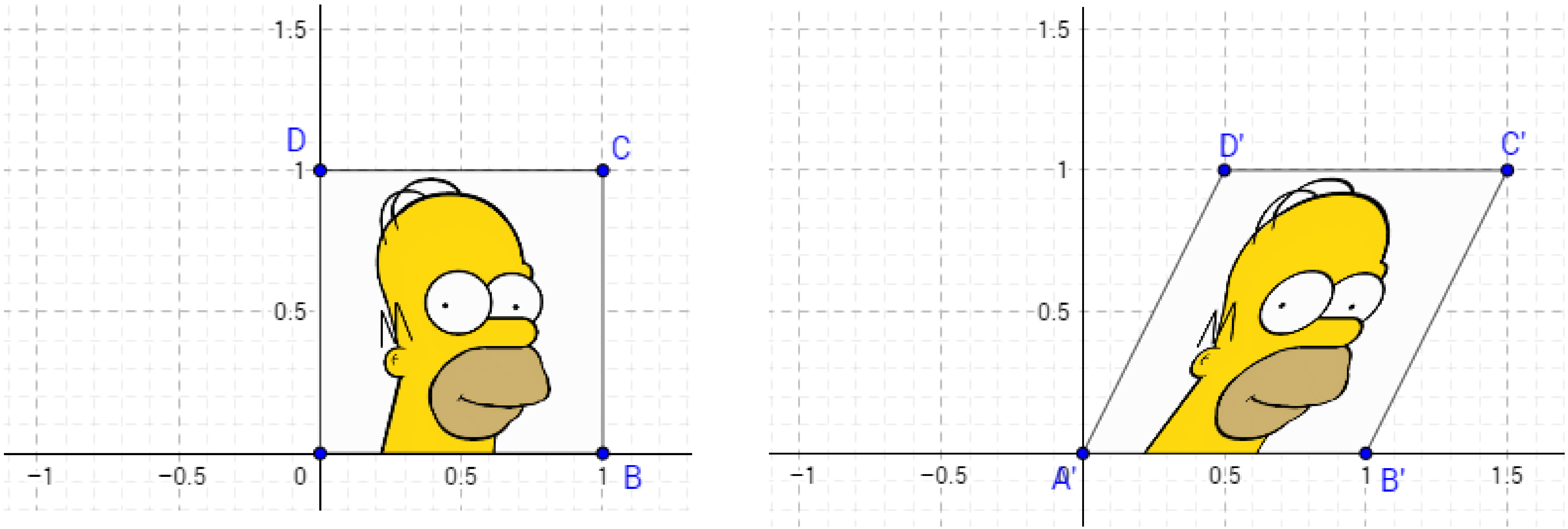


2D Linear Transformations

Shear

- $\begin{pmatrix} 1 & \alpha \\ 0 & 1 \end{pmatrix}\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x + \alpha y \\ y \end{pmatrix}$

$$\begin{pmatrix} 1 & 0.5 \\ 0 & 1 \end{pmatrix}$$



2D Linear Transformations

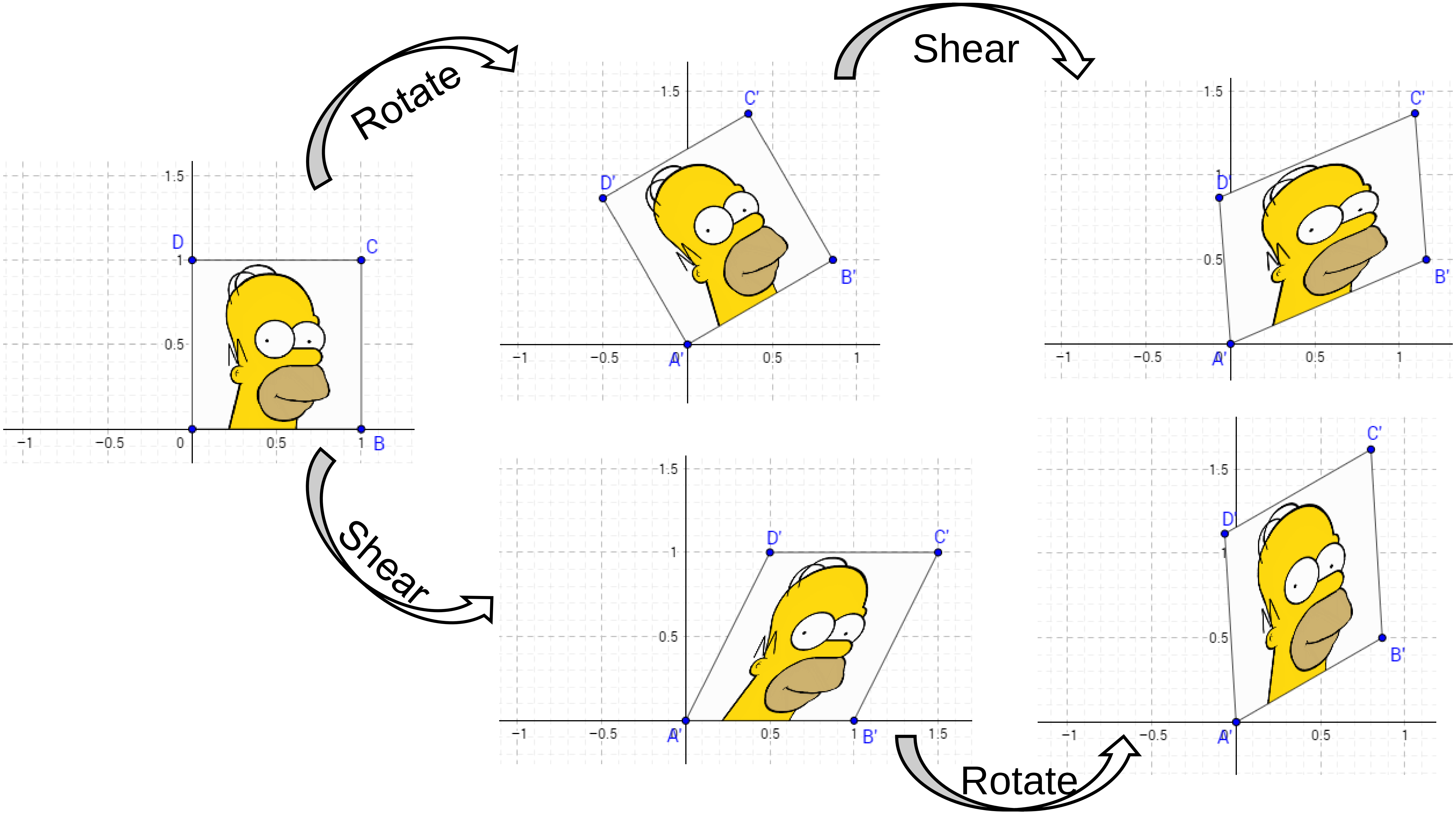
Composition

- You moved an object but you didn't like the position, then you move it again
- Can we do it at once?
- Translation:
  - $\boldsymbol{x}' = \boldsymbol{x} + \boldsymbol{t}$ and then $\boldsymbol{x}'' = \boldsymbol{x}' + \boldsymbol{v}$
  - Easy to compose $\boldsymbol{x}'' = \boldsymbol{x} + \boldsymbol{t} + \boldsymbol{v}$= $\boldsymbol{x} + \boldsymbol{v} + \boldsymbol{t}$
- 2D linear transformation
  - $\boldsymbol{x}' = \boldsymbol{M}\boldsymbol{x}$ and then $\boldsymbol{x}'' = \boldsymbol{P}\boldsymbol{x}'$
  - How do we compose?
  - $\boldsymbol{x}'' = \boldsymbol{P}\boldsymbol{M}\boldsymbol{x}$?
  - $\boldsymbol{x}'' = \boldsymbol{M}\boldsymbol{P}\boldsymbol{x}$?
  - $\boldsymbol{P}\boldsymbol{M}\boldsymbol{x}$ =? $\mathbf{M}\boldsymbol{P}\boldsymbol{x}$

2D Linear Transformations

Composition

Rotate + Shear =? Shear + Rotate



Combining linear with translation

- $\boldsymbol{x}_{\boldsymbol{S}}' = \boldsymbol{M}_{\boldsymbol{S}}\boldsymbol{x}_{\boldsymbol{S}} + \boldsymbol{t}_{\boldsymbol{s}}$ and then $\boldsymbol{x}'_{\boldsymbol{P}} = \boldsymbol{M}_{\boldsymbol{P}}\boldsymbol{x}_{\boldsymbol{S}}' + \boldsymbol{t}_{\boldsymbol{p}}$
- Let's compose it

$$\boldsymbol{x}'_{\boldsymbol{P}} = \boldsymbol{M}_{\boldsymbol{P}}\boldsymbol{x}_{\boldsymbol{S}}' + \boldsymbol{t}_{\boldsymbol{p}}$$
$$\boldsymbol{x}'_{\boldsymbol{P}} = \boldsymbol{M}_{\boldsymbol{P}}(\boldsymbol{M}_{\boldsymbol{S}}\boldsymbol{x}_{\boldsymbol{S}} + \boldsymbol{t}_{\boldsymbol{s}}) + \boldsymbol{t}_{\boldsymbol{p}}$$
$$\boldsymbol{x}'_{\boldsymbol{P}} = \boldsymbol{M}_{\boldsymbol{P}}\boldsymbol{M}_{\boldsymbol{S}}\boldsymbol{x}_{\boldsymbol{S}} + \boldsymbol{M}_{\boldsymbol{S}}\boldsymbol{t}_{\boldsymbol{s}} + \boldsymbol{t}_{\boldsymbol{p}}$$

- This will work but it is messy.
- Can we use a single transformation matrix to model all?

Combining linear with translation

Homogenous coordinates

- A cheap trick for elegance
- Add an extra
  - component for vectors
  - row and column vector for matrices

- $\begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} ax + by \\ cx + dy \end{pmatrix}$

Homogenous coordinates

2D linear transformations

- Translation: $$\begin{pmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{pmatrix}\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} x + t_x \\ y + t_y \\ 1 \end{pmatrix}$$

- Rotation: $$\begin{pmatrix} cos\theta & -sin\theta & 0 \\ sin\theta & cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix}\begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

- Scaling: $$\begin{pmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{pmatrix}\begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

Homogenous coordinates

Composition

- Let's go back to composition with translation

$\boldsymbol{x}'_S = \boldsymbol{M}_S \boldsymbol{x}_S + \boldsymbol{t}_s$ and then $\boldsymbol{x}'_P = \boldsymbol{M}_P \boldsymbol{x}'_S + \boldsymbol{t}_p$

- Composition with 3x3 matrix multiplication

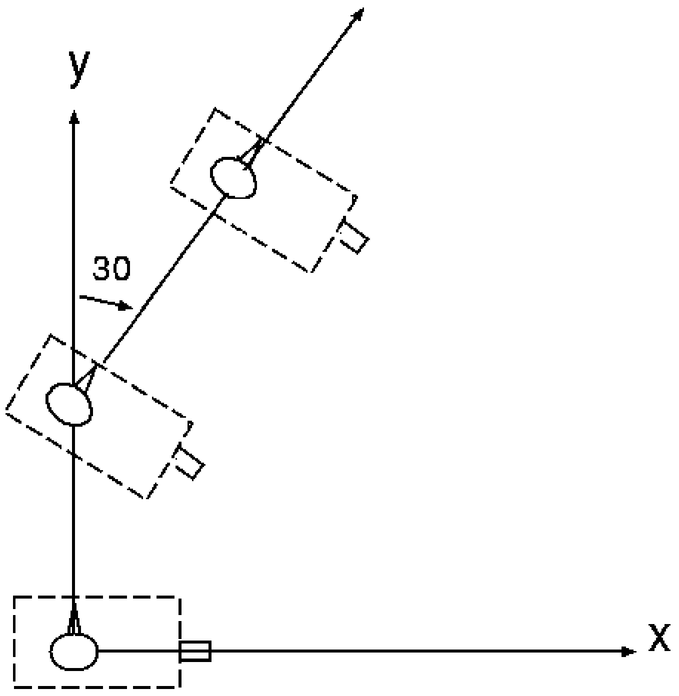
$$\begin{pmatrix} \boldsymbol{M}_p & \boldsymbol{t}_p \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \boldsymbol{M}_s & \boldsymbol{t}_s \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \boldsymbol{x}_S \\ 1 \end{pmatrix}$$
$$= \begin{pmatrix} \boldsymbol{M}_P \boldsymbol{M}_S \boldsymbol{x}_S + (\boldsymbol{M}_S \boldsymbol{t}_s + \boldsymbol{t}_p) \\ 1 \end{pmatrix}$$

- Exactly the same but cleaner and more generic

Why is this useful?

Car Example

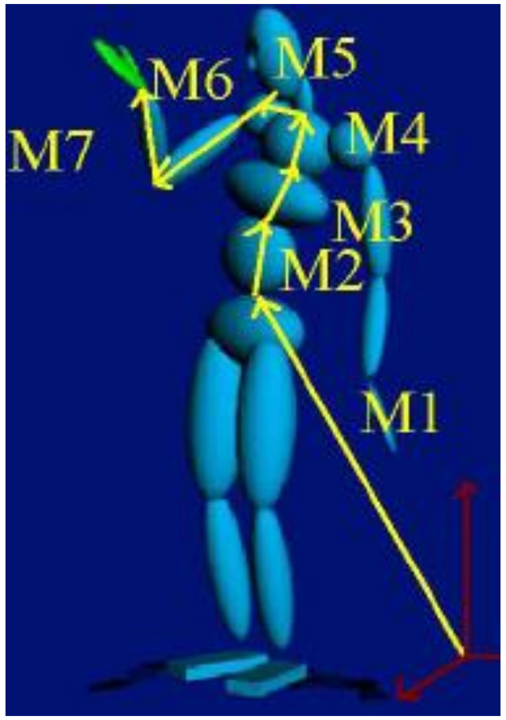
1. I sat in the car and find the side mirror is 0.4m on my right
2. I started my car and drove 5m forward
3. turned 30 degrees to right ($cos30^0 = \frac{\sqrt{3}}{2}, sin30^0 = 1/2$)
4. moved 5m forward again



What is the position of the side mirror now, relative to where I was sitting in the beginning?

$$\boldsymbol{Mx} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 5 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \sqrt{3}/2 & 1/2 & 0 \\ -1/2 & \sqrt{3}/2 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 5 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0.4 \\ 0 \\ 1 \end{pmatrix}$$

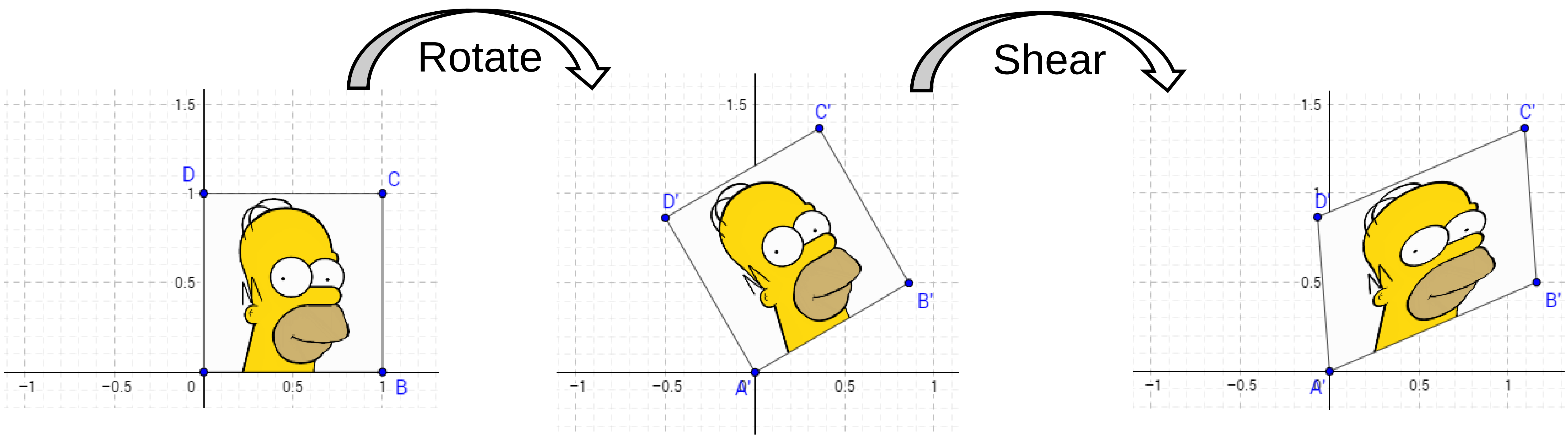
$$= \begin{pmatrix} 2.85 \\ 9.13 \\ 1 \end{pmatrix}$$



$$\boldsymbol{Mx} = \boldsymbol{M^1 M^2} \ldots \boldsymbol{M^7 x}$$

Affine transformations

- So far the set of transformations we have seen is called “affine” transformations
  - Straight lines, planes preserved
  - Parallel lines, planes preserved
  - Midpoints preserved



Properties of Matrices

- Translations: linear part is the identity
  - $$\begin{pmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{pmatrix}^{-1} = \begin{pmatrix} 1 & 0 & -t_x \\ 0 & 1 & -t_y \\ 0 & 0 & 1 \end{pmatrix}$$
- Scales: linear part is diagonal
  - $$\begin{pmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{pmatrix}^{-1} = \begin{pmatrix} 1/s_x & 0 & 0 \\ 0 & 1/s_y & 0 \\ 0 & 0 & 1 \end{pmatrix}$$
- Rotations: linear part is orthogonal
  - Columns of R are mutually orthonormal: $RR^{\mathrm{T}} = R^{\mathrm{T}}R = I$
  - Determinant of R is 1.0 $[\det(R) = \mathrm{I}]$
  - $$\begin{pmatrix} cos\theta & -sin\theta & 0 \\ sin\theta & cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix}^{-1} = \begin{pmatrix} cos\theta & sin\theta & 0 \\ -sin\theta & cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Slide credits: S. Marschner

Transforming points and vectors

- Difference between points and vectors
  - vectors are just offsets $(p - q)$
  - points have a location (vector offset from a fixed origin)
- Points and vectors transform differently
  - points can be translated but vectors cannot be

$$
\begin{aligned}
v &= p - q \\
A(p) &= Mp + t \\
A(p - q) &= Mp + t - (Mq + t) \\
&= M(p - q) = Mv
\end{aligned}
$$

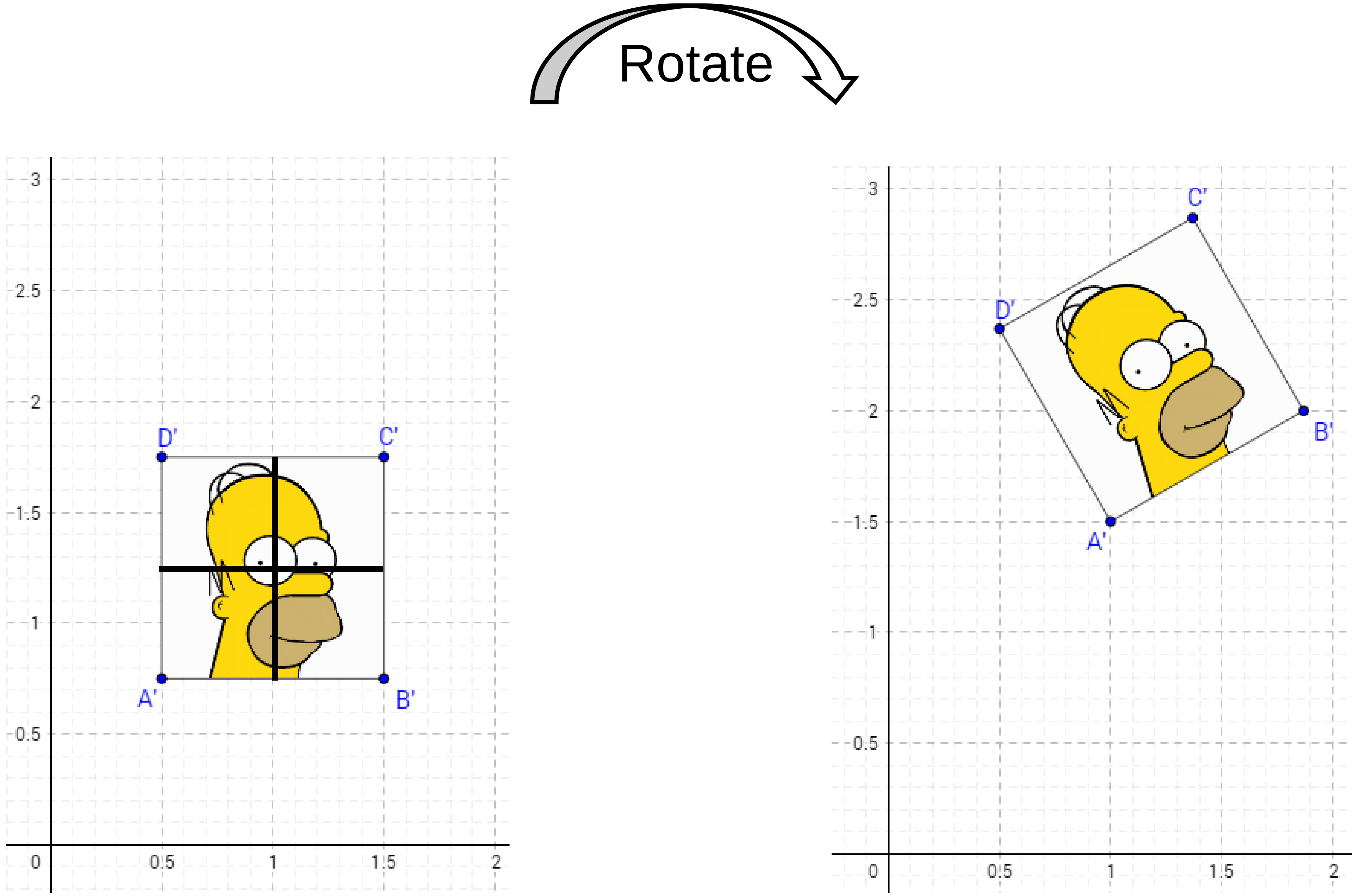
- Homogenous cords. have 0 instead of 1

Slide credits: S. Marschner

Change of coordinates

Rotate about a particular point

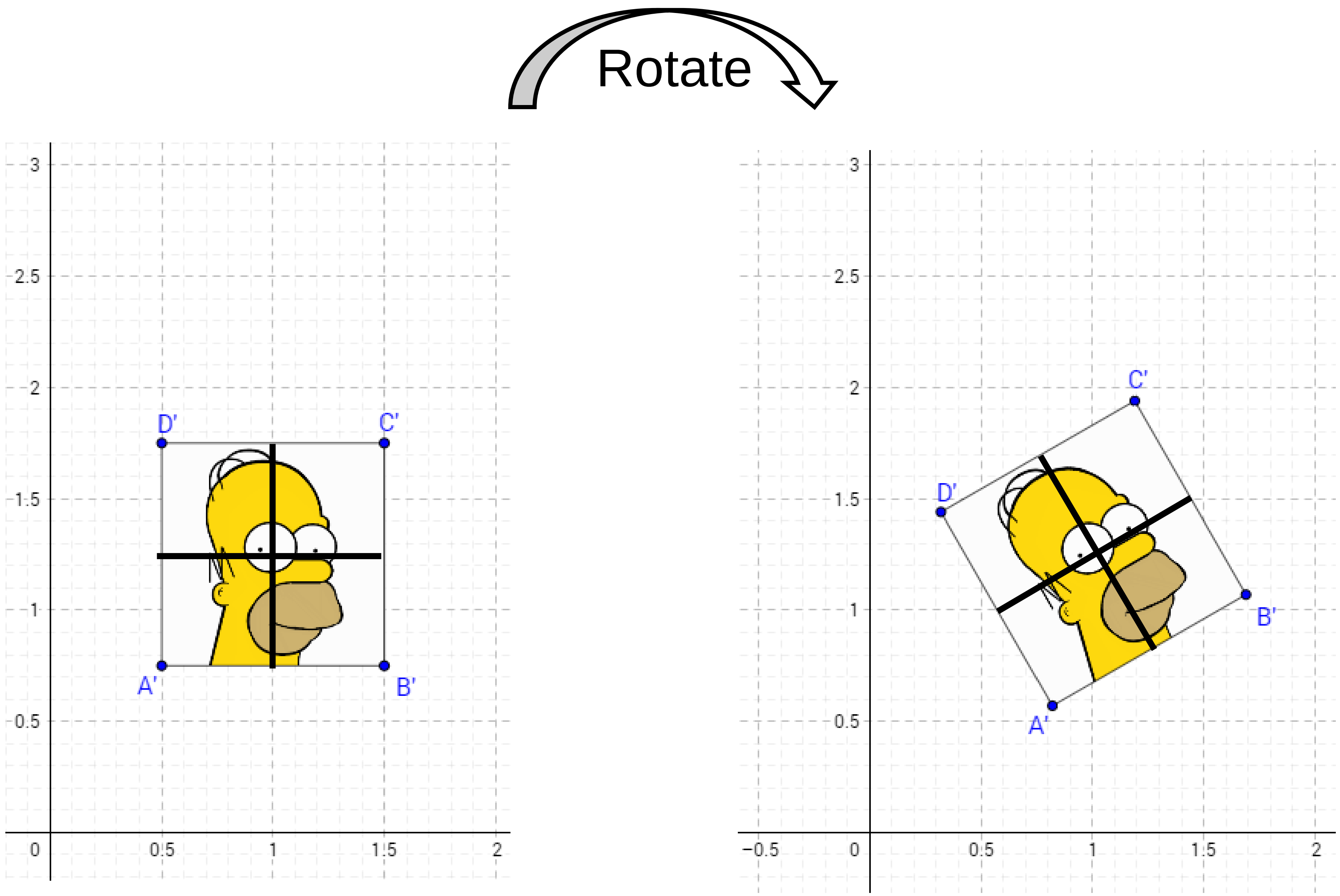
What you get by rotating



Change of coordinates

Rotate about a particular point

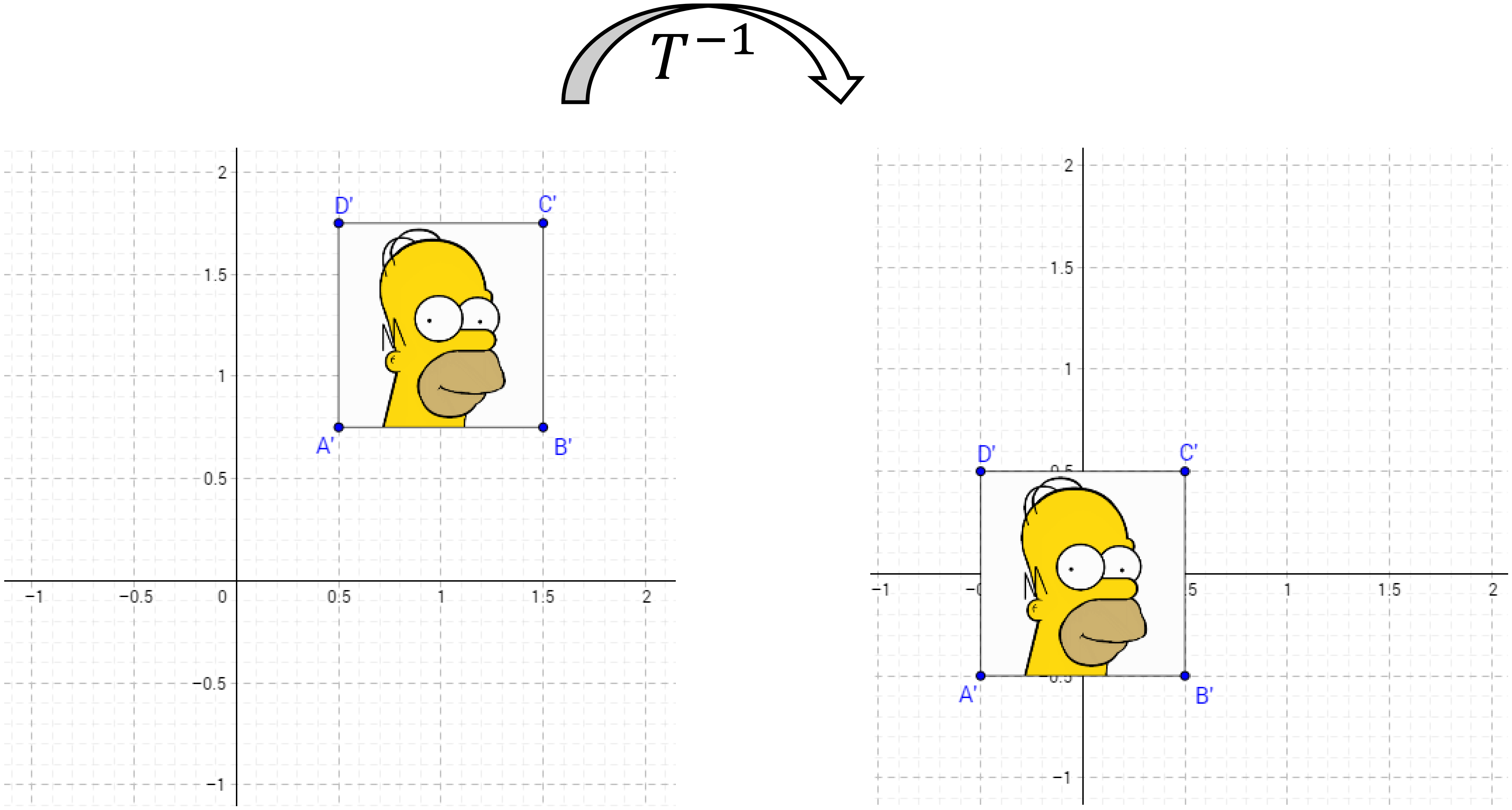
What you want is



Change of coordinates

Rotate about a particular point

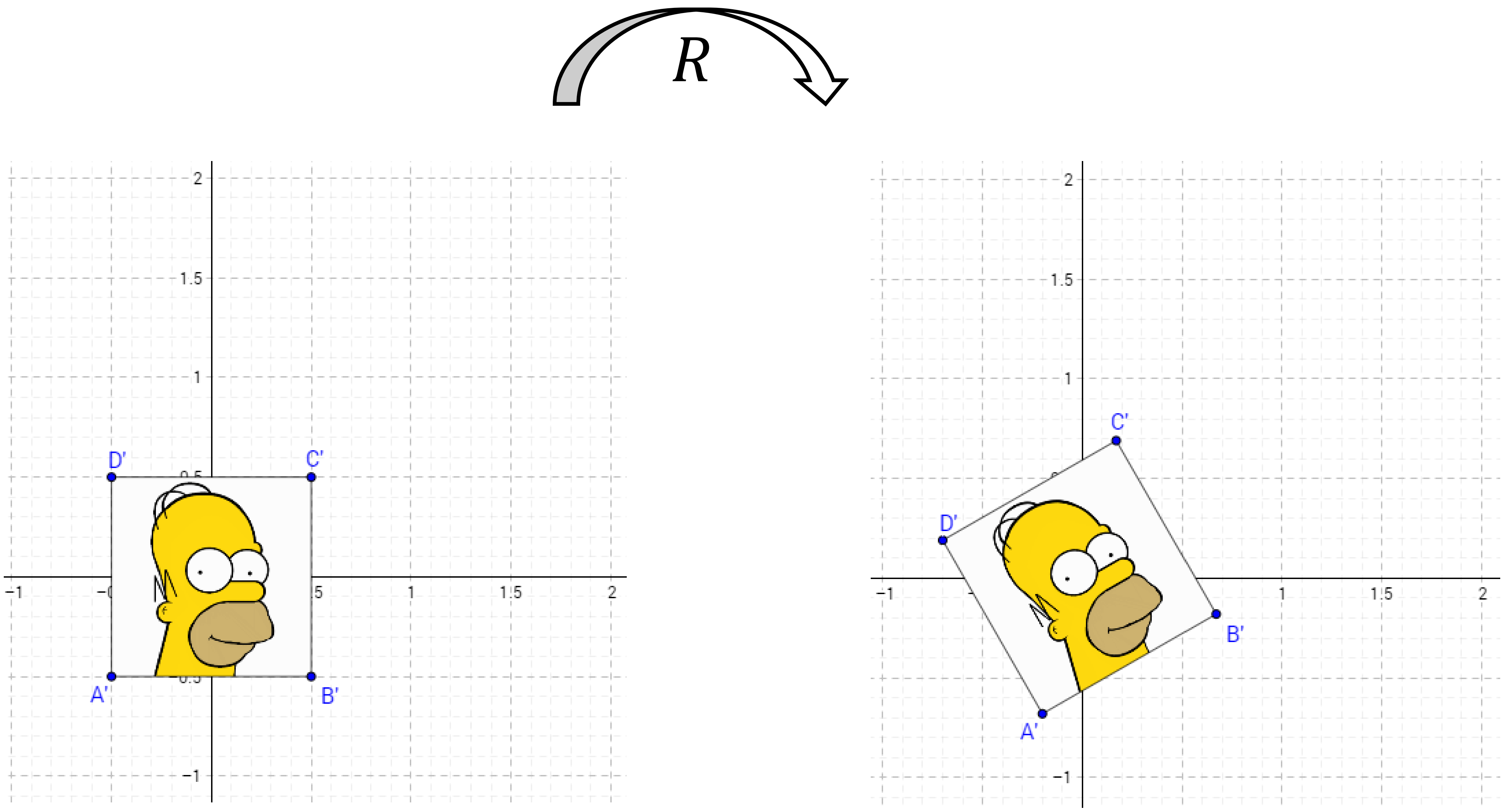
$$M = T^{-1}$$



Change of coordinates

Rotate about a particular point

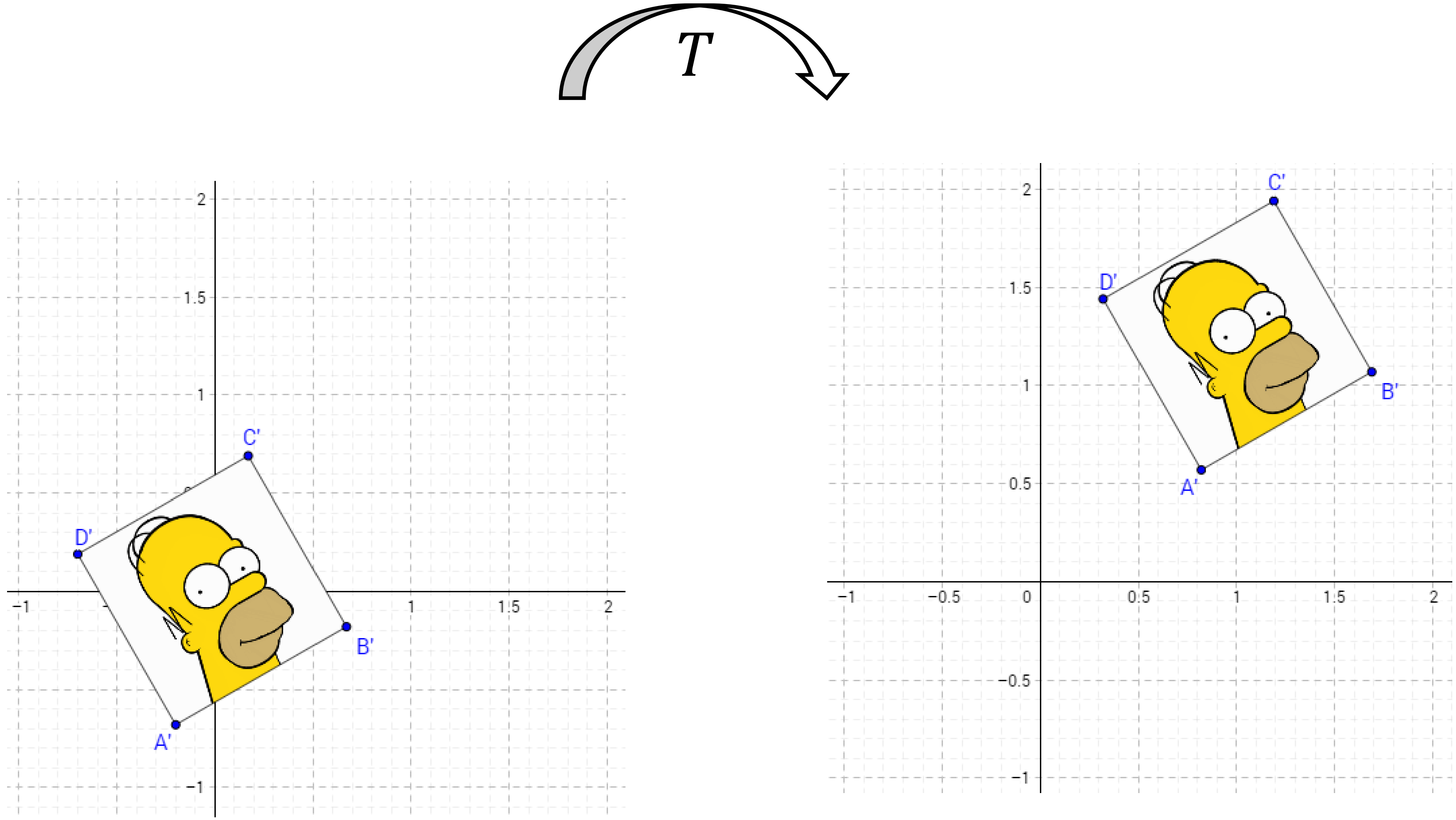
$$M = RT^{-1}$$



Change of coordinates

Rotate about a particular point

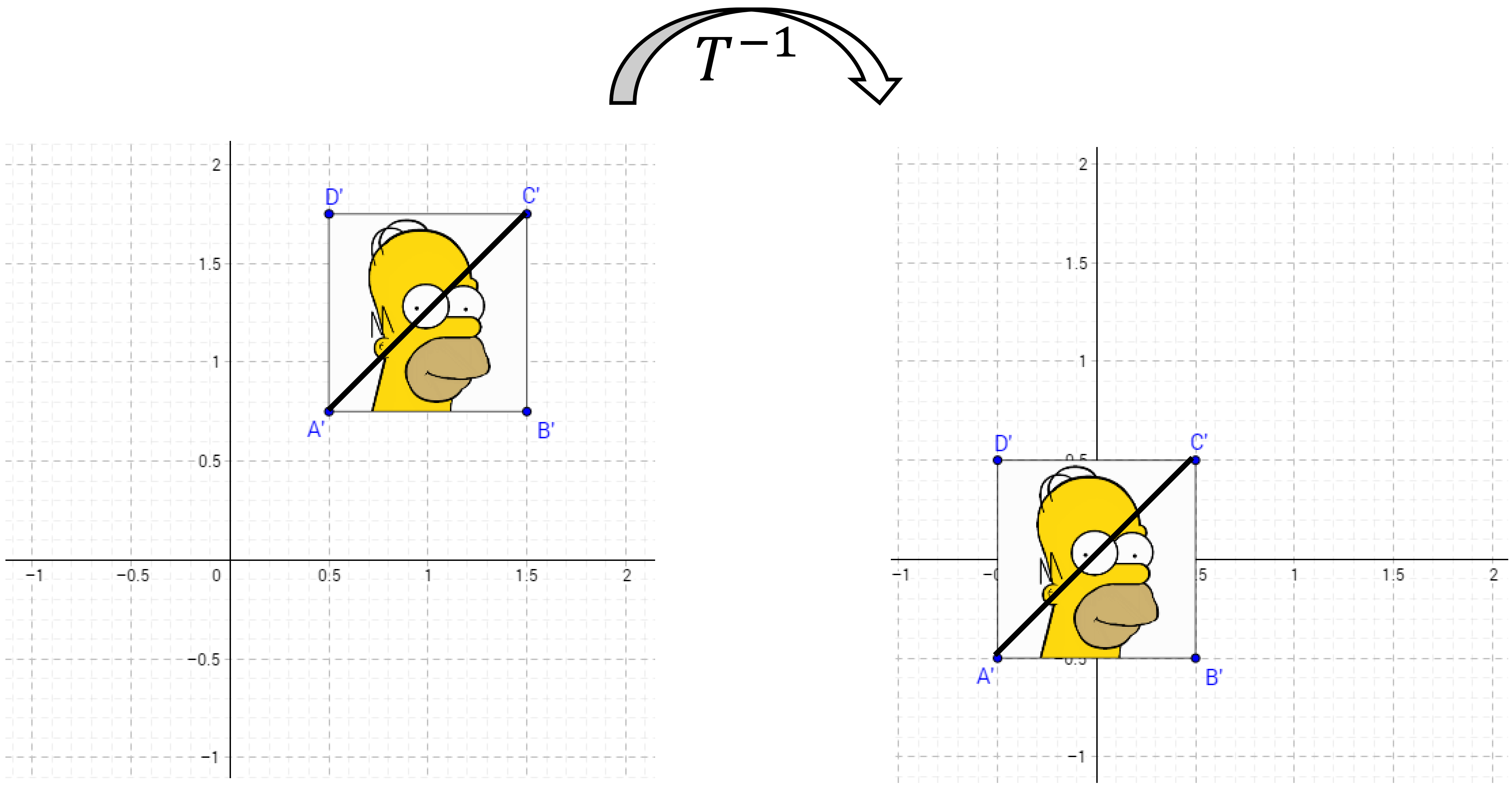
$$M = TRT^{-1}$$



Change of coordinates

Scale along a particular axis

$$M = T^{-1}$$

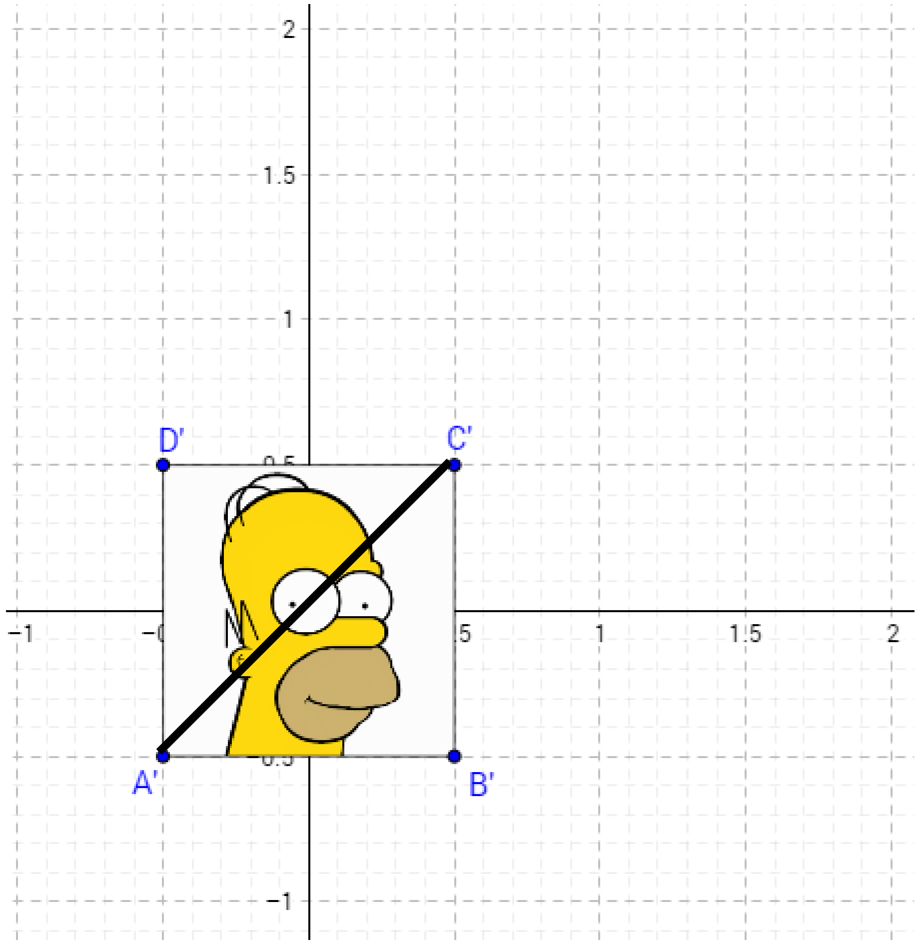


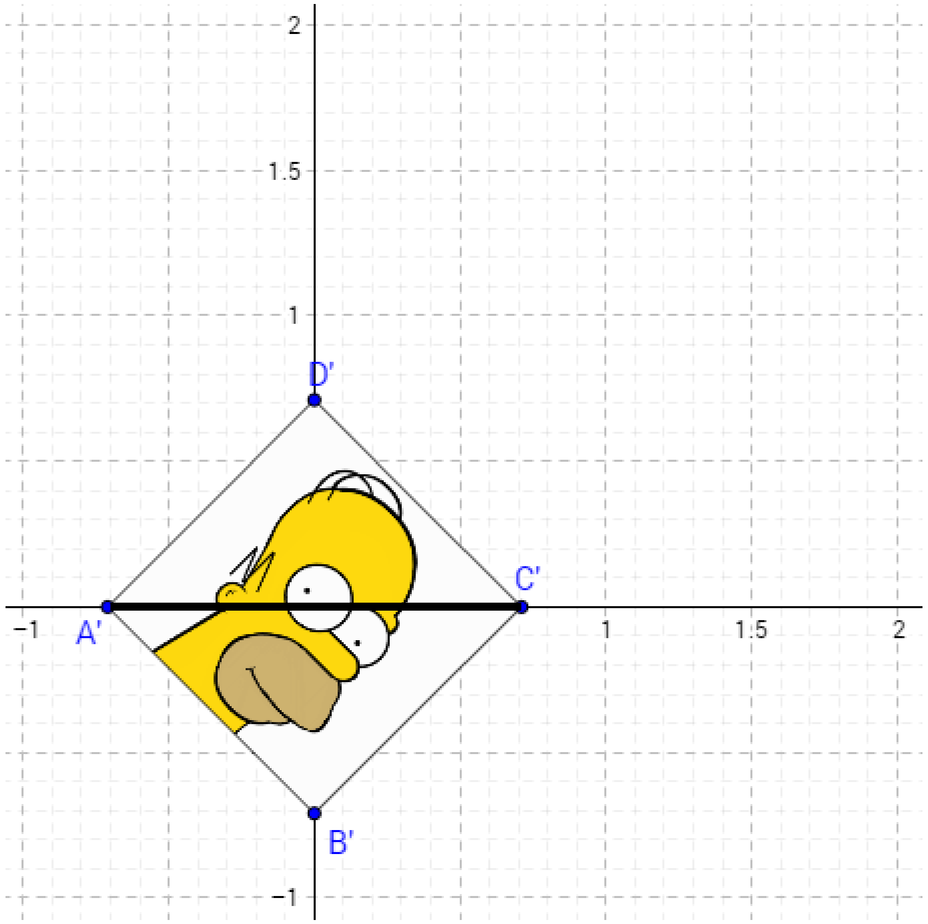
Change of coordinates

Scale along a particular axis

$$M = R^{-1}T^{-1}$$

$$R^{-1}$$

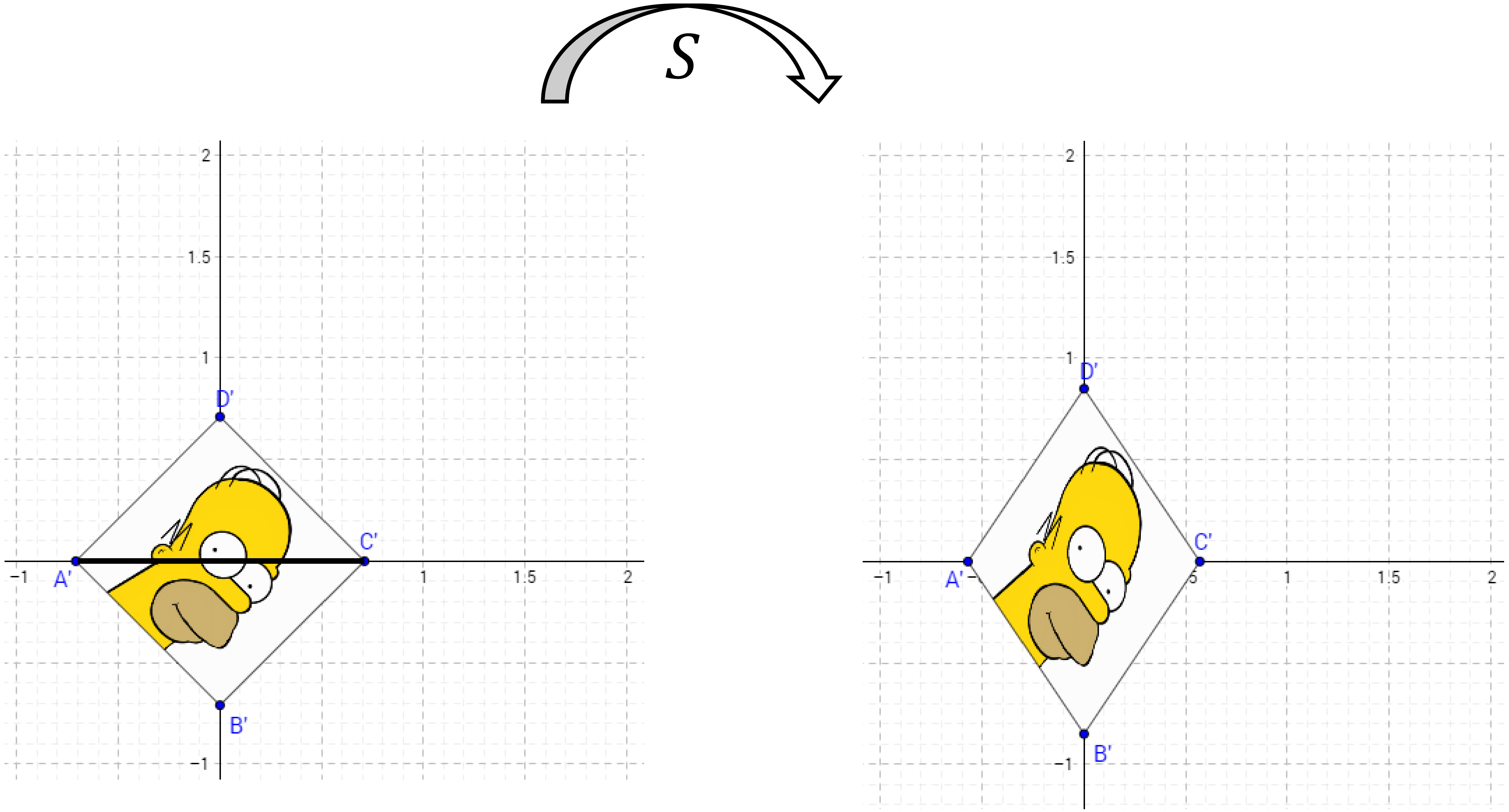




Change of coordinates

Scale along a particular axis

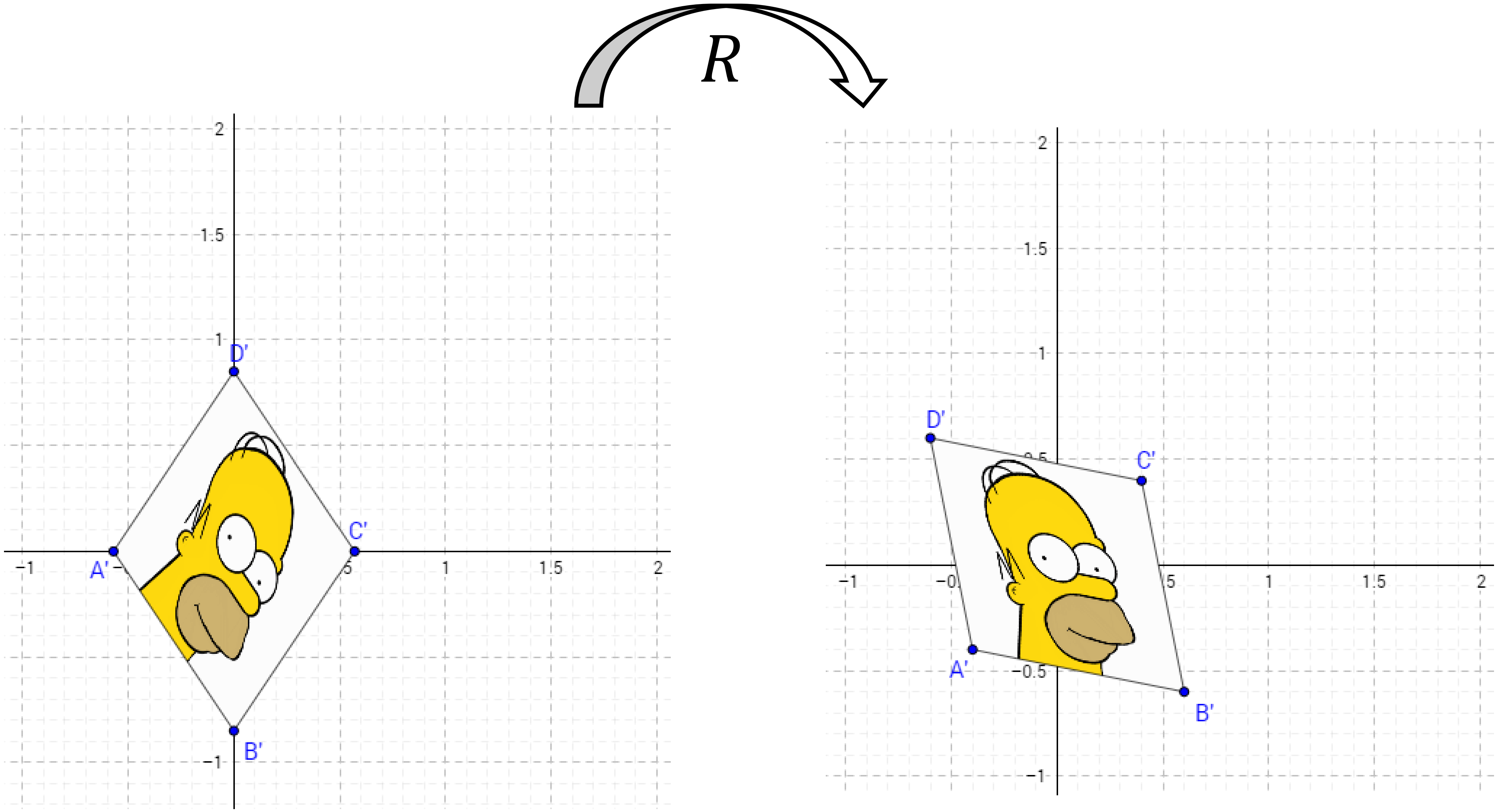
$$M = SR^{-1}T^{-1}$$



Change of coordinates

Scale along a particular axis

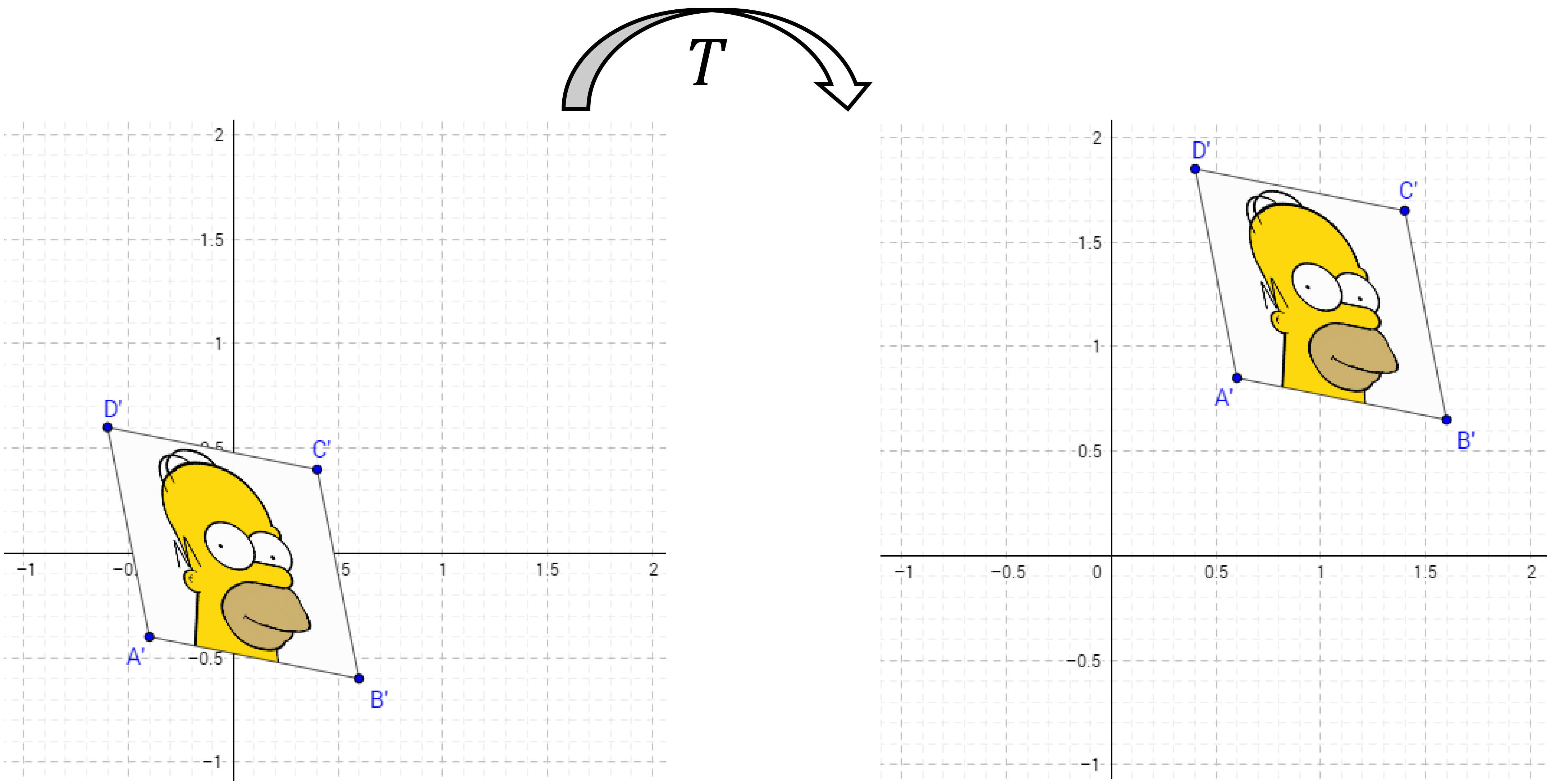
$$M = RSR^{-1}T^{-1}$$



Change of coordinates

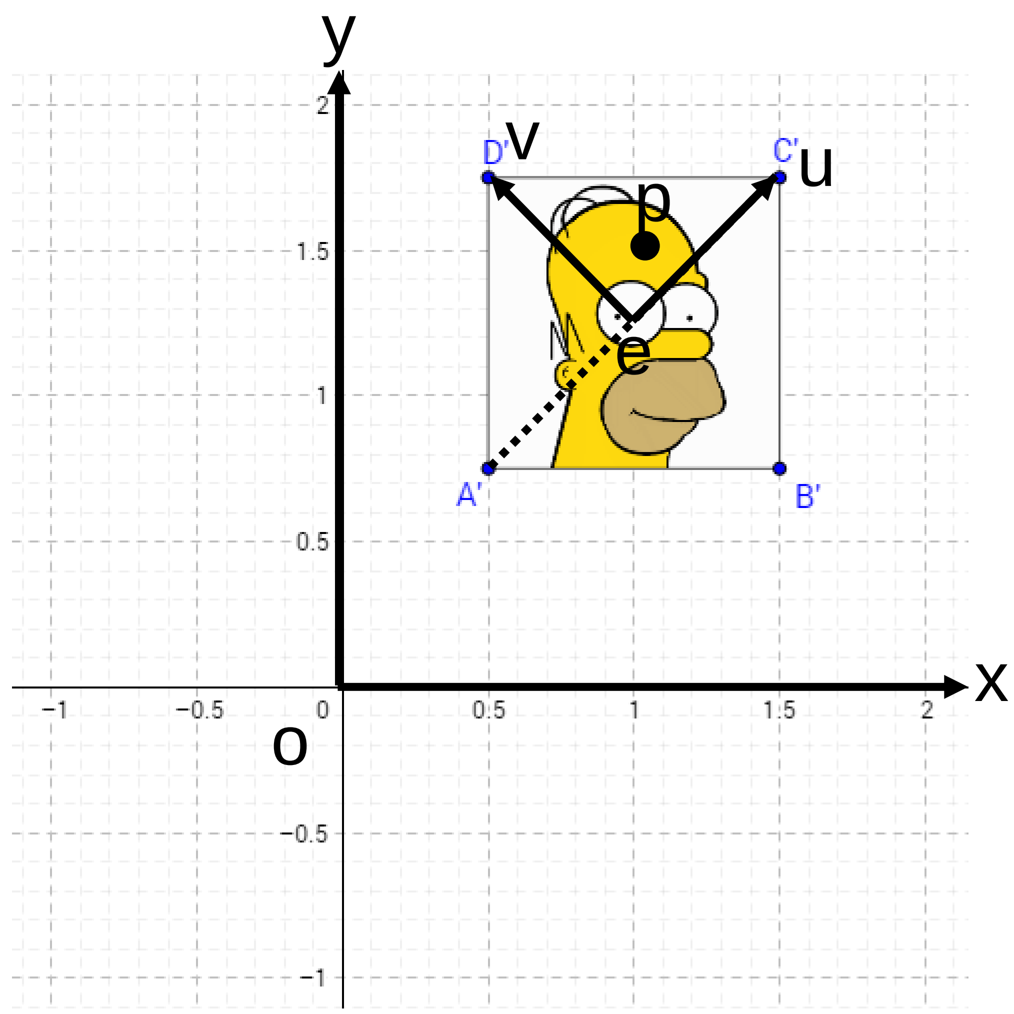
Scale along a particular axis

Finally $$M = TRSR^{-1}T^{-1}$$



Change of coordinates

General case



$$\boldsymbol{p} = (x_p, y_p) = \boldsymbol{o} + x_p\boldsymbol{x} + y_p\boldsymbol{y}$$
$$\boldsymbol{p} = (u_p, v_p) = \boldsymbol{e} + u_p\boldsymbol{u} + v_p\boldsymbol{v}$$

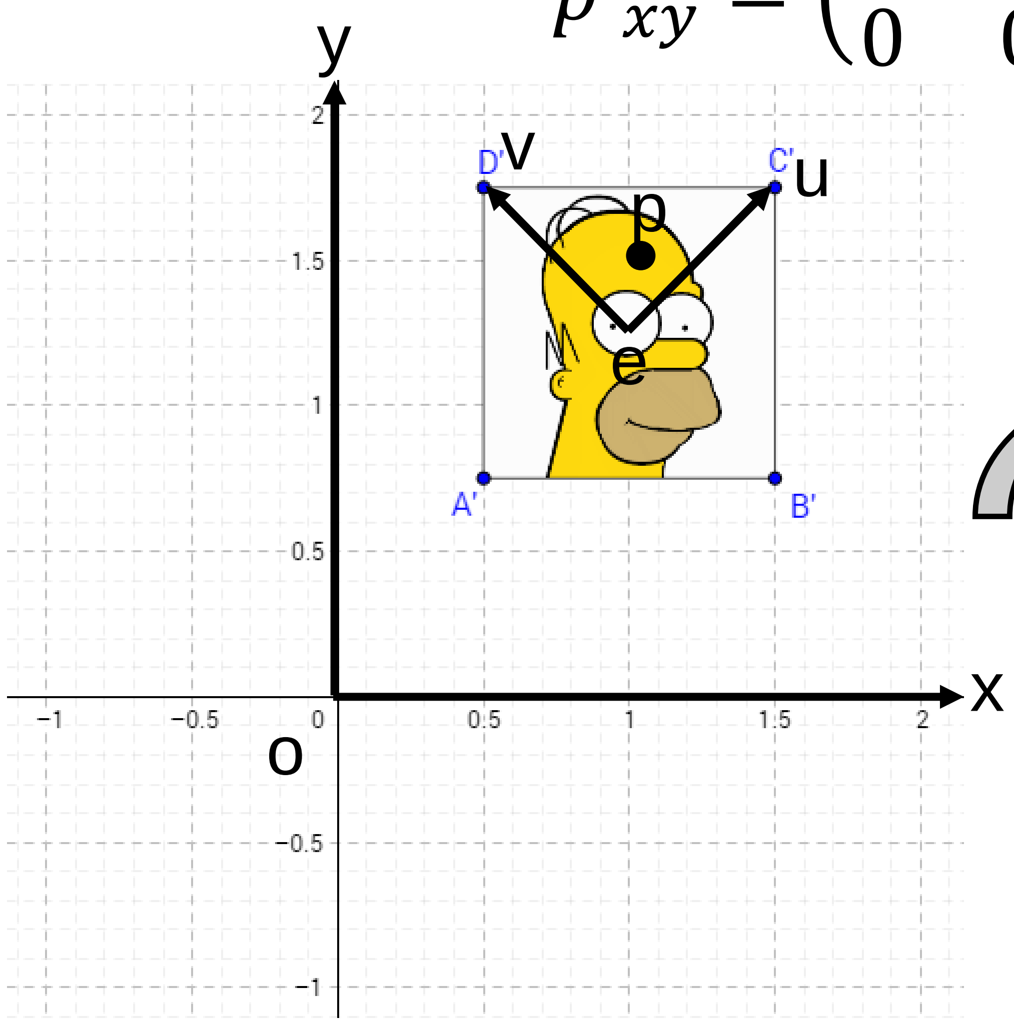
- $\boldsymbol{o} + x_p\boldsymbol{x} + y_p\boldsymbol{y} = \boldsymbol{e} + u_p\boldsymbol{u} + v_p\boldsymbol{v}$
- Assuming x and y are canonical
  - $\begin{pmatrix} x_p \\ y_p \\ 1 \end{pmatrix} = \begin{pmatrix} x_u & x_v & x_e \\ y_u & y_v & y_e \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} u_p \\ v_p \\ 1 \end{pmatrix}$
  - $p_{xy} = \begin{pmatrix} \boldsymbol{u} & \boldsymbol{v} & \boldsymbol{e} \\ 0 & 0 & 1 \end{pmatrix} p_{uv}$
  - $p_{uv} = \begin{pmatrix} \boldsymbol{u} & \boldsymbol{v} & \boldsymbol{e} \\ 0 & 0 & 1 \end{pmatrix}^{-1} p_{xy}$

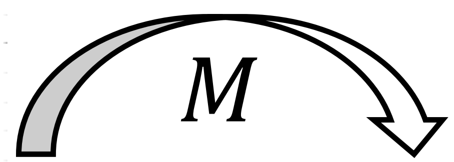
Change of coordinates

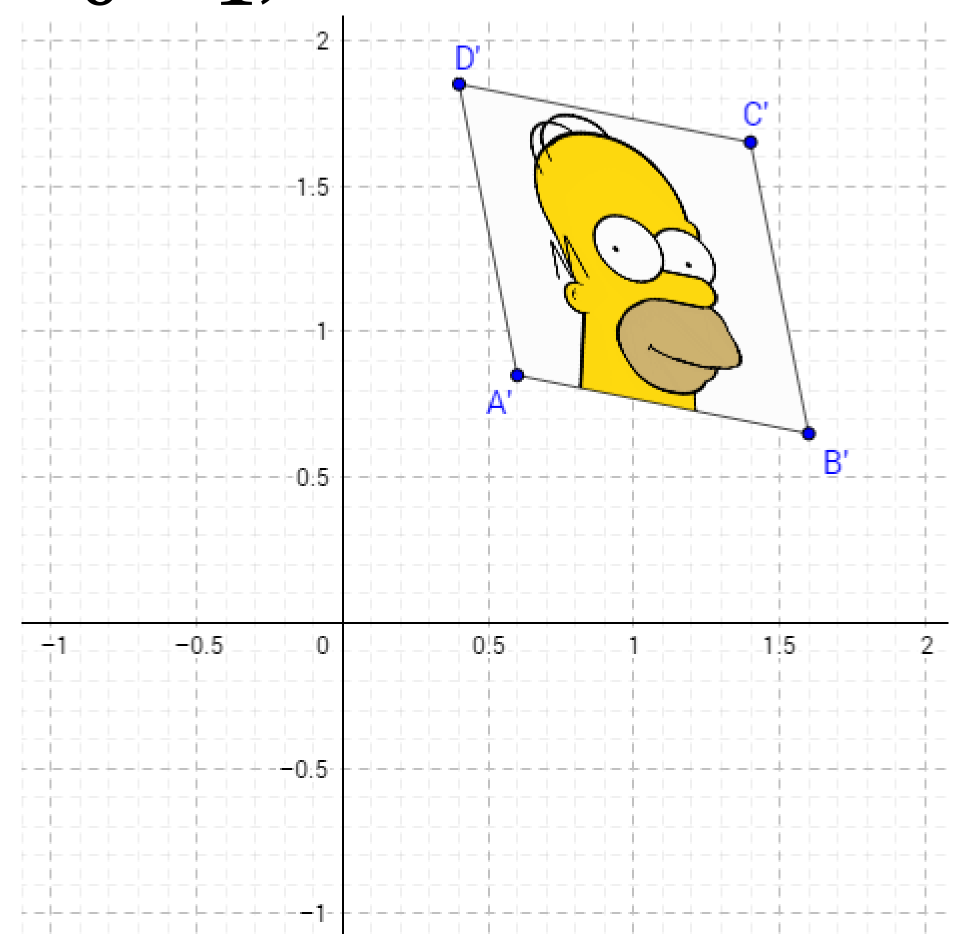
General case

$$M = TRSR^{-1}T^{-1}$$

$$p'_{xy} = \begin{pmatrix} \boldsymbol{u} & \boldsymbol{v} & \boldsymbol{e} \\ 0 & 0 & 1 \end{pmatrix} S \begin{pmatrix} \boldsymbol{u} & \boldsymbol{v} & \boldsymbol{e} \\ 0 & 0 & 1 \end{pmatrix}^{-1} p_{xy}$$



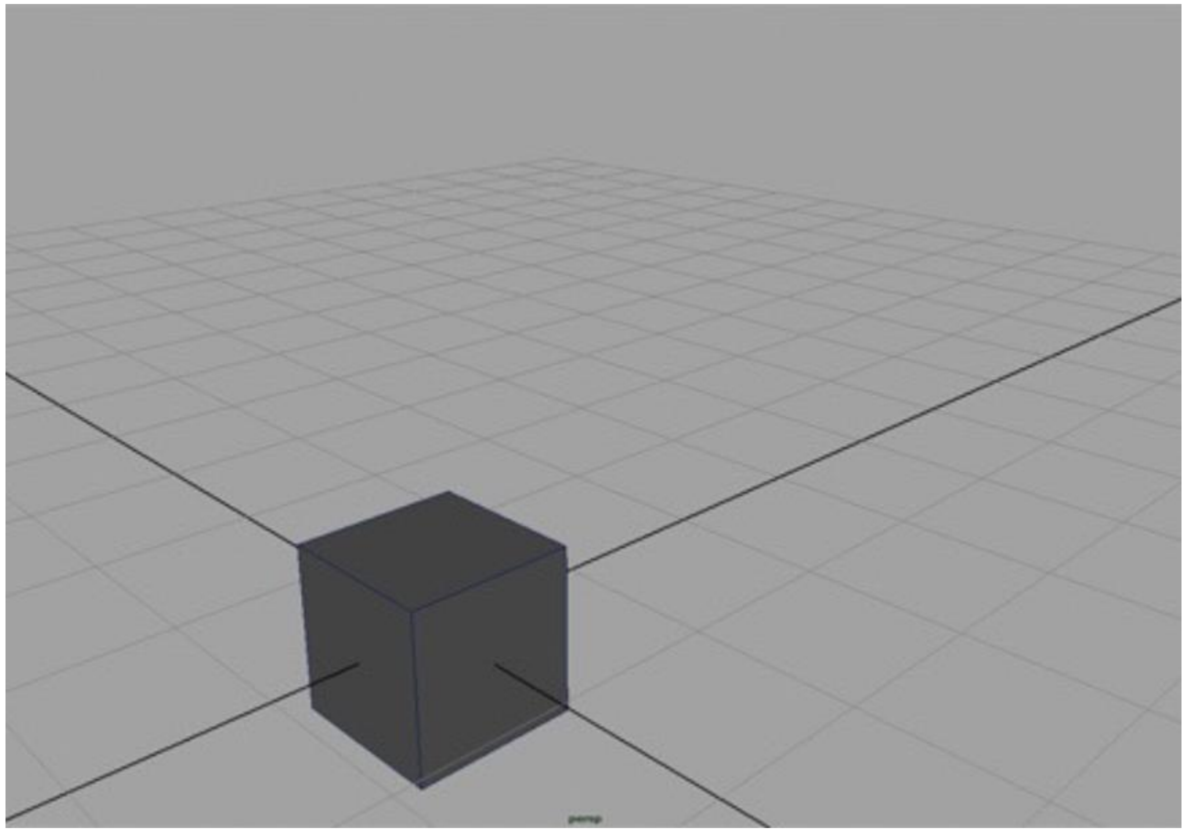




3D Affine transformations

Translation

$$\begin{pmatrix} x' \\ y' \\ z' \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

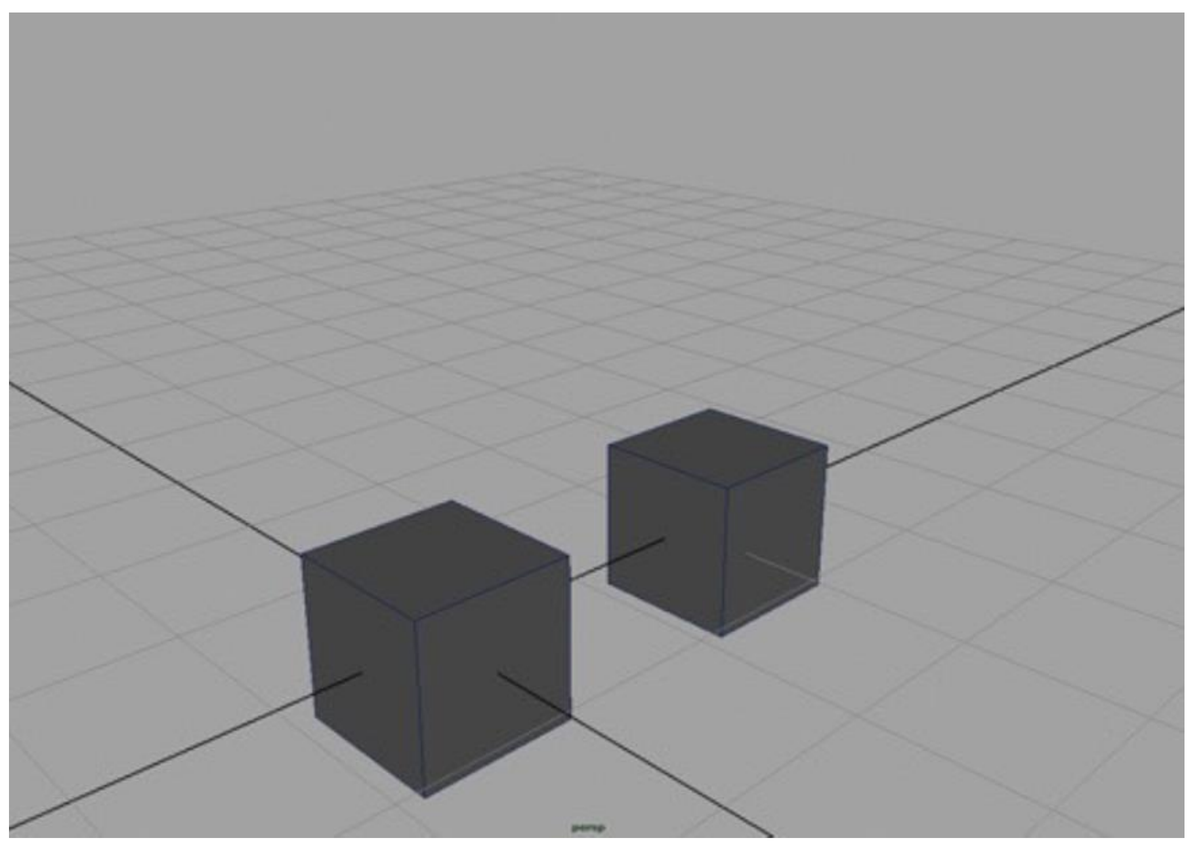


Slide credits: S. Marschner

3D Affine transformations

Translation

$$\begin{pmatrix} x' \\ y' \\ z' \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

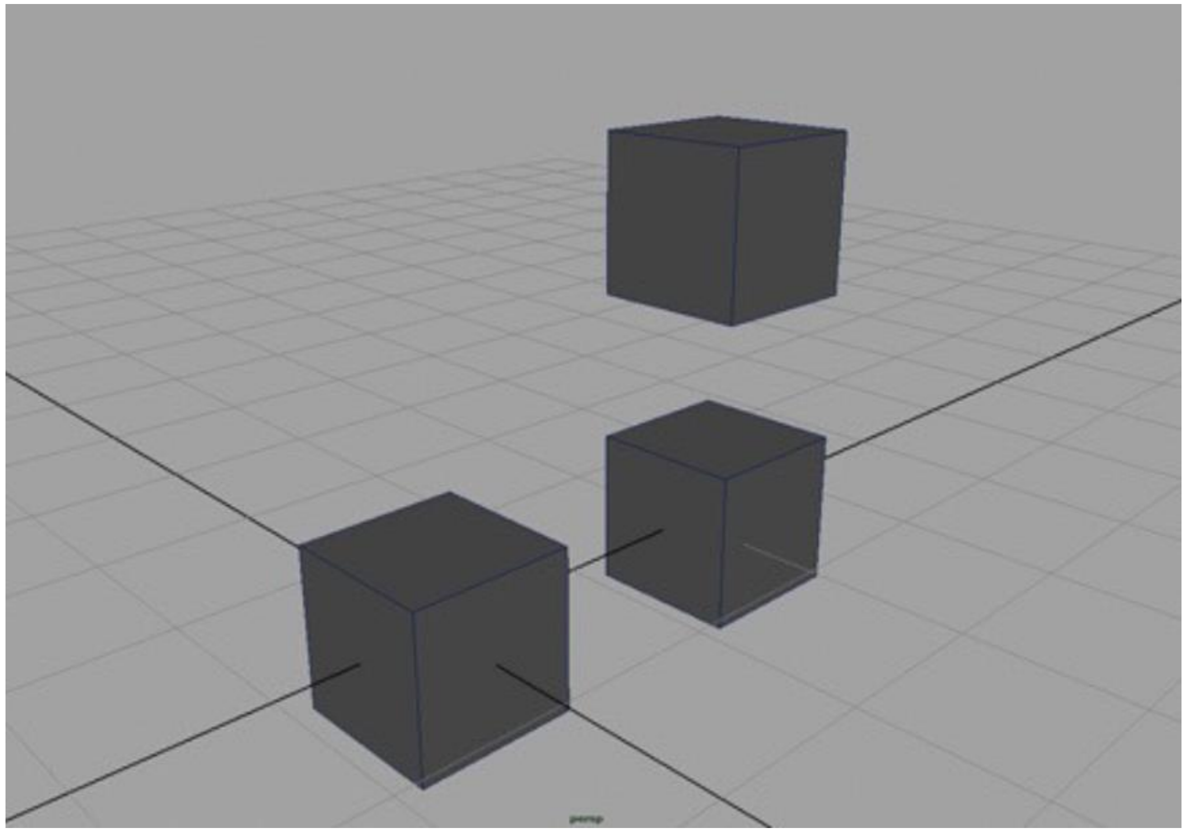


Slide credits: S. Marschner

3D Affine transformations

Translation

$$\begin{pmatrix} x' \\ y' \\ z' \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

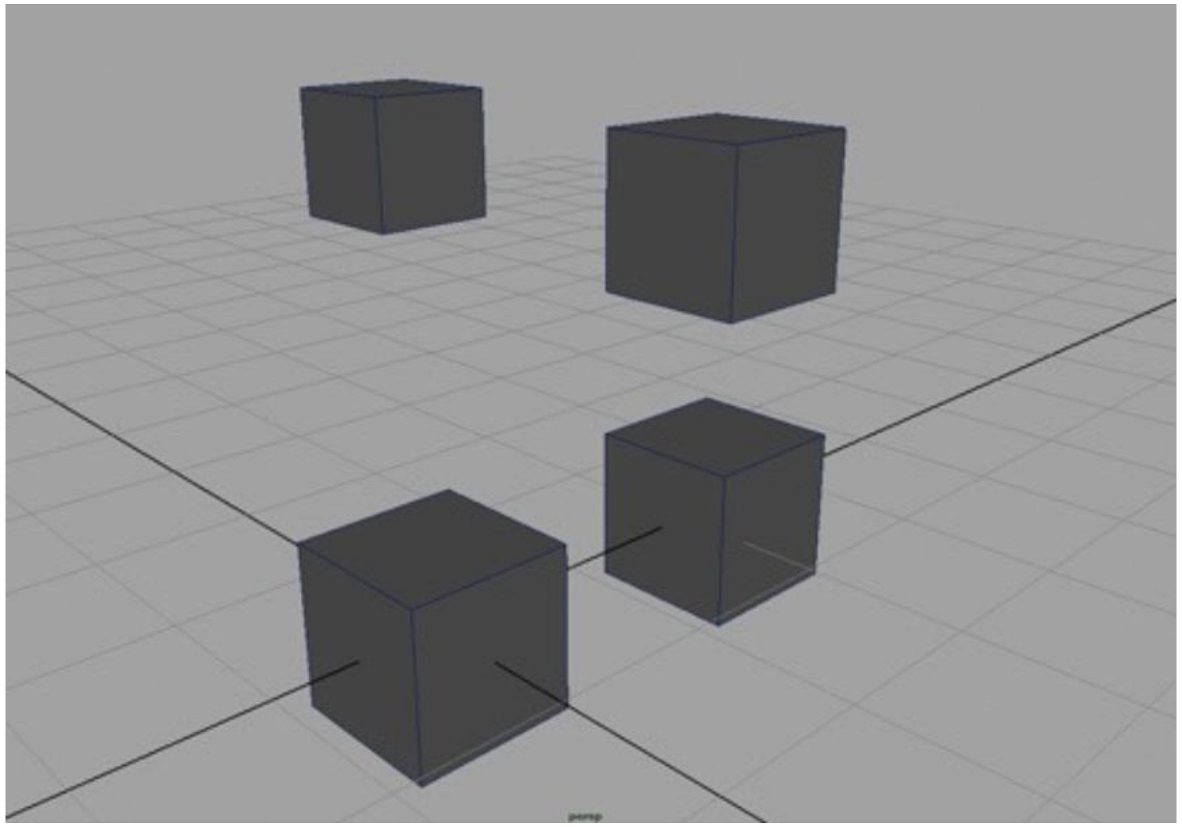


Slide credits: S. Marschner

3D Affine transformations

Translation

$$\begin{pmatrix} x' \\ y' \\ z' \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

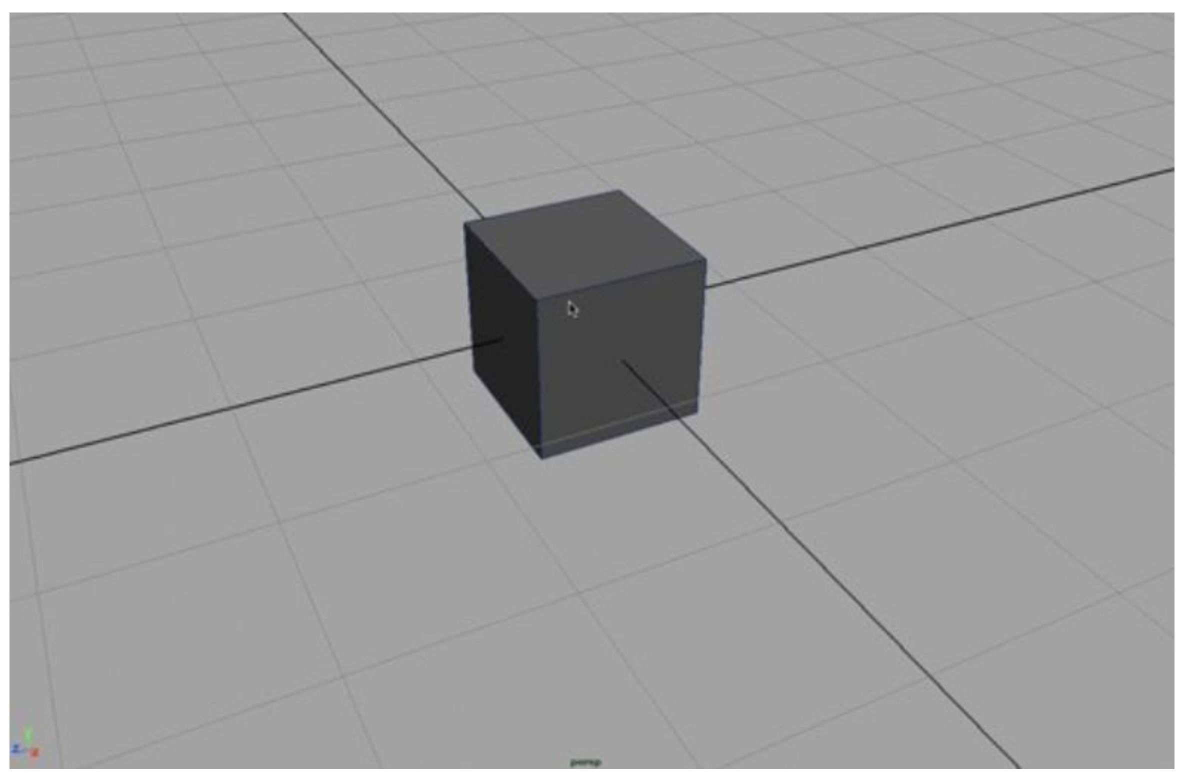


Slide credits: S. Marschner

3D Affine transformations

Scaling

$$\begin{pmatrix} x' \\ y' \\ z' \\ 1 \end{pmatrix} = \begin{pmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

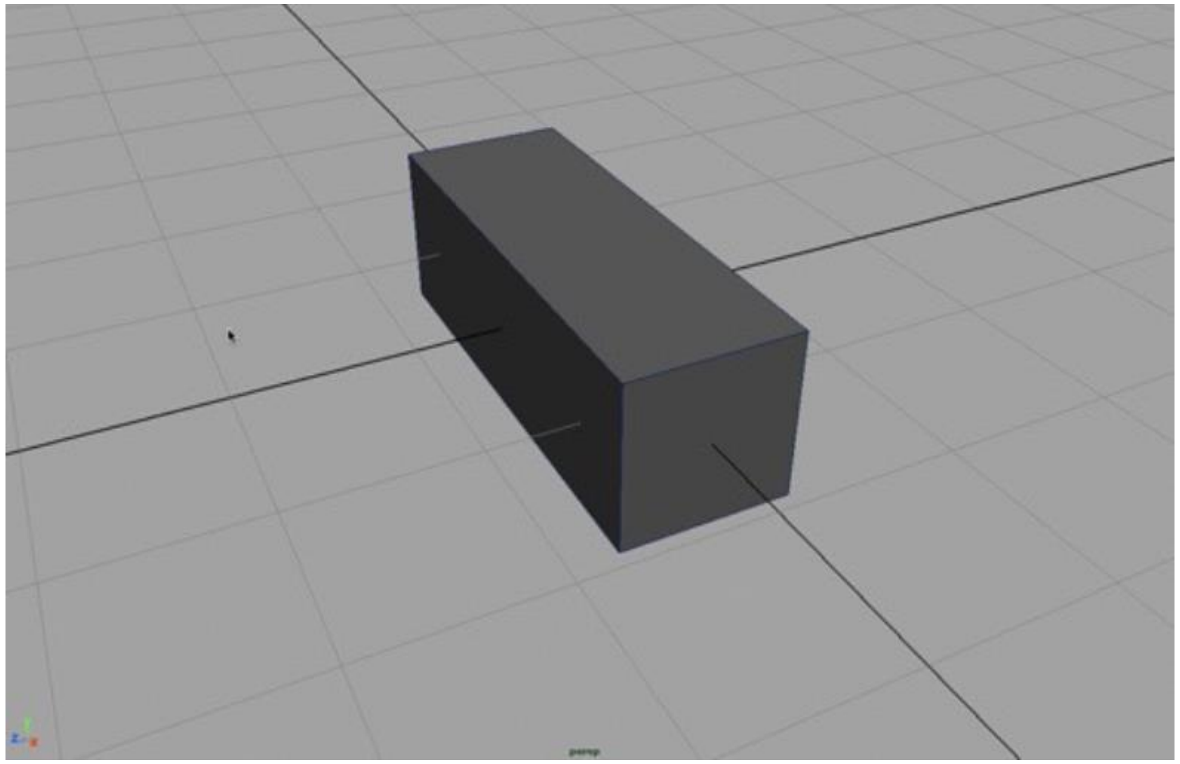


Slide credits: S. Marschner

3D Affine transformations

Scaling

$$\begin{pmatrix} x' \\ y' \\ z' \\ 1 \end{pmatrix} = \begin{pmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

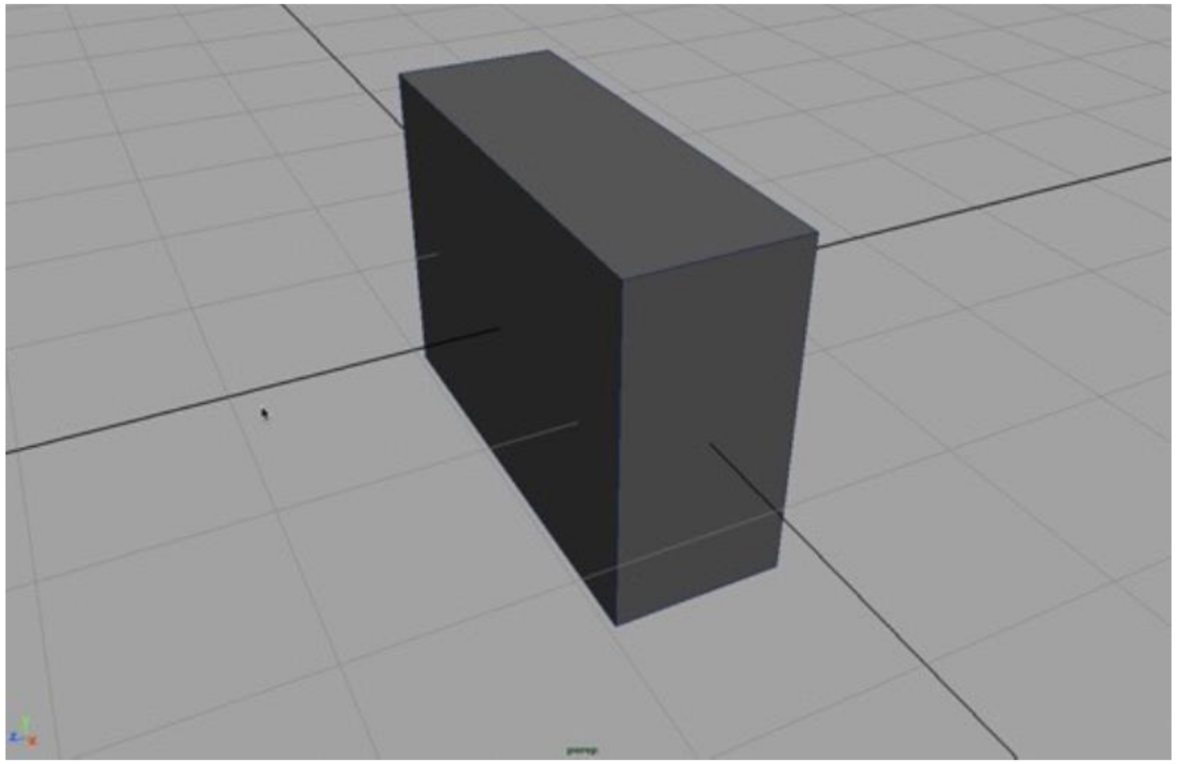


Slide credits: S. Marschner

3D Affine transformations

Scaling

$$\begin{pmatrix} x' \\ y' \\ z' \\ 1 \end{pmatrix} = \begin{pmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

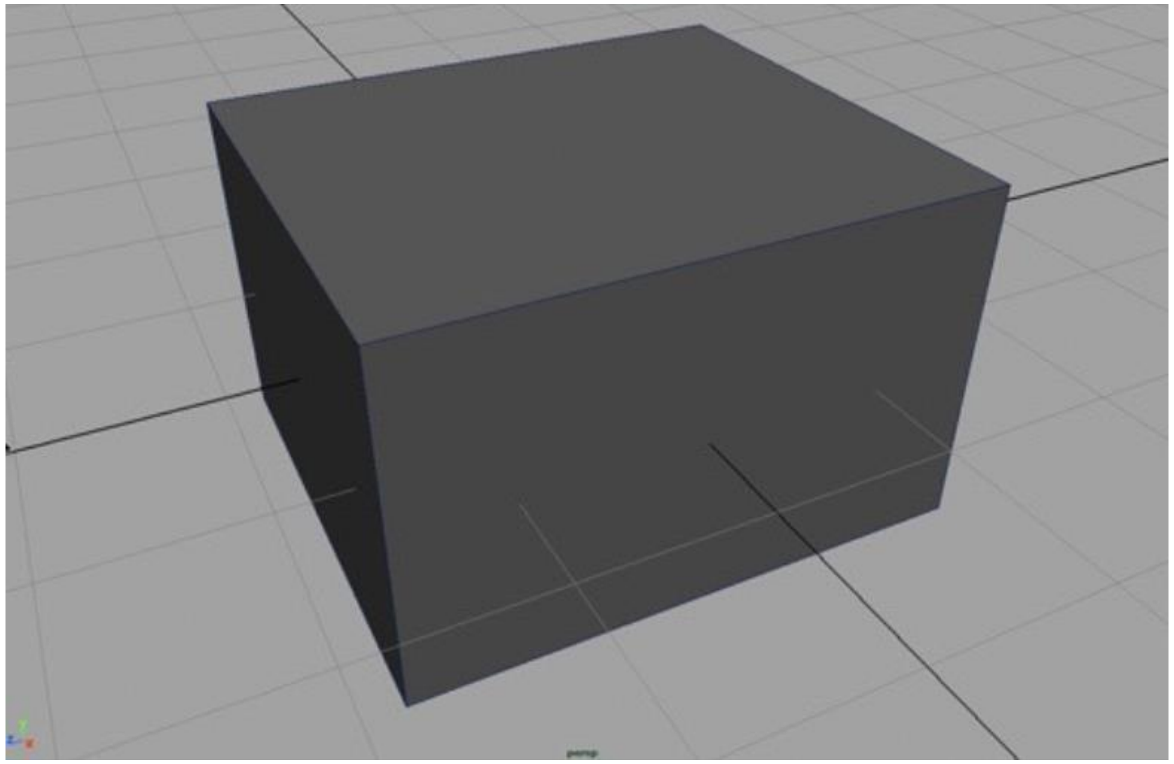


Slide credits: S. Marschner

3D Affine transformations

Scaling

$$\begin{pmatrix} x' \\ y' \\ z' \\ 1 \end{pmatrix} = \begin{pmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

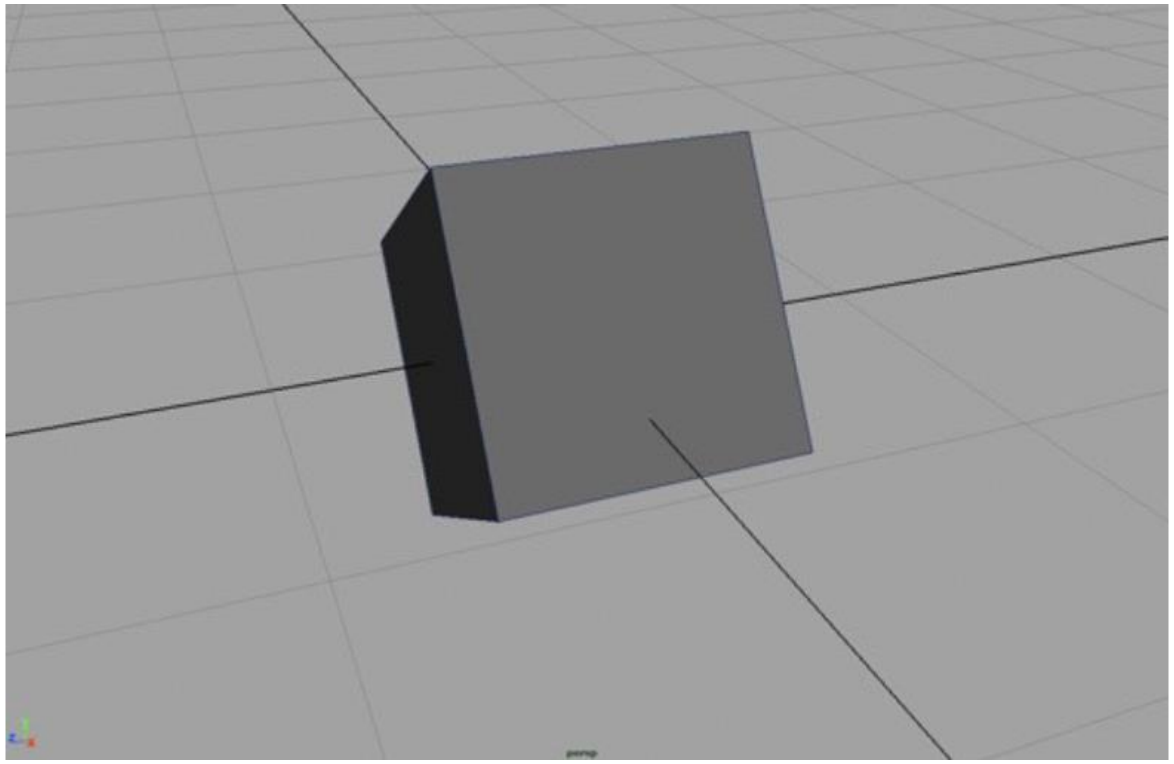


Slide credits: S. Marschner

3D Affine transformations

Rotation around z axis

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

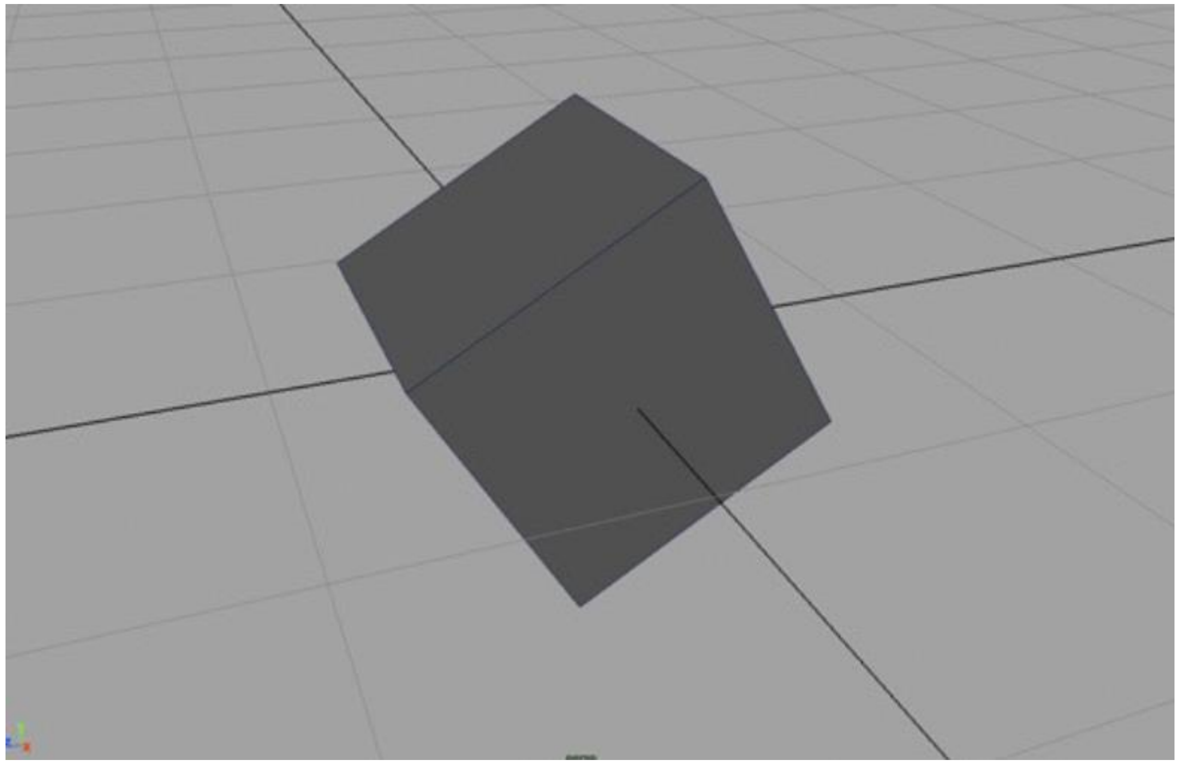


Slide credits: S. Marschner

3D Affine transformations

Rotation around x axis

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

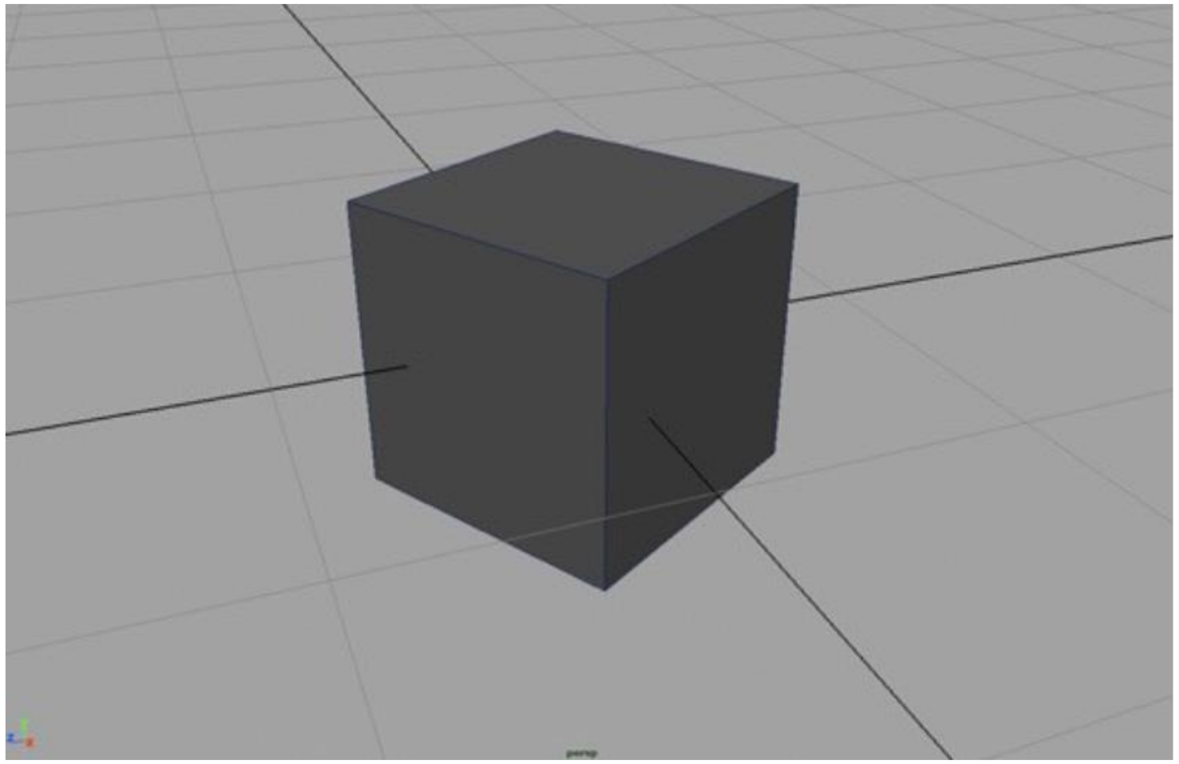


Slide credits: S. Marschner

3D Affine transformations

Rotation around y axis

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



Slide credits: S. Marschner

Summary

- Transformations: translation, rotation, scaling and shearing
- Using homogeneous transformation, 2D (3D) transformations can be represented by multiplication of a 3x3 (4x4) matrix
- Change of coordinates
- 3D transformations

Reading

B1: Chapter 6