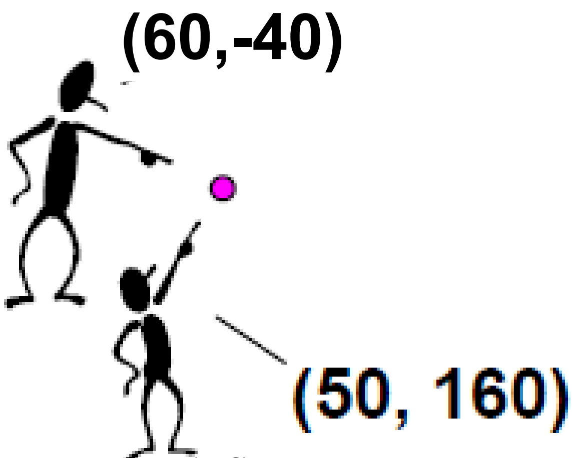
Vector Algebra
Transformations

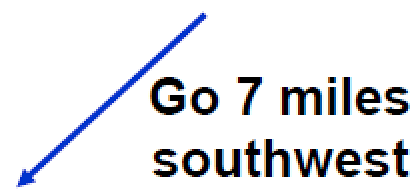
Lecture 4

Geometry

- **A part of mathematics concerned with questions of size, shape, and relative positions of figures**
- **Coordinates are used to represent points and vectors**
  - **a naming scheme**
  - **The same point can be described by different coordinates**

Points **vectors**





Coordinates

[KAIST]

Points

- **Conceptually, points and vectors are very different**
  - **A point $\dot{p}$ is a place in space**
  - **A vector $\vec{v}$ describes a direction independent of position (pay attentions notations)**

Geometry

- Linear Algebra
  - Scalar
  - Vector
  - Linear independence
  - Linear transformations
- Frames (=Coordinates)
  - Points
  - Vectors
- Affine transformations
  - Translation
  - Rotation

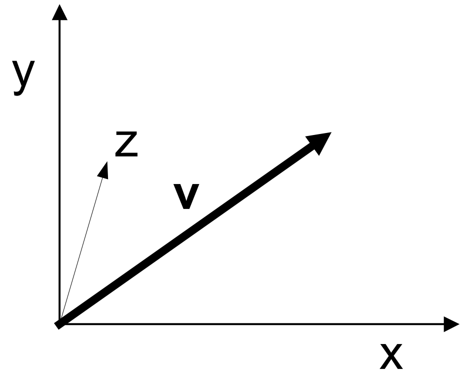
Vector Spaces

- **A vector (or linear) space V over a scalar field S consists of a set on which the following two operators are defined and the following conditions hold:**
- **Two operators for vectors:**
  - **Vector-vector addition**

$$\forall \vec{u}, \vec{v} \in V \quad \vec{u} + \vec{v} \in V$$

  - **Scalar-vector multiplication**

$$\forall \vec{u} \in V, \forall a \in S \quad a\vec{u} \in V$$



- **Notation:**
  - **Vector** $\vec{v} = \begin{bmatrix} a \\ b \\ c \end{bmatrix}$ = [a b c]^T

Vector Spaces

- **Vector-vector addition**
  - **Commutes and associates**

$$\vec{u}+\vec{v}=\vec{v}+\vec{u} \quad \vec{u}+(\vec{v}+\vec{w})=(\vec{u}+\vec{v})+\vec{w}$$

  - **An additive identity and an additive inverse for each vector**

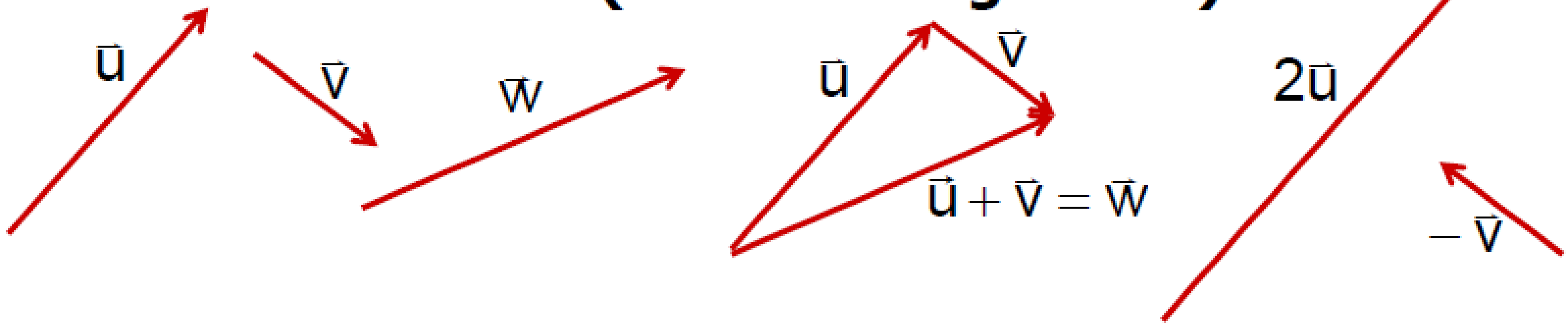
$$\vec{u}+\vec{0}=\vec{u} \quad \vec{u}+(-\vec{u})=\vec{0}$$

- **Scalar-vector multiplication distributes**

$$(a+b)\vec{u}=a\vec{u}+b\vec{u} \quad a(\vec{u}+\vec{v})=a\vec{u}+a\vec{v}$$

Example Vector Spaces

- **Geometric vectors (directed segments)**



- **N-tuples of scalars**

$$\vec{u}=(1,3,7)^t \qquad \vec{u}+\vec{v}=(3,5,4)^t=\vec{w}$$

$$\vec{v}=(2,2,-3)^t \qquad 2\vec{u}=(2,6,14)^t$$

$$\vec{w}=(3,5,4)^t \qquad -\vec{v}=(-2,-2,3)^t$$

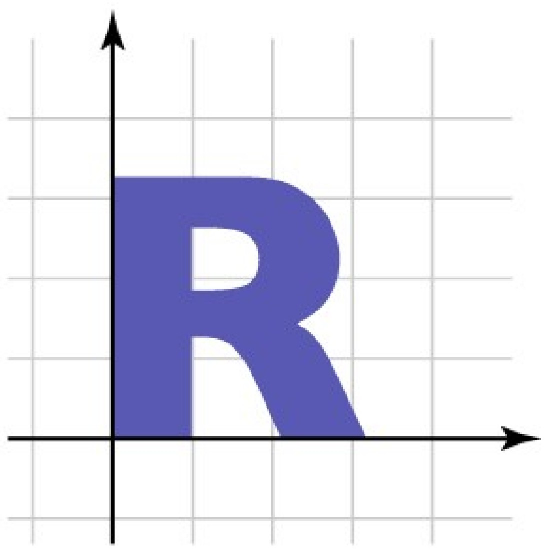
t denotes "transpose"

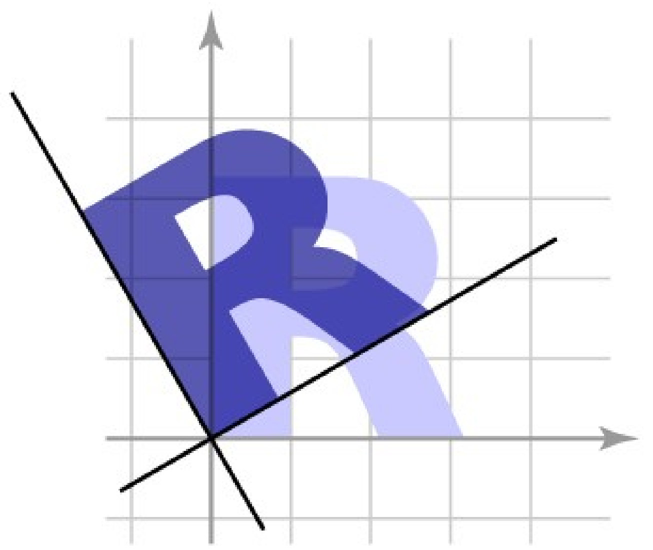
- **We can use N-tuples to represent vectors**

Transforming geometry

- Move a subset of the space (in 2D case, plane) using a mapping from the plane to itself

$$S \rightarrow \{T(\mathbf{v}) \mid \mathbf{v} \in S\}$$





Linear transformations

- One way to define a transformation is by matrix multiplication:

$$T(\mathbf{v}) = M\mathbf{v}$$

- Such transformations are *linear*, which is to say:

$$T(a\mathbf{u} + \mathbf{v}) = aT(\mathbf{u}) + T(\mathbf{v})$$

(and in fact all linear transformations can be written this way)

What is a Matrix?

- A matrix is a set of elements, organized into rows and columns

$$\begin{array}{cc} & \text{rows} \\ \text{columns} & \begin{bmatrix} a & b \\ c & d \end{bmatrix} \end{array}$$

Matrix operating on vectors

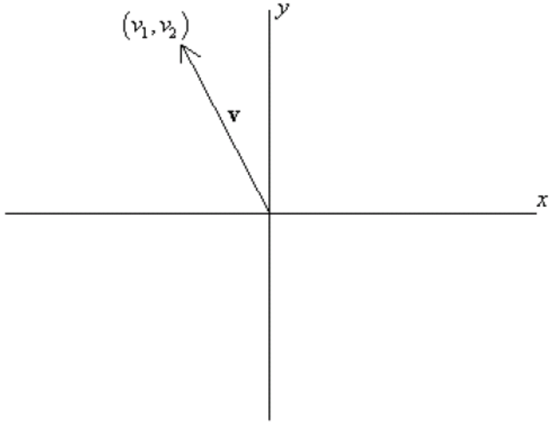
- Matrix is like a function that transforms the vectors on a plane
- Matrix operating on a general point => transforms x and y components
- *System of linear equations*: matrix is just the bunch of coefficients

- x' = ax + by
- y' = cx + dy

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x' \\ y' \end{bmatrix}$$

Bases & Orthonormal Bases

- Basis vectors (or axes): frame of reference



Any point in a space is a *linear combination* of the basis vectors
Usually, orthonormal matrices are used for defining coordinate frames
Ortho-Normal: orthogonal + normal

[**Orthogonal**: dot product is zero
Normal: magnitude is one]

$$x = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}^T \qquad x \cdot y = 0$$

$$y = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix}^T \qquad x \cdot z = 0$$

$$z = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T \qquad y \cdot z = 0$$

Geometry of 2D linear trans.

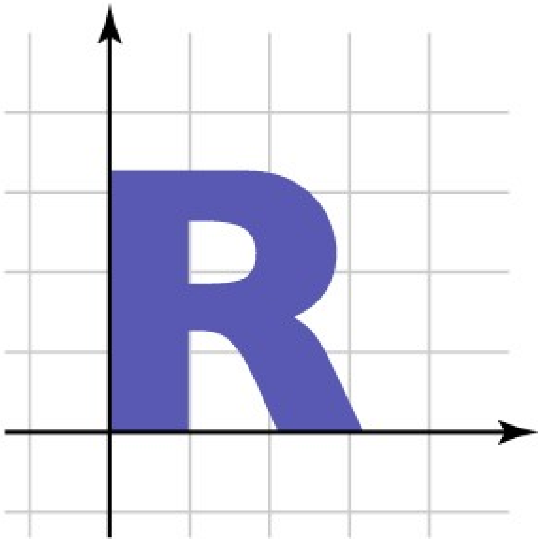
- 2x2 matrices have simple geometric interpretations
  - uniform scale
  - non-uniform scale
  - rotation
  - shear
  - reflection

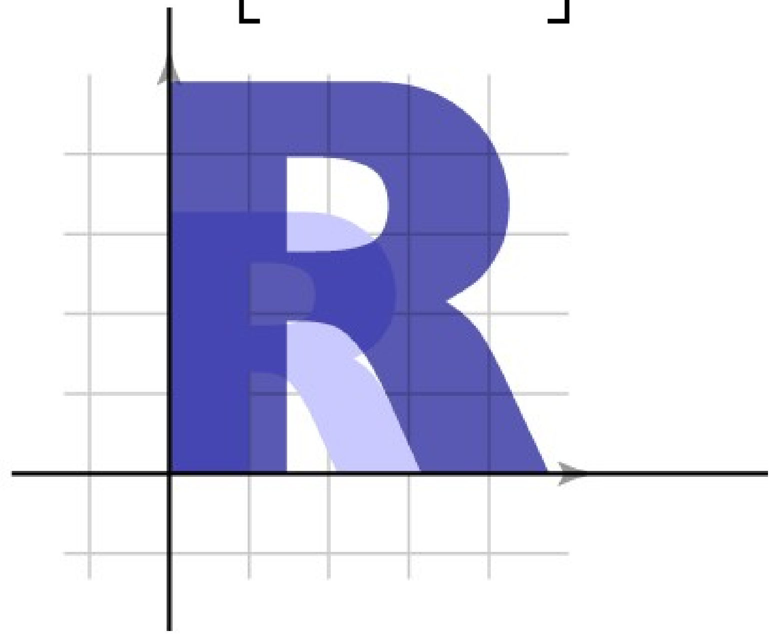
Linear transformation gallery

- Uniform scale

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

$$\begin{bmatrix} 1.5 & 0 \\ 0 & 1.5 \end{bmatrix}$$



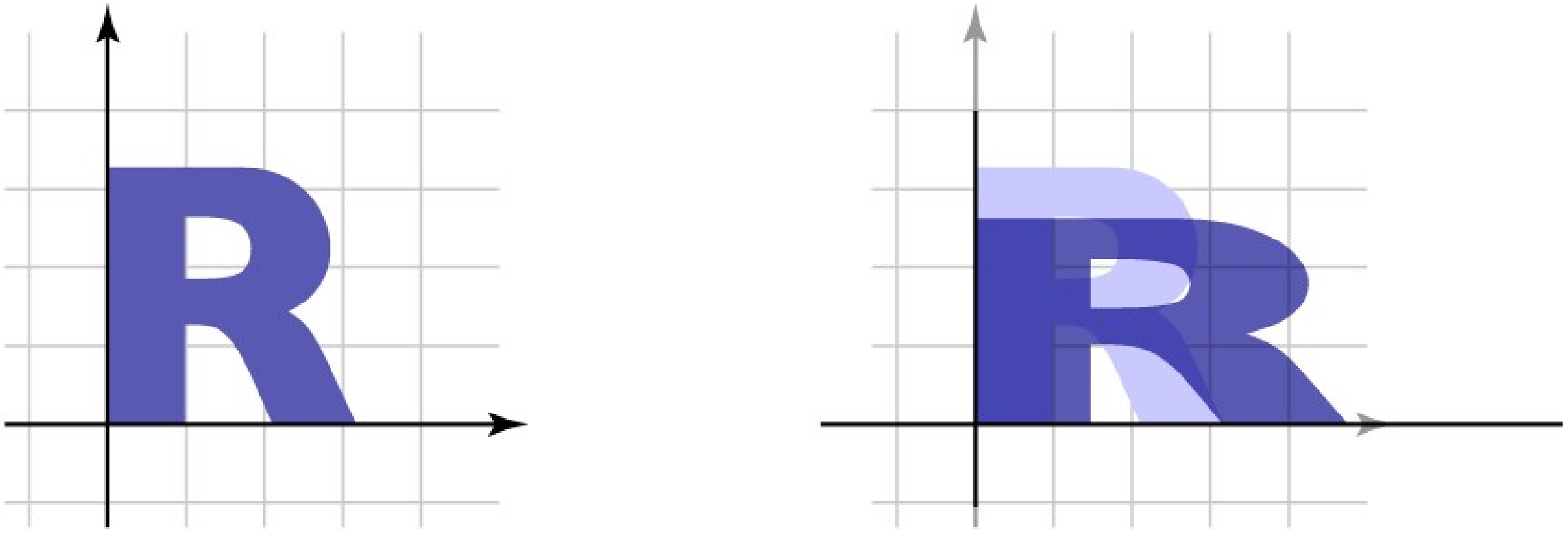


Linear transformation gallery

- Nonuniform scale

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

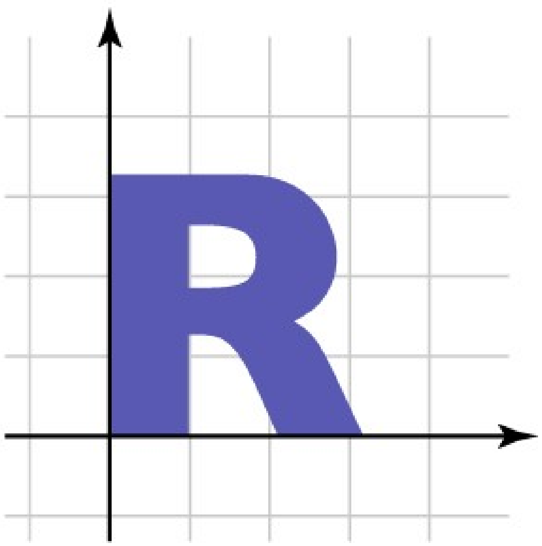
$$\begin{bmatrix} 1.5 & 0 \\ 0 & 0.8 \end{bmatrix}$$

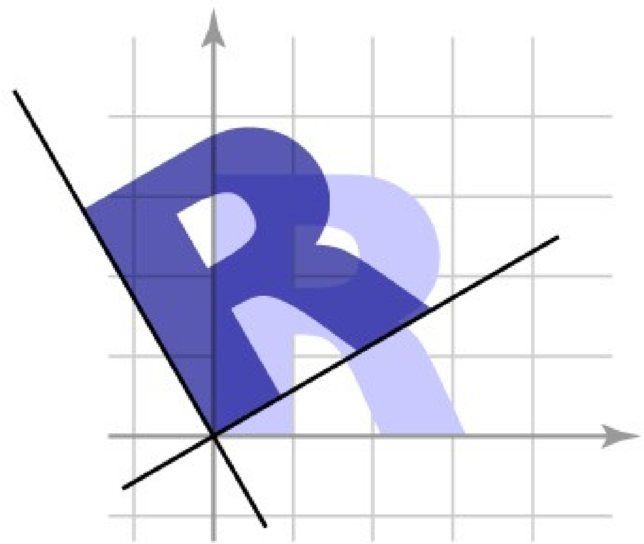


Linear transformation gallery

- Rotation $\begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x\cos\theta - y\sin\theta \\ x\sin\theta + y\cos\theta \end{bmatrix}$

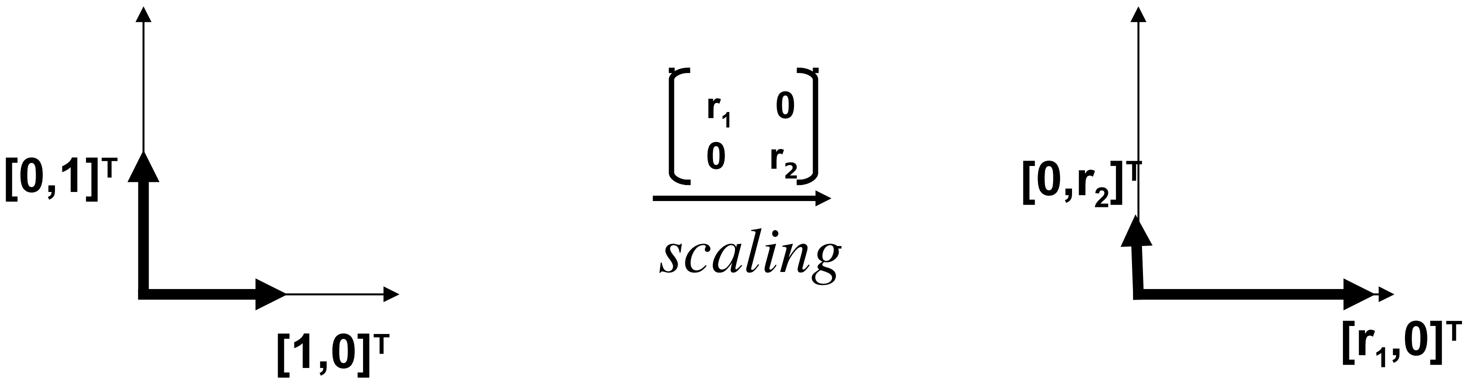
$$\begin{bmatrix} 0.866 & -.05 \\ 0.5 & 0.866 \end{bmatrix}$$

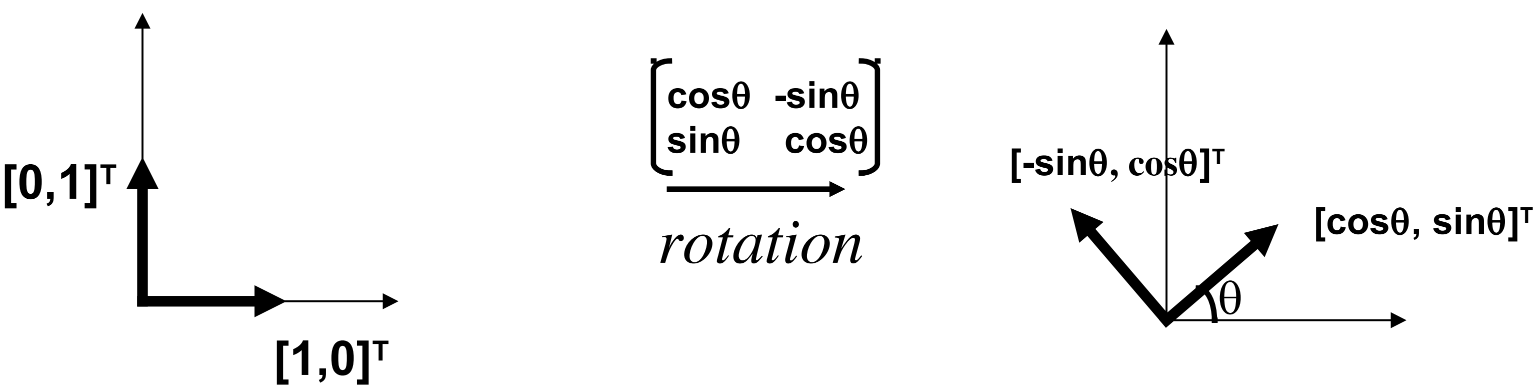




Matrices: Scaling, Rotation, Identity

- Scaling without rotation => "**<u>diagonal</u>** matrix"
- Rotation without stretching => "**<u>orthonormal</u>** matrix" **O**
- **<u>Identity</u>** ("do nothing") matrix = unit scaling, no rotation

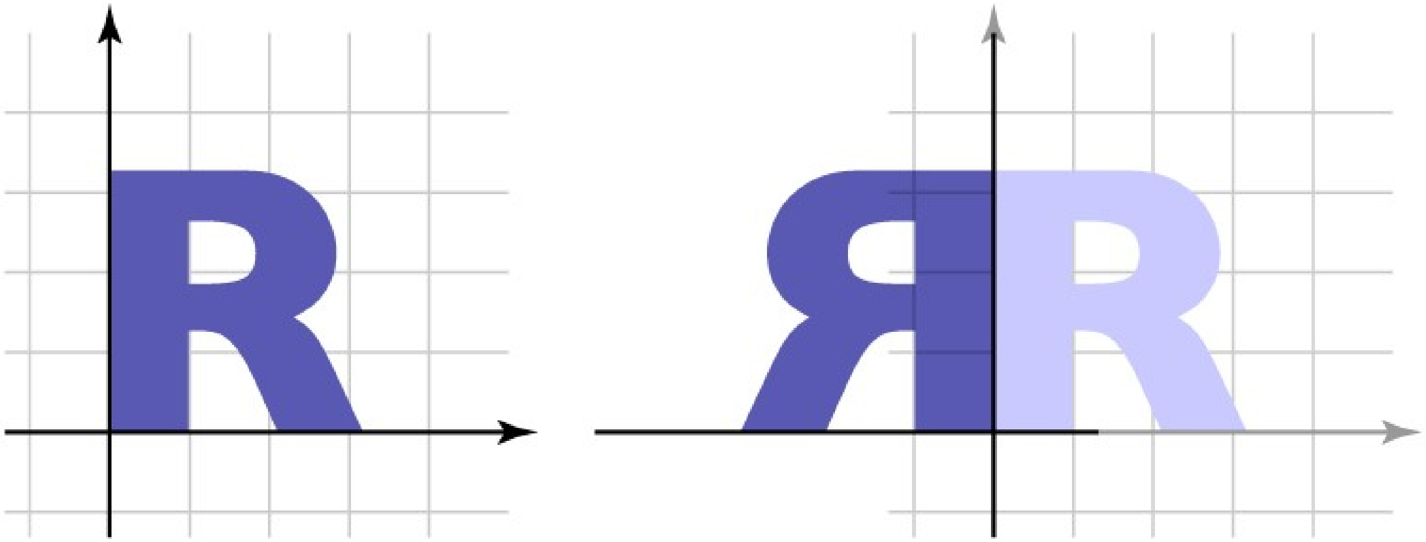




Linear transformation gallery

- Reflection
  - can consider it a special case of nonuniform scale

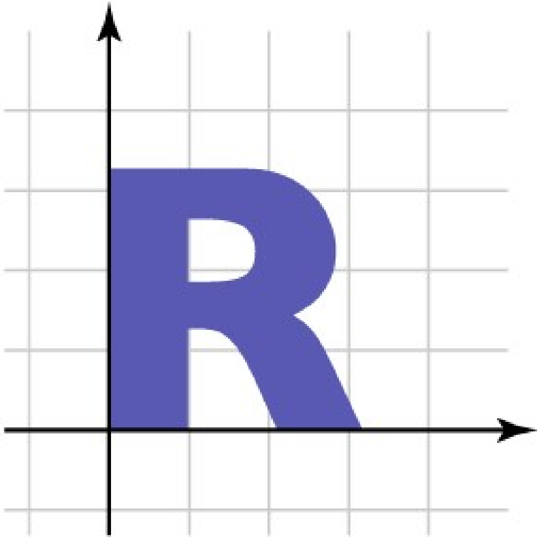
$$\begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}$$

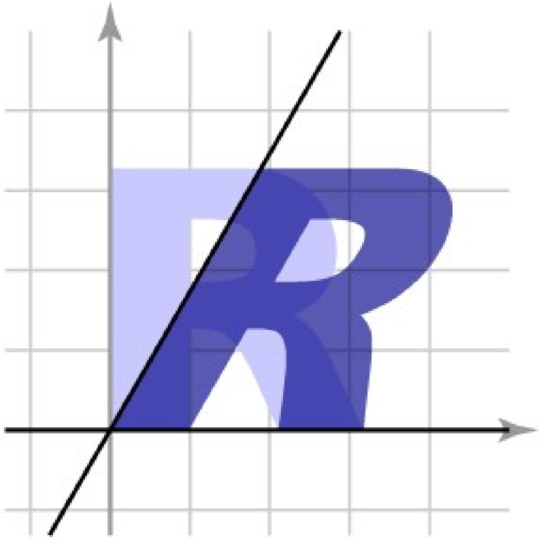


Linear transformation gallery

- Shear $$\begin{bmatrix} 1 & a \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x + ay \\ y \end{bmatrix}$$

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





Composing transformations

- Want to move an object, then move it some more
  - $\mathbf{p} \to T(\mathbf{p}) \to S(T(\mathbf{p})) = (S \circ T)(\mathbf{p})$
- We need to represent *S* o *T* ("S compose T")
  - and would like to use the same representation as for *S* and *T*

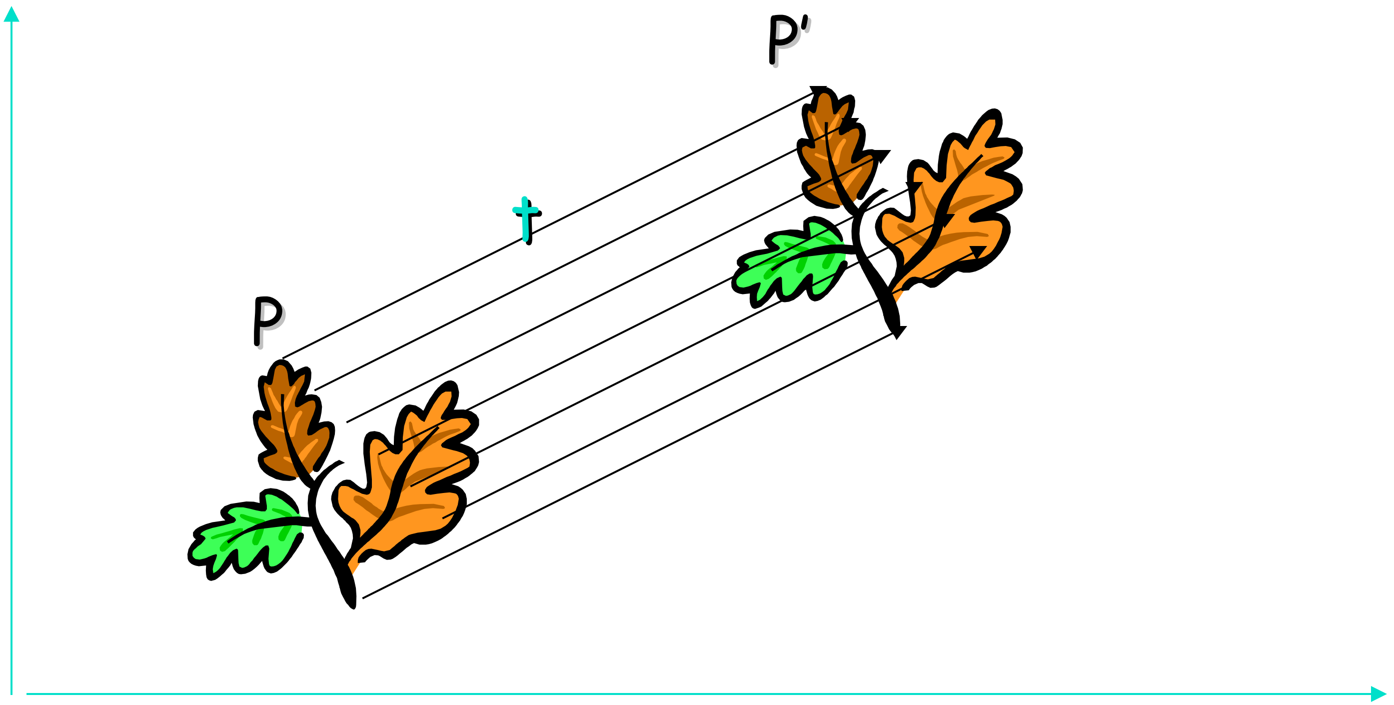
Composing transformations

- Composing linear transformations is straightforward
  - $T(\mathbf{p}) = M_T\mathbf{p}; S(\mathbf{p}) = M_S\mathbf{p}$

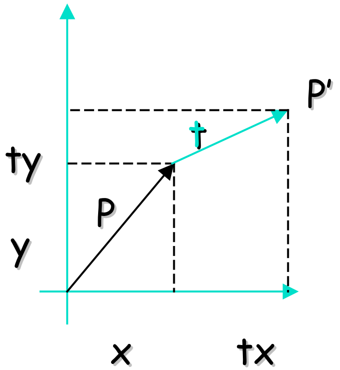
$$(S \circ T)(\mathbf{p}) = M_S M_T \mathbf{p}$$

- Transforming first by M_T then by M_S is the same as transforming by $M_S M_T$
  - only sometimes commutative
    - e.g. 2D rotations & uniform scales
    - e.g. non-uniform scales w/o rotation
  - Note $M_S M_T$, or S o T, is T first, then S

Translation



$$\mathbf{P'} = (x + t_x, y + t_y) = \mathbf{P} + \mathbf{t}$$



Composing translations

- Composing translations is easy
  - $\mathbf{p} \to T(\mathbf{p}) \to S(T(\mathbf{p})) = (S \circ T)(\mathbf{p})$
- Translation by $\mathbf{u}_T$ then by $\mathbf{u}_S$ is translation by $\mathbf{u}_T + \mathbf{u}_S$
- commutative!

$$T(\mathbf{p}) = \mathbf{p} + \mathbf{u}_T; S(\mathbf{p}) = \mathbf{p} + \mathbf{u}_S$$

$$(S \circ T)(\mathbf{p}) = \mathbf{p} + (\mathbf{u}_T + \mathbf{u}_S)$$

Is translation linear transformation?

- Translation is the simplest transformation:
- Inverse:

$$T(\mathbf{v}) = \mathbf{v} + \mathbf{u}$$

$$T^{-1}(\mathbf{v}) = \mathbf{v} - \mathbf{u}$$

Is translation linear transformation?

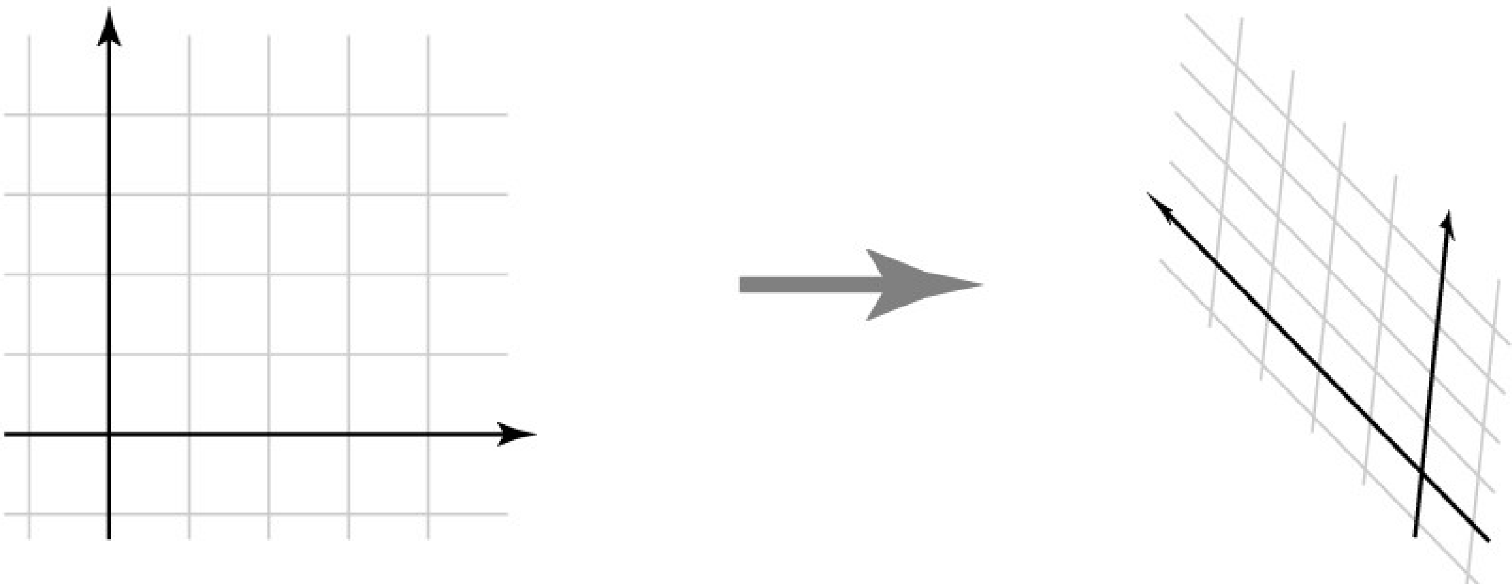
- No!   because T(v)= I v + u != I v

- Affine transformation

  T(v)=Mv + u

Affine transformations

- straight lines preserved; parallel lines preserved
- ratios of lengths along lines preserved (midpoints preserved)
- Origin



Combining linear with translation

- Need to use both in single framework
- Can represent arbitrary seq. as
  - $T(\mathbf{p}) = M_T\mathbf{p} + \mathbf{u}_T$
  - $S(\mathbf{p}) = M_S\mathbf{p} + \mathbf{u}_S$
  - $(S \circ T)(\mathbf{p}) = M_S(M_T\mathbf{p} + \mathbf{u}_T) + \mathbf{u}_S$
  - $= (M_S M_T)\mathbf{p} + (M_S\mathbf{u}_T + \mathbf{u}_S)$
- Transforming by M_T and $\mathbf{u}_T$, then by M_S and $\mathbf{u}_S$, is the same as transforming by $M_S M_T$ and $M_s\mathbf{u}_T + \mathbf{u}_S$
- This will work but is a little awkward

Homogeneous coordinates

- A trick for representing the foregoing more elegantly
- Extra component *w* for vectors, extra row/column for matrices
  - for affine, can always keep $w = 1$
- Represent linear transformations with dummy extra row and column

$$\begin{bmatrix} a & b & 0 \\ c & d & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} ax + by \\ cx + dy \\ 1 \end{bmatrix}$$

Homogeneous coordinates

- Represent translation using the extra column

$$\begin{bmatrix} 1 & 0 & t \\ 0 & 1 & s \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} x + t \\ y + s \\ 1 \end{bmatrix}$$

Homogeneous coordinates

- Composition just works, by 3x3 matrix multiplication

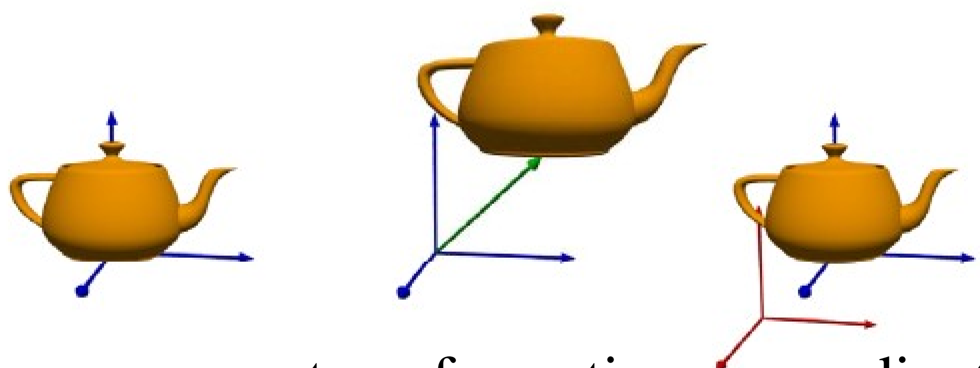
$$\begin{bmatrix} M_S & \mathbf{u}_S \\ 0 & 1 \end{bmatrix} \begin{bmatrix} M_T & \mathbf{u}_T \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{p} \\ 1 \end{bmatrix}$$

$$= \begin{bmatrix} (M_S M_T)\mathbf{p} + (M_S \mathbf{u}_T + \mathbf{u}_S) \\ 1 \end{bmatrix}$$

- This is exactly the same result
  - but cleaner
  - and generalizes in useful ways as we'll see later

Homogeneous coordinates in 3D (Affine transformation)

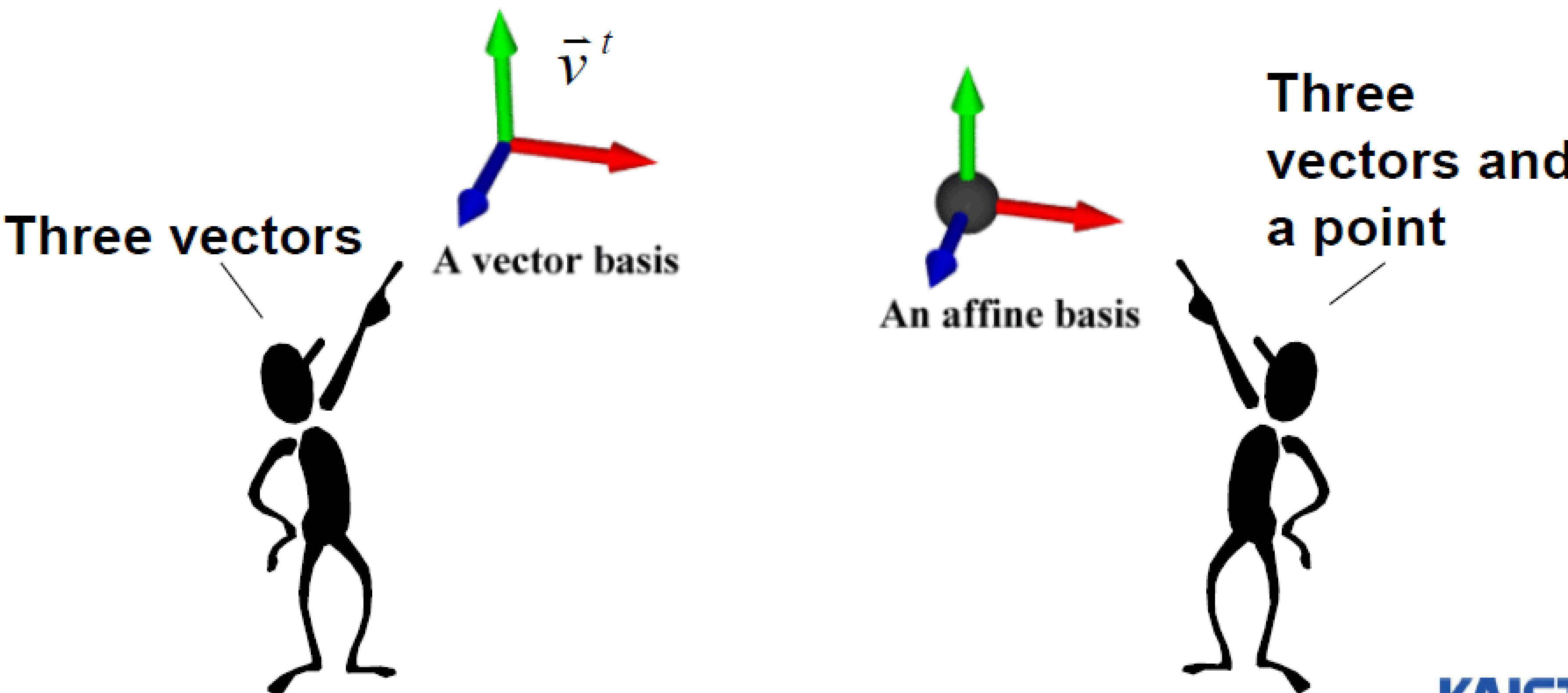
$$\begin{pmatrix} v_{1x} & v_{2x} & v_{3x} & o_x \\ v_{1y} & v_{2y} & v_{3y} & o_y \\ v_{1z} & v_{2z} & v_{3z} & o_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} c_1 \\ c_2 \\ c_3 \\ 1 \end{pmatrix} = \begin{pmatrix} \boldsymbol{v}_{\boldsymbol{x}} \cdot \boldsymbol{c} + o_x \\ \boldsymbol{v}_{\boldsymbol{y}} \cdot \boldsymbol{c} + o_y \\ \boldsymbol{v}_{\boldsymbol{z}} \cdot \boldsymbol{c} + o_z \\ 1 \end{pmatrix} = \begin{pmatrix} \boldsymbol{V}\boldsymbol{c} + \boldsymbol{o} \\ 1 \end{pmatrix}$$



transformation coordinates

Pictures of Frames

- **Graphically, we will distinguish between vector bases and affine bases (frames) using the following convention**



Coordinates :

A Basis for Points

- **Key distinction between vectors and points: points are *absolute*, vectors are *relative***
- **Vector space is completely defined by a set of basis vectors**
- **The space that points live in requires the specification of an absolute origin**

$$p = \dot{o} + \sum_i \vec{v}_i c_i = \begin{bmatrix} \vec{v}_1 & \vec{v}_2 & \vec{v}_3 & \dot{o} \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ 1 \end{bmatrix}$$

Notice how 4 scalars (one of which is 1) are required to identify a 3D point

KAIST

Frames

- **Points live in *Affine* spaces**
- **Affine-basis-sets are called *frames***

$$\dot{\mathrm{f}}^{t} = \begin{bmatrix} \vec{\mathrm{v}}_1 & \vec{\mathrm{v}}_2 & \vec{\mathrm{v}}_3 & \dot{\mathrm{o}} \end{bmatrix}$$

- **Frames can describe vectors as well as points**

$$\dot{p} = \begin{bmatrix} \vec{v}_1 & \vec{v}_2 & \vec{v}_3 & \dot{o} \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ 1 \end{bmatrix} \qquad \vec{x} = \begin{bmatrix} \vec{v}_1 & \vec{v}_2 & \vec{v}_3 & \dot{o} \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ 0 \end{bmatrix}$$

A Consistent Model

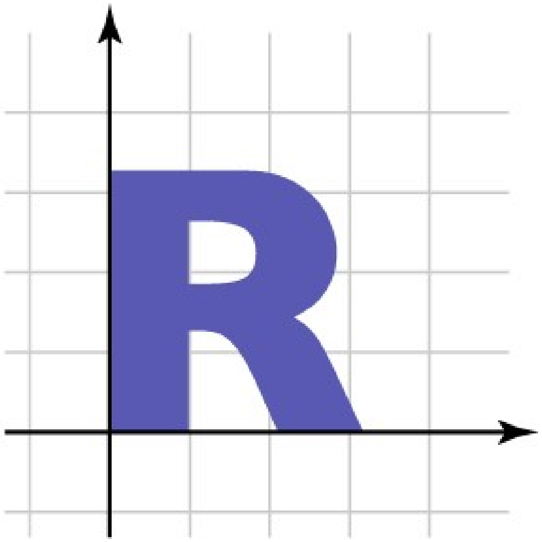
- **Behavior of affine frame coordinates is completely consistent with our intuition**
  - **Subtracting two points yields a vector**
  - **Adding a vector to a point produces a point**
  - **If you multiply a vector by a scalar you still get a vector**
  - **Scaling points gives a nonsense 4th coordinate element in most cases**

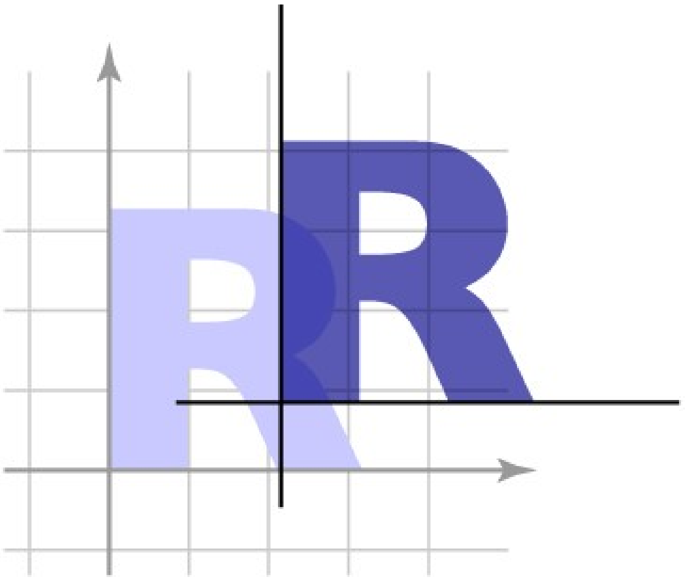
$$\begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ 1 \end{bmatrix} - \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ 1 \end{bmatrix} = \begin{bmatrix} a_1 - b_1 \\ a_2 - b_2 \\ a_3 - b_3 \\ 0 \end{bmatrix} \qquad \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ 1 \end{bmatrix} + \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ 0 \end{bmatrix} = \begin{bmatrix} a_1 + v_1 \\ a_2 + v_2 \\ a_3 + v_3 \\ 1 \end{bmatrix}$$

2D Affine transformation gallery

- Translation

$$\begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \qquad \begin{bmatrix} 1 & 0 & 2.15 \\ 0 & 1 & 0.85 \\ 0 & 0 & 1 \end{bmatrix}$$

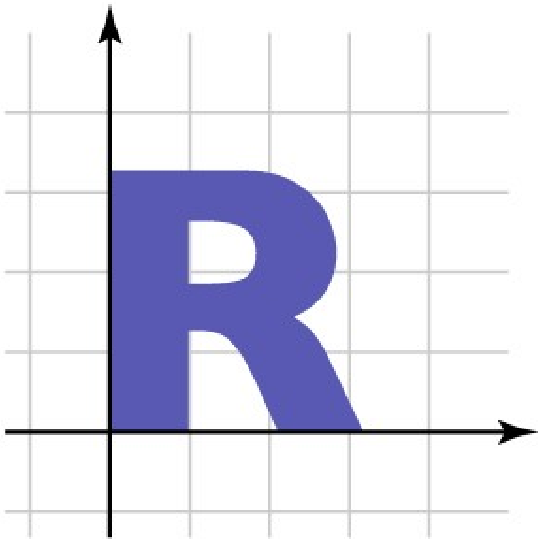


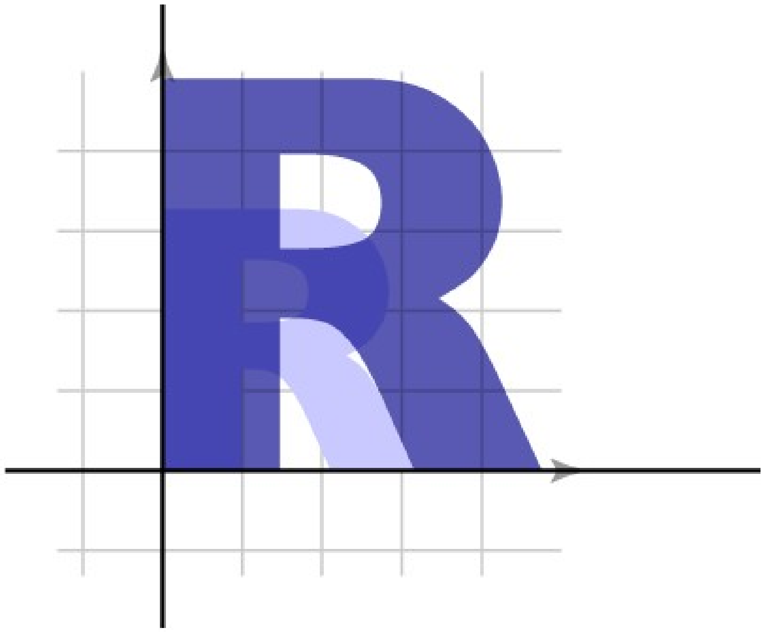


Affine transformation gallery

- Uniform scale

$$\begin{bmatrix} s & 0 & 0 \\ 0 & s & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad \begin{bmatrix} 1.5 & 0 & 0 \\ 0 & 1.5 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

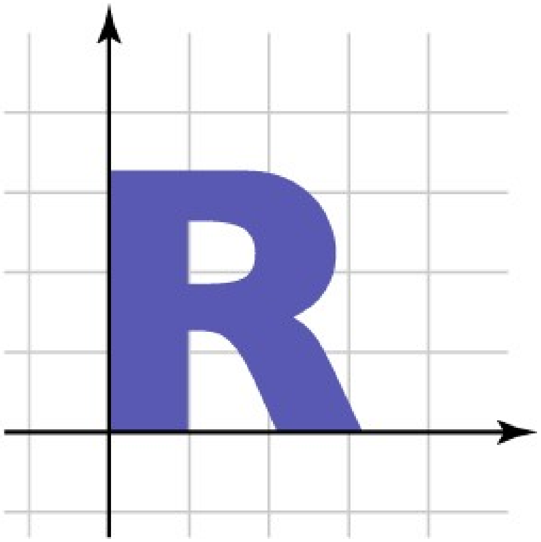


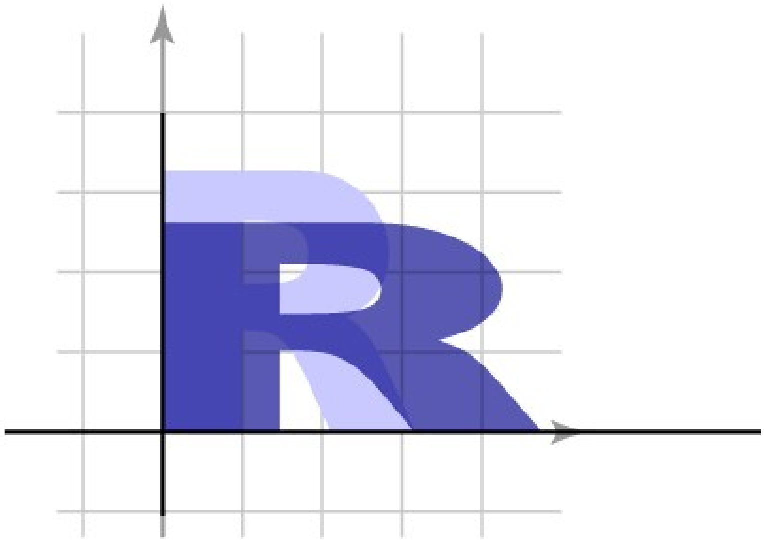


Affine transformation gallery

- Nonuniform sca

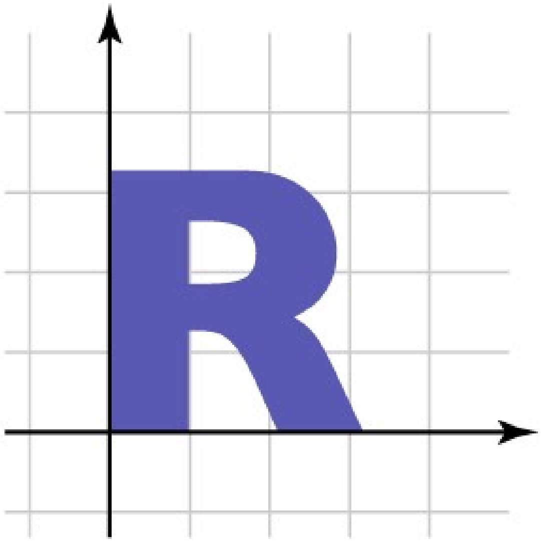
$$\begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad \begin{bmatrix} 1.5 & 0 & 0 \\ 0 & 0.8 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

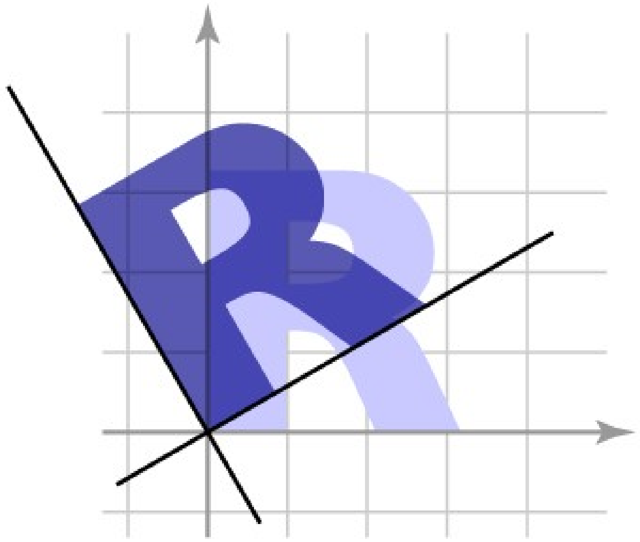




Affine transformation gallery

- Rotation $\begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$ $\begin{bmatrix} 0.866 & -0.5 & 0 \\ 0.5 & 0.866 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

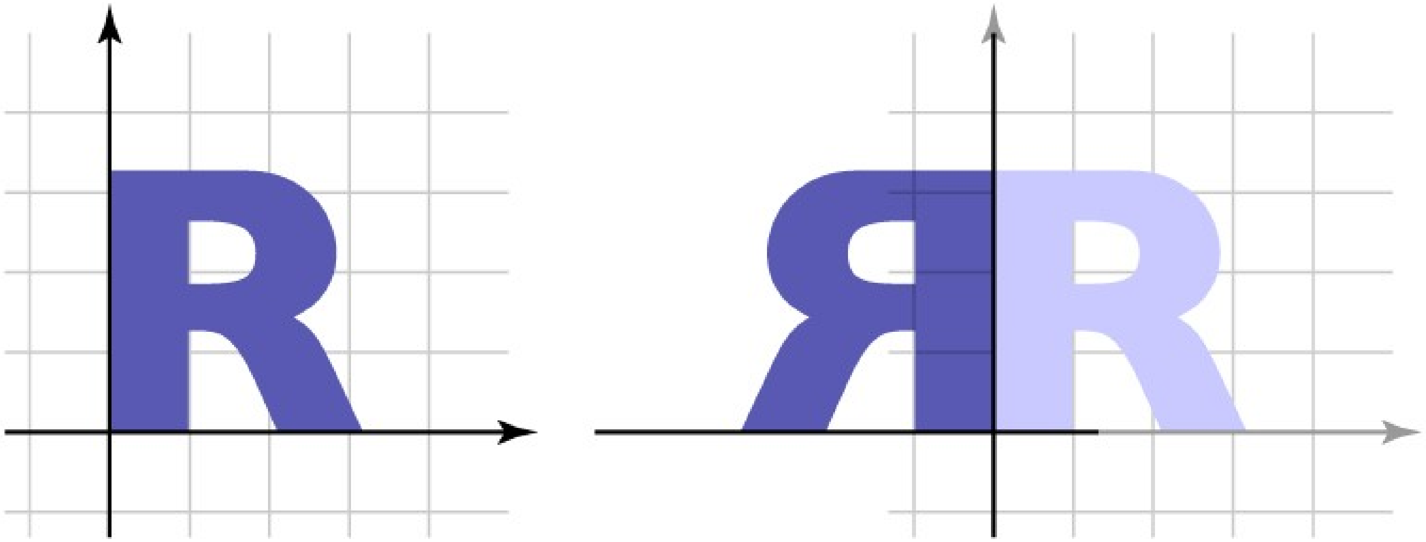




Affine transformation gallery

- Reflection
  - can consider it a special case of nonuniform scale

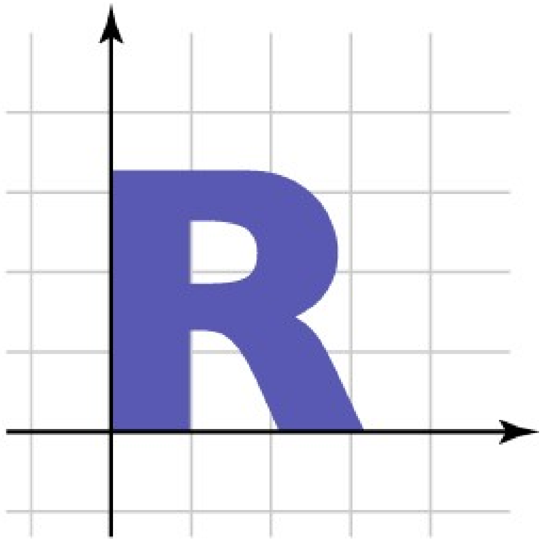
$$\begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

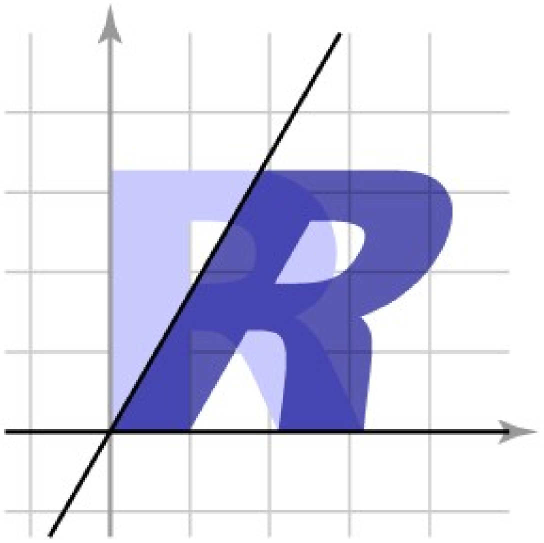


Affine transformation gallery

- Shear

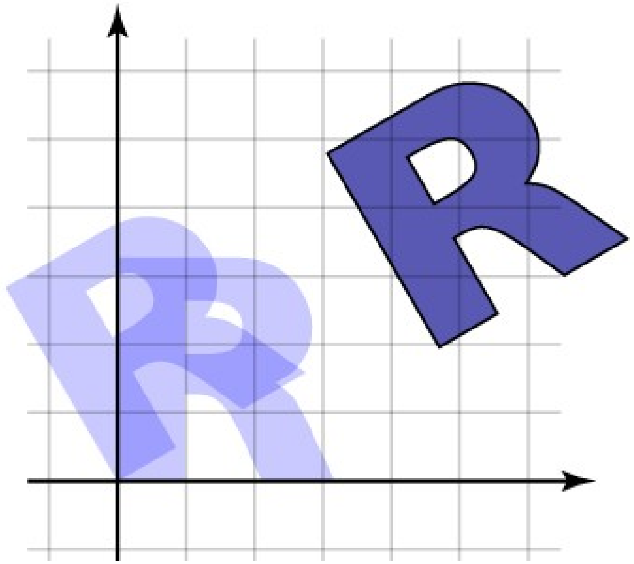
$$\begin{bmatrix} 1 & a & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad \begin{bmatrix} 1 & 0.5 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$



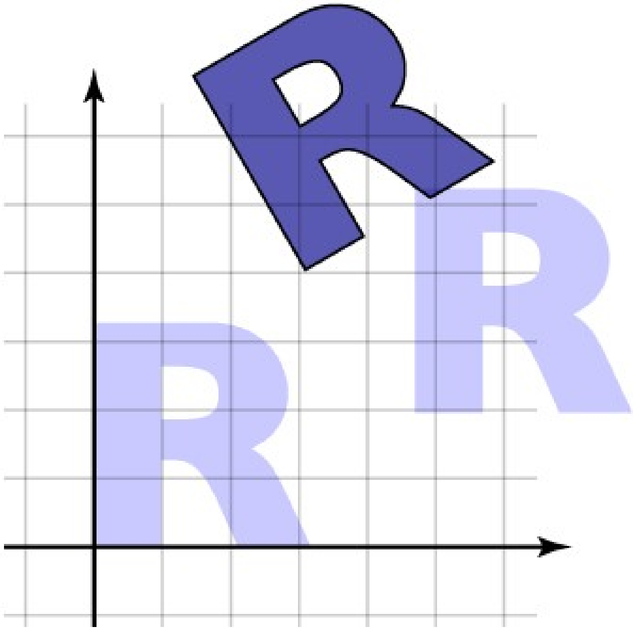


Composite affine transformations

- In general **not** commutative: order matters!



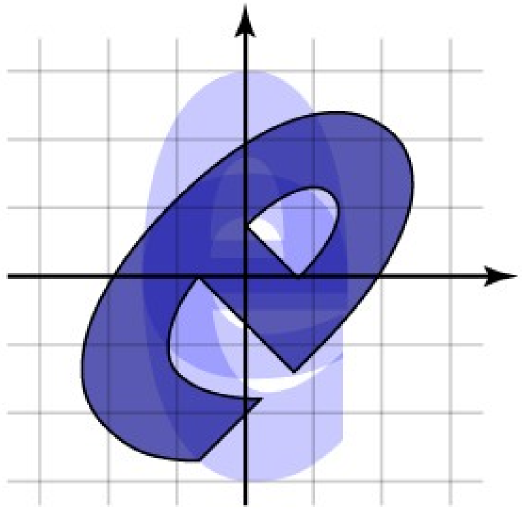
rotate, then translate



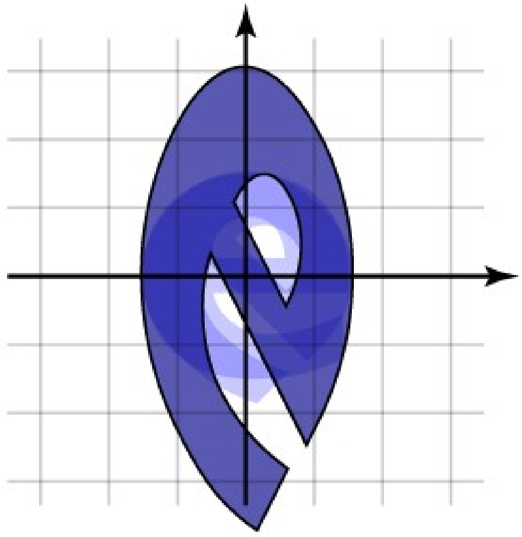
translate, then rotate

Composite affine transformations

- Another example



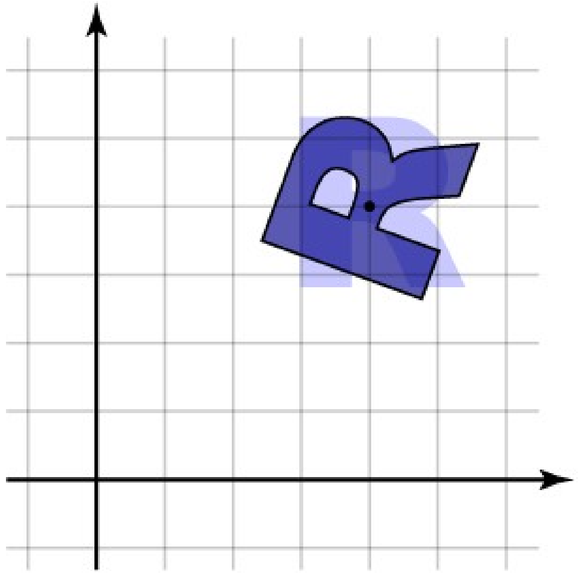
scale, then rotate



rotate, then scale

Composing to change axes

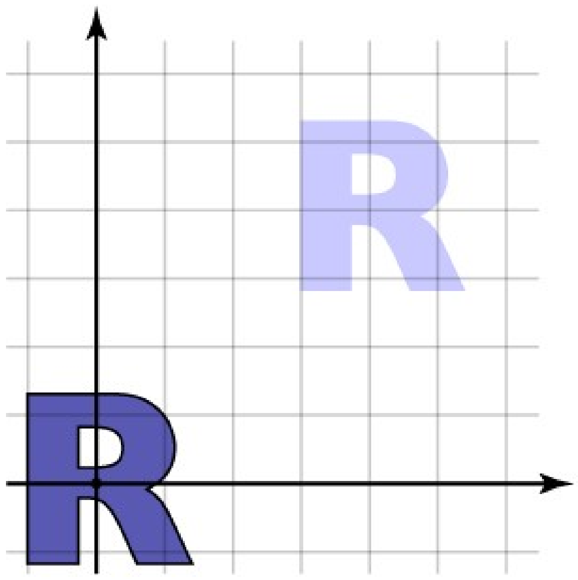
- Want to rotate about a particular point
  - could work out formulas directly…
- Know how to rotate about the origin
  - so translate that point to the origin



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

Composing to change axes

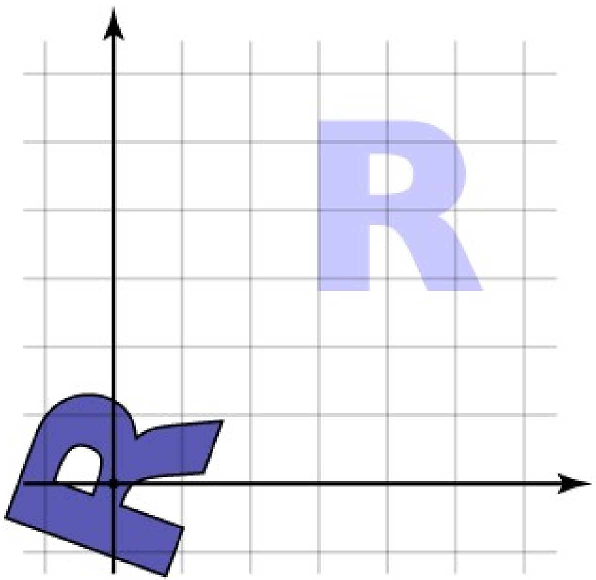
- Want to rotate about a particular point
  - could work out formulas directly…
- Know how to rotate about the origin
  - so translate that point to the origin



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

Composing to change axes

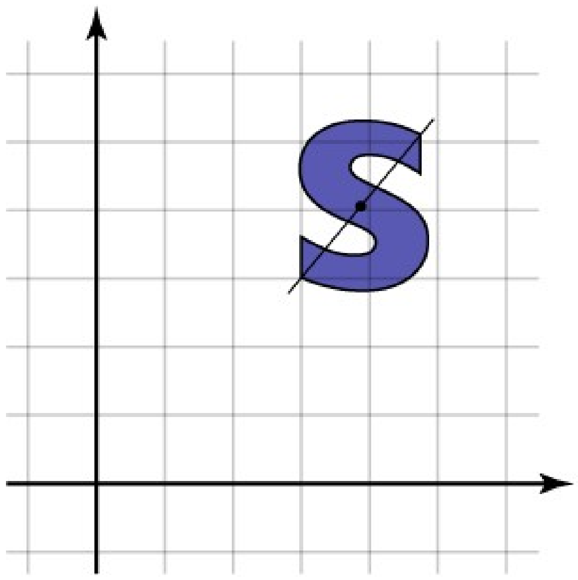
- Want to rotate about a particular point
  - could work out formulas directly…
- Know how to rotate about the origin
  - so translate that point to the origin



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

Composing to change axes

- Want to scale along a particular axis and point
- Know how to scale along the y axis at the origin

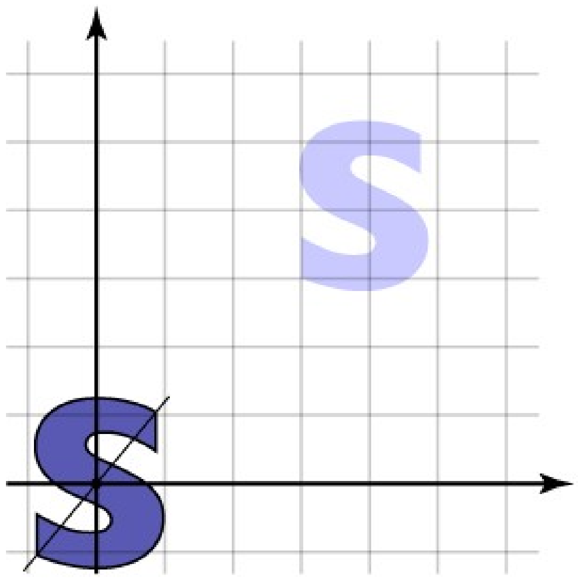


igin and rotate to align axes

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

Composing to change axes

- Want to scale along a particular axis and point
- Know how to scale along the y axis at the origin



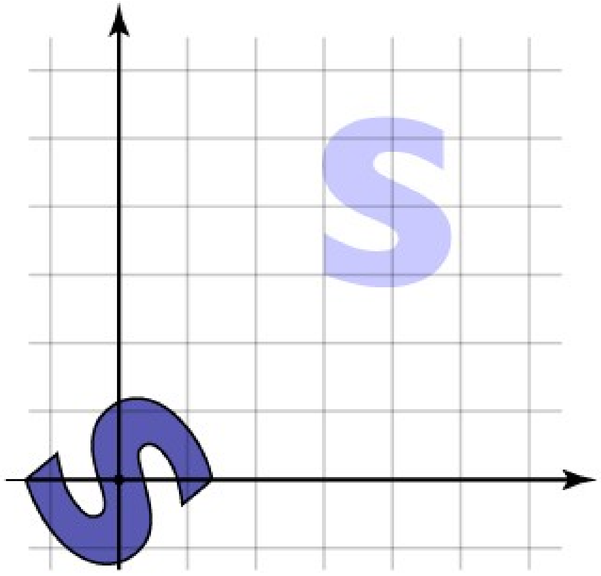
igin and rotate to align axes

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

Composing to change axes

- Want to scale along a particular axis and point
- Know how to scale along the y axis at the origin

igin and rotate to align axes

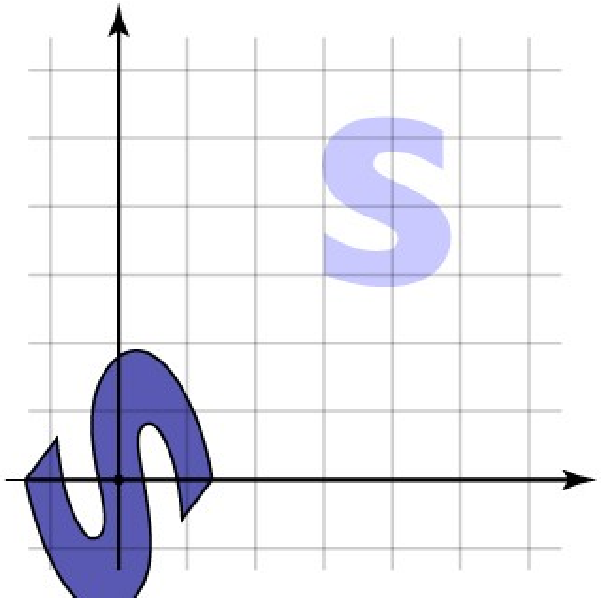


$$M = T^{-1} R^{-1} S R T$$

Composing to change axes

- Want to scale along a particular axis and point
- Know how to scale along the y axis at the origin

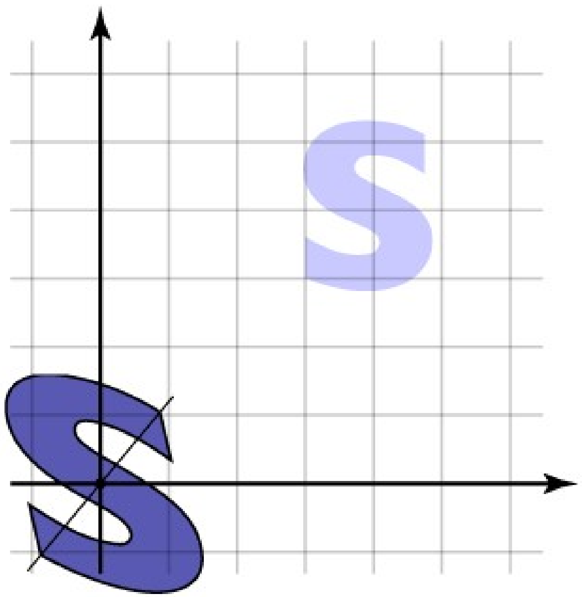
igin and rotate to align axes



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

Composing to change axes

- Want to scale along a particular axis and point
- Know how to scale along the y axis at the origin



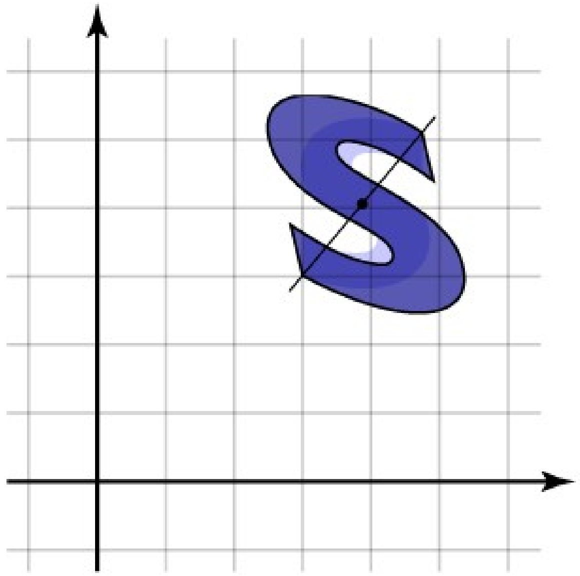
igin and rotate to align axes

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

Composing to change axes

- Want to scale along a particular axis and point
- Know how to scale along the y axis at the origin

igin and rotate to align axes



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

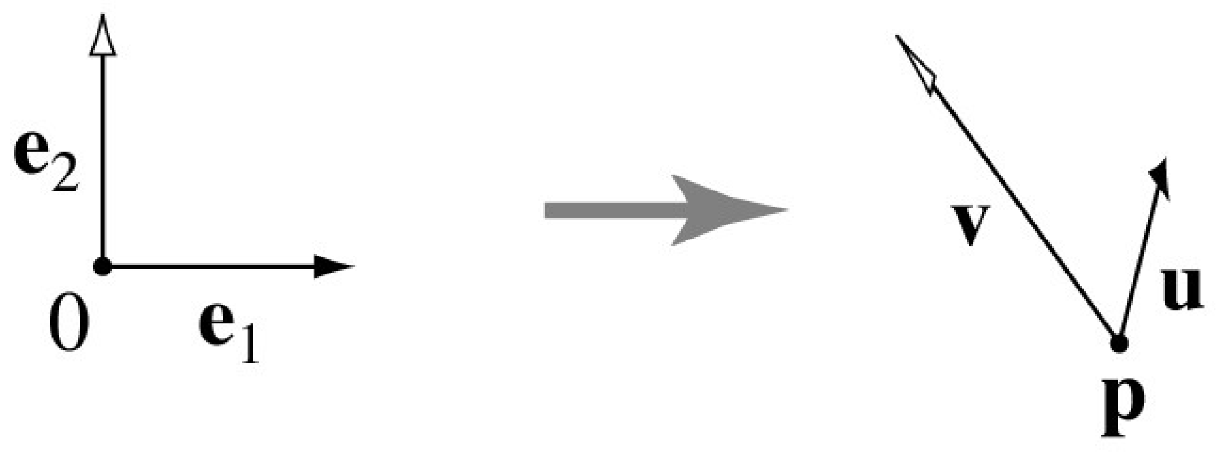
More math background

- Coordinate systems
  - Expressing vectors with respect to bases
  - Linear transformations as changes of basis

Affine change of coordinates

- Six degrees of freedom

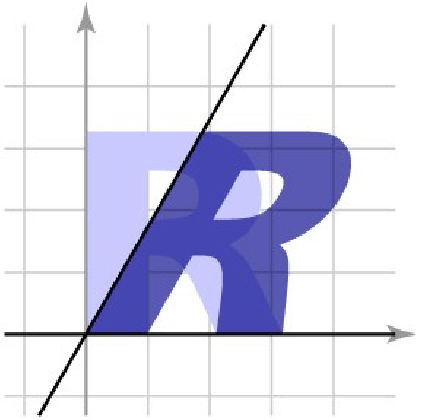
$$\begin{bmatrix} a_1 & a_2 & a_3 \\ a_4 & a_5 & a_6 \\ 0 & 0 & 1 \end{bmatrix} \quad \text{or} \quad \begin{bmatrix} \mathbf{u} & \mathbf{v} & \mathbf{p} \\ 0 & 0 & 1 \end{bmatrix}$$



Affine change of coordinates

- A new way to "read off" the matri
  - e.g. shear from earlier
  - can look at picture, see ef on basis vectors, write down matrix

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



- Also an easy way to construct transforms
  - e. g. scale by 2 across direction (1,2)

Rigid motions (Proper Euclidean space)

- A transform made up of only translation and rotation is a *rigid motion* or a *rigid body transformation*
- The linear part is an orthonormal matrix

$$R = \begin{bmatrix} Q & \mathbf{u} \\ 0 & 1 \end{bmatrix}$$

- Inverse of orthonormal matrix is transpose
  - so inverse of rigid motion is easy:

$$R^{-1}R = \begin{bmatrix} Q^T & -Q^T\mathbf{u} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} Q & \mathbf{u} \\ 0 & 1 \end{bmatrix}$$

Transforming points and vectors

- Recall distinction points vs. vectors
  - vectors are just offsets (differences between points)
  - points have a location
    - represented by vector offset from a fixed origin
- Points and vectors transform differently
  - points respond to translation; vectors do not

$$\mathbf{v} = \mathbf{p} - \mathbf{q}$$

$$T(\mathbf{x}) = M\mathbf{x} + \mathbf{t}$$

$$\begin{aligned} T(\mathbf{p} - \mathbf{q}) &= M\mathbf{p} + \mathbf{t} - (M\mathbf{q} + \mathbf{t}) \\ &= M(\mathbf{p} - \mathbf{q}) + (\mathbf{t} - \mathbf{t}) = M\mathbf{v} \end{aligned}$$

Transforming points and vectors

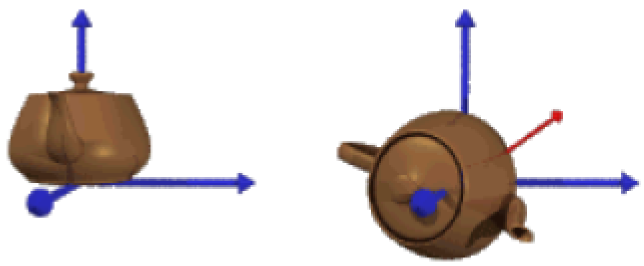
- Homogeneous coords. let us exclude translation
  - just put 0 rather than 1 in the last place

$$\begin{bmatrix} M & \mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix} \begin{bmatrix} \mathbf{p} \\ 1 \end{bmatrix} = \begin{bmatrix} M\mathbf{p} + \mathbf{t} \\ 1 \end{bmatrix} \qquad \begin{bmatrix} M & \mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix} \begin{bmatrix} \mathbf{v} \\ 0 \end{bmatrix} = \begin{bmatrix} M\mathbf{v} \\ 0 \end{bmatrix}$$

  - and note that subtracting two points cancels the extra coordinate, resulting in a vector!

3D Rotations

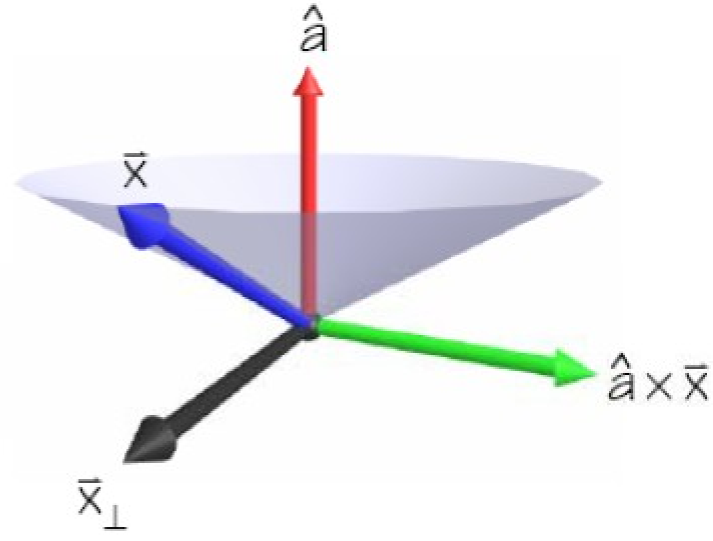
- More complicated than 2D rotations
  - Rotate objects along a rotation axis



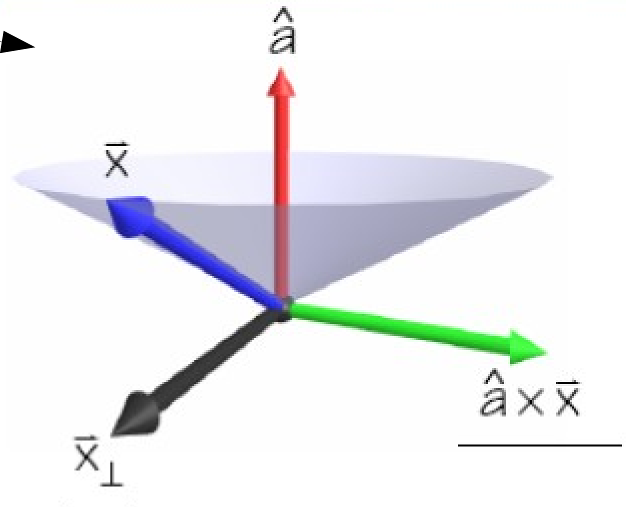
- Several approaches
  - Compose three canonical rotations about the axes
  - Quaternions

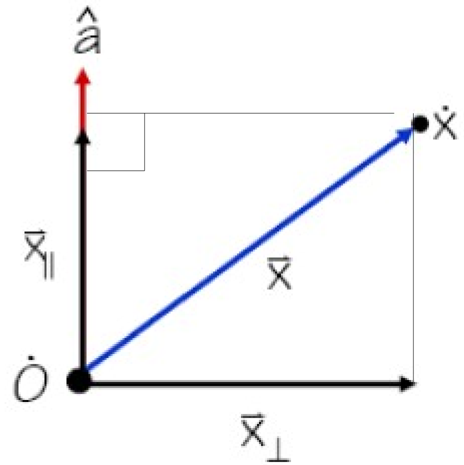
Geometry of a Rotation

- **Natural basis for rotation of a vector about a specified axis:**
  - $\hat{a}$ - rotation axis (normalized)
  - $\hat{a} \times \vec{x}$ - vector perpendicular to
  - $\vec{x}_{\perp}$ - perpendicular component of $\vec{x}$ relative to $\hat{a}$



Geometry of a Rotation

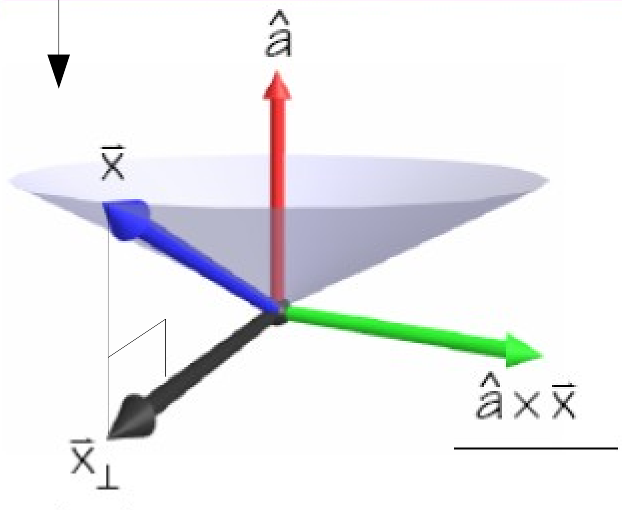


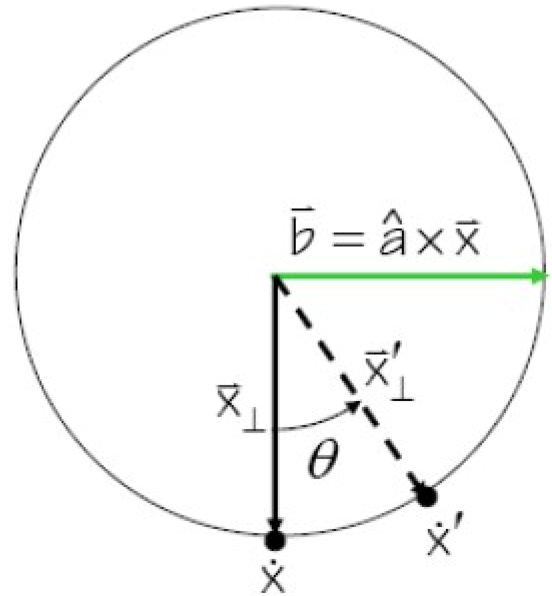


$$\vec{x}_\parallel = \hat{a}(\hat{a} \cdot \vec{x})$$

$$\vec{x}_\perp = \vec{x} - \vec{x}_\parallel$$

Geometry of a Rotation





$$\dot{x}' = \dot{O} + x_{\parallel} + \vec{x}'_{\perp}$$

$$\vec{x}'_{\perp} = \cos\theta\,\vec{x}_{\perp} + \sin\theta\,\vec{b}$$

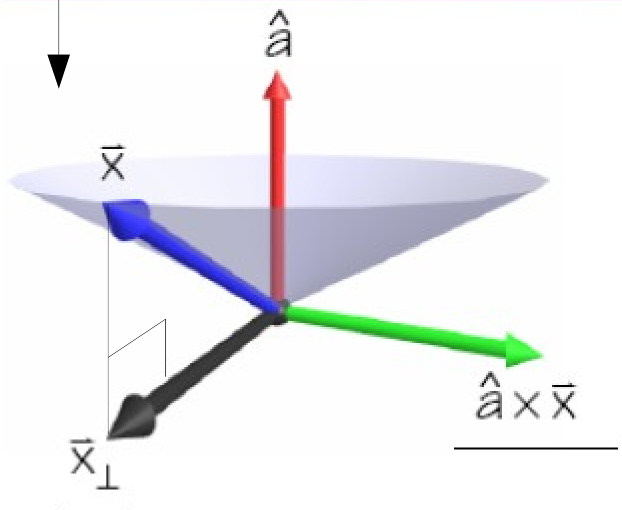
$$\vec{x}_{\parallel} = \hat{a}(\hat{a} \cdot \vec{x})$$

$$\vec{x}_{\perp} = \vec{x} - \vec{x}_{\parallel}$$

$$\dot{x}' = \dot{O} + \cos\theta\,\vec{x} + (1 - \cos\theta)(\hat{a}(\hat{a} \cdot \vec{x})) + \sin\theta(\hat{a} \times \vec{x})$$

Note that $\|\vec{x}_{\perp}\| = \|\hat{a} \times \vec{x}\|$

Geometry of a Rotation



$$\dot{x}' = \dot{O} + x_{\|} + \vec{x}'_{\perp}$$

$$\vec{x}'_{\perp} = \cos\theta\,\vec{x}_{\perp} + \sin\theta\,\vec{b}$$

$$\vec{x}_{\|} = \hat{a}(\hat{a}\cdot\vec{x})$$

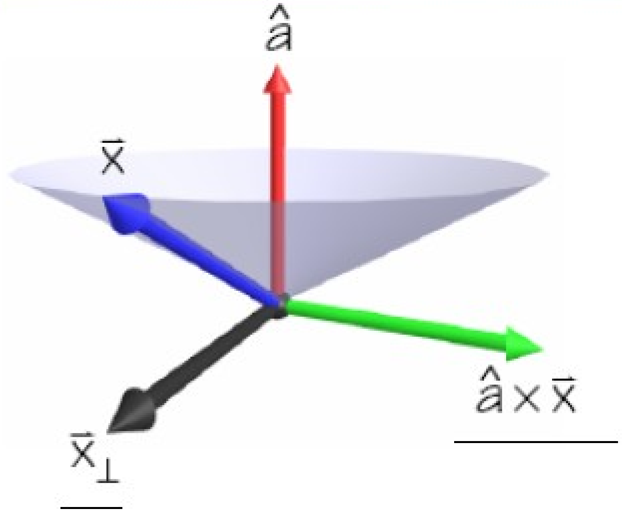
$$\vec{x}_{\perp} = \vec{x} - \vec{x}_{\|}$$

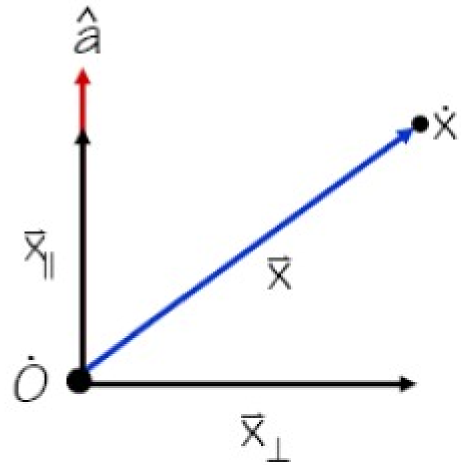
$$\dot{x}' = \dot{O} + \cos\theta\,\vec{x} + (1-\cos\theta)(\hat{a}(\hat{a}\cdot\vec{x})) + \sin\theta(\hat{a}\times\vec{x})$$

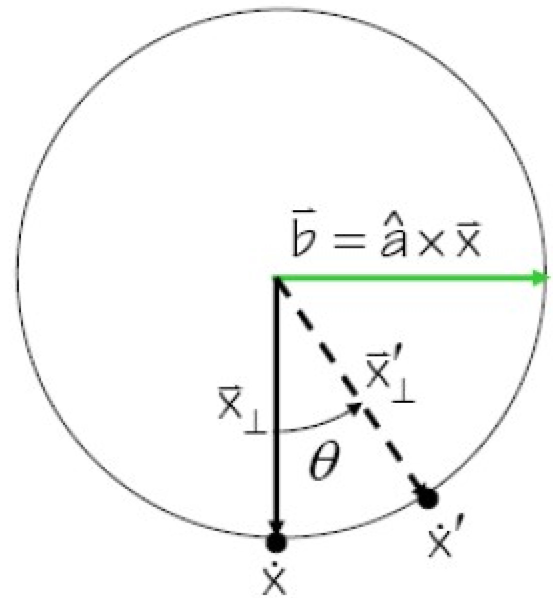
$$\mathbf{c}_{\dot{x}'} = \mathbf{M}\mathbf{c}_{\dot{x}}$$

$$\mathbf{M} = diag(\dot{O}) + \cos\theta\, diag([1 \quad 1 \quad 1 \quad O]^t) + (1-\cos\theta)\mathbf{A}_{\otimes} + \sin\theta\,\mathbf{A}_{\times}$$

Geometry of a Rotation







$$\dot{x}' = \dot{O} + x_{\parallel} + \vec{x}'_{\perp}$$

$$\vec{x}'_{\perp} = \cos\theta\,\vec{x}_{\perp} + \sin\theta\,\vec{b}$$

$$\vec{x}_{\parallel} = \hat{a}(\hat{a}\cdot\vec{x})$$

$$\vec{x}_{\perp} = \vec{x} - \vec{x}_{\parallel}$$

$$\dot{x}' = \dot{O} + \cos\theta\,\vec{x} + (1-\cos\theta)(\hat{a}(\hat{a}\cdot\vec{x})) + \sin\theta(\hat{a}\times\vec{x})$$

$$\mathbf{c}_{\dot{x}'} = \mathbf{M}\mathbf{c}_{\dot{x}}$$

$$\mathbf{M} = diag(\dot{O}) + \cos\theta\, diag([1 \quad 1 \quad 1 \quad 0]^{t}) + (1-\cos\theta)\mathbf{A}_{\otimes} + \sin\theta\,\mathbf{A}_{\times}$$

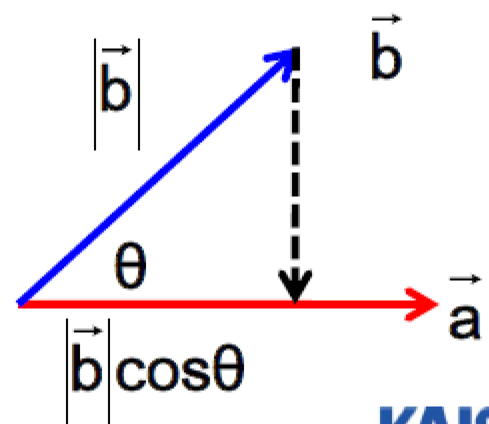
Vector Algebra

- We already saw vector addition and multiplications by a scalar
- Will study three kinds of vector multiplications
  - Dot product ($\cdot$) - returns a scalar
  - Cross product ($\times$) - returns a vector
  - Tensor product ($\otimes$) - returns a matrix

Dot Product (·)

$$\vec{a}\cdot\vec{b} \equiv \vec{a}^T\vec{b} = \begin{bmatrix} a_x & a_y & a_z & 0 \end{bmatrix} \begin{bmatrix} b_x \\ b_y \\ b_z \\ 0 \end{bmatrix} = s, \qquad \vec{a}\cdot\dot{b} \equiv \vec{a}^T\dot{b} = \begin{bmatrix} a_x & a_y & a_z & 0 \end{bmatrix} \begin{bmatrix} b_x \\ b_y \\ b_z \\ 1 \end{bmatrix} = s$$

- **Returns a scalar s**
- **Geometric interpretations *s*:**
  - $\vec{a}\cdot\vec{b} = |a||b|\cos\theta$
  - **Length of** $\vec{b}$ **projected onto and** $\vec{a}$ **or vice versa**
  - **Distance of** $\dot{b}$ **from the origin in the direction of** $\vec{a}$

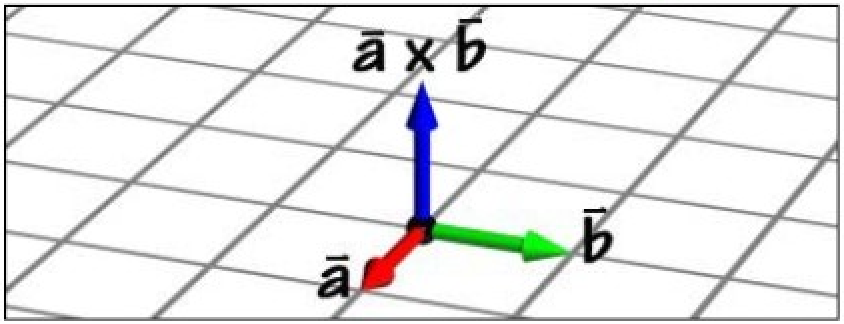


Cross Product ($\times$)

$$\vec{a} \times \vec{b} \equiv \begin{bmatrix} 0 & -a_z & a_y & 0 \\ a_z & 0 & -a_x & 0 \\ -a_y & a_x & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} b_x \\ b_y \\ b_z \\ 0 \end{bmatrix} = \vec{c} \qquad \begin{aligned} \vec{a} \cdot \vec{c} &= 0 \\ \vec{b} \cdot \vec{c} &= 0 \end{aligned}$$

$$\vec{c} = \begin{bmatrix} a_y b_z - a_z b_y & a_z b_x - a_x b_z & a_x b_y - a_y b_x \end{bmatrix}$$

- **Return a vector $\vec{c}$ that is perpendicular to both $\vec{a}$ and $\vec{b}$, oriented according to the right-hand rule**
- **The matrix is called the skew-symmetric matrix of $\vec{a}$**



Cross Product (×)

- **A mnemonic device for remembering the cross-product**

$$\vec{a} \times \vec{b} \equiv det \begin{bmatrix} \vec{i} & \vec{j} & \vec{k} \\ a_x & a_y & a_z \\ b_x & b_y & b_z \end{bmatrix}$$

$$= (a_y b_z - a_z b_y)\vec{i} + (a_z b_x - a_x b_z)\vec{j} + (a_x b_y - a_y b_x)\vec{k}$$

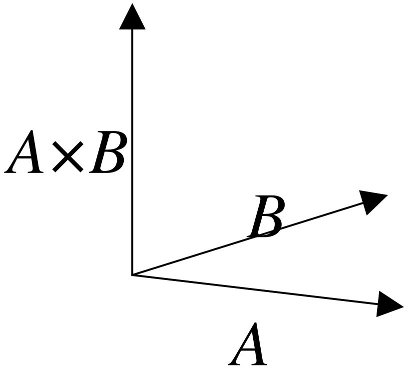
$$\vec{i} = [1 \quad 0 \quad 0]$$

$$\vec{j} = [0 \quad 1 \quad 0]$$

$$\vec{k} = [0 \quad 0 \quad 1]$$

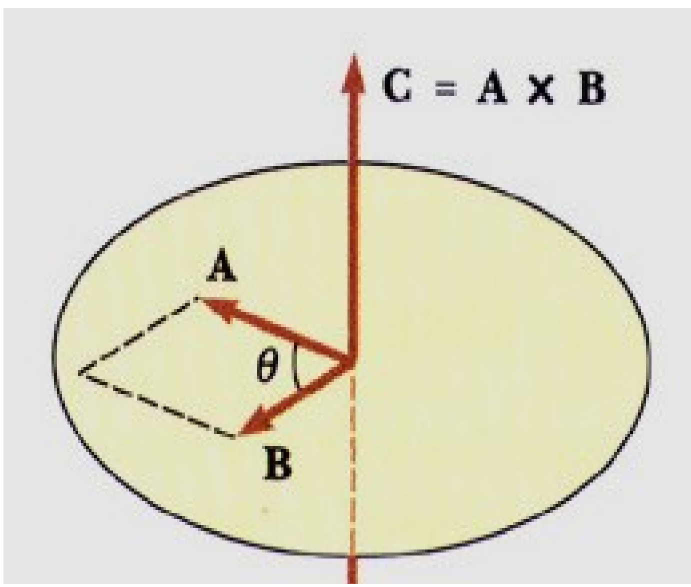
Vectors: Cross Product

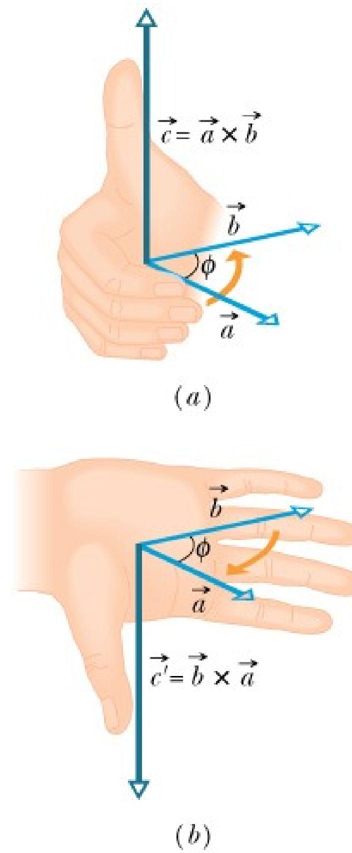
- The cross product of vectors A and B is a vector C which is perpendicular to A and B
- The magnitude of C is proportional to the sin of the angle between A and B
- The direction of C follows the **right hand rule** if we are working in a right-handed coordinate system



$$\|A \times B\| = \|A\| \, \|B\| \sin(\theta)$$

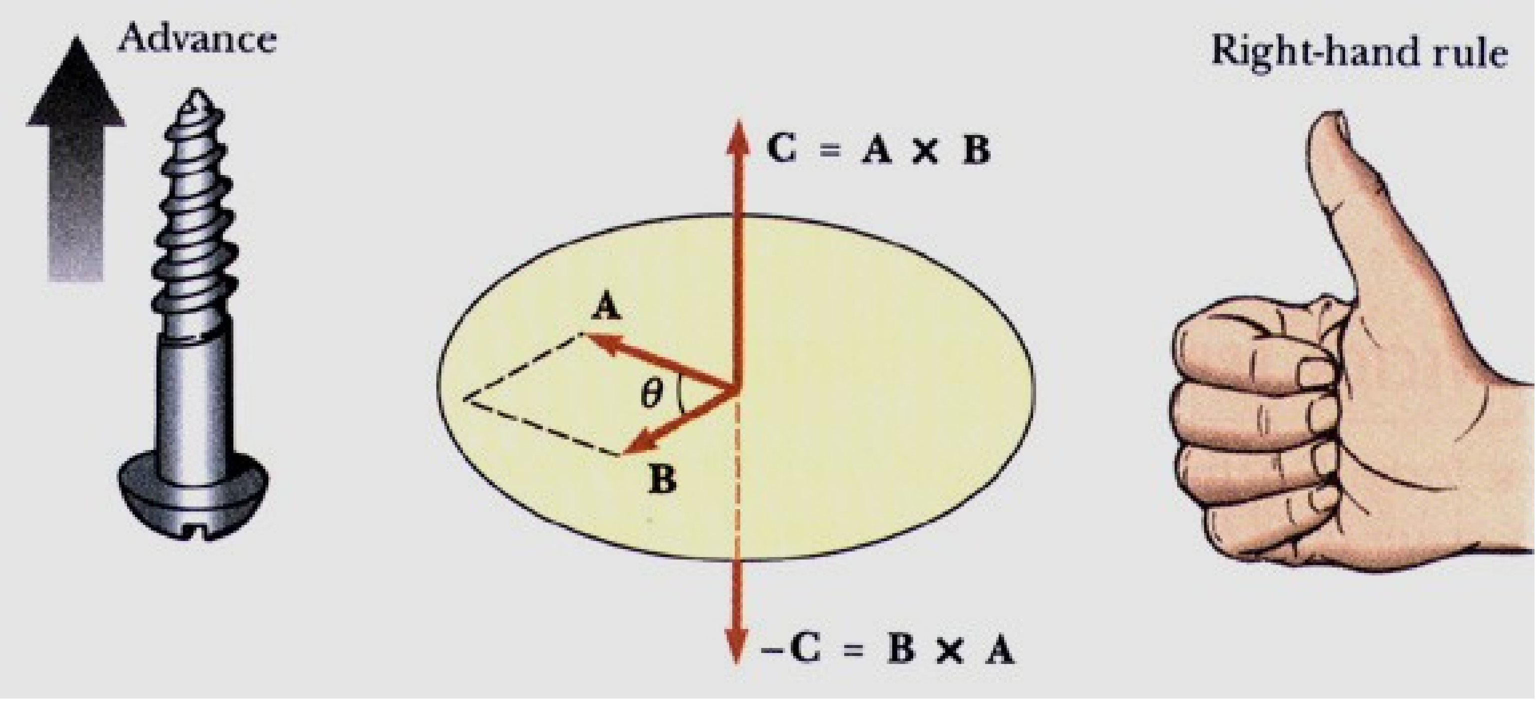
MAGNITUDE OF THE CROSS PRODUCT





DIRECTION OF THE CROSS PRODUCT

- The right hand rule determines the direction of the cross product



Tensor Product ($\otimes$)

$$\vec{a}\otimes\vec{b} \equiv \vec{a}\vec{b}^t = \begin{bmatrix} a_x \\ a_y \\ a_z \\ 0 \end{bmatrix}\begin{bmatrix} b_x & b_y & b_z & 0 \end{bmatrix} = \begin{bmatrix} a_x b_x & a_x b_y & a_x b_z & 0 \\ a_y b_x & a_y b_y & a_y b_z & 0 \\ a_z b_x & a_z b_y & a_z b_z & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$(\vec{a}\otimes\vec{b})\vec{c} = \begin{bmatrix} (b_x c_x + b_y c_y + b_z c_z)a_x \\ (b_x c_x + b_y c_y + b_z c_z)a_y \\ (b_x c_x + b_y c_y + b_z c_z)a_z \end{bmatrix} = \vec{a}(\vec{b}\cdot\vec{c})$$

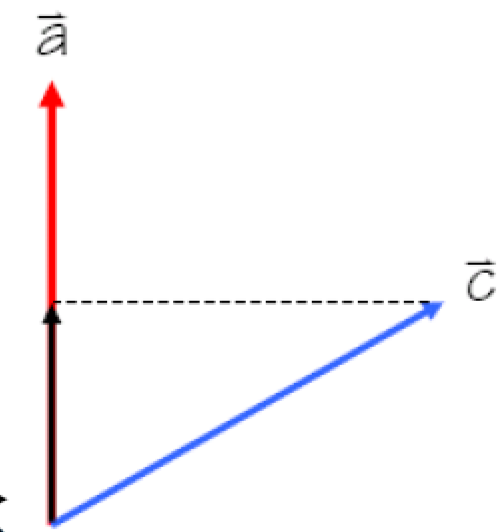
- **Creates a matrix that when applied to a vector $\vec{c}$ return $\vec{a}$ scaled by the project of $\vec{c}$ onto $\vec{b}$**

KAIST

Tensor Product ($\otimes$)

- **Useful when** $\vec{b} = \vec{a}$
- **The matrix** $\vec{a} \otimes \vec{a}$ **is called the symmetric matrix of** $\vec{a}$
  - **We shall denote this** $A_{\otimes}$

$$\boldsymbol{A}_{\otimes} = \vec{a} \otimes \vec{a} = \begin{bmatrix} a_x a_x & a_x a_y & a_x a_z & 0 \\ a_y a_x & a_y a_y & a_y a_z & 0 \\ a_z a_x & a_z a_y & a_z a_z & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$



$$\boldsymbol{A}_{\otimes}\vec{c}$$
$$= (\vec{a} \otimes \vec{a})\vec{c}$$
$$= \vec{a}(\vec{a} \cdot \vec{c})$$

KAIST

Sanity Check

- **Consider a rotation by about the x-axis**

$$Rotate(\begin{bmatrix}1\\0\\0\\0\end{bmatrix},\theta)=\begin{bmatrix}0&0&0&0\\0&0&0&0\\0&0&0&0\\0&0&0&1\end{bmatrix}+\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&0&0\end{bmatrix}\cos\theta+\begin{bmatrix}1&0&0&0\\0&0&0&0\\0&0&0&0\\0&0&0&0\end{bmatrix}(1-\cos\theta)+\begin{bmatrix}0&0&0&0\\0&0&-1&0\\0&1&0&0\\0&0&0&0\end{bmatrix}\sin\theta$$

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

- **You can check it in any computer graphics book, but you don't need to memorize it**