Topic: Basic Transformations

Today's problem

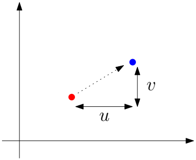
- **Goal:** Programming a simple animation of Sun, Earth and Moon (see short demo)

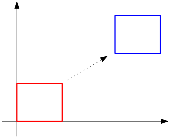
Topic: Basic transformations (translation, scaling, rotation)

1. Recap linear algebra
2. Concept: homogeneous coordinates
   (including: expressing basic transformations via homogeneous coordinates)
3. Application: animations

1. Recap linear algebra

1. Translation

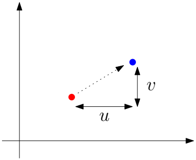


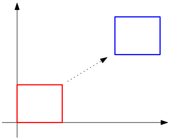


1. Recap linear algebra

1. Translation

$$\begin{pmatrix} x_{\text{new}} \\ y_{\text{new}} \end{pmatrix} = \begin{pmatrix} x_{\text{old}} \\ y_{\text{old}} \end{pmatrix} + \begin{pmatrix} u \\ v \end{pmatrix}$$

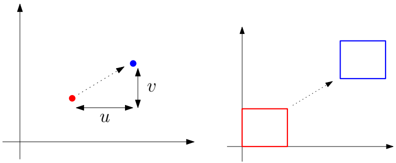




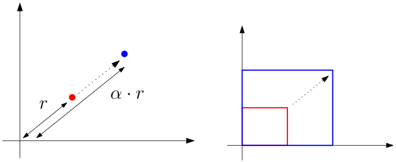
1. Recap linear algebra

1. Translation

$$\begin{pmatrix} x_{\text{new}} \\ y_{\text{new}} \end{pmatrix} = \begin{pmatrix} x_{\text{old}} \\ y_{\text{old}} \end{pmatrix} + \begin{pmatrix} u \\ v \end{pmatrix}$$



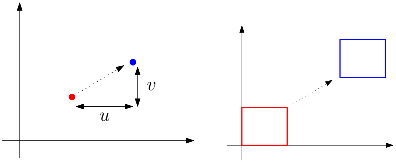
2. Scaling



1. Recap linear algebra

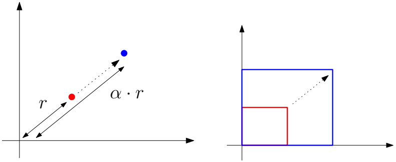
1. Translation

$$\begin{pmatrix} x_{\text{new}} \\ y_{\text{new}} \end{pmatrix} = \begin{pmatrix} x_{\text{old}} \\ y_{\text{old}} \end{pmatrix} + \begin{pmatrix} u \\ v \end{pmatrix}$$



2. Scaling

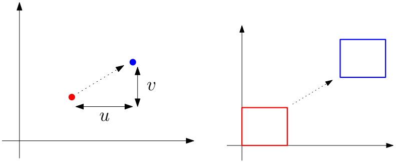
$$\begin{pmatrix} x_{\text{new}} \\ y_{\text{new}} \end{pmatrix} = \begin{pmatrix} \alpha & 0 \\ 0 & \alpha \end{pmatrix} \cdot \begin{pmatrix} x_{\text{old}} \\ y_{\text{old}} \end{pmatrix}$$



1. Recap linear algebra

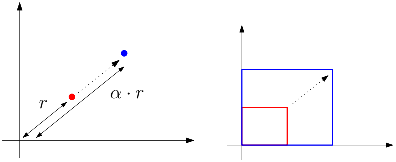
1. Translation

$$\begin{pmatrix} x_{\text{new}} \\ y_{\text{new}} \end{pmatrix} = \begin{pmatrix} x_{\text{old}} \\ y_{\text{old}} \end{pmatrix} + \begin{pmatrix} u \\ v \end{pmatrix}$$

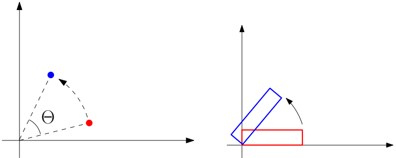


2. Scaling

$$\begin{pmatrix} x_{\text{new}} \\ y_{\text{new}} \end{pmatrix} = \begin{pmatrix} \alpha & 0 \\ 0 & \alpha \end{pmatrix} \cdot \begin{pmatrix} x_{\text{old}} \\ y_{\text{old}} \end{pmatrix}$$



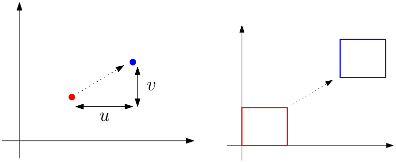
3. Rotation



1. Recap linear algebra

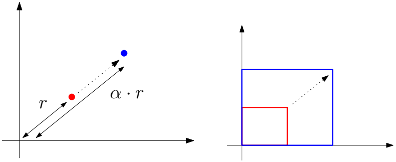
1. Translation

$$\begin{pmatrix} x_{\text{new}} \\ y_{\text{new}} \end{pmatrix} = \begin{pmatrix} x_{\text{old}} \\ y_{\text{old}} \end{pmatrix} + \begin{pmatrix} u \\ v \end{pmatrix}$$



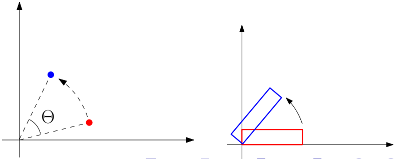
2. Scaling

$$\begin{pmatrix} x_{\text{new}} \\ y_{\text{new}} \end{pmatrix} = \begin{pmatrix} \alpha & 0 \\ 0 & \alpha \end{pmatrix} \cdot \begin{pmatrix} x_{\text{old}} \\ y_{\text{old}} \end{pmatrix}$$



3. Rotation

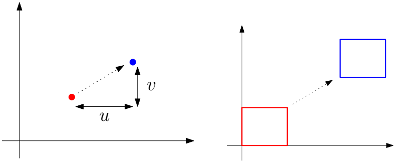
$$\begin{pmatrix} x_{\text{new}} \\ y_{\text{new}} \end{pmatrix} = \begin{pmatrix} \cos(\Theta) & -\sin(\Theta) \\ \sin(\Theta) & \cos(\Theta) \end{pmatrix} \cdot \begin{pmatrix} x_{\text{old}} \\ y_{\text{old}} \end{pmatrix}$$



1. Recap linear algebra

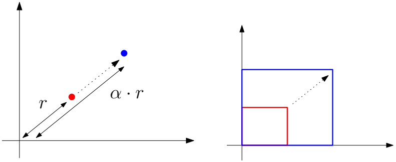
1. Translation

$$\begin{pmatrix} x_{\text{new}} \\ y_{\text{new}} \end{pmatrix} = \begin{pmatrix} x_{\text{old}} \\ y_{\text{old}} \end{pmatrix} + \begin{pmatrix} u \\ v \end{pmatrix}$$



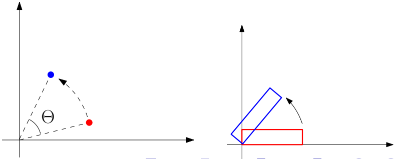
2. Scaling

$$\begin{pmatrix} x_{\text{new}} \\ y_{\text{new}} \end{pmatrix} = \begin{pmatrix} \alpha & 0 \\ 0 & \alpha \end{pmatrix} \cdot \begin{pmatrix} x_{\text{old}} \\ y_{\text{old}} \end{pmatrix}$$



3. Rotation

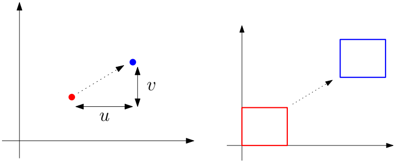
$$\begin{pmatrix} x_{\text{new}} \\ y_{\text{new}} \end{pmatrix} = \begin{pmatrix} \cos(\Theta) & -\sin(\Theta) \\ \sin(\Theta) & \cos(\Theta) \end{pmatrix} \cdot \begin{pmatrix} x_{\text{old}} \\ y_{\text{old}} \end{pmatrix}$$



1. Recap linear algebra

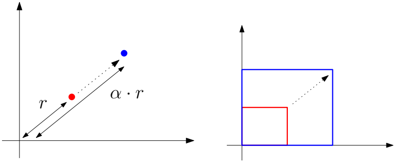
1. Translation

$$\begin{pmatrix} x_{\text{new}} \\ y_{\text{new}} \end{pmatrix} = \begin{pmatrix} x_{\text{old}} \\ y_{\text{old}} \end{pmatrix} + \begin{pmatrix} u \\ v \end{pmatrix}$$



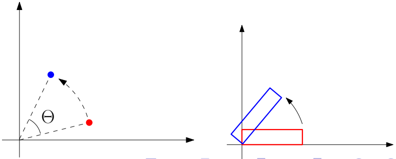
2. Scaling

$$\begin{pmatrix} x_{\text{new}} \\ y_{\text{new}} \end{pmatrix} = \begin{pmatrix} \alpha & 0 \\ 0 & \alpha \end{pmatrix} \cdot \begin{pmatrix} x_{\text{old}} \\ y_{\text{old}} \end{pmatrix}$$



3. Rotation

$$\begin{pmatrix} x_{\text{new}} \\ y_{\text{new}} \end{pmatrix} = \begin{pmatrix} \cos(\Theta) & -\sin(\Theta) \\ \sin(\Theta) & \cos(\Theta) \end{pmatrix} \cdot \begin{pmatrix} x_{\text{old}} \\ y_{\text{old}} \end{pmatrix}$$



matrix multiplications!

2. Concept: homogeneous coordinates

Goal: Representing translations as matrix multiplications

Advantages:

- composition of 2 operations = multiplication of 2 matrices
- computational efficiency
  (many known optimizations for matrix-operations)
- fewer case distinctions

2. Concept: homogeneous coordinates

Goal: Representing translations as matrix multiplications

’Tool’: homogeneous coordinates: $\begin{pmatrix} x \\ y \end{pmatrix} \rightsquigarrow \begin{pmatrix} x \\ y \\ \color{red}{1} \end{pmatrix}$

2. Concept: homogeneous coordinates

Goal: Representing translations as matrix multiplications

'Tool': homogeneous coordinates: $\begin{pmatrix} x \\ y \end{pmatrix} \rightsquigarrow \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$

- Scaling: $$\begin{pmatrix} x_{\text{new}} \\ y_{\text{new}} \end{pmatrix} = \begin{pmatrix} \alpha & 0 \\ 0 & \alpha \end{pmatrix} \cdot \begin{pmatrix} x_{\text{old}} \\ y_{\text{old}} \end{pmatrix}$$
- Rotation: $$\begin{pmatrix} x_{\text{new}} \\ y_{\text{new}} \end{pmatrix} = \begin{pmatrix} \cos(\Theta) & -\sin(\Theta) \\ \sin(\Theta) & \cos(\Theta) \end{pmatrix} \cdot \begin{pmatrix} x_{\text{old}} \\ y_{\text{old}} \end{pmatrix}$$

2. Concept: homogeneous coordinates

Goal: Representing translations as matrix multiplications

'Tool': homogeneous coordinates: $\begin{pmatrix} x \\ y \end{pmatrix} \rightsquigarrow \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$

- Scaling: $\begin{pmatrix} x_{\text{new}} \\ y_{\text{new}} \\ 1 \end{pmatrix} = \begin{pmatrix} \alpha & 0 & 0 \\ 0 & \alpha & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x_{\text{old}} \\ y_{\text{old}} \\ 1 \end{pmatrix}$
- Rotation: $\begin{pmatrix} x_{\text{new}} \\ y_{\text{new}} \\ \end{pmatrix} = \begin{pmatrix} \cos(\Theta) & -\sin(\Theta) & \\ \sin(\Theta) & \cos(\Theta) & \\ & & \end{pmatrix} \cdot \begin{pmatrix} x_{\text{old}} \\ y_{\text{old}} \\ \end{pmatrix}$

2. Concept: homogeneous coordinates

Goal: Representing translations as matrix multiplications

'Tool': homogeneous coordinates: $\begin{pmatrix} x \\ y \end{pmatrix} \rightsquigarrow \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$

- Scaling: $\begin{pmatrix} x_{\text{new}} \\ y_{\text{new}} \\ 1 \end{pmatrix} = \begin{pmatrix} \alpha & 0 & 0 \\ 0 & \alpha & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x_{\text{old}} \\ y_{\text{old}} \\ 1 \end{pmatrix}$
- Rotation: $\begin{pmatrix} x_{\text{new}} \\ y_{\text{new}} \\ 1 \end{pmatrix} = \begin{pmatrix} \cos(\Theta) & -\sin(\Theta) & 0 \\ \sin(\Theta) & \cos(\Theta) & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x_{\text{old}} \\ y_{\text{old}} \\ 1 \end{pmatrix}$

2. Concept: homogeneous coordinates

Goal: Representing translations as matrix multiplications

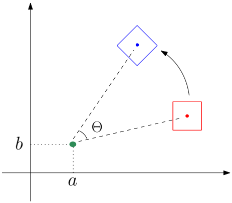
'Tool': homogeneous coordinates: $\begin{pmatrix} x \\ y \end{pmatrix} \rightsquigarrow \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$

- Scaling: $\begin{pmatrix} x_{\text{new}} \\ y_{\text{new}} \\ 1 \end{pmatrix} = \begin{pmatrix} \alpha & 0 & 0 \\ 0 & \alpha & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x_{\text{old}} \\ y_{\text{old}} \\ 1 \end{pmatrix}$
- Rotation: $\begin{pmatrix} x_{\text{new}} \\ y_{\text{new}} \\ 1 \end{pmatrix} = \begin{pmatrix} \cos(\Theta) & -\sin(\Theta) & 0 \\ \sin(\Theta) & \cos(\Theta) & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x_{\text{old}} \\ y_{\text{old}} \\ 1 \end{pmatrix}$
- Translation: $\begin{pmatrix} x_{\text{new}} \\ y_{\text{new}} \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & u \\ 0 & 1 & v \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x_{\text{old}} \\ y_{\text{old}} \\ 1 \end{pmatrix}$

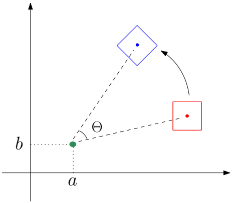
Remark 1: There are many more beautiful and also practical aspects of homogeneous coordinates
(relating, e.g., to projective geometry, algebra and topology)

Remark 2: The formulas of the previous slides can be adapted to corresponding operations in 3-D.

Often used transformation:
Rotation around a given point.

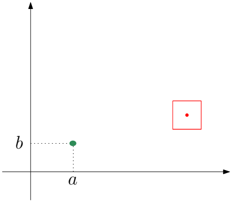


2. Concept: homogeneous coordinates



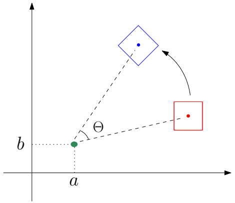
Often used transformation:
Rotation around a given point.

Decomposed into basic operations:



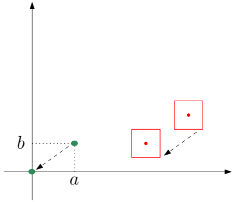
2. Concept: homogeneous coordinates

Often used transformation:
Rotation around a given point.



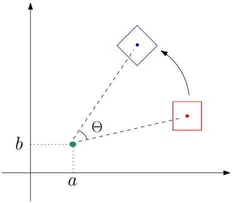
Decomposed into basic operations:

$$\begin{pmatrix} x_{\text{new}} \\ y_{\text{new}} \\ 1 \end{pmatrix} = \underbrace{\begin{pmatrix} 1 & 0 & -a \\ 0 & 1 & -b \\ 0 & 0 & 1 \end{pmatrix}}_{\text{translation by } (-a;\, -b)} \cdot \begin{pmatrix} x_{\text{old}} \\ y_{\text{old}} \\ 1 \end{pmatrix}$$



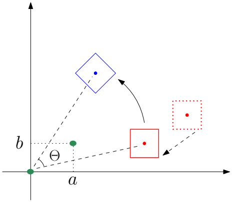
2. Concept: homogeneous coordinates

Often used transformation:
Rotation around a given point.



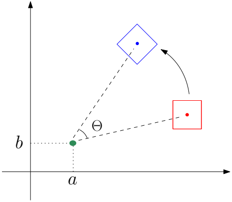
Decomposed into basic operations:

$$\begin{pmatrix} x_{\text{new}} \\ y_{\text{new}} \\ 1 \end{pmatrix} = \underbrace{\begin{pmatrix} \cos(\Theta) & -\sin(\Theta) & 0 \\ \sin(\Theta) & \cos(\Theta) & 0 \\ 0 & 0 & 1 \end{pmatrix}}_{\text{rotation around origin}} \cdot \underbrace{\begin{pmatrix} 1 & 0 & -a \\ 0 & 1 & -b \\ 0 & 0 & 1 \end{pmatrix}}_{\text{translation by } (-a;\, -b)} \cdot \begin{pmatrix} x_{\text{old}} \\ y_{\text{old}} \\ 1 \end{pmatrix}$$



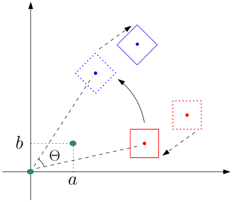
2. Concept: homogeneous coordinates

Often used transformation:
Rotation around a given point.



Decomposed into basic operations:

$$\begin{pmatrix} x_{\text{new}} \\ y_{\text{new}} \\ 1 \end{pmatrix} = \underbrace{\begin{pmatrix} 1 & 0 & a \\ 0 & 1 & b \\ 0 & 0 & 1 \end{pmatrix}}_{\text{translation by } (a;\, b)} \cdot \underbrace{\begin{pmatrix} \cos(\Theta) & -\sin(\Theta) & 0 \\ \sin(\Theta) & \cos(\Theta) & 0 \\ 0 & 0 & 1 \end{pmatrix}}_{\text{rotation around origin}} \cdot \underbrace{\begin{pmatrix} 1 & 0 & -a \\ 0 & 1 & -b \\ 0 & 0 & 1 \end{pmatrix}}_{\text{translation by } (-a;\, -b)} \cdot \begin{pmatrix} x_{\text{old}} \\ y_{\text{old}} \\ 1 \end{pmatrix}$$



2. Concept: homogeneous coordinates

Short Summary: Operations with homogeneous coordinates

1. Translation by $(u; v)$: $\begin{pmatrix} x_{\text{new}} \\ y_{\text{new}} \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & u \\ 0 & 1 & v \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x_{\text{old}} \\ y_{\text{old}} \\ 1 \end{pmatrix}$

2. Scaling by a factor α: $\begin{pmatrix} x_{\text{new}} \\ y_{\text{new}} \\ 1 \end{pmatrix} = \begin{pmatrix} \alpha & 0 & 0 \\ 0 & \alpha & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x_{\text{old}} \\ y_{\text{old}} \\ 1 \end{pmatrix}$

3. Rotation around a point $(a; b)$

$$\begin{pmatrix} x_{\text{new}} \\ y_{\text{new}} \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & a \\ 0 & 1 & b \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} \cos(\Theta) & -\sin(\Theta) & 0 \\ \sin(\Theta) & \cos(\Theta) & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 & -a \\ 0 & 1 & -b \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_{\text{old}} \\ y_{\text{old}} \\ 1 \end{pmatrix}$$

3. Application: animations

Next Steps: Implementing some animations in Python using the techniques from the previous slides:

1. Warmup – two short examples
2. Animation of Earth, Sun and Moon

Animations: warmup

Example 1: An object starting at the origin and moving by $(0.1, 0.05)$ in every time step
Result: see short demo

Pseudo-Code

$(x, y) = (0, 0)$ // initialization
$n = 1000$ // number of iterations (arbitrarily chosen number)
$x\_step := 0.1, \quad y\_step := 0.05$
for $i = 1$ **to** n
$\{$
$(x_{\text{new}}, y_{\text{new}}) := (\ x_{\text{old}} + x\_step,\ y_{\text{old}} + y\_step\ )$
plot$(x_{\text{new}}, y_{\text{new}})$
wait for a short moment
$\}$
end for

Animations: warmup

Example 1: An object starting at the origin and moving by $(0.1, 0.05)$ in every time step
Python Code: see file `moving_point` (available on moodle)
Some Python-Commands

- plt.plot(x, y, 'b$*$'): displays a blue star at point (x, y)

Some specifiers for color/shape

Shape

$*$	asterisk
o	circle
.	point
x	cross

Color

b	blue
r	red
m	magenta
k	black

More options can be found at matplotlib.org/stable/contents.html
Hint: Comment 'cla' out and consider the result

Animations: warmup

Exercise

For references look at matplotlib.org/stable/api/animation_api.html

1. Adapt the code of `moving_point` such that ...
   - the speed is twice as large as before
   - the moving object has a different shape (e.g. a circle or a cross) and a different color
2. What happens if you remove the axis-command?

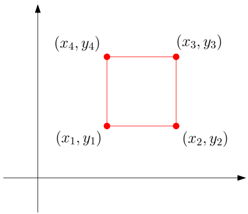
Solutions: see blackboard

Animations: warmup

Example 2: A square of a given side length moving by $[0.1, 0.05]$ in every time step.
Result: see short demo

Exercise: Adapt the program for Example 1 such that a square is moving by $[0.1, 0.05]$ in every time step.
A code skeleton can be found in the file `moving_square_skeleton`

Hint: plotting the rectangle depicted below



can be obtained via the Python command
plt.plot($[x_1 \;\; x_2 \;\; x_3 \;\; x_4 \;\; x_1]$, $[y_1 \;\; y_2 \;\; y_3 \;\; y_4 \;\; y_1]$, 'r')

Animation of Sun, Earth and Moon

- You will implement (step by step) an animation of Sun, Earth and Moon.
- A detailed description can be found on the Exercise Sheet.