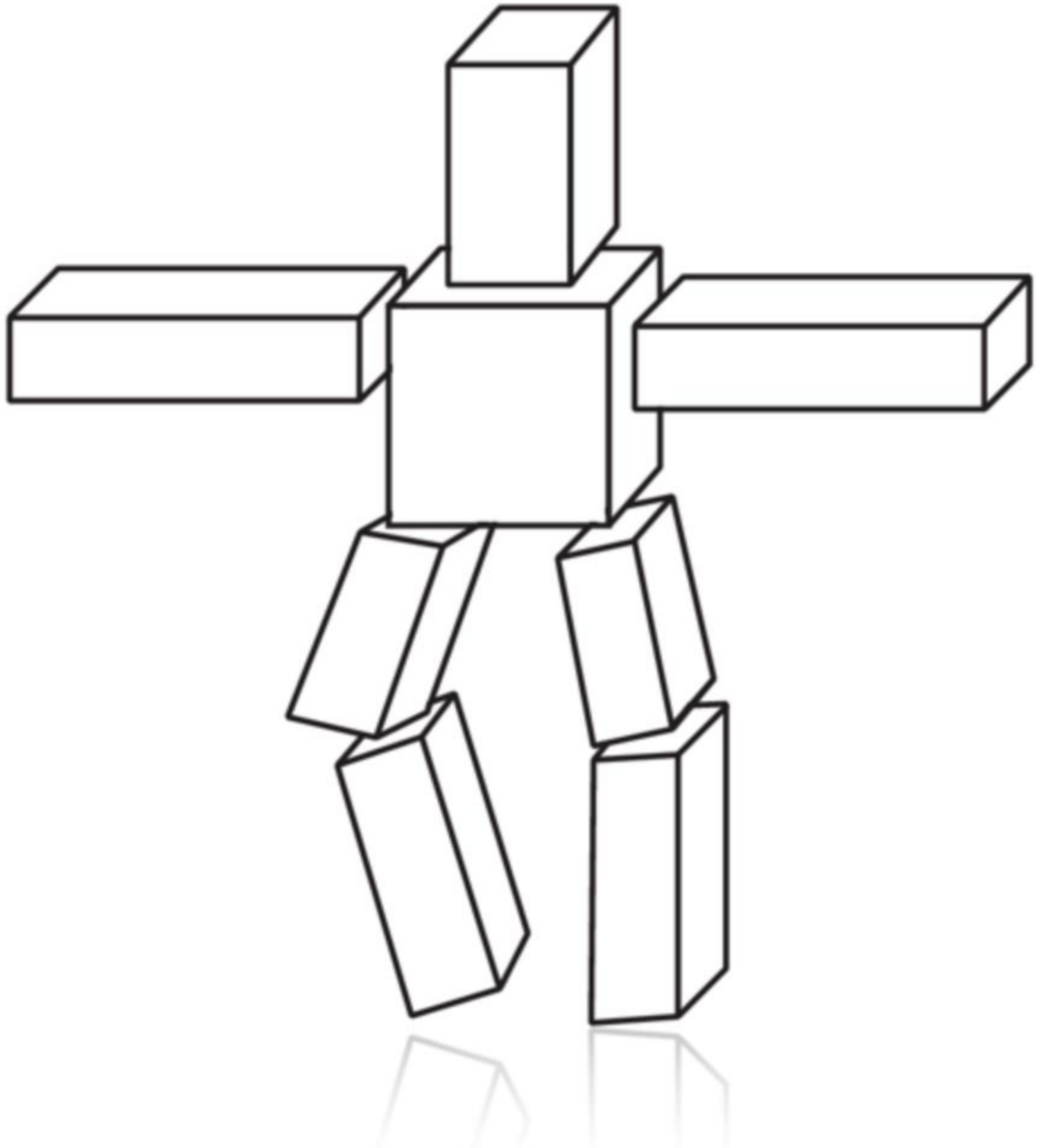
Lecture 4:
Transforms

Computer Graphics
CMU 15-462/15-662, Fall 2016

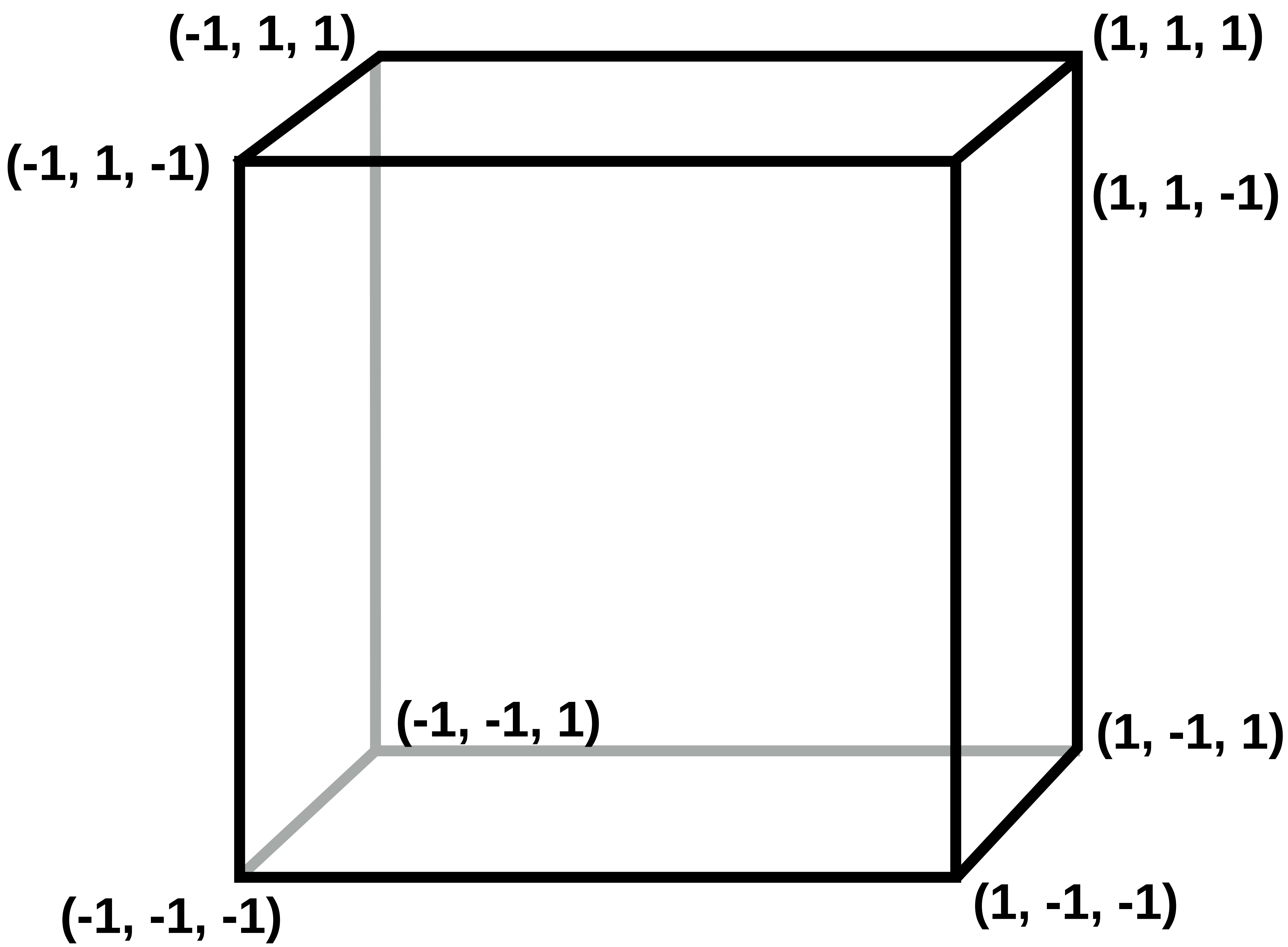
Brief recap from last class

- How to draw a triangle
  - Why focus on triangles, and not quads, pentagons, etc?
  - What was specific to triangles in what we discussed last class?

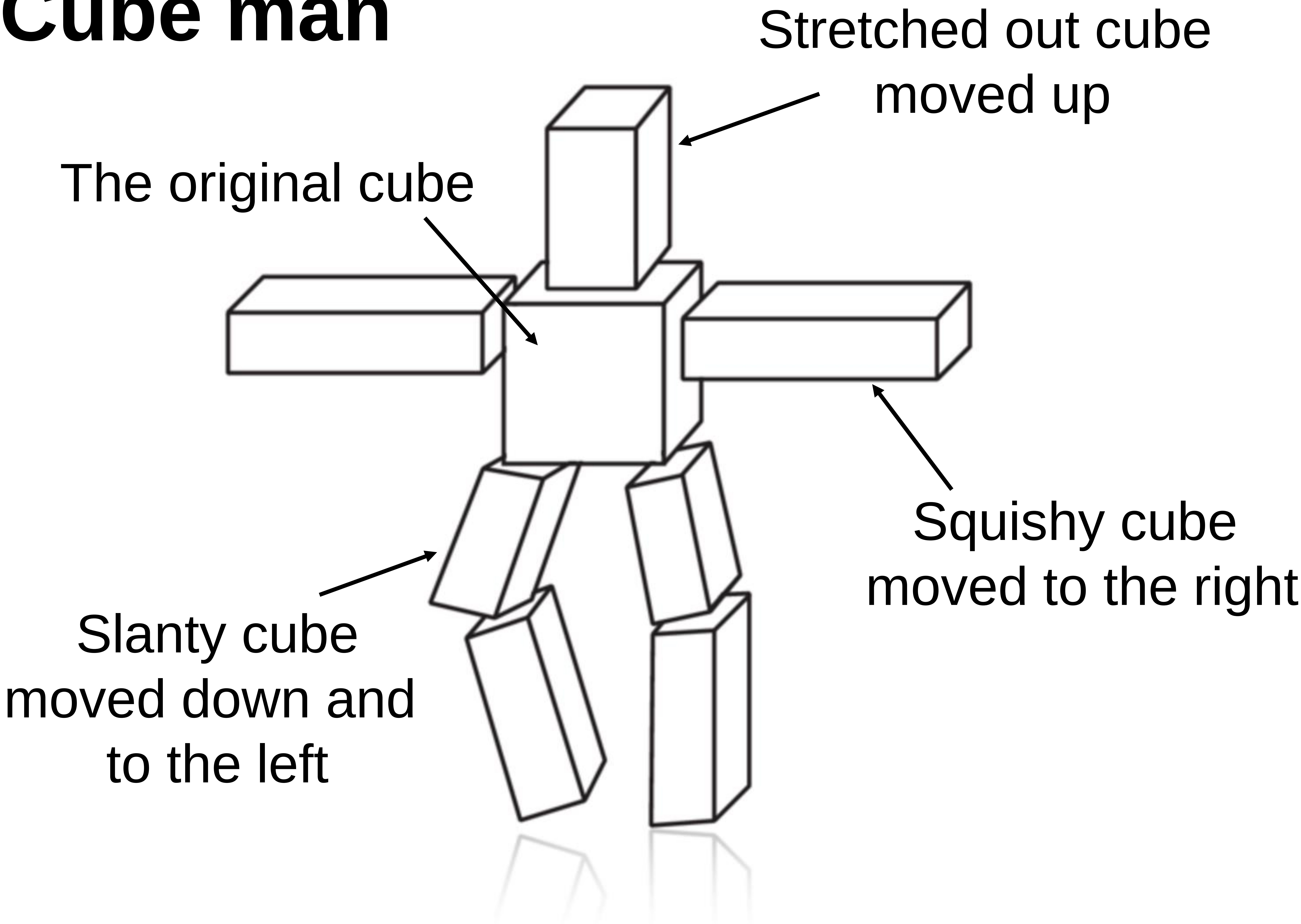
What in the world is this?



Cube

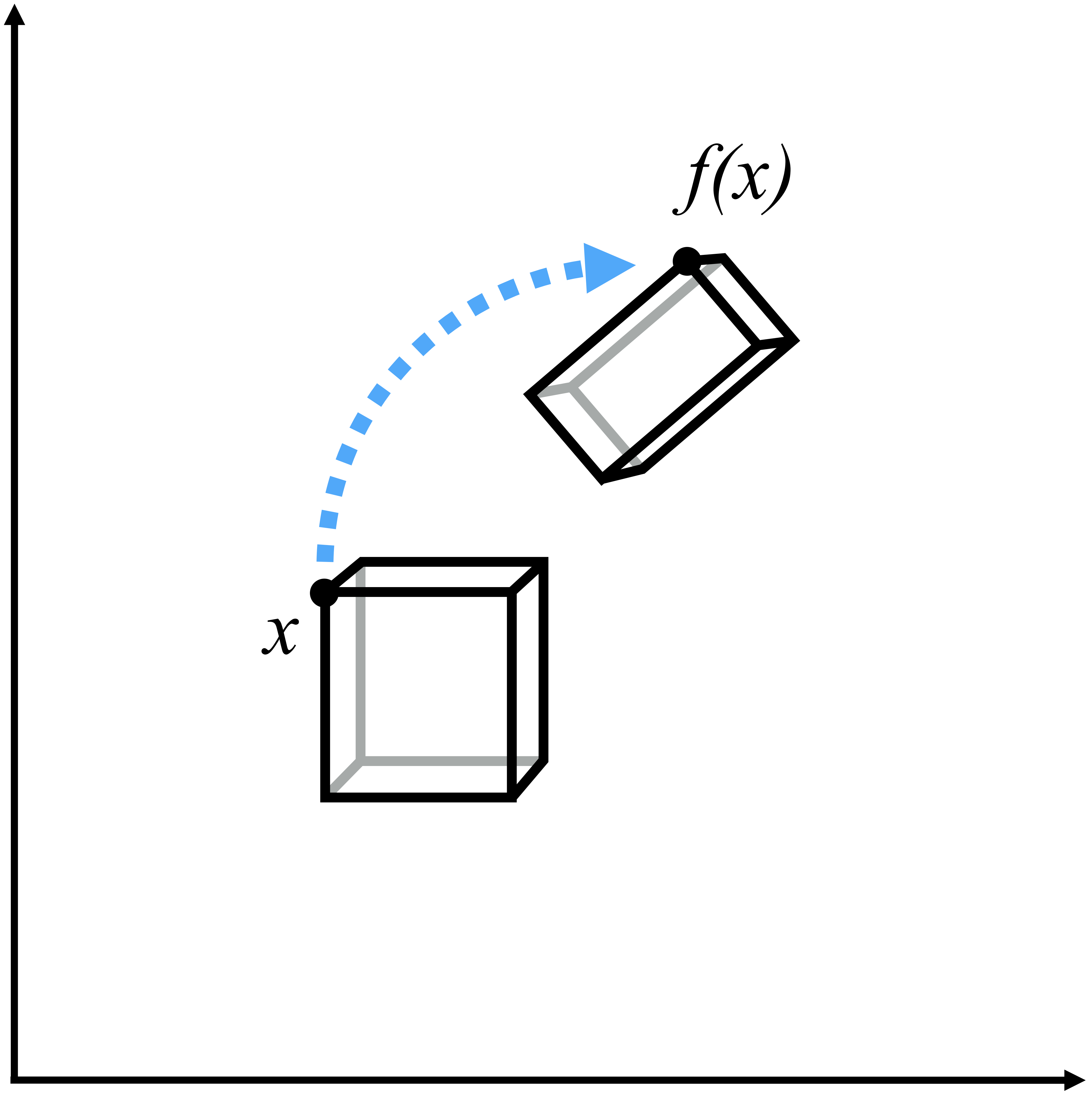


Cube man



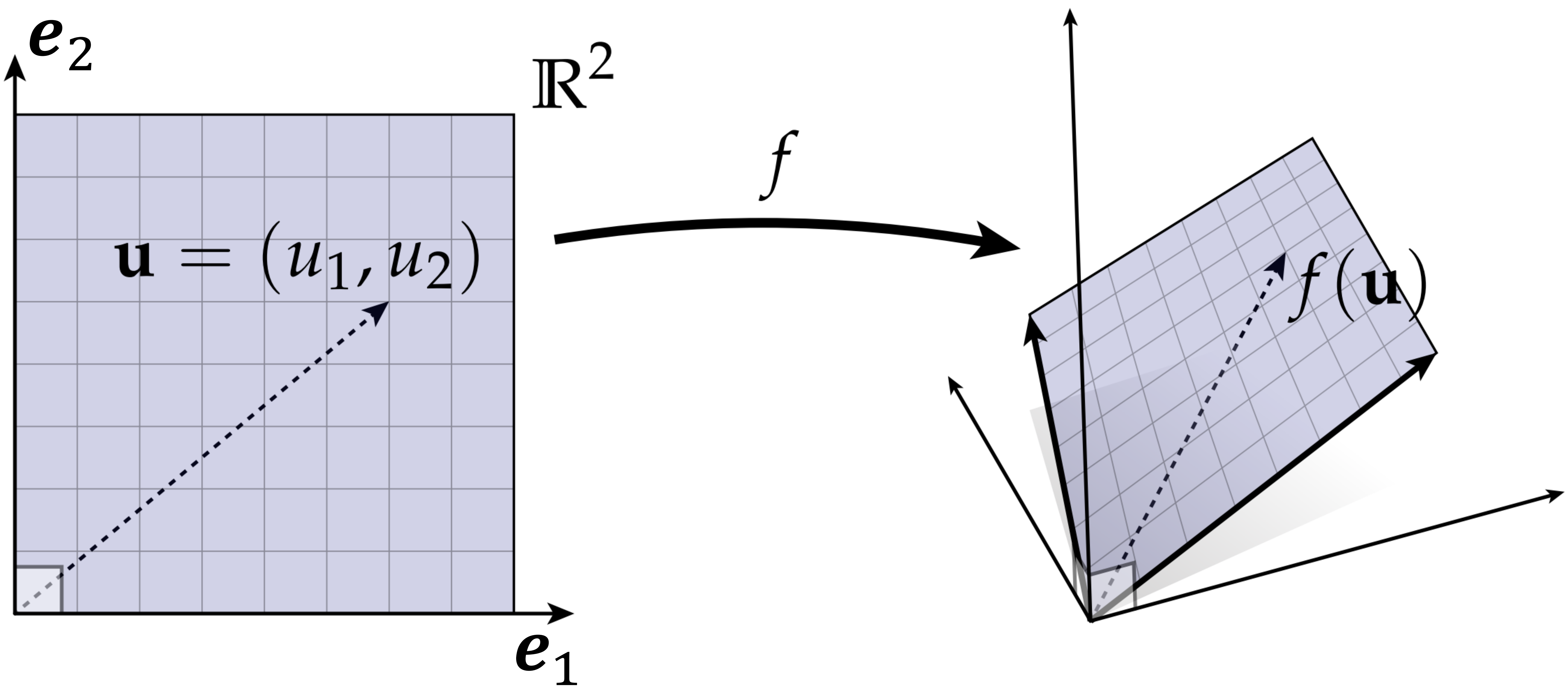
Stretched out cube
moved up
The original cube
Squishy cube
moved to the right
Slanty cube
moved down and
to the left

f transforms x to $f(x)$



And what is our favorite type of transformation?

Linear transforms



- **But what does it mean?**

Linear transforms

$$f(\boldsymbol{u} + \boldsymbol{v}) = f(\boldsymbol{u}) + f(\boldsymbol{v})$$

$$f(a\boldsymbol{u}) = af(\boldsymbol{u})$$

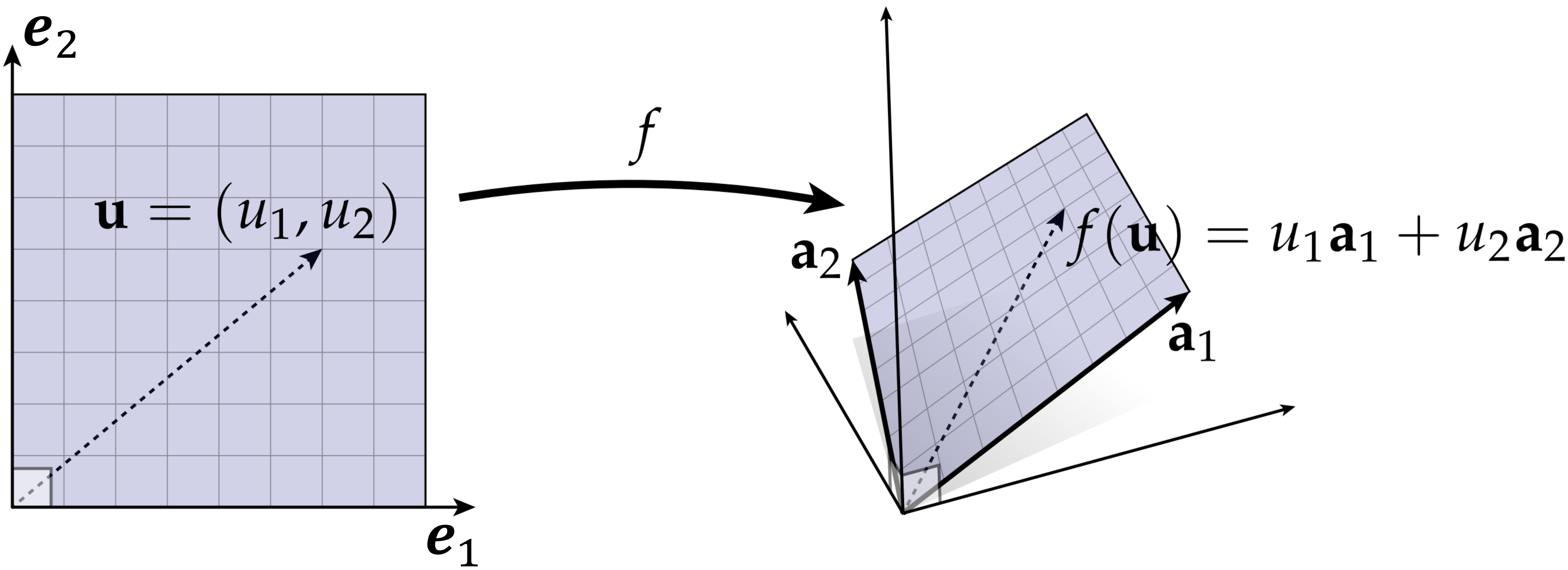
Linear transforms

If a map can be expressed as

$$f(\boldsymbol{u}) = \sum_{i=1}^{m} u_i \boldsymbol{a}_i$$

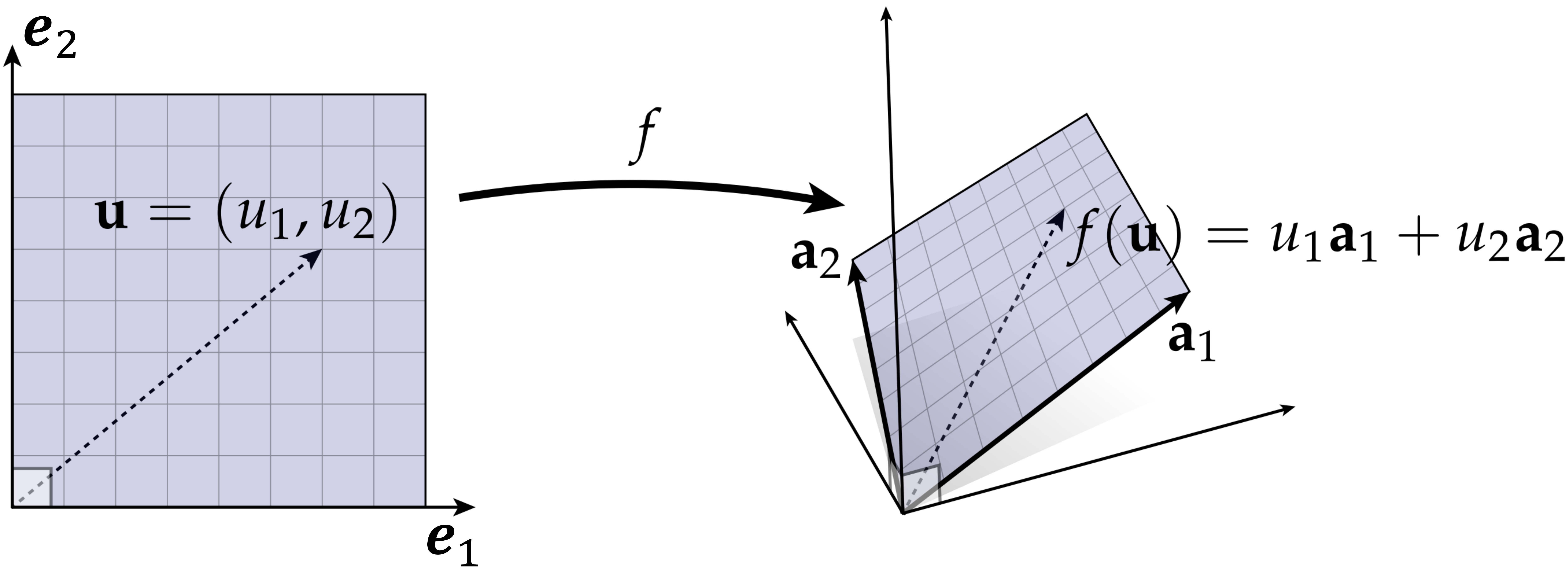
with fixed vectors $\boldsymbol{a}_i$, then it is linear

Linear transforms



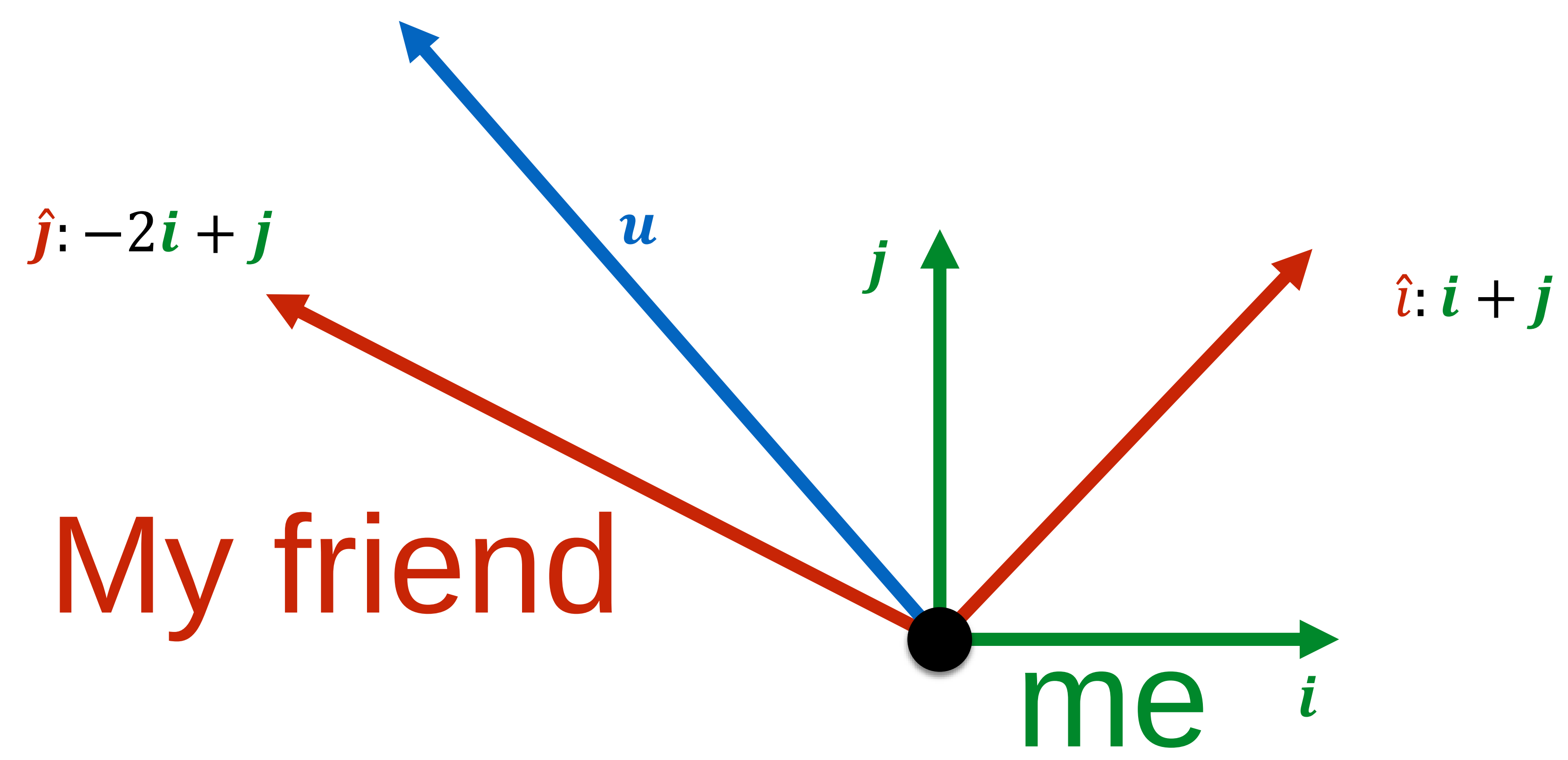
- **Do you know…**
  - **what u_1 and u_2 are?**
  - **what a_1 and a_2 are?**

Linear transforms



- $\boldsymbol{u}$ **is a linear combination of** $\boldsymbol{e}_1$ **and** $\boldsymbol{e}_2$
- $\boldsymbol{f}(\boldsymbol{u})$ **is that same linear combination of** $\boldsymbol{a}_1$ **and** $\boldsymbol{a}_2$
- $\boldsymbol{a}_1$ **and** $\boldsymbol{a}_2$ **are** $\boldsymbol{f}(\boldsymbol{e}_1)$ **and** $\mathbf{f}(\boldsymbol{e}_2)$
- **by knowing what** $\boldsymbol{e}_1$ **and** $\boldsymbol{e}_2$ **map to, you know how to map the entire space!**

An example: Coordinate transformations



My friend says, look at 3 o'clock (in their coordinate frame that means one "forward" and one to the "right")!

Where should I look?

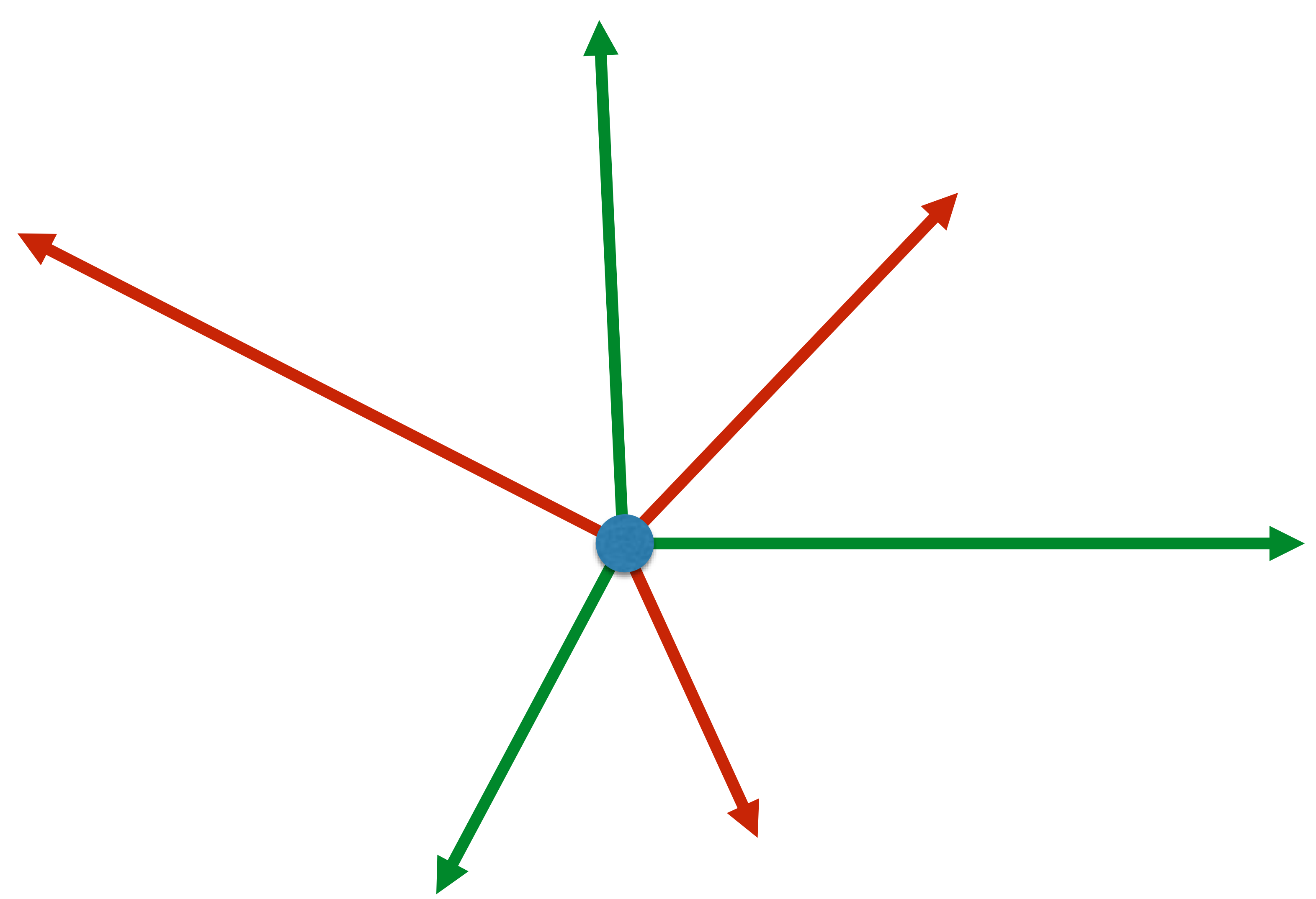
Direction in my friend's coordinate frame

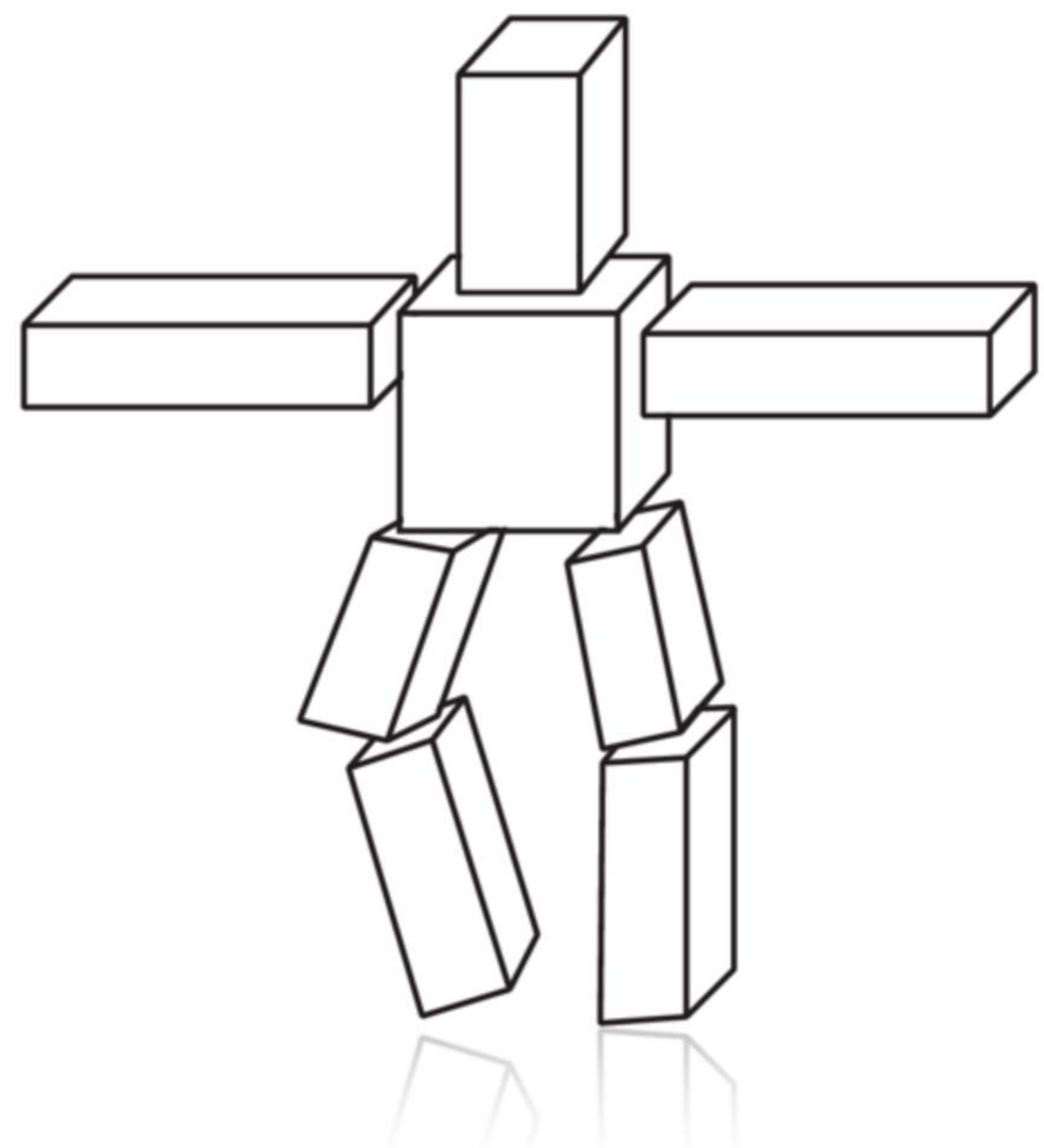
$$\boldsymbol{f}(\boldsymbol{u}) = \boldsymbol{f}(u_1 \hat{\boldsymbol{\imath}} + u_2 \hat{\boldsymbol{\jmath}}) = u_1 \boldsymbol{f}(\hat{\boldsymbol{\imath}}) + u_2 \boldsymbol{f}(\hat{\boldsymbol{\jmath}}) = u_1 \begin{bmatrix} 1 \\ 1 \end{bmatrix} + u_2 \begin{bmatrix} -2 \\ 1 \end{bmatrix}$$

Same direction in my coordinate frame

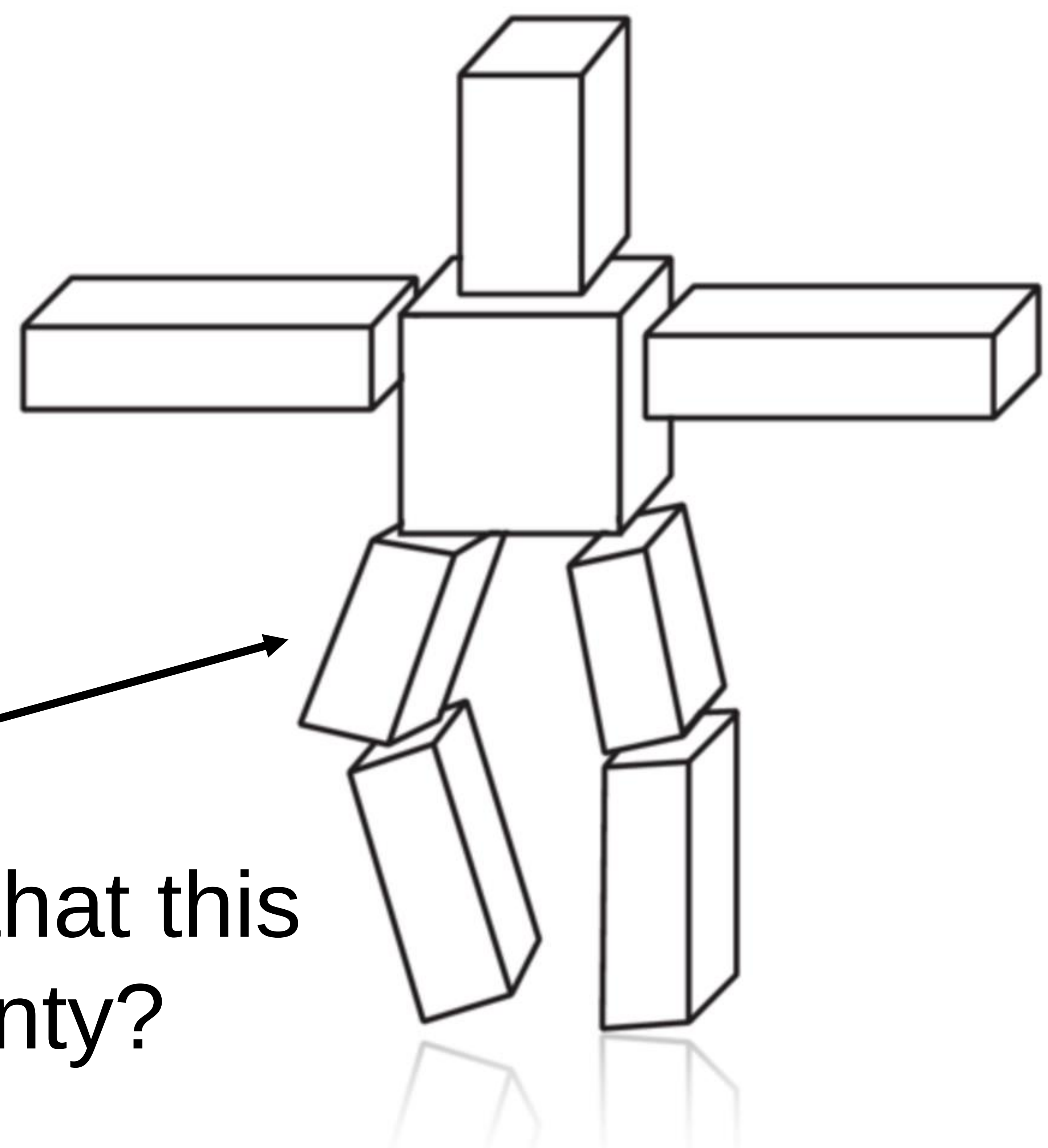
Linear maps

- In graphics we often talk about changing coordinate frames (go from local to world to camera to screen coordinates)
- Equally useful to think about maps transforming a space (and everything in it!)



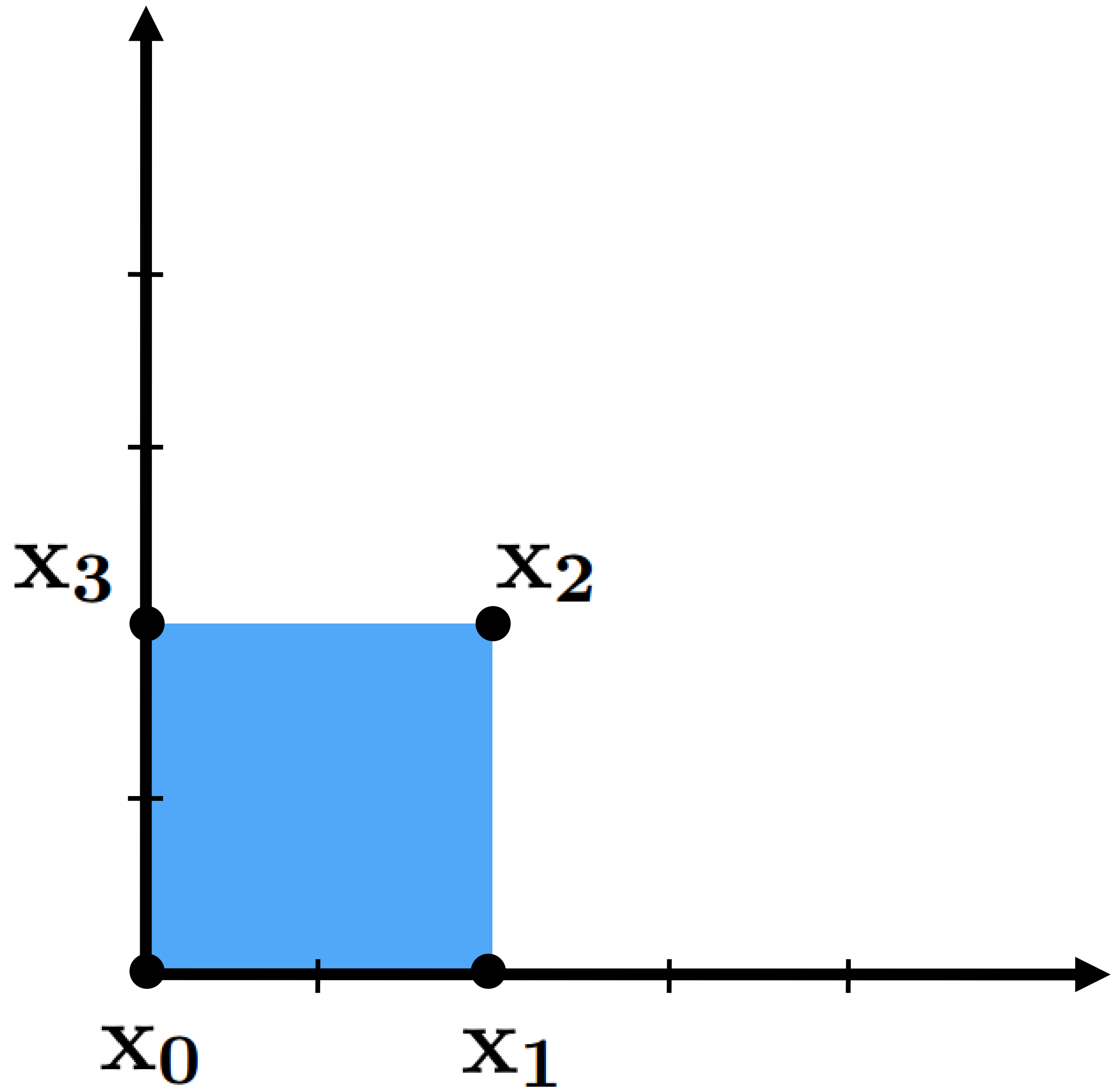


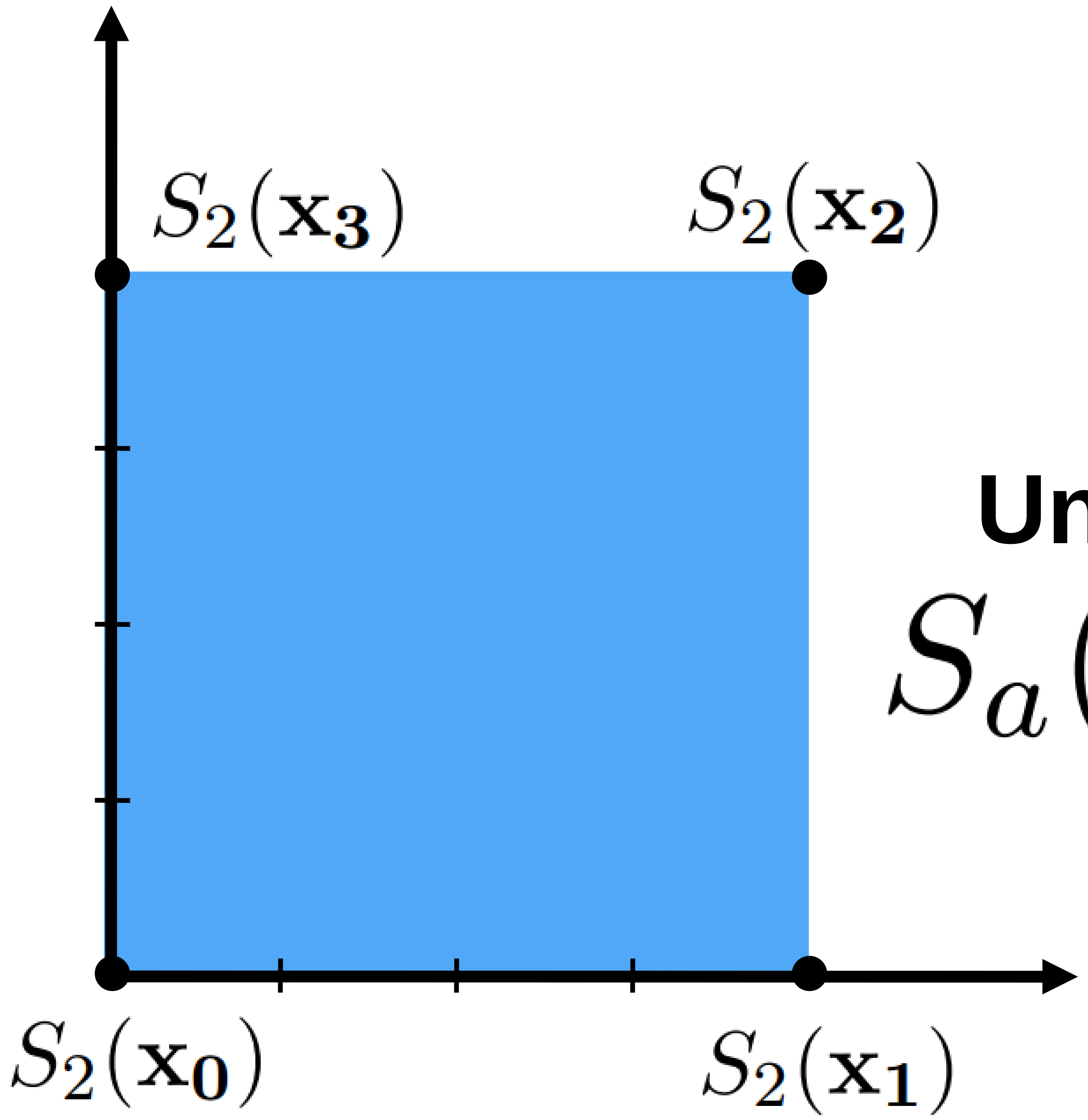
Let's look at some transforms that are important in graphics…



How do you formally tell a computer that this cube should be squished and slanty?

Scale



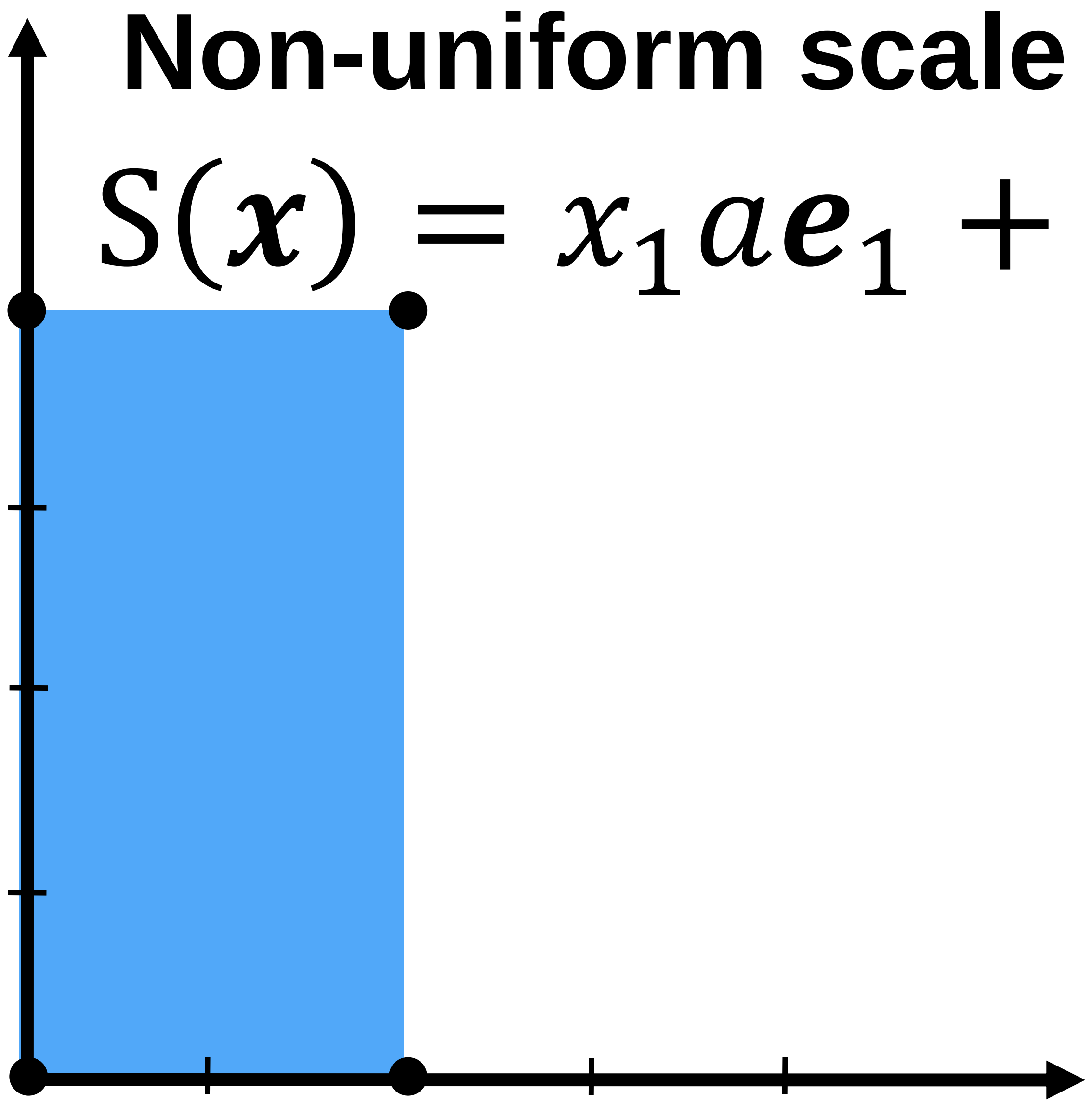


Uniform scale:

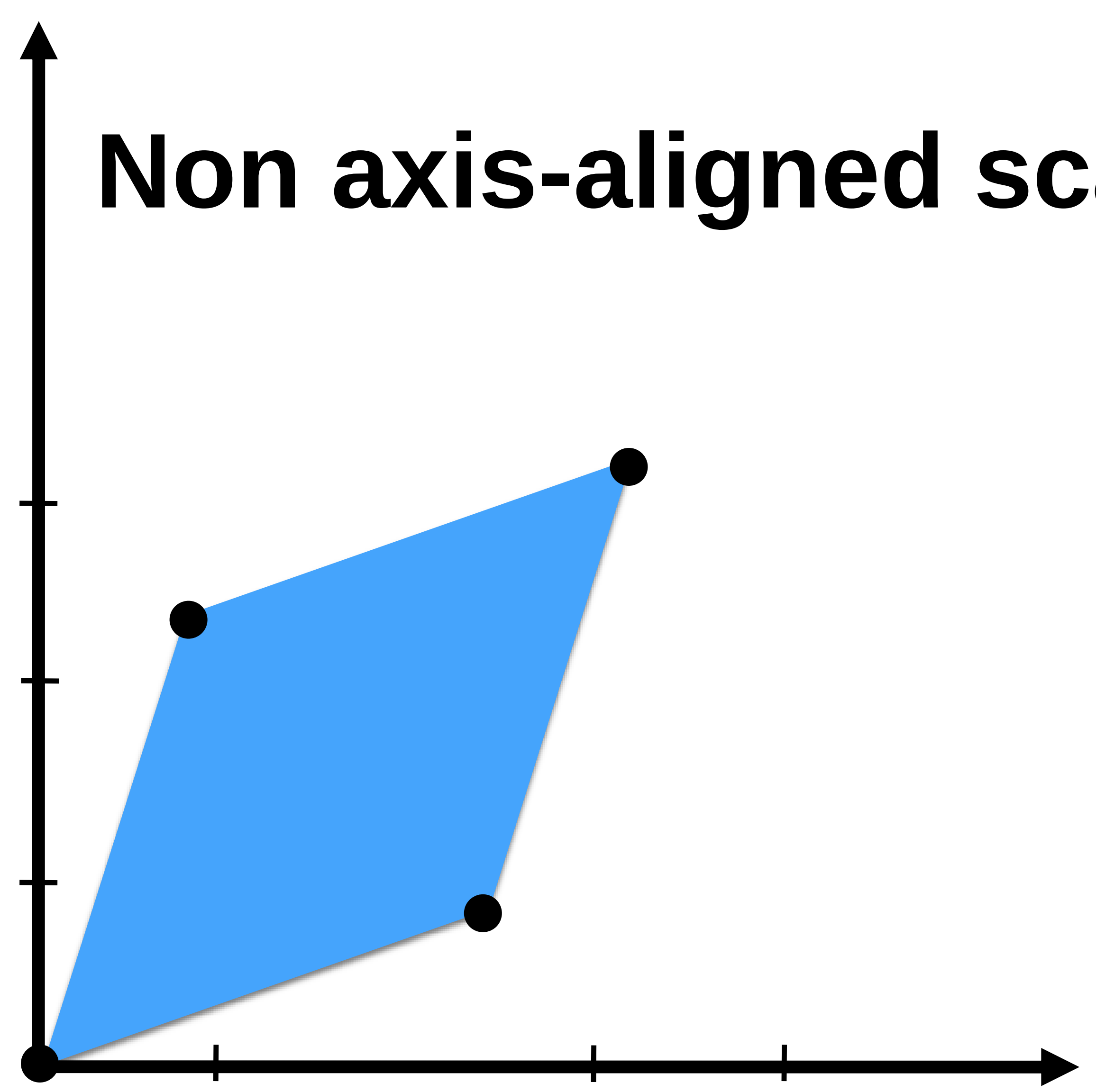
$$S_a(\mathbf{x}) = a\mathbf{x}$$

Non-uniform scale

$$S(\boldsymbol{x}) = x_1 a\boldsymbol{e}_1 + x_2 b\boldsymbol{e}_2$$

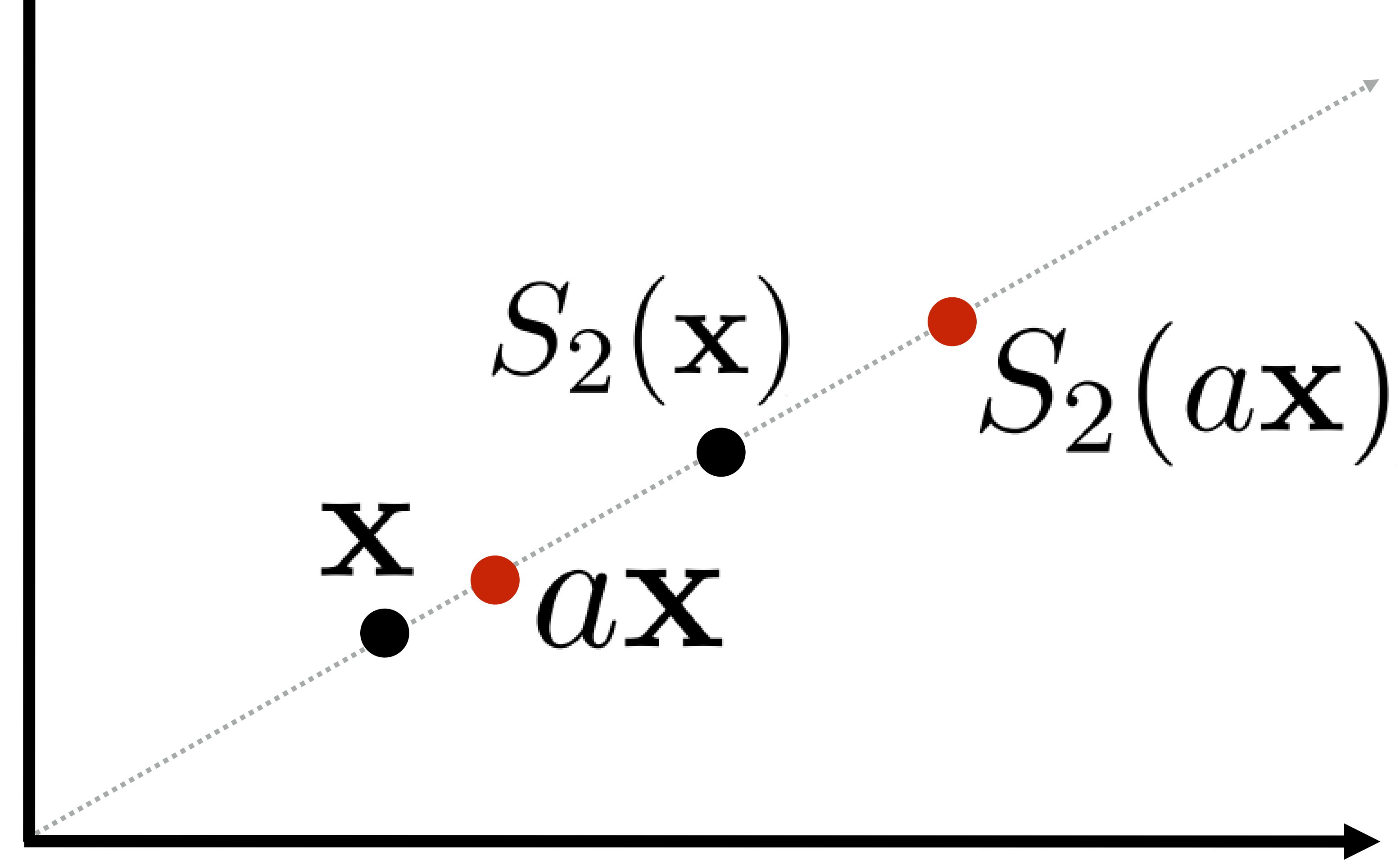


Non axis-aligned scaling??

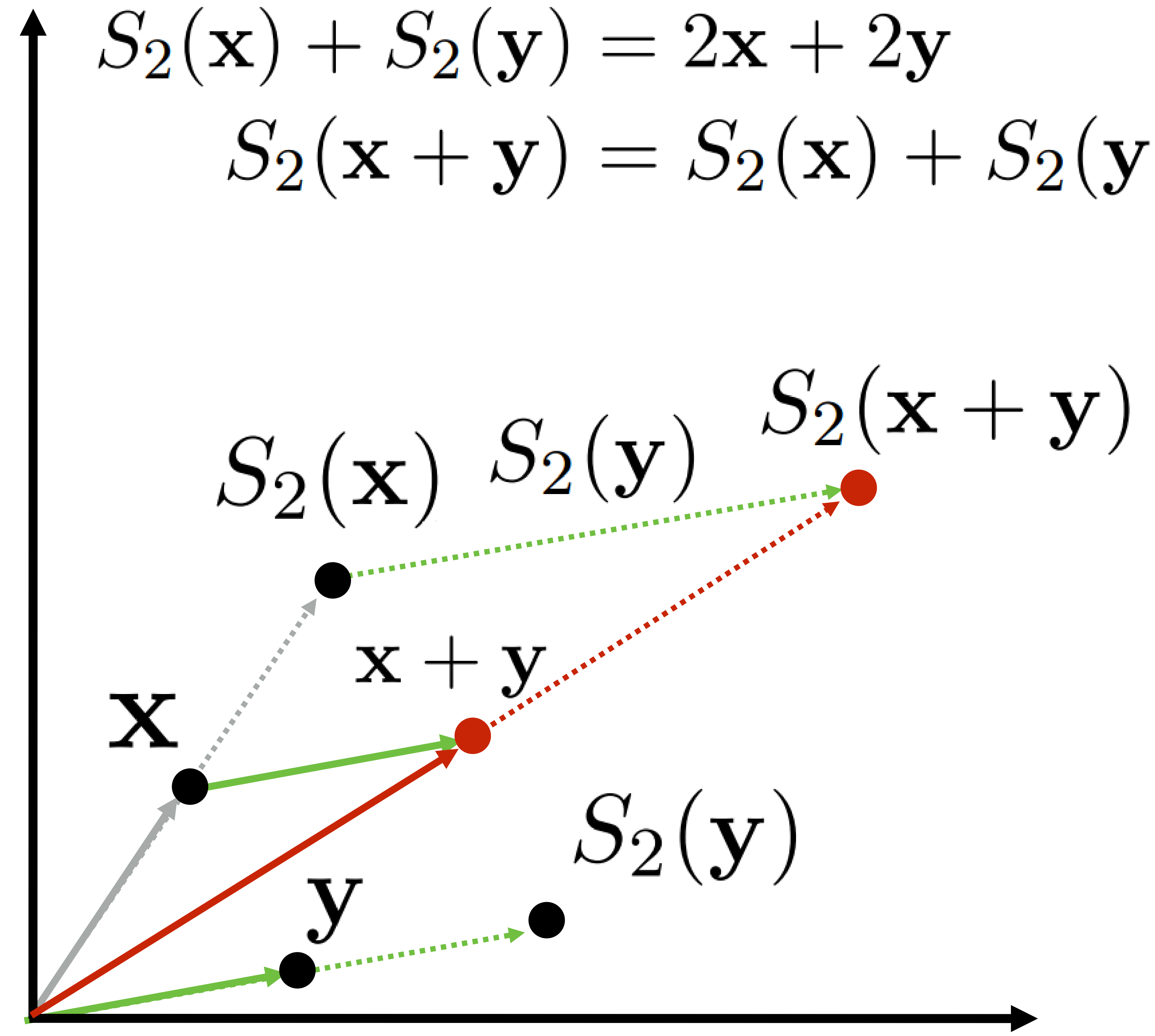


Is uniform scale a linear transform?

$$S_2(\mathbf{x}) = 2\mathbf{x}$$
$$S_2(a\mathbf{x}) = 2a\mathbf{x}$$
$$aS_2(\mathbf{x}) = 2a\mathbf{x}$$
$$S_2(a\mathbf{x}) = aS_2(\mathbf{x})$$

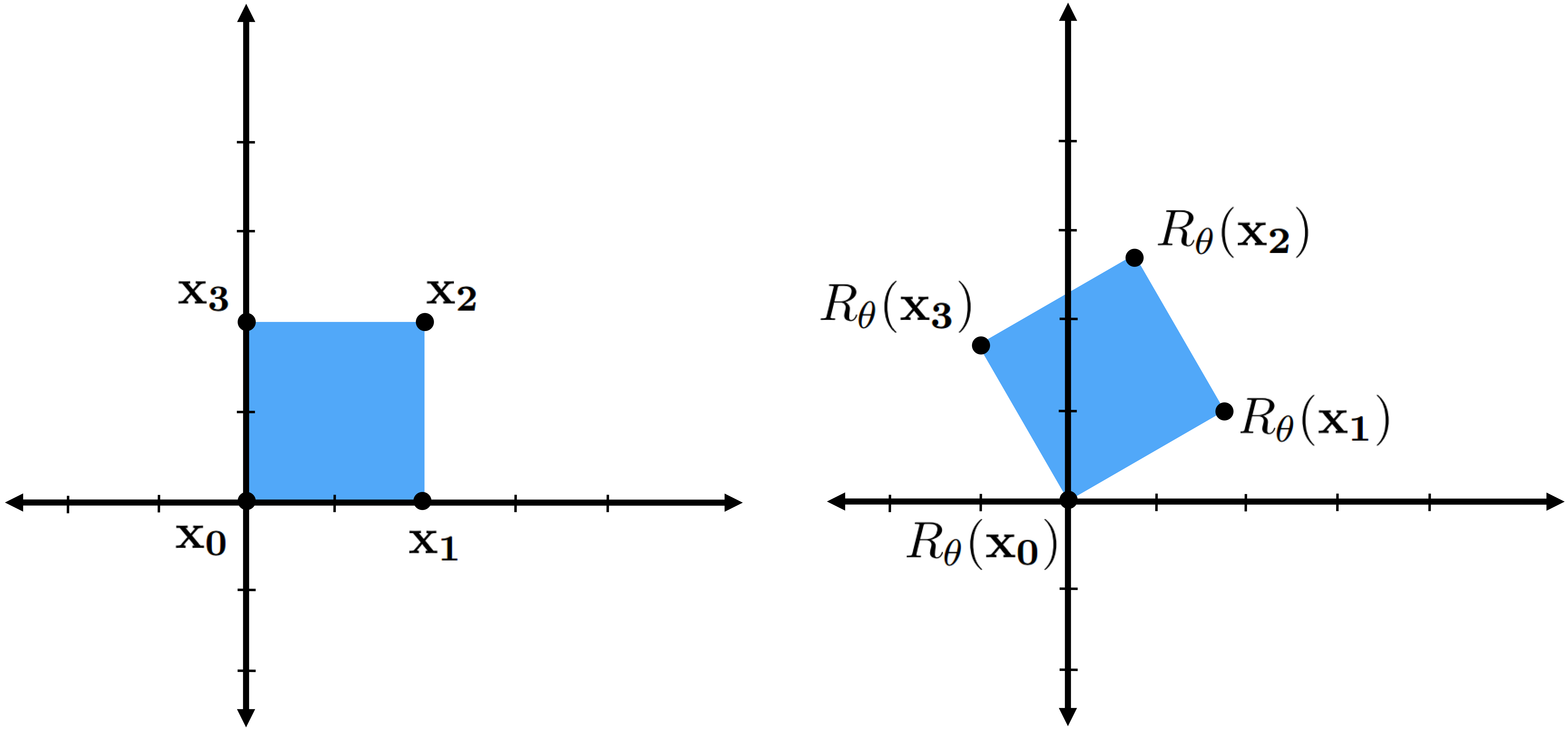


$$S_2(\mathbf{x}+\mathbf{y}) = 2(\mathbf{x}+\mathbf{y})$$
$$S_2(\mathbf{x}) + S_2(\mathbf{y}) = 2\mathbf{x} + 2\mathbf{y}$$
$$S_2(\mathbf{x}+\mathbf{y}) = S_2(\mathbf{x}) + S_2(\mathbf{y})$$



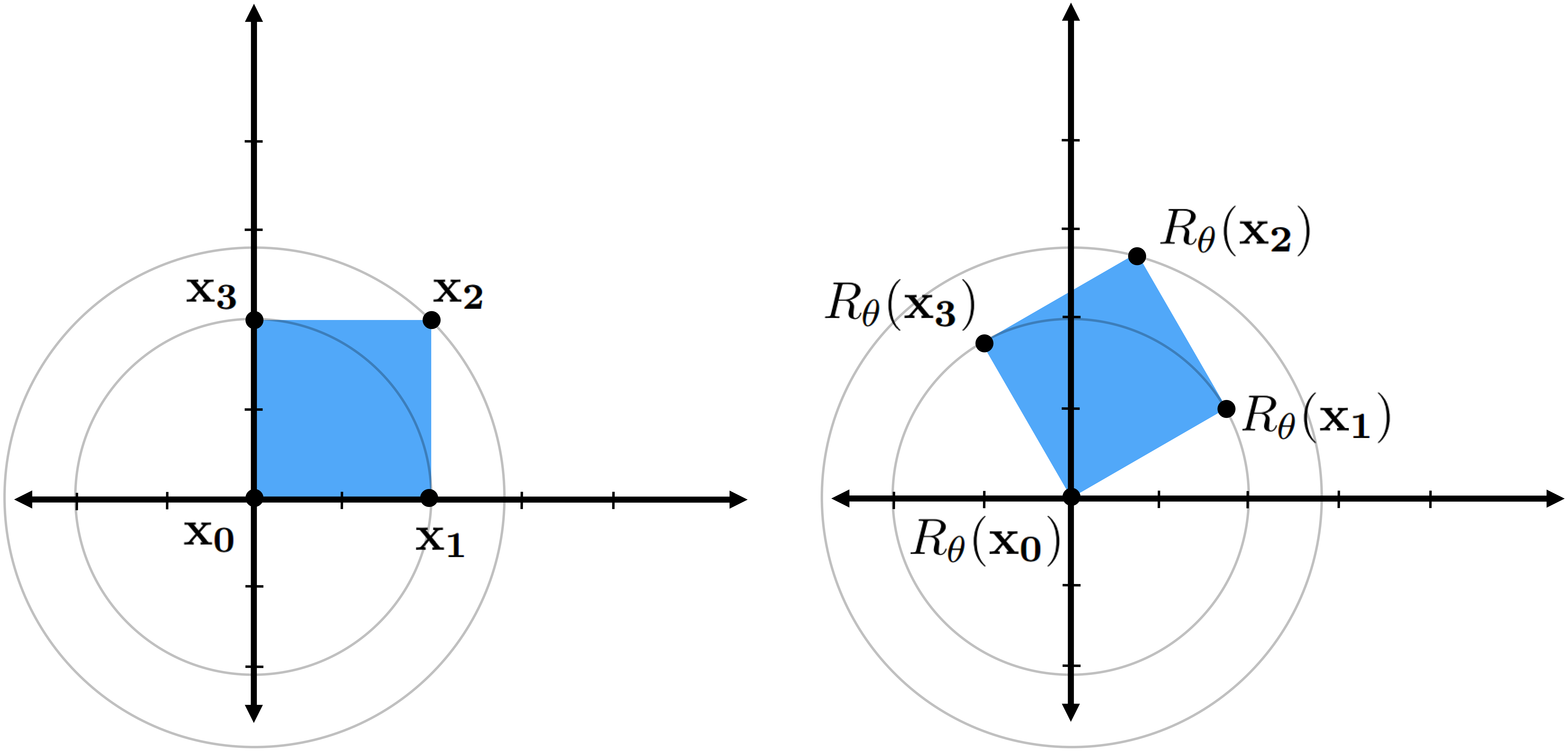
Yes!

Rotation



R_θ **= rotate counter-clockwise by** θ

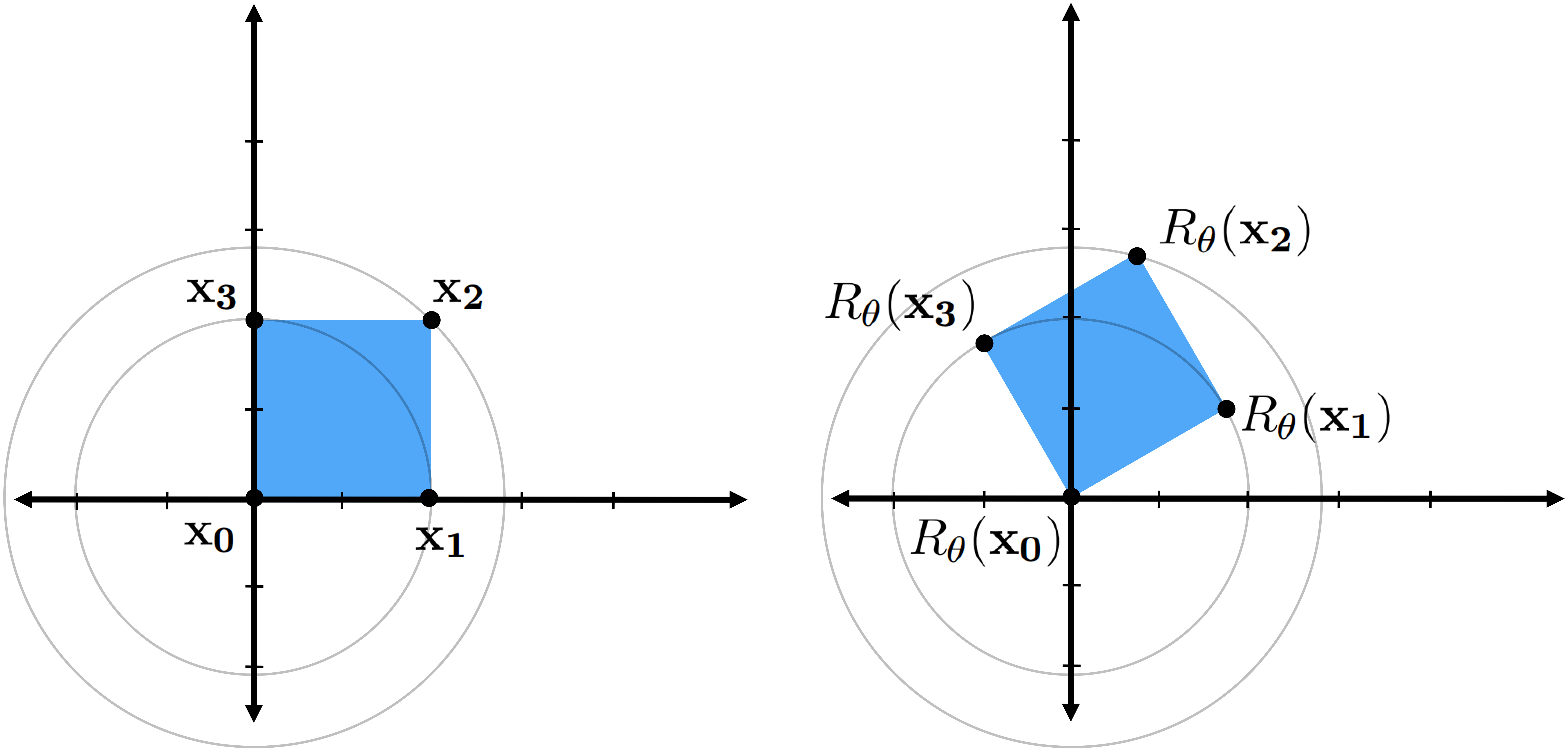
Rotation



R_θ = rotate counter-clockwise by θ

As angle changes, points move along *circular* trajectories.

Rotation



R_θ = rotate counter-clockwise by θ

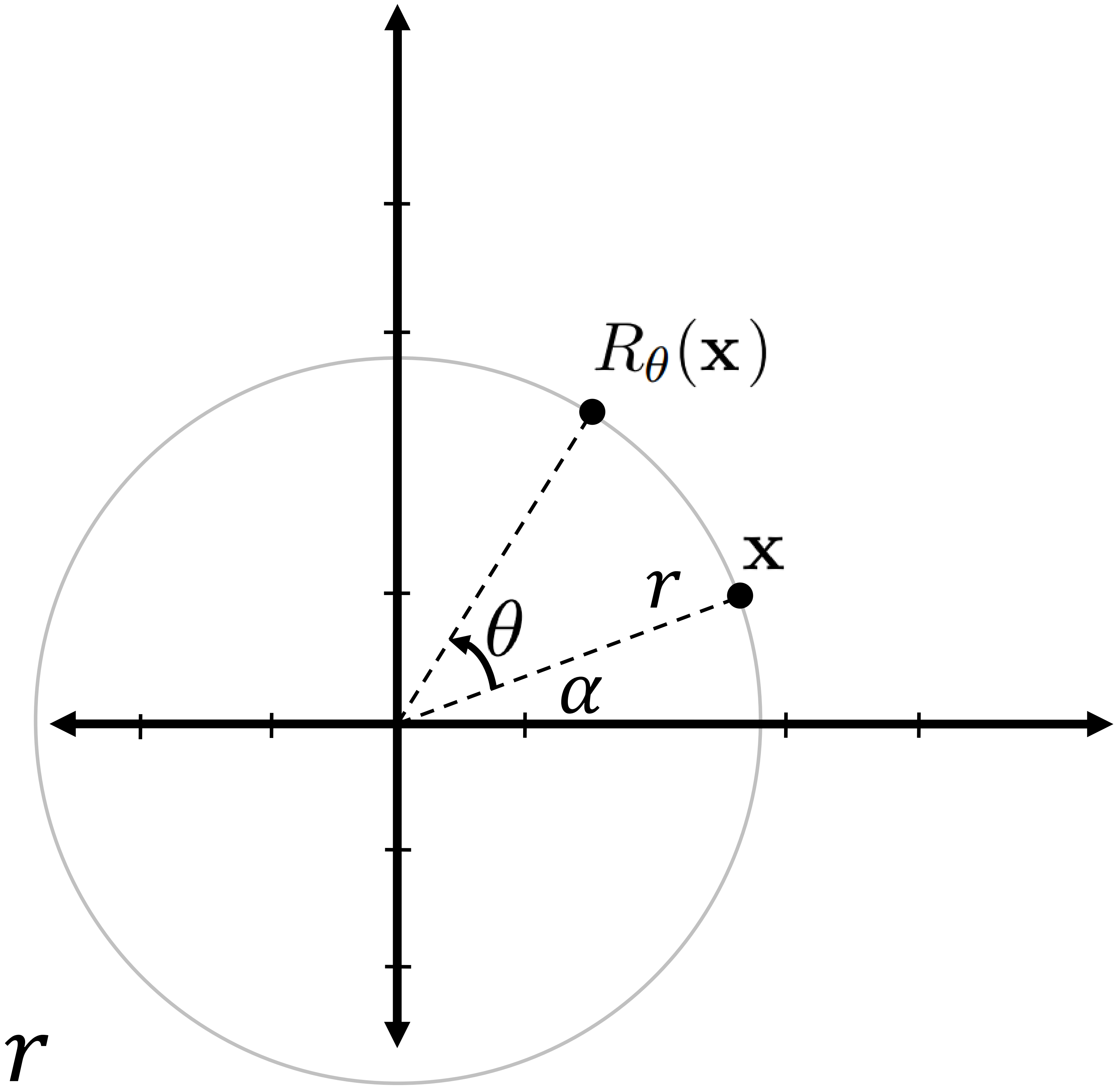
As angle changes, points move along *circular* trajectories.

Shape (distance between any two points) does not change!

(Rigid or isometric transformation)

Rotation

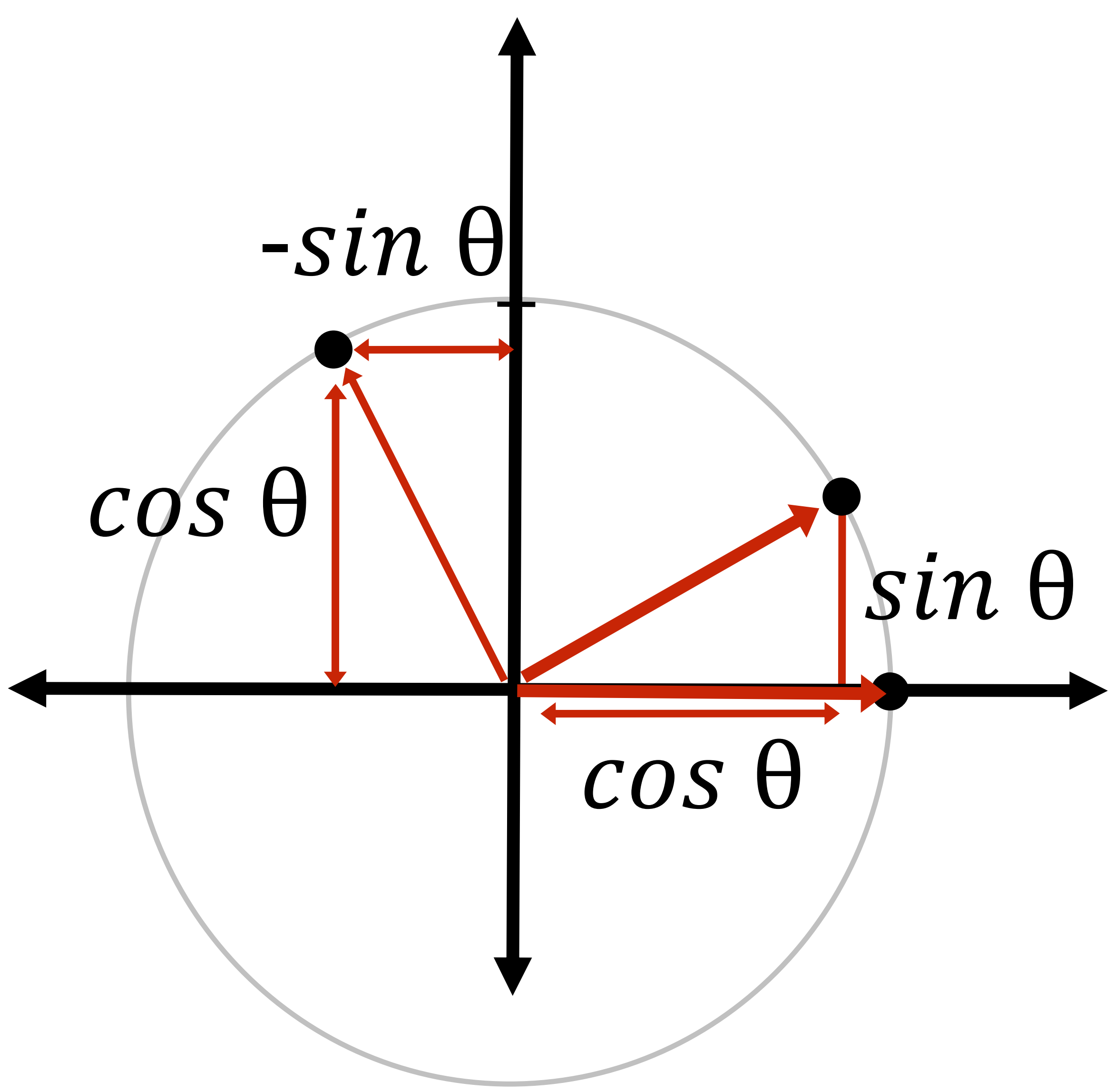
What does R_θ look like?



- **From x, compute α and r**
- **Write down $R_\theta(x)$ as a function of α, θ and r** (i.e. vector (r,0) rotated by $\alpha + \theta$)
- **Apply sum of angle formulae…**
- **Fine, but remember, we only need to know how e_1 and e_2 are transformed!**

Rotation

So, what happens to vectors (1, 0) and (0, 1) after rotation by θ?



Answer:

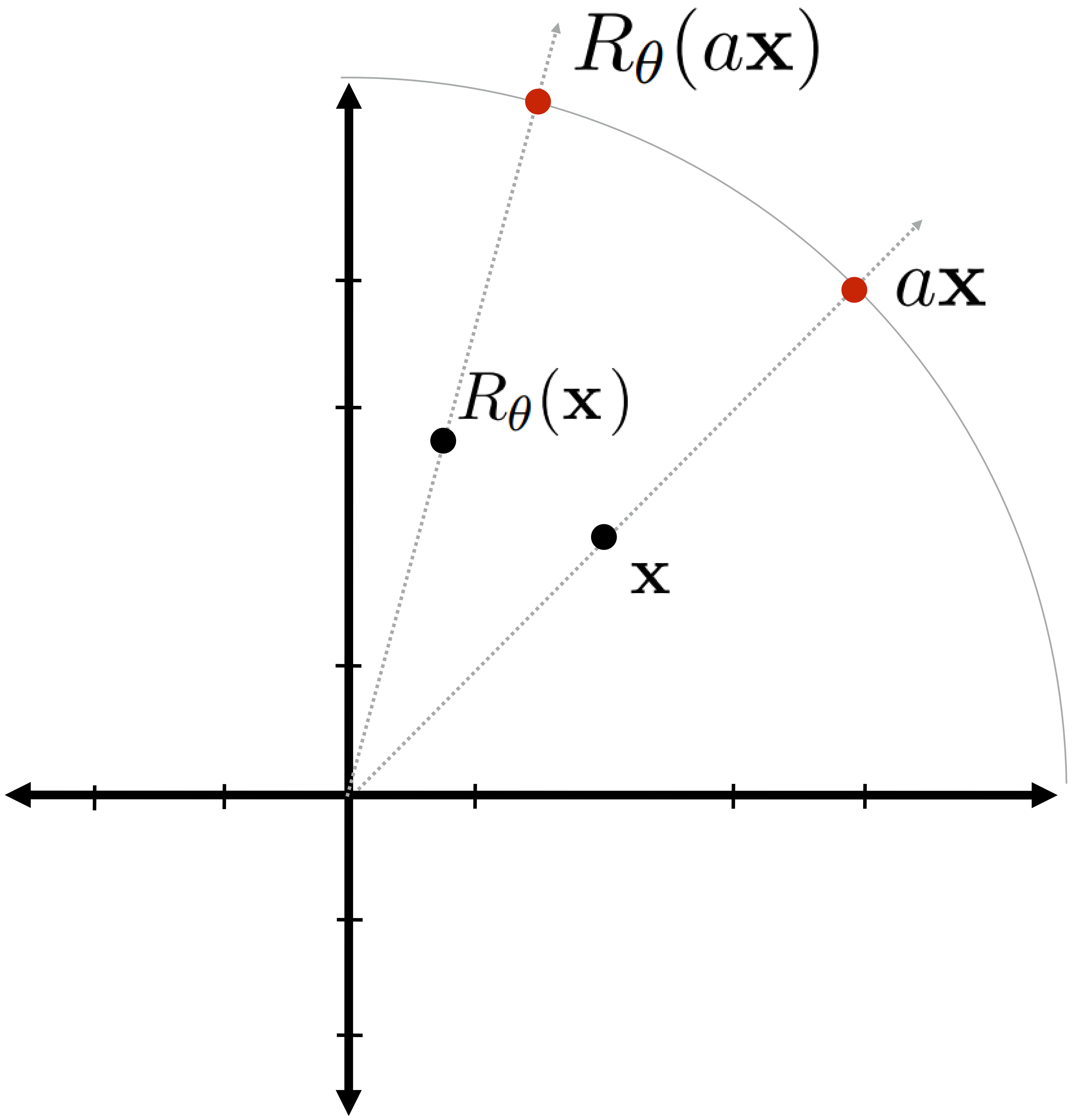
$$R_\theta(\boldsymbol{e}_1) = (cos\ \theta, sin\ \theta) = \boldsymbol{a}_1$$

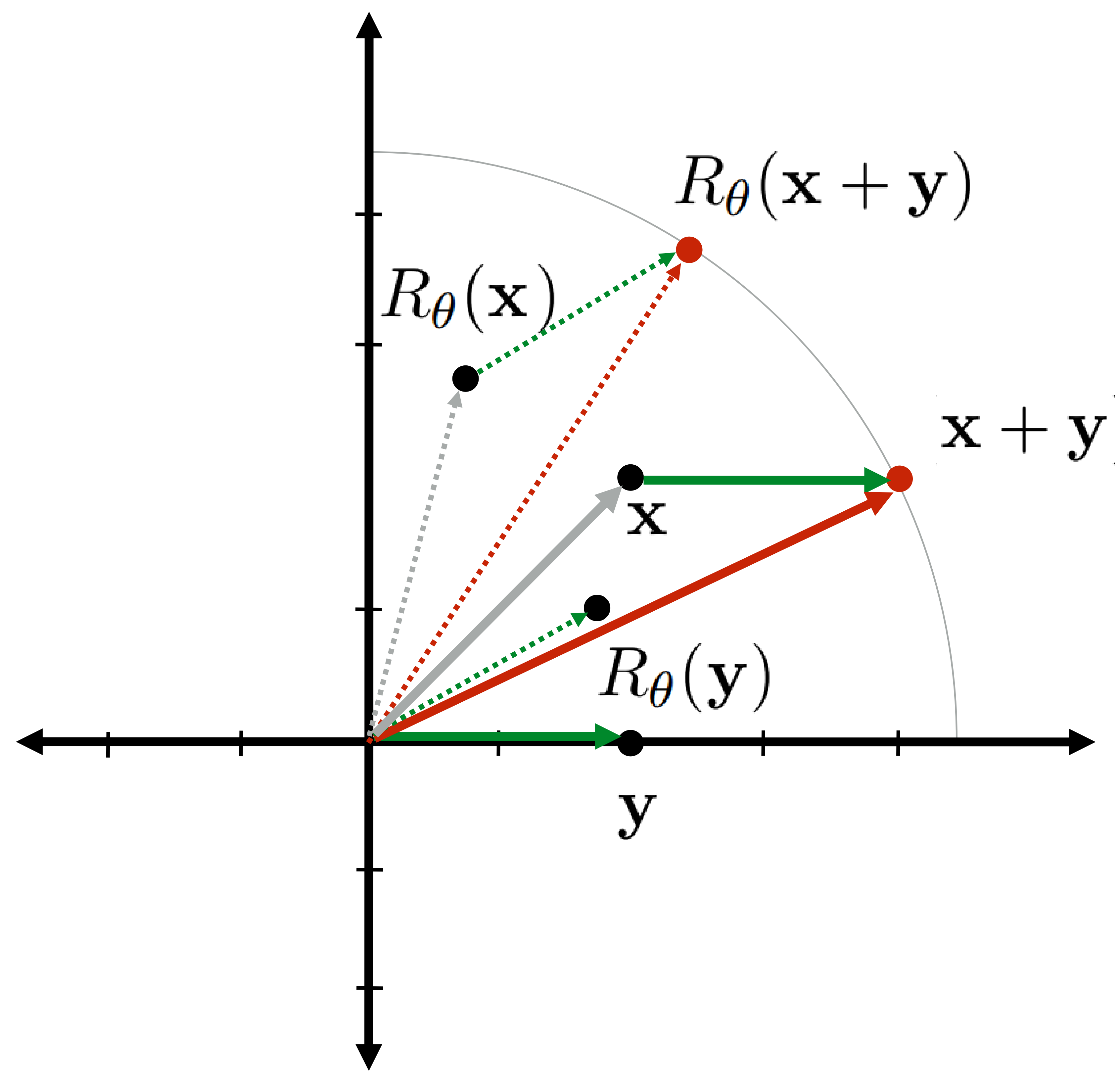
$$R_\theta(\boldsymbol{e}_2) = (-sin\ \theta, cos\ \theta) = \boldsymbol{a}_2$$

So:

$$R_\theta(\boldsymbol{x}) = x_1\boldsymbol{a}_1 + x_2\boldsymbol{a}_2$$

Is rotation linear?

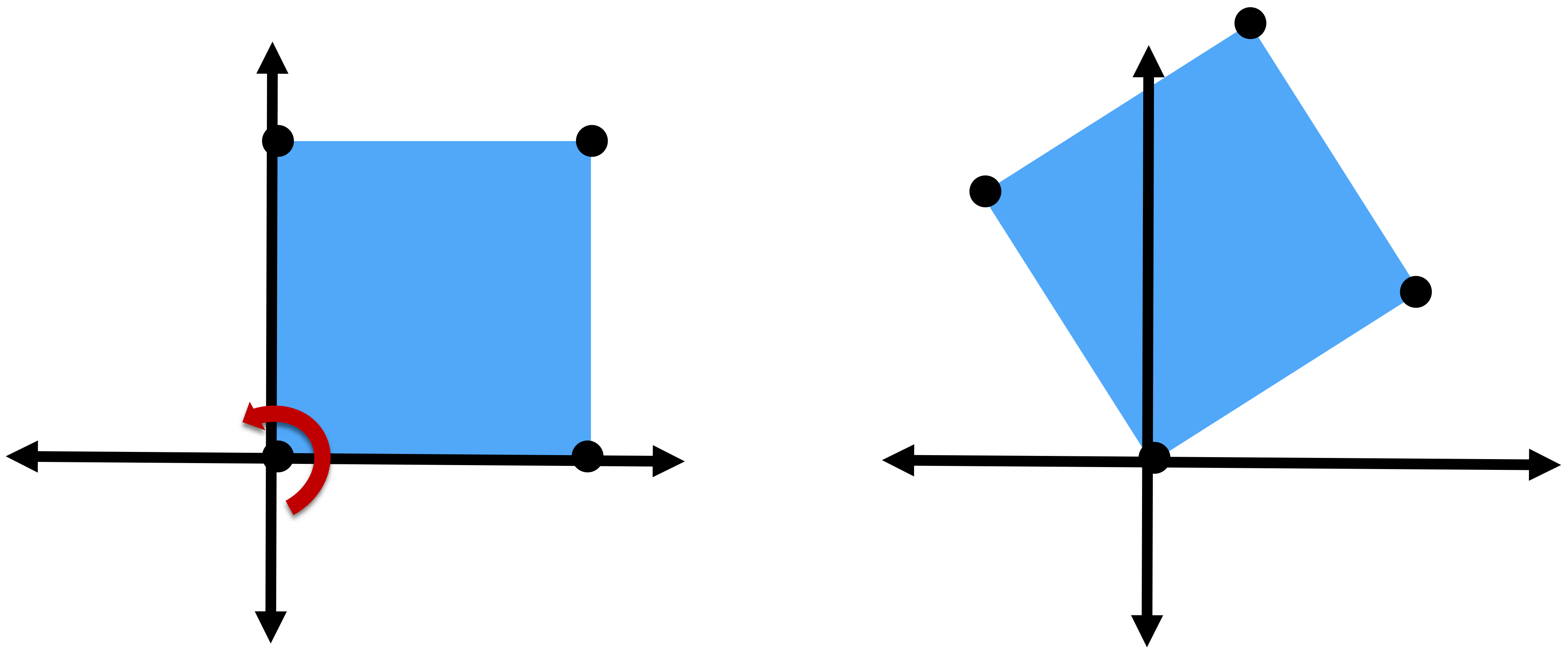


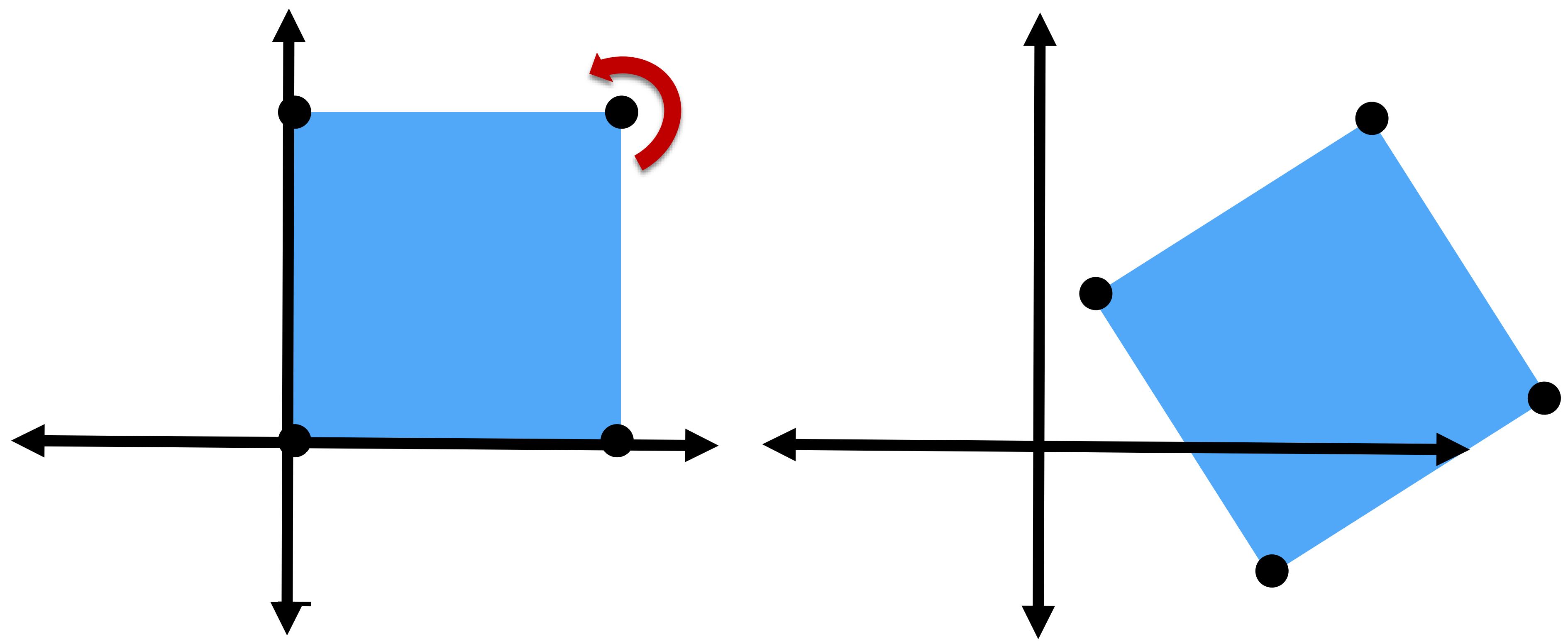


Yes!

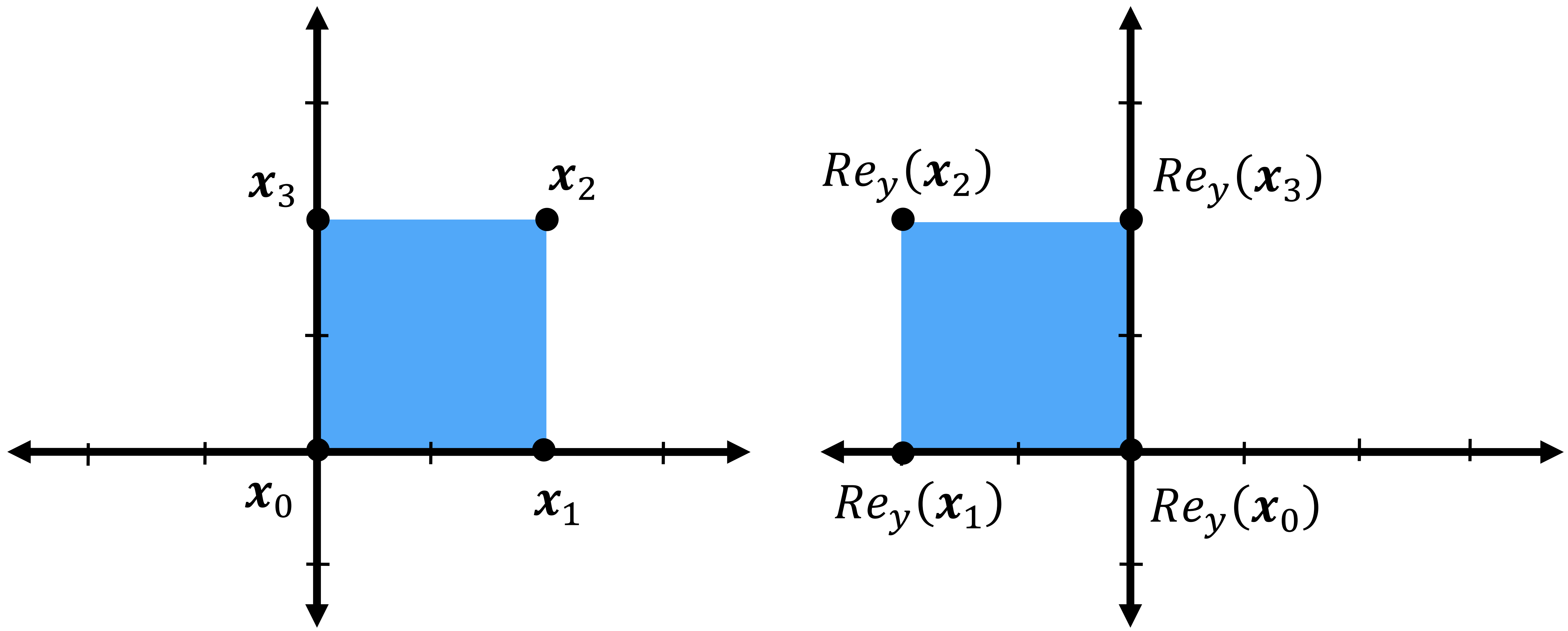
Rotation

- **Note: all points are rotated about the origin**
  - **By the way, what are we actually transforming here?**
- **What if we want to rotate about another point?**





Reflection



$Re_y(\boldsymbol{x})$ **: reflection about y-axis**

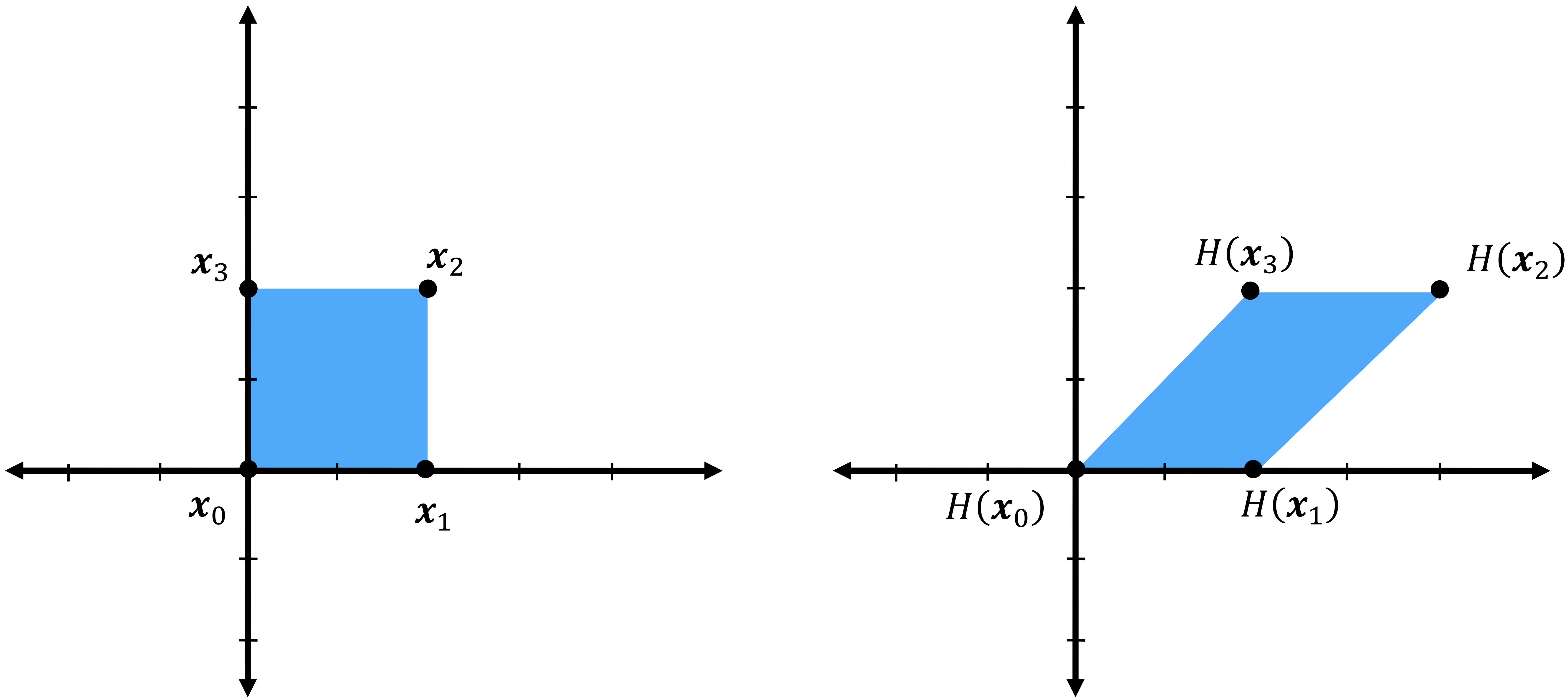
Reflections change "handedness"…

Do you know what $Re_y(\boldsymbol{x})$ **looks like?**

Is reflection a linear transform?

Do you know how to reflect about an arbitrary axis?

Shear (in x direction)

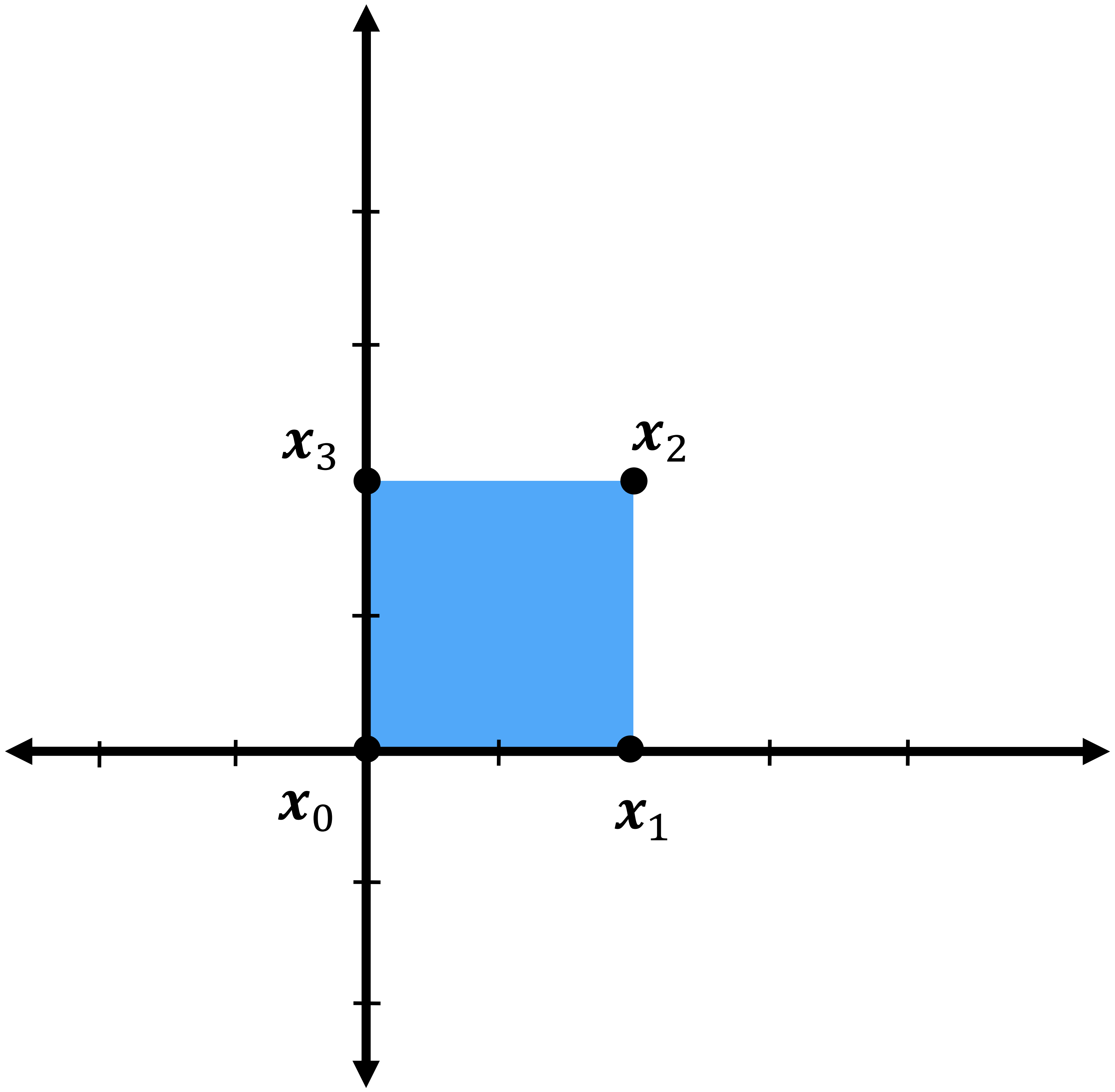


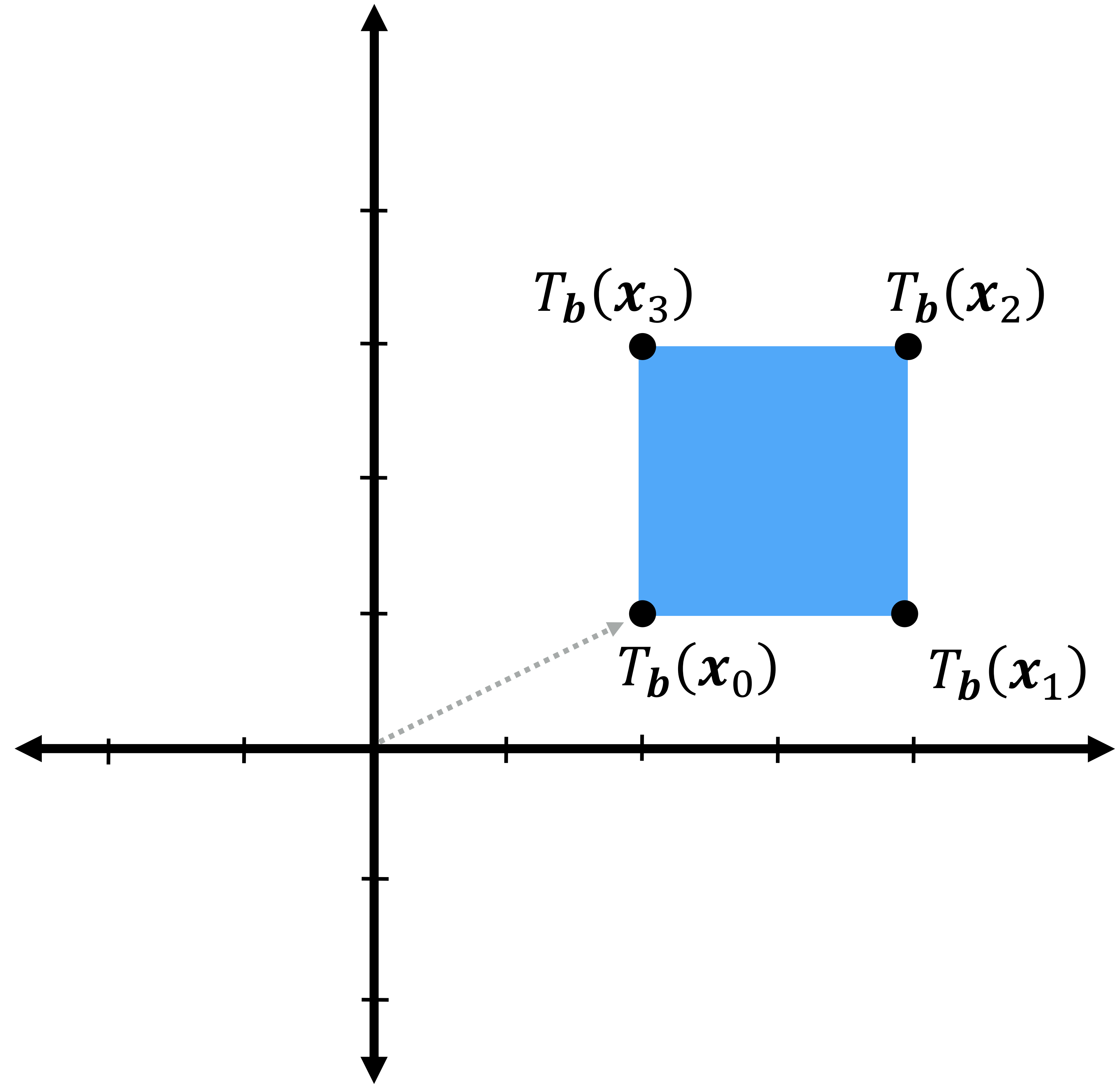
What does $H(\boldsymbol{x})$ look like?

$$\boldsymbol{H}_a(\boldsymbol{x}) = x_1 \begin{bmatrix} 1 \\ 0 \end{bmatrix} + x_2 \begin{bmatrix} a \\ 1 \end{bmatrix}$$

Is shearing a linear transformation?

Translation



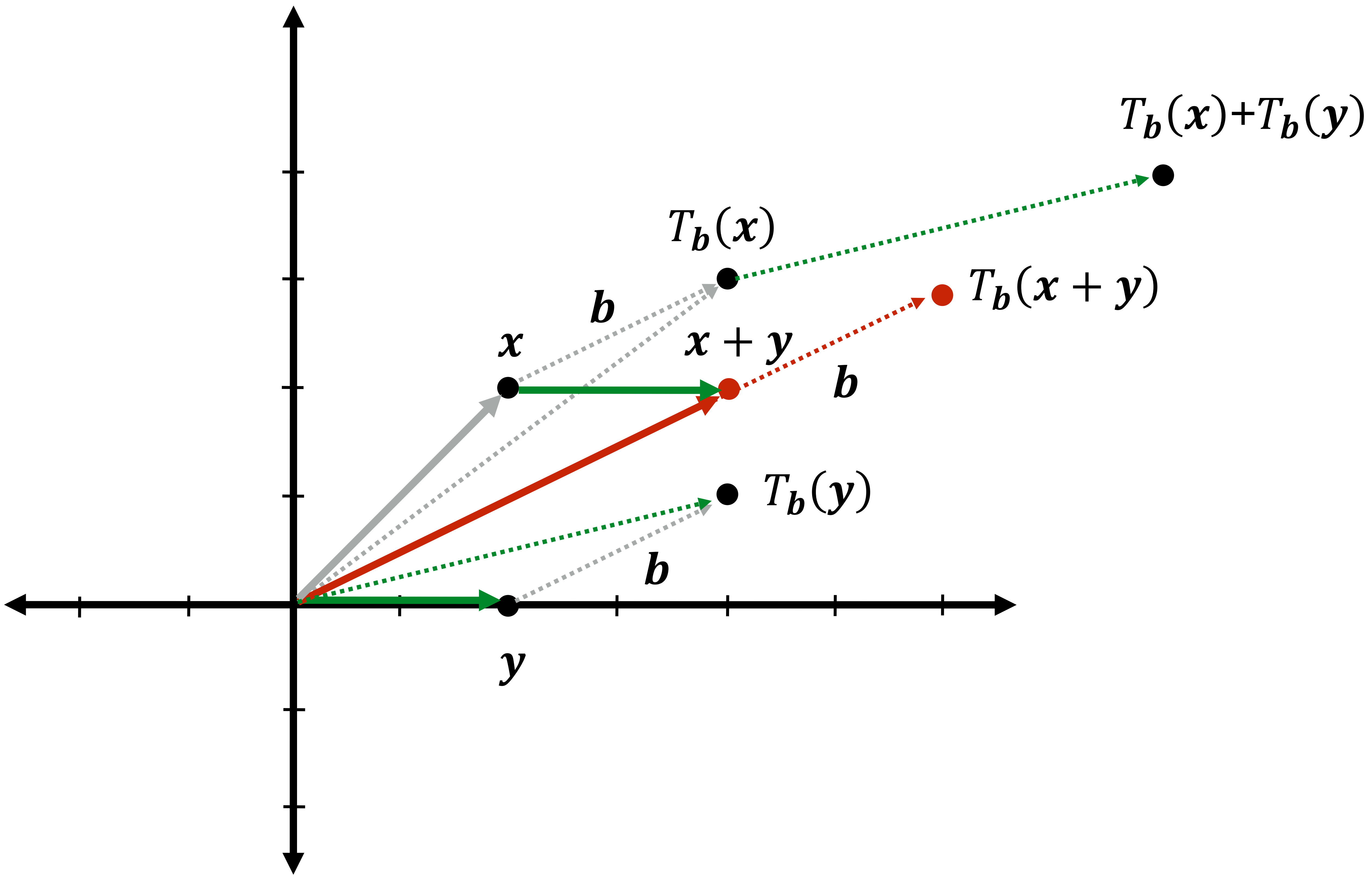


Let's write $T_b(\boldsymbol{x})$ in the form

$$T_{\boldsymbol{b}}(\boldsymbol{x}) = x_1 \begin{bmatrix} ? \\ ? \end{bmatrix} + x_2 \begin{bmatrix} ? \\ ? \end{bmatrix}$$

such that $T_{\boldsymbol{b}}(\boldsymbol{x}) = \boldsymbol{x} + \boldsymbol{b}$

Is translation linear?



No. Translation is affine.

Summary of basic transforms

Linear:

$$f(\mathbf{x}+\mathbf{y}) = f(\mathbf{x}) + f(\mathbf{y})$$
$$f(a\mathbf{x}) = af(\mathbf{x})$$

Scale
Rotation
Reflection
Shear

Not linear:

Translation

Affine:

Composition of linear transform + translation
(all examples on previous two slides)

$$f(\mathbf{x}) = g(\mathbf{x}) + \mathbf{b}$$

Not affine: perspective projection (will discuss later)

Euclidean: (Isometries)

Preserve distance between points (preserves length)

$$|f(\mathbf{x}) - f(\mathbf{y})| = |\mathbf{x} - \mathbf{y}|$$

Translation
Rotation
Reflection

"Rigid body" transforms are Euclidean transforms that also preserve "winding" (does not include reflection)

When at first you don’t succeed…

- **We’ll turn affine transformations into linear ones via**

Homogeneous coordinates
(aka projective coordinates)

- **But first, let’s use matrix notation to represent linear transforms**

Linear transforms as matrix-vector products

$$\underbrace{\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}}_{\boldsymbol{A}} * \underbrace{\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}}_{\boldsymbol{x}} = \begin{bmatrix} a_{11}x_1 + a_{12}x_2 \\ a_{21}x_1 + a_{22}x_2 \end{bmatrix}$$

$$= x_1 \begin{bmatrix} a_{11} \\ a_{21} \end{bmatrix} + x_2 \begin{bmatrix} a_{12} \\ a_{22} \end{bmatrix} = \underbrace{x_1 \boldsymbol{a_1} + x_2 \boldsymbol{a_2}}_{}$$

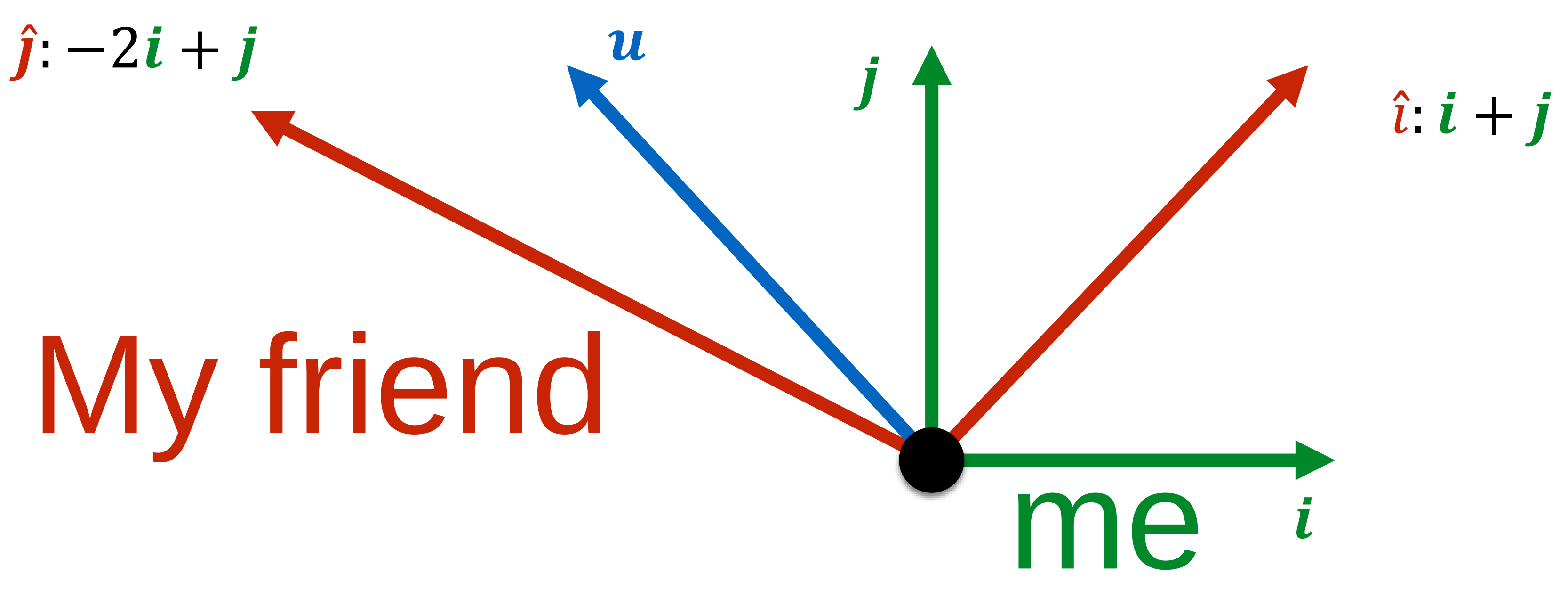
$$f(\boldsymbol{x}) = \sum_{i=1}^{m} x_i \boldsymbol{a_i} = \boldsymbol{Ax}$$

Linear transforms as matrix-vector products

Change of coordinate systems

$$f(\boldsymbol{x}) = x_1 \begin{bmatrix} 1 \\ 1 \end{bmatrix} + x_2 \begin{bmatrix} -2 \\ 1 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & -2 \\ 1 & 1 \end{bmatrix} \boldsymbol{x}$$

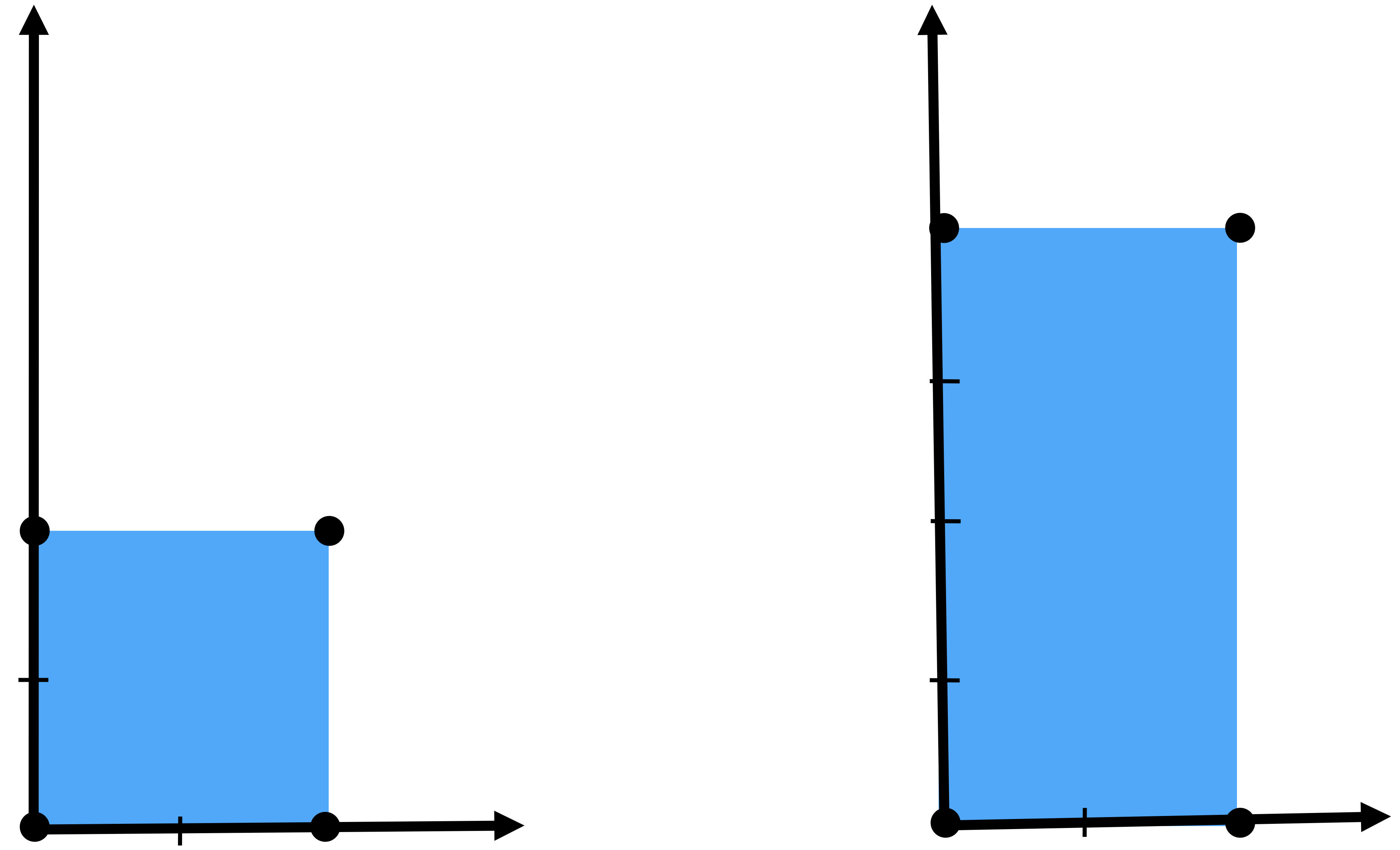


Linear transforms as matrix-vector products

Non-uniform scale

$$S(\boldsymbol{x}) = x_1 a \boldsymbol{e}_1 + x_2 b \boldsymbol{e}_2$$

$$= \begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix} \boldsymbol{x}$$



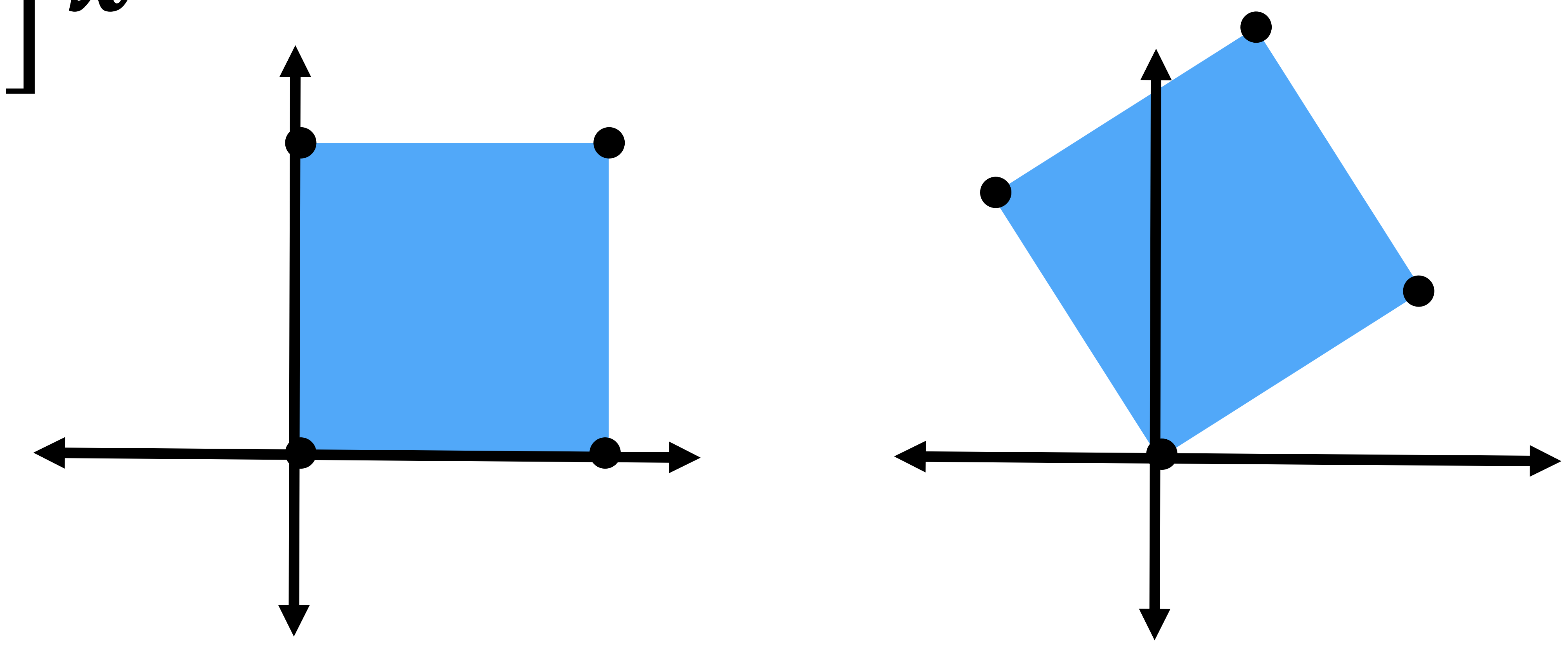
Linear transforms as matrix-vector products

Rotation

$$R_\theta(\boldsymbol{e}_1) = (cos\,\theta, sin\,\theta) = \boldsymbol{a}_1$$
$$R_\theta(\boldsymbol{e}_2) = (-sin\,\theta, cos\,\theta) = \boldsymbol{a}_2$$

$$R_\theta(\boldsymbol{x}) = x_1\boldsymbol{a}_1 + x_2\boldsymbol{a}_2$$

$$= \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \boldsymbol{x}$$

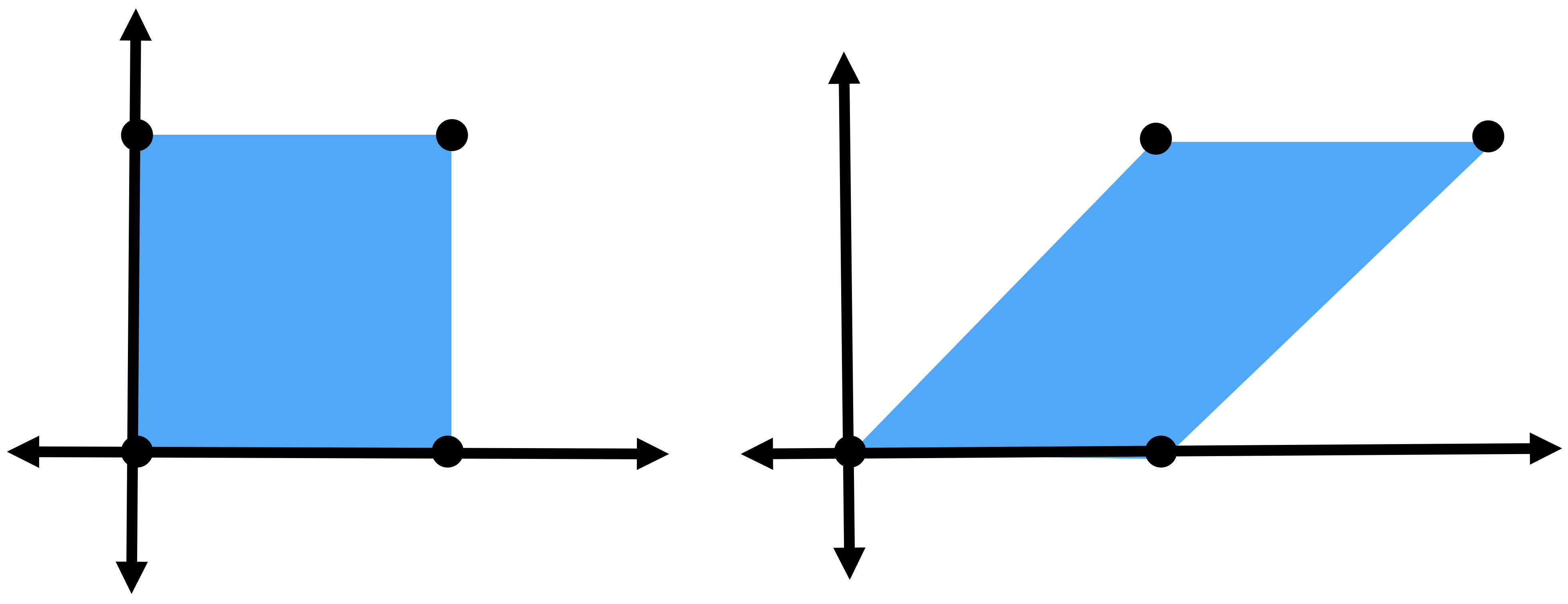


Linear transforms as matrix-vector products

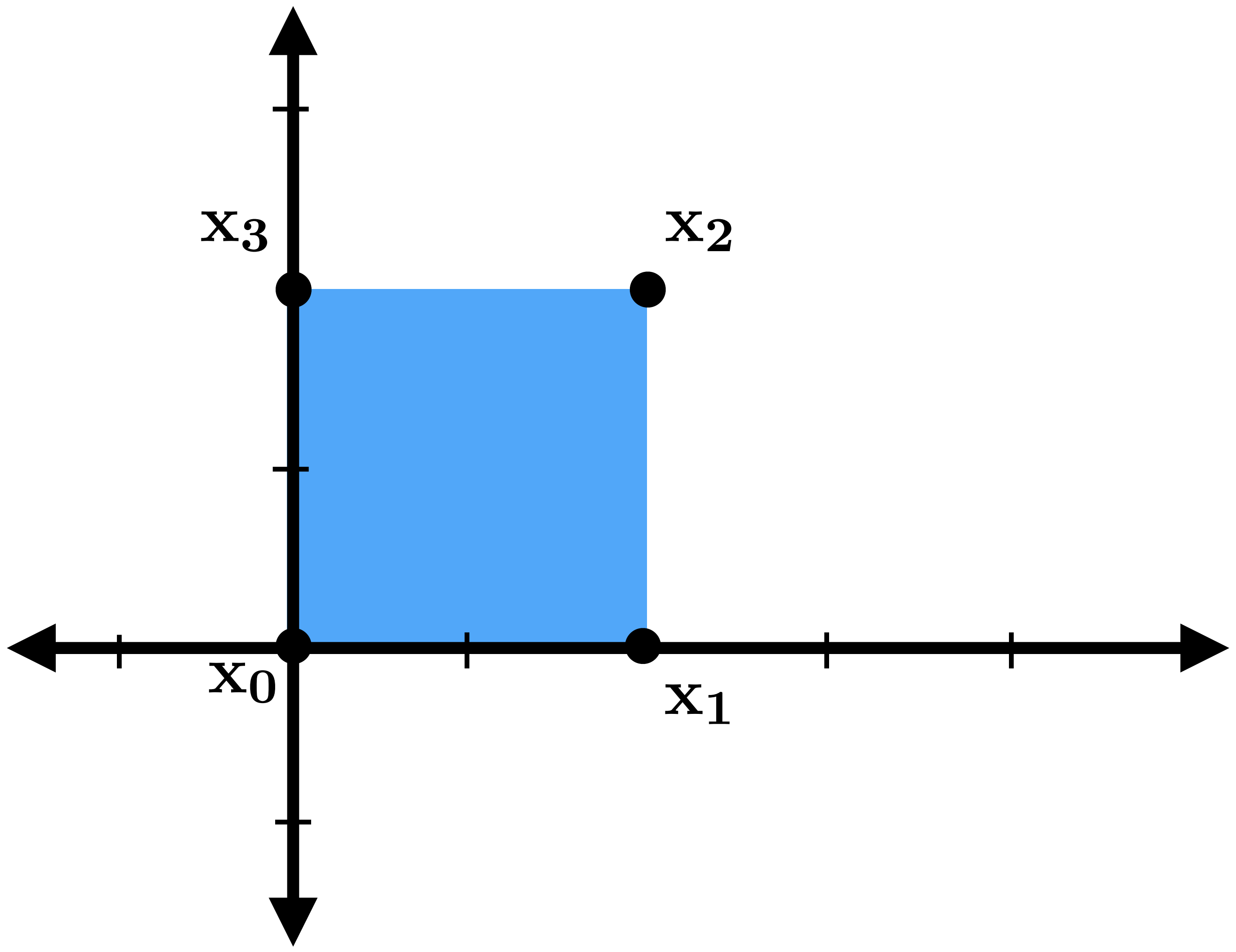
Shear

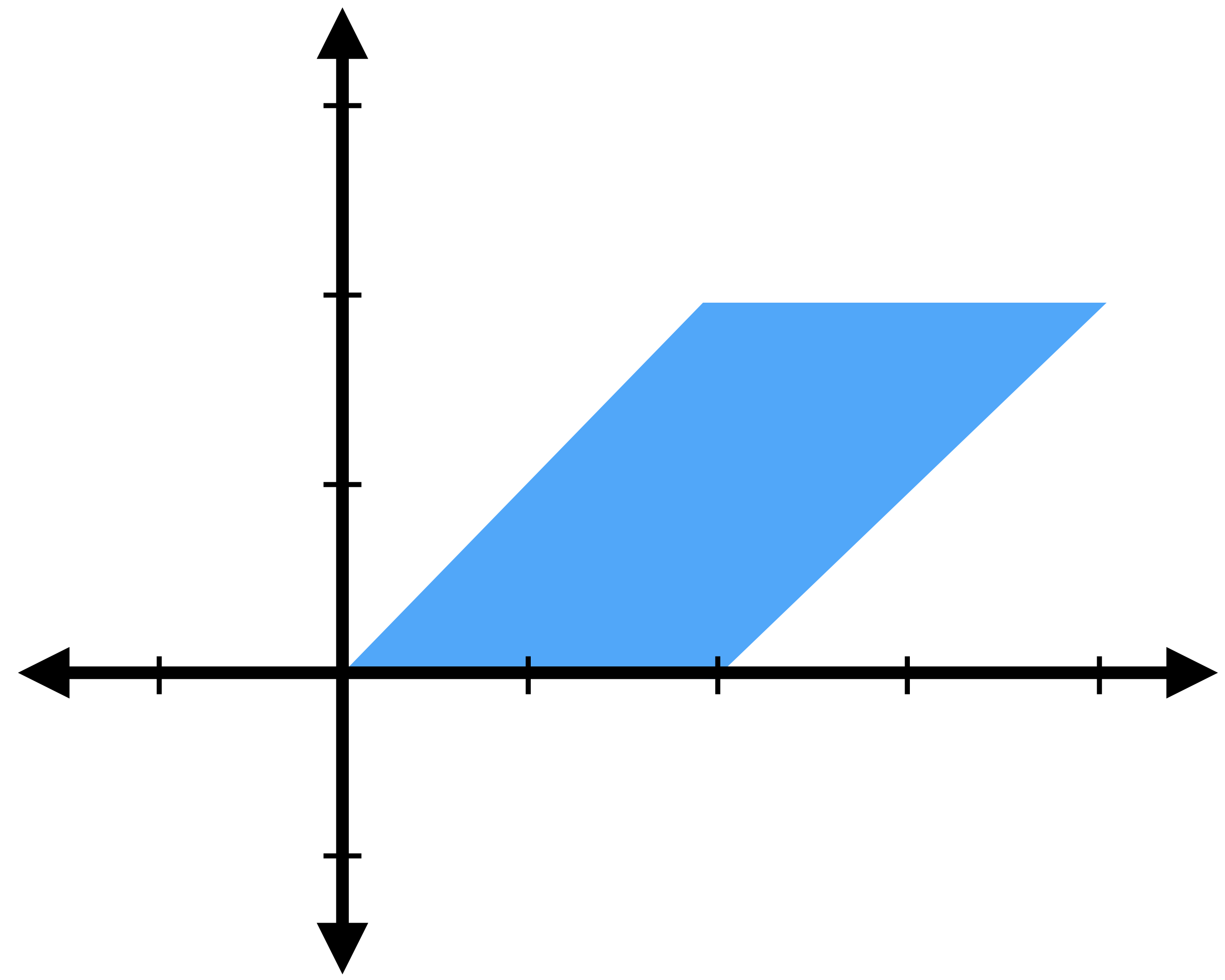
$$H(\boldsymbol{x}) = x_1 \begin{bmatrix} 1 \\ 0 \end{bmatrix} + x_2 \begin{bmatrix} a \\ 1 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & a \\ 0 & 1 \end{bmatrix} \boldsymbol{x}$$



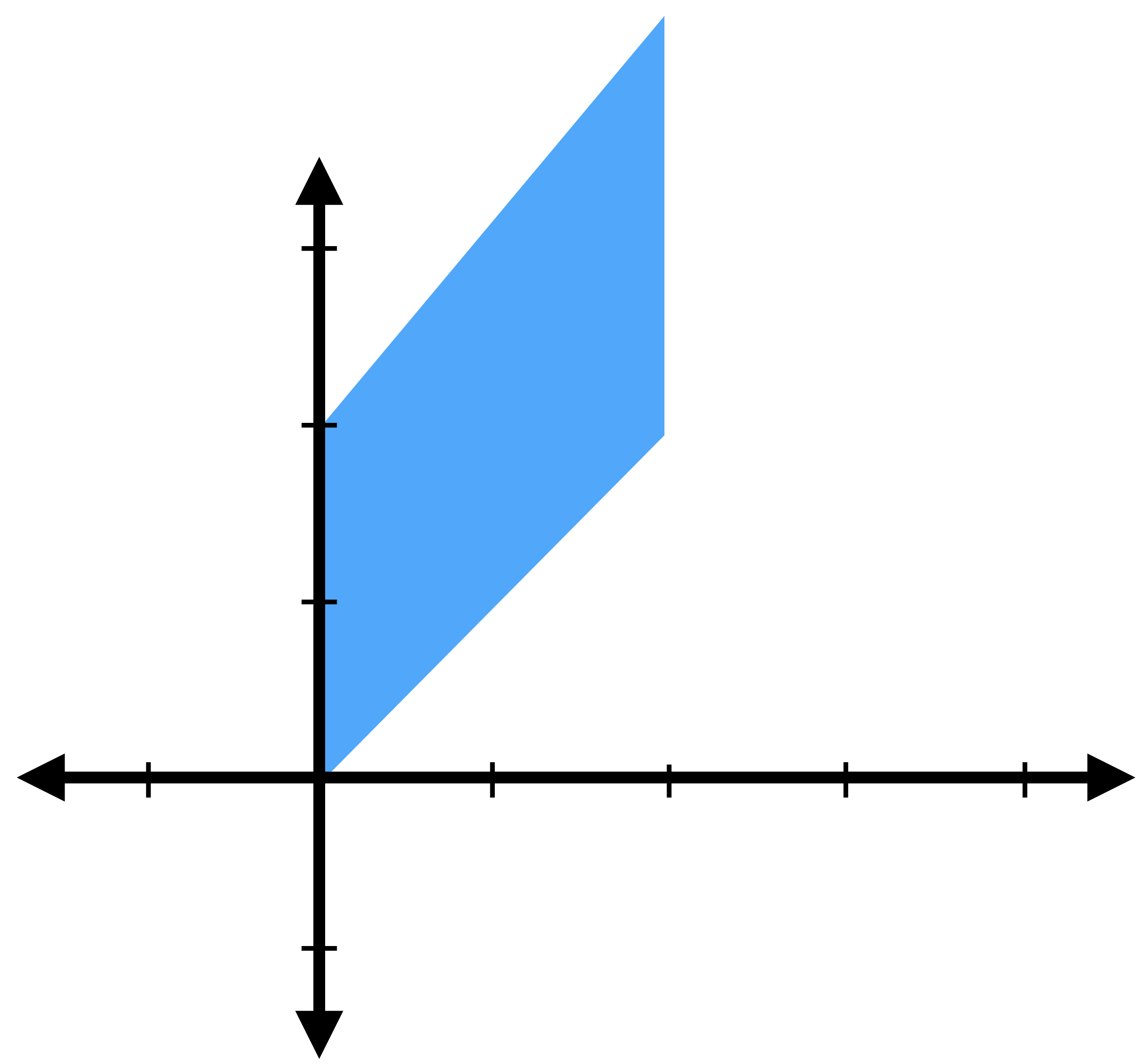
Shear





Shear in x:

$$\mathbf{H}_{xs} = \begin{bmatrix} 1 & s \\ 0 & 1 \end{bmatrix}$$



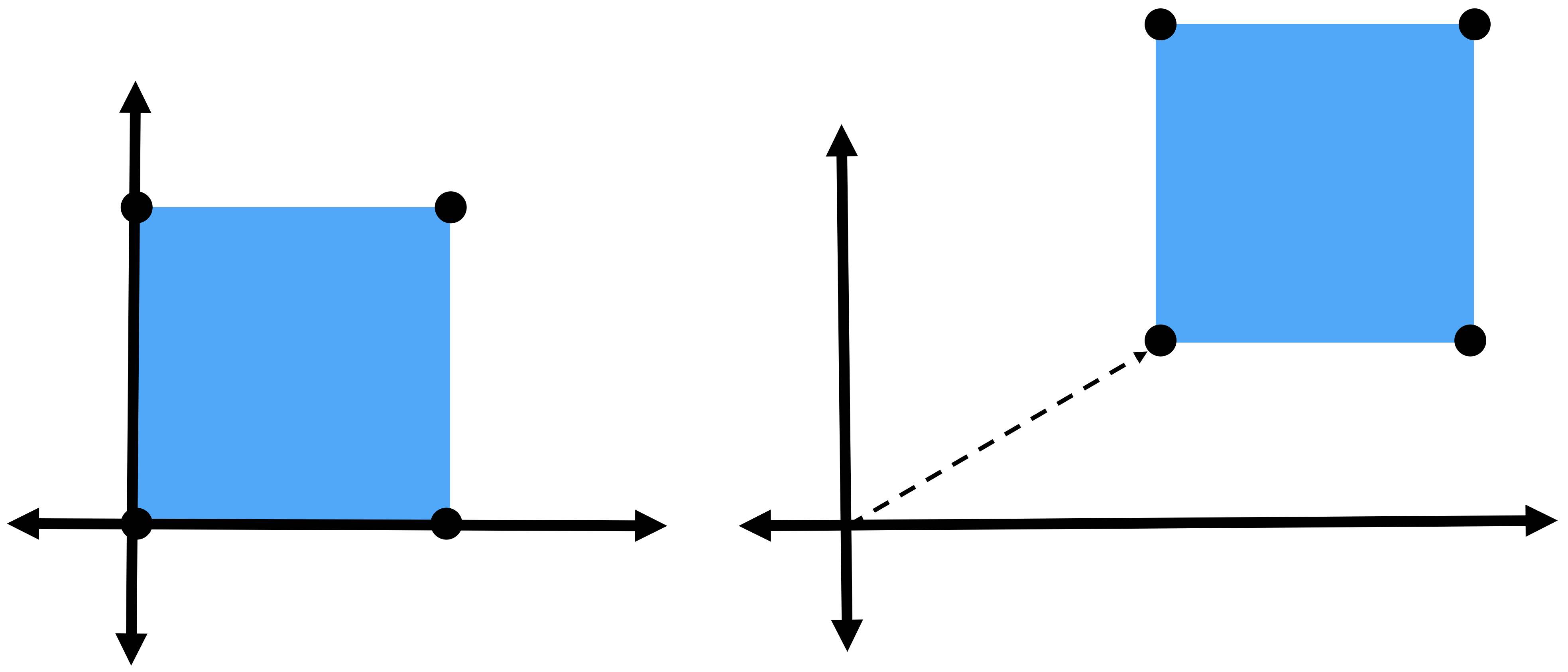
Shear in y:

$$\mathbf{H}_{ys} = \begin{bmatrix} 1 & 0 \\ s & 1 \end{bmatrix}$$

Linear transforms as matrix-vector products

Translation

Not a linear map*…



***when we're using Cartesian coordinates**

2D homogeneous coordinates (2D-H)

Key idea: lift 2D points to a 3D space

So the point (x_1, x_2) **is represented as the 3-vector:** $\begin{bmatrix} x_1 \\ x_2 \\ 1 \end{bmatrix}$

And 2D transforms are represented by 3x3 matrices

For example: 2D rotation in homogeneous coordinates:

$$\begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ 1 \end{bmatrix}$$

Q: how do the transforms we've seen so far affect the last coordinate?

Translation in 2D-H coords

Translation expressed as 3x3 matrix multiplication:

$$\boldsymbol{T}(\boldsymbol{x}) = \boldsymbol{x} + \boldsymbol{b} = \begin{bmatrix} 1 & 0 & b_1 \\ 0 & 1 & b_2 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ 1 \end{bmatrix} = \begin{bmatrix} x_1 + b_1 \\ x_2 + b_2 \\ 1 \end{bmatrix}$$

In homogeneous coordinates, translation is a linear transformation!

Translation in 2D-H coords

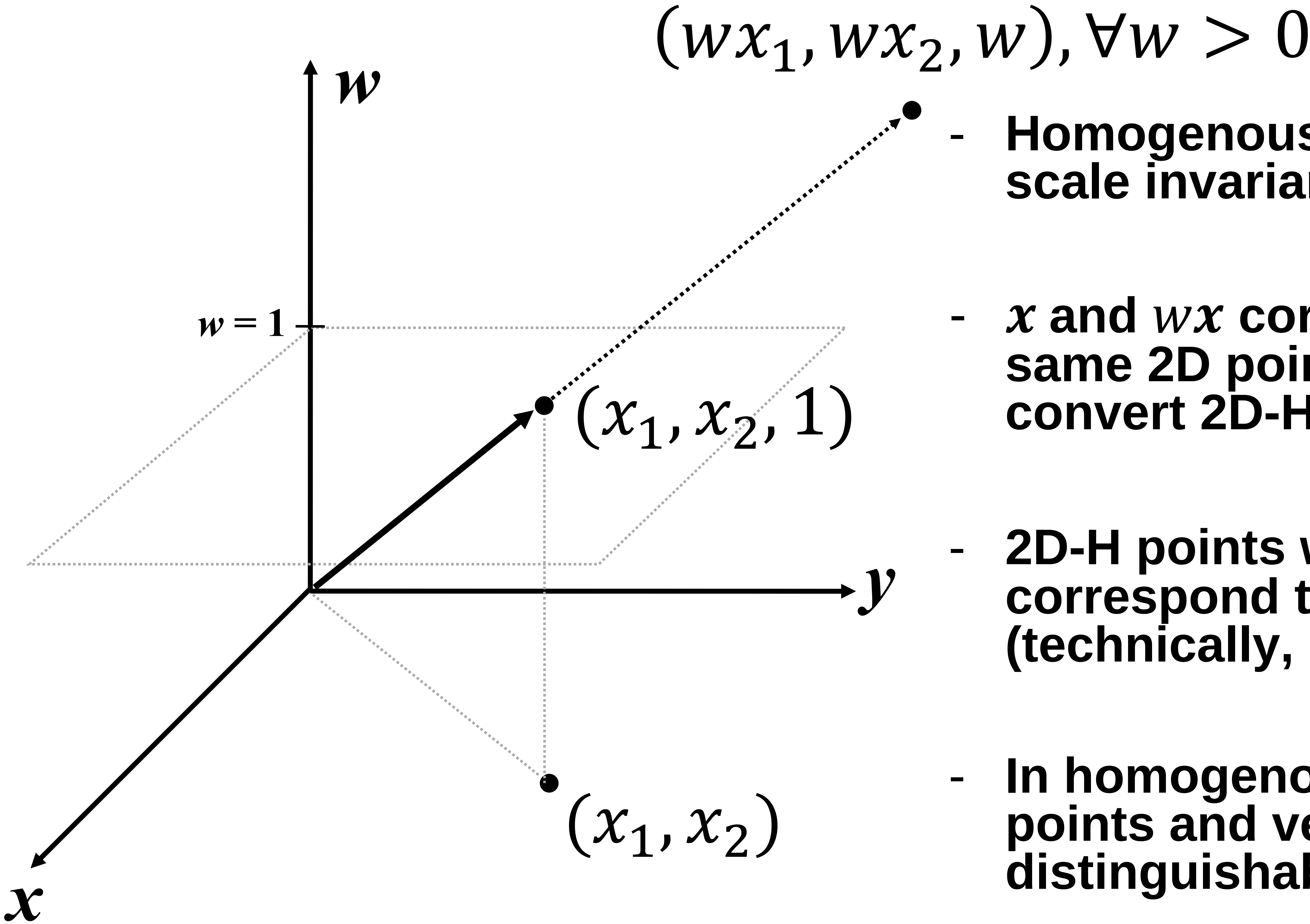
What is this magic?

$$\begin{bmatrix} 1 & 0 & b_1 \\ 0 & 1 & b_2 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} x_1 + b_1 \\ x_2 + b_2 \\ x_3 \end{bmatrix}$$

Translation in 2D homogeneous coordinates is equivalent to shearing along x & y axes - a linear operation

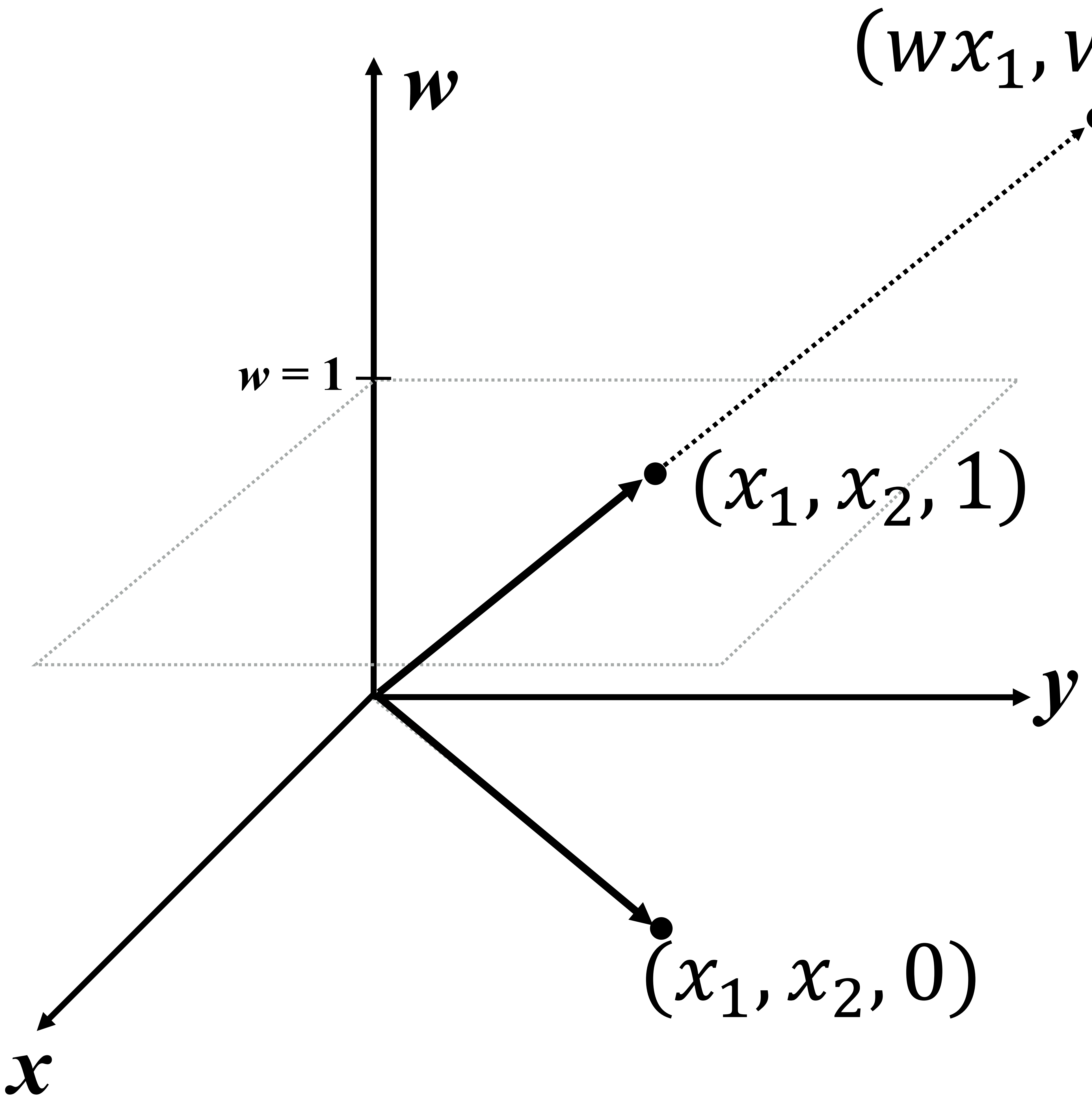
But why is x_3 set to 1? Could it not be 3.4182 instead?

Homogeneous coordinates



- Homogenous coordinates are scale invariant
- $\boldsymbol{x}$ and $w\boldsymbol{x}$ correspond to the same 2D point (divide by w to convert 2D-H back to 2D)
- 2D-H points with $w = 0$ correspond to 2D vectors (technically, points at infinity)
- In homogenous coordinates, points and vectors are distinguishable from each other!

Homogeneous coordinates: points vs. vectors



$$\begin{bmatrix} 1 & 0 & b_1 \\ 0 & 1 & b_2 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ 1 \end{bmatrix}$$

vs

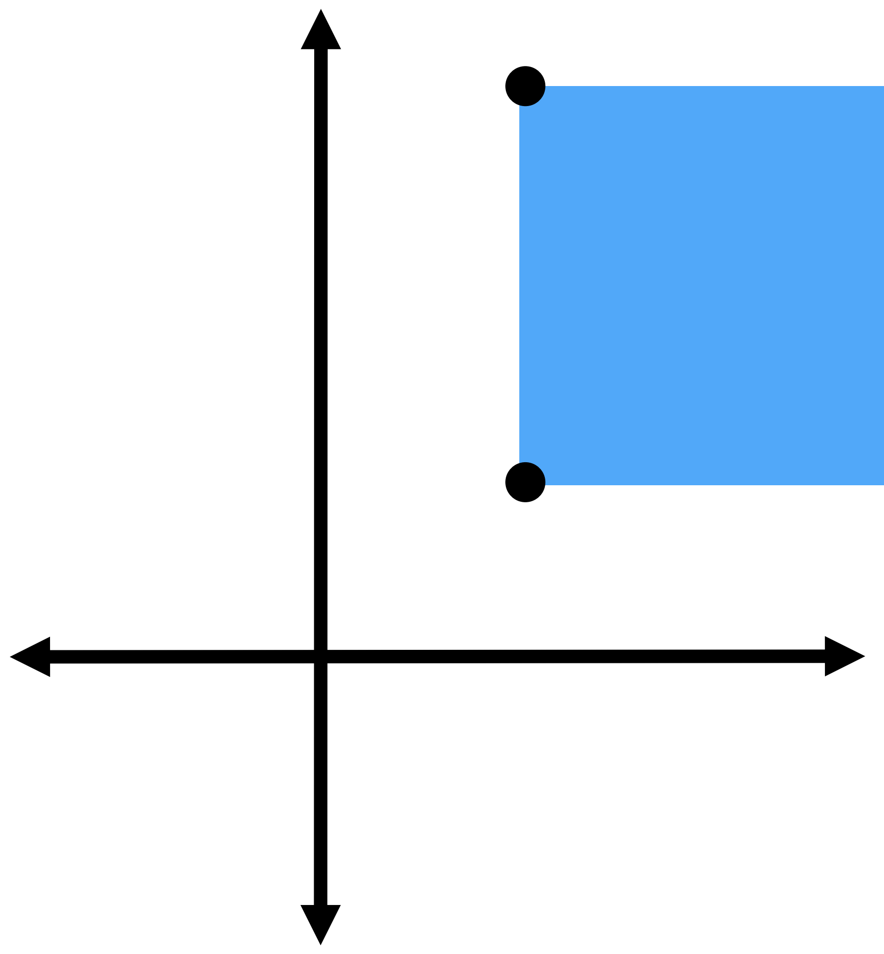
$$\begin{bmatrix} 1 & 0 & b_1 \\ 0 & 1 & b_2 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ 0 \end{bmatrix}$$

Summary so far…

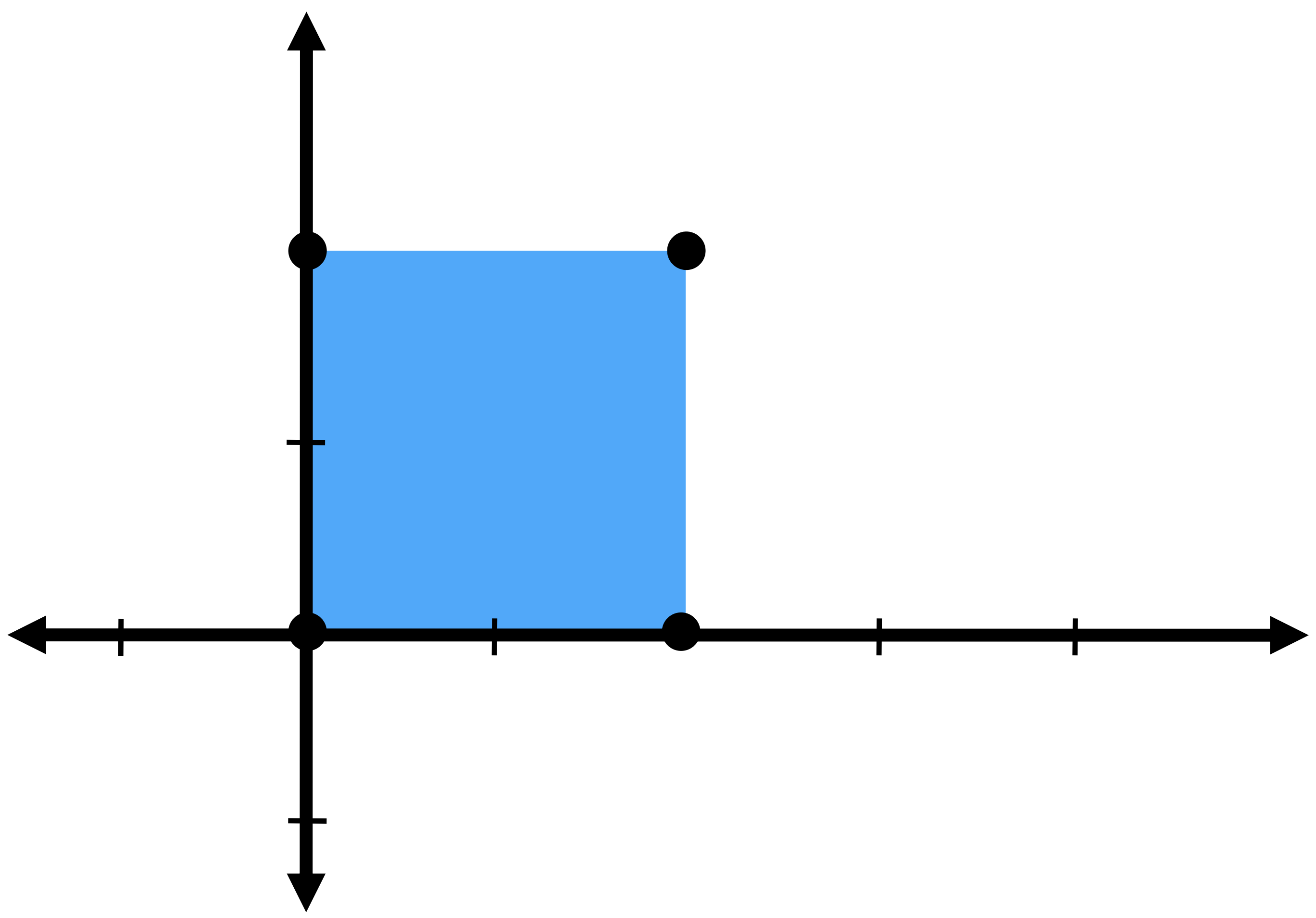
- We know how to transform (scale, rotate, reflect, shear, translate) 2D points and vectors
  - All these transforms are linear maps expressed as matrix-vector products when using (slightly) higher-dimensional homogenous coordinates
  - How about other types of transforms (e.g. rotate about an arbitrary point)?
  - How about 3D transforms?

Onto more complex transforms

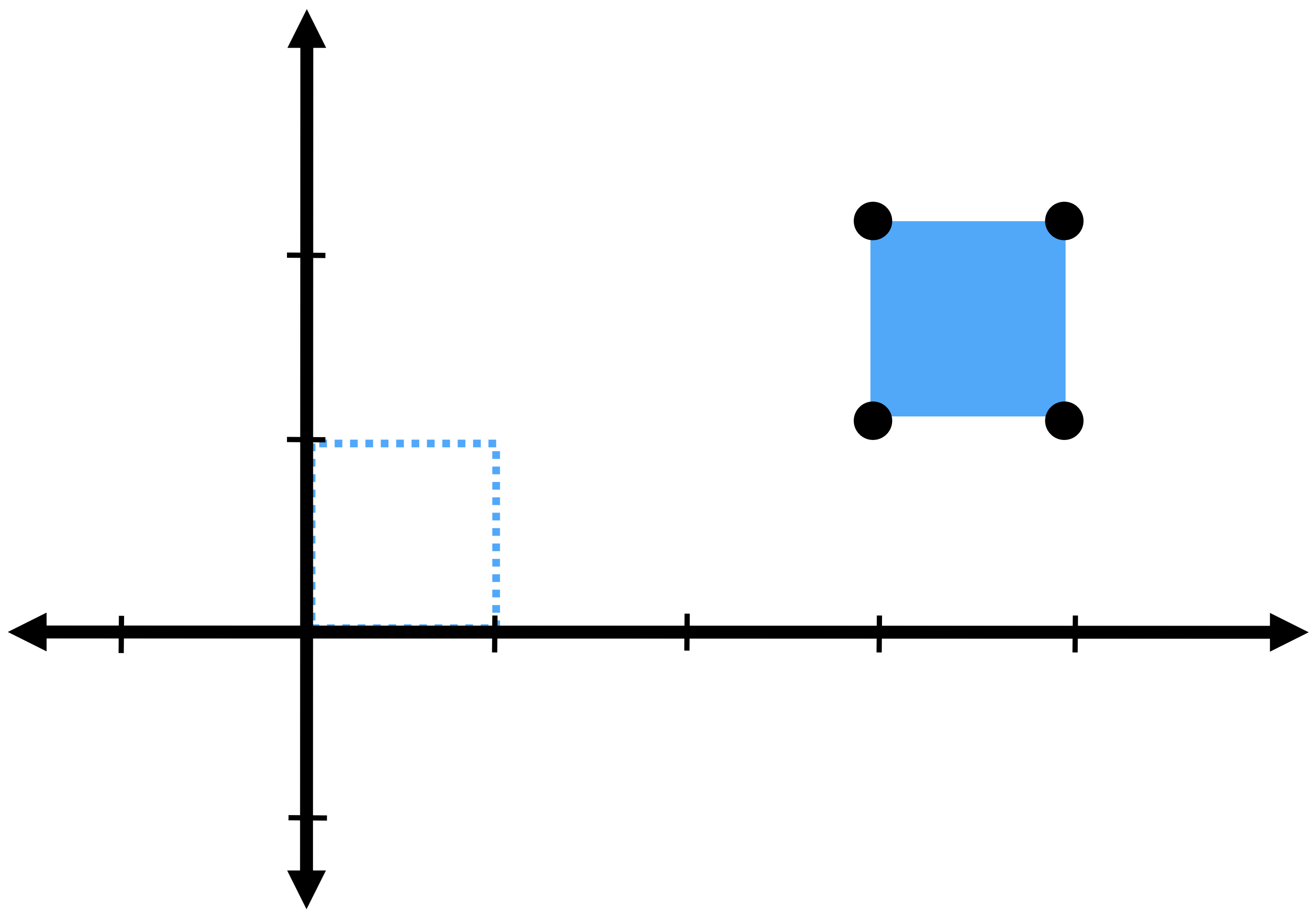
- **How would you transform this object such that it gets twice as large?**
  - **but remains where it is…**



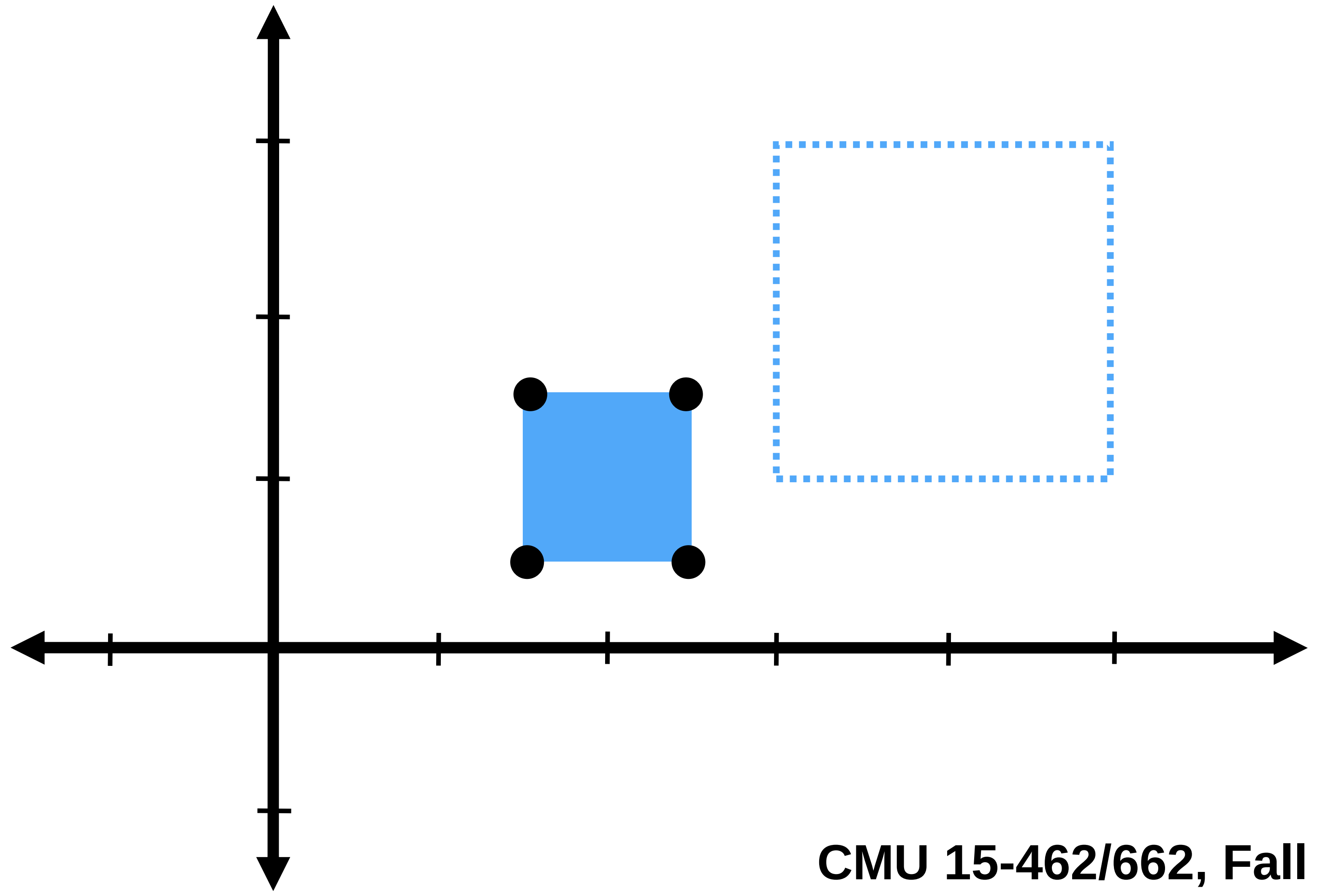
Composition of basic transforms



Scale by 0.5, then translate by (3,1)

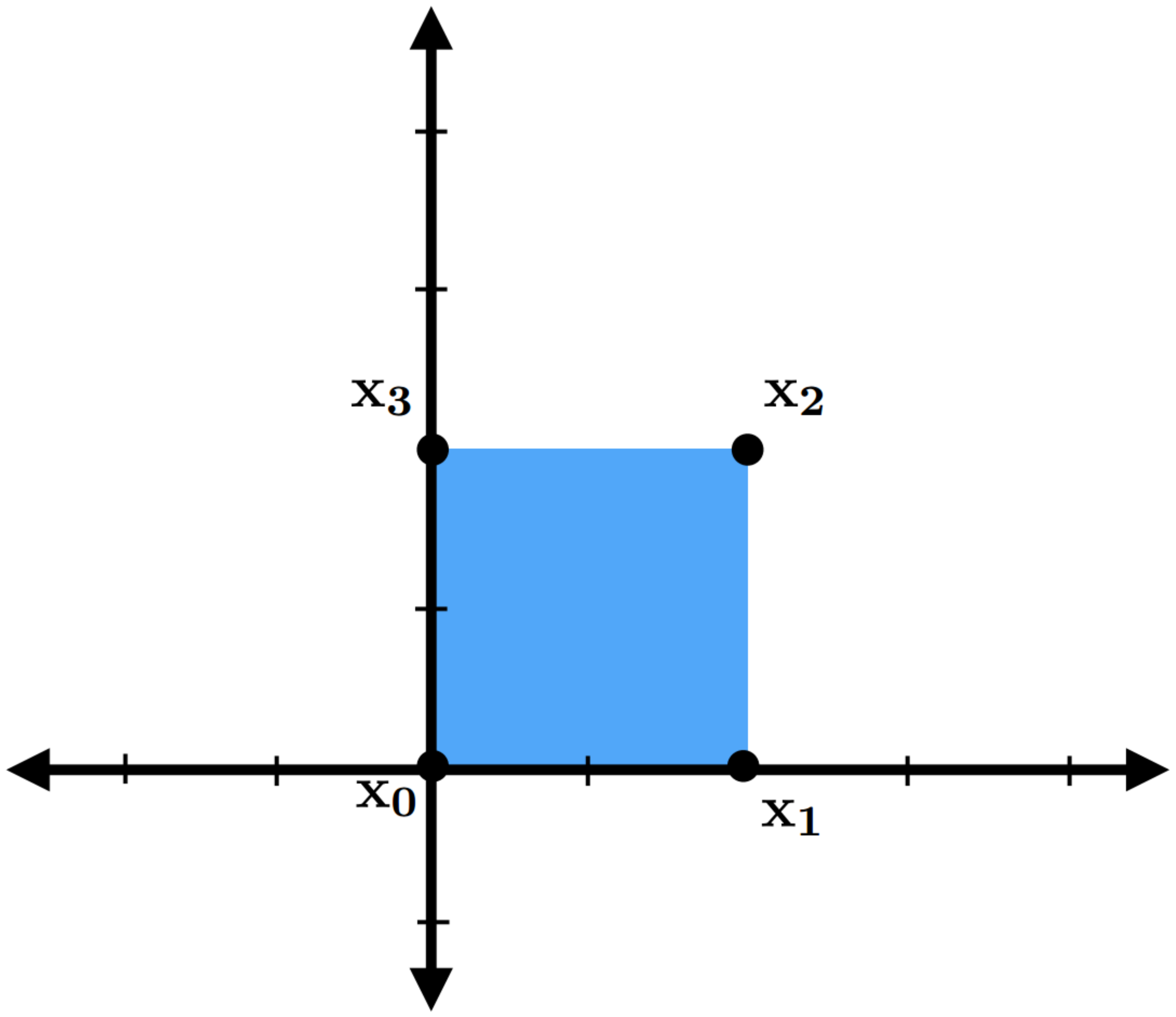


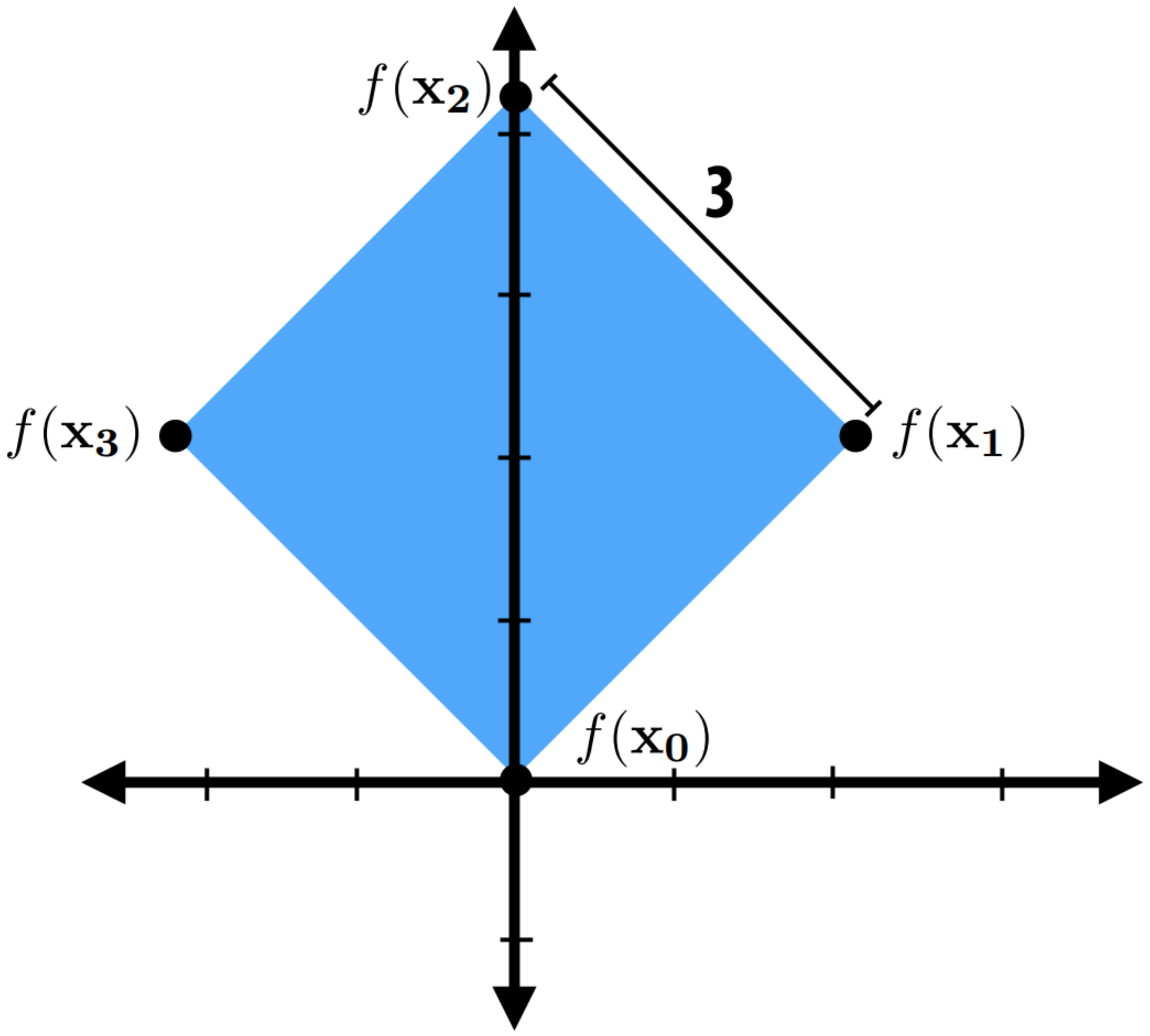
Translate by (3,1), then scale by 0.5



Note 1: order of composition matters!
Note 2: common source of bugs!

How do we compose linear transforms?



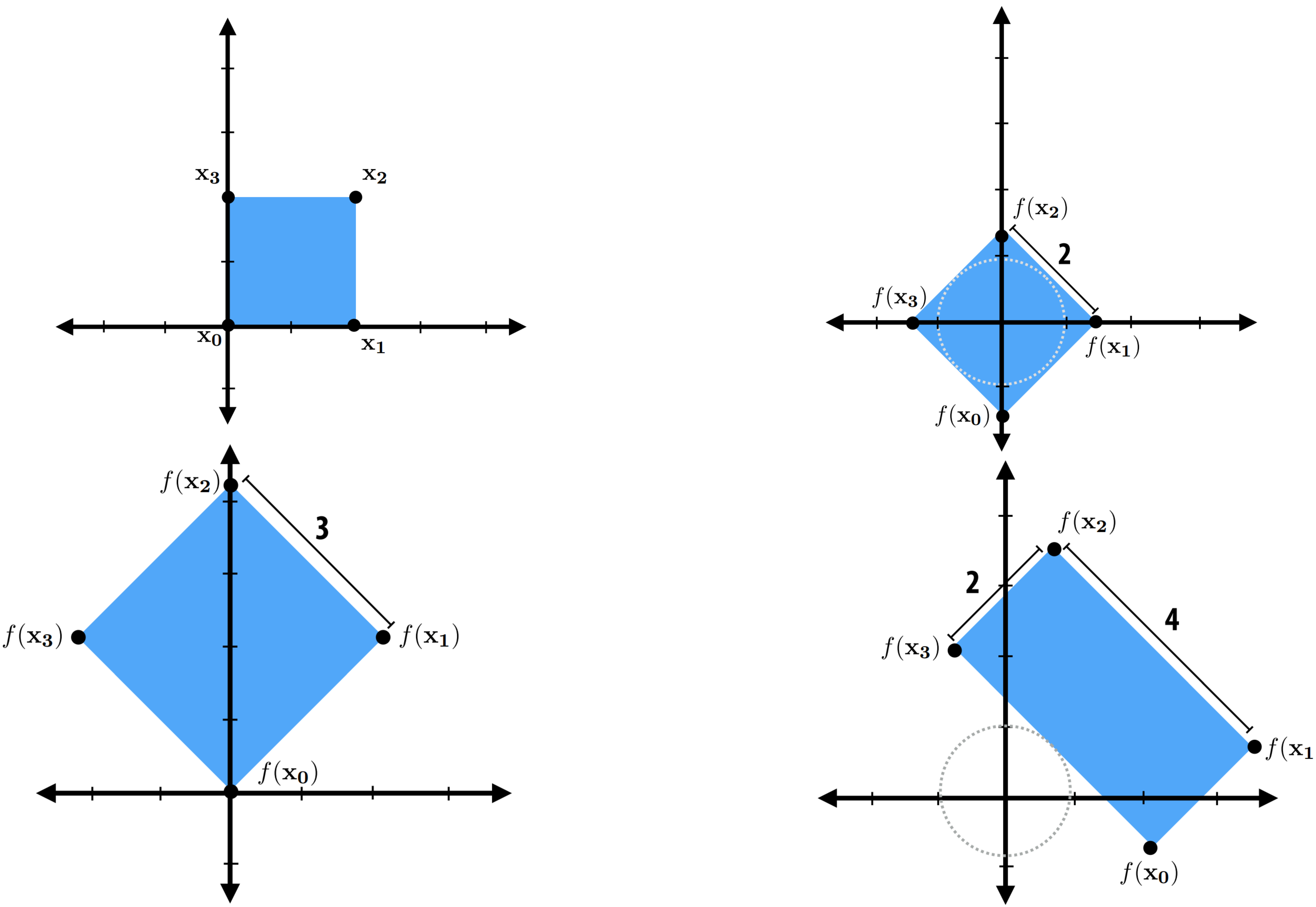


$$f(\mathbf{x}) = R_{\pi/4}\mathbf{S}_{[1.5,1.5]}\mathbf{x}$$

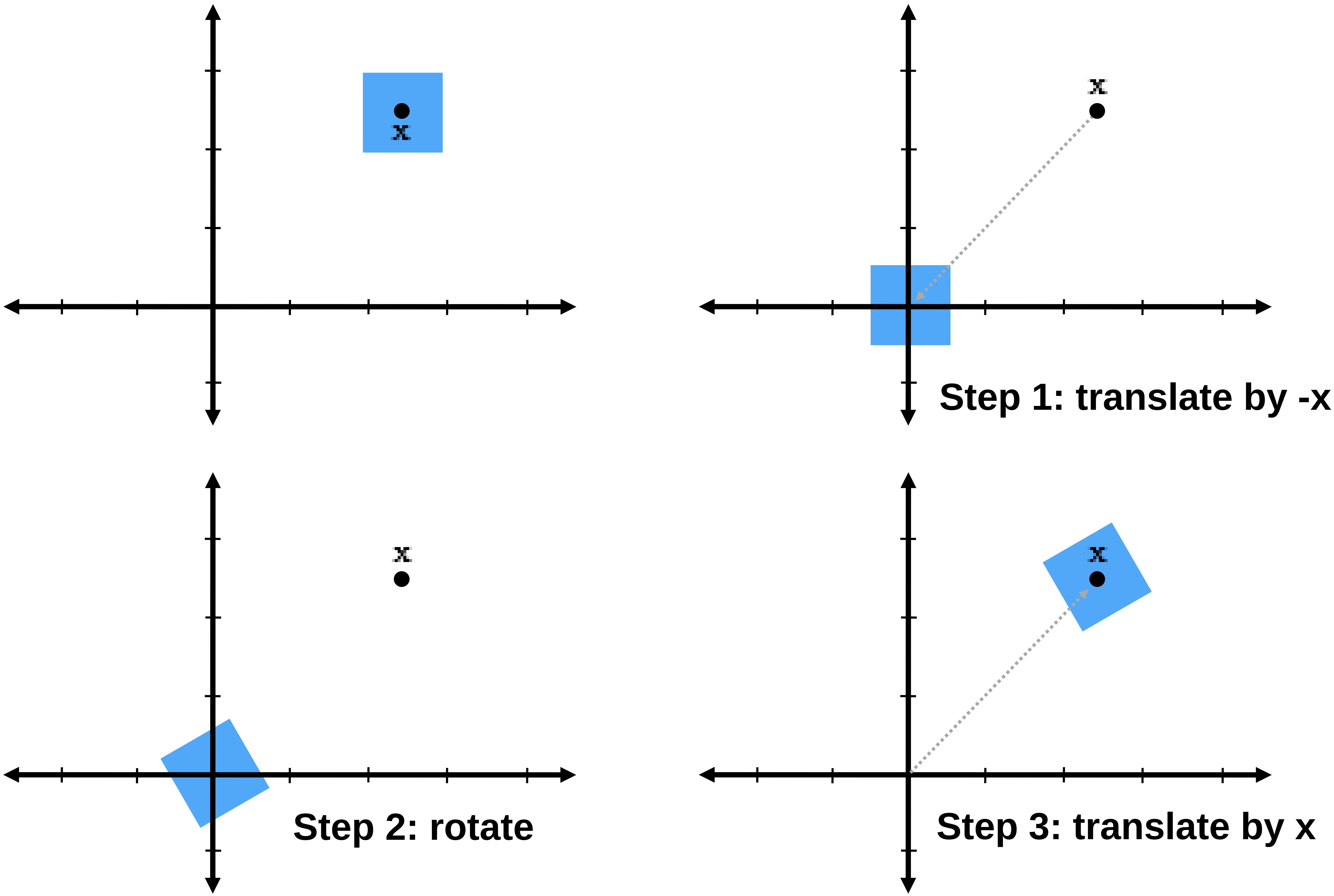
Compose linear transforms via matrix multiplication.

Enables simple & efficient implementation: reduce complex chain of transforms to a single matrix.

How would you perform these transformations?



Common pattern: rotation about point x



Q: In homogenous coordinates, what does the corresponding transformation matrix look like?

Exercise

- **Reflection about an arbitrary line**