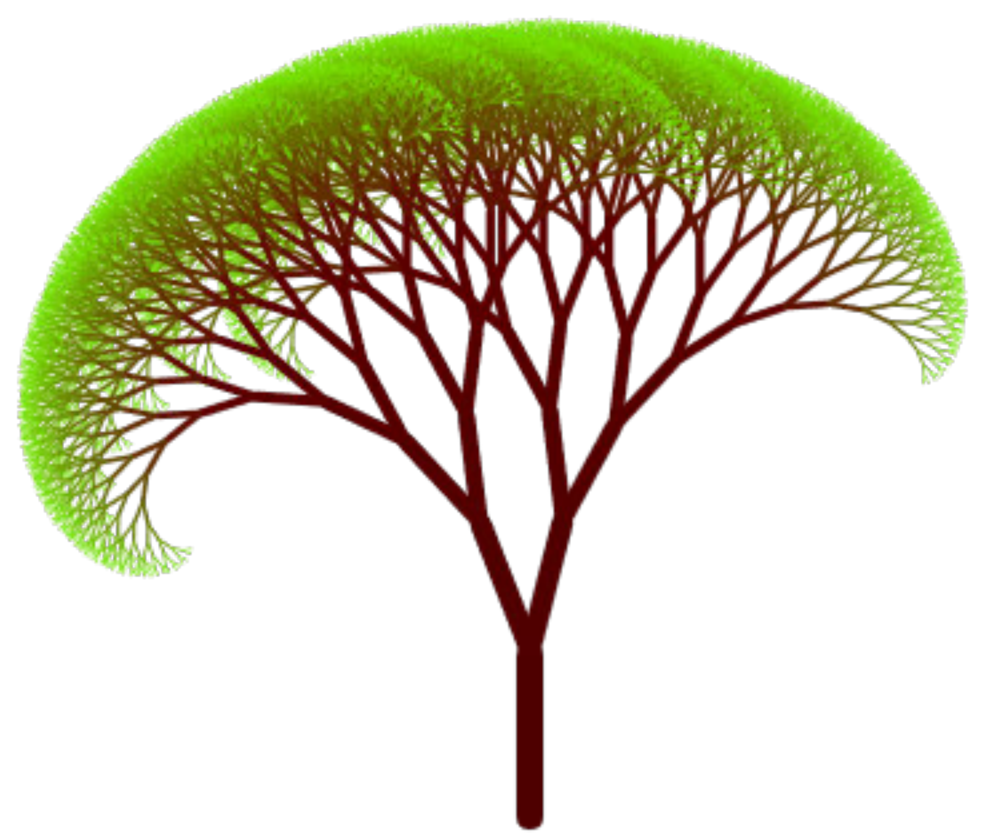
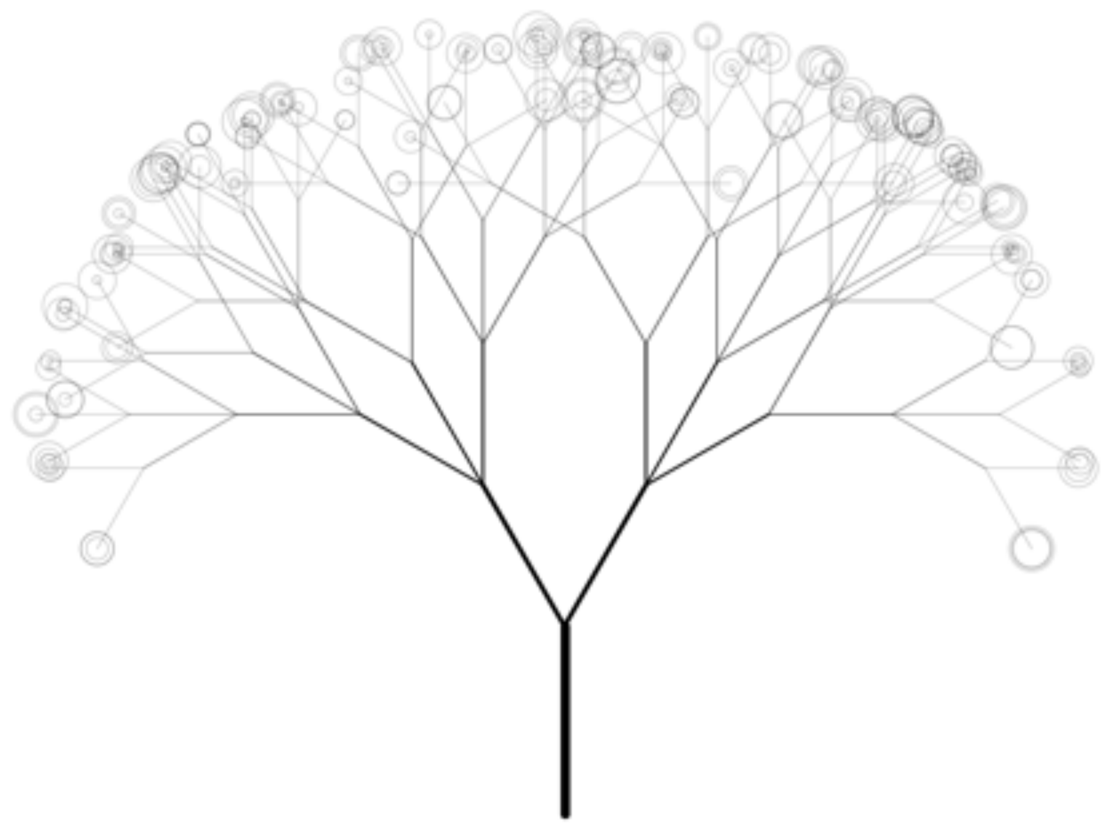
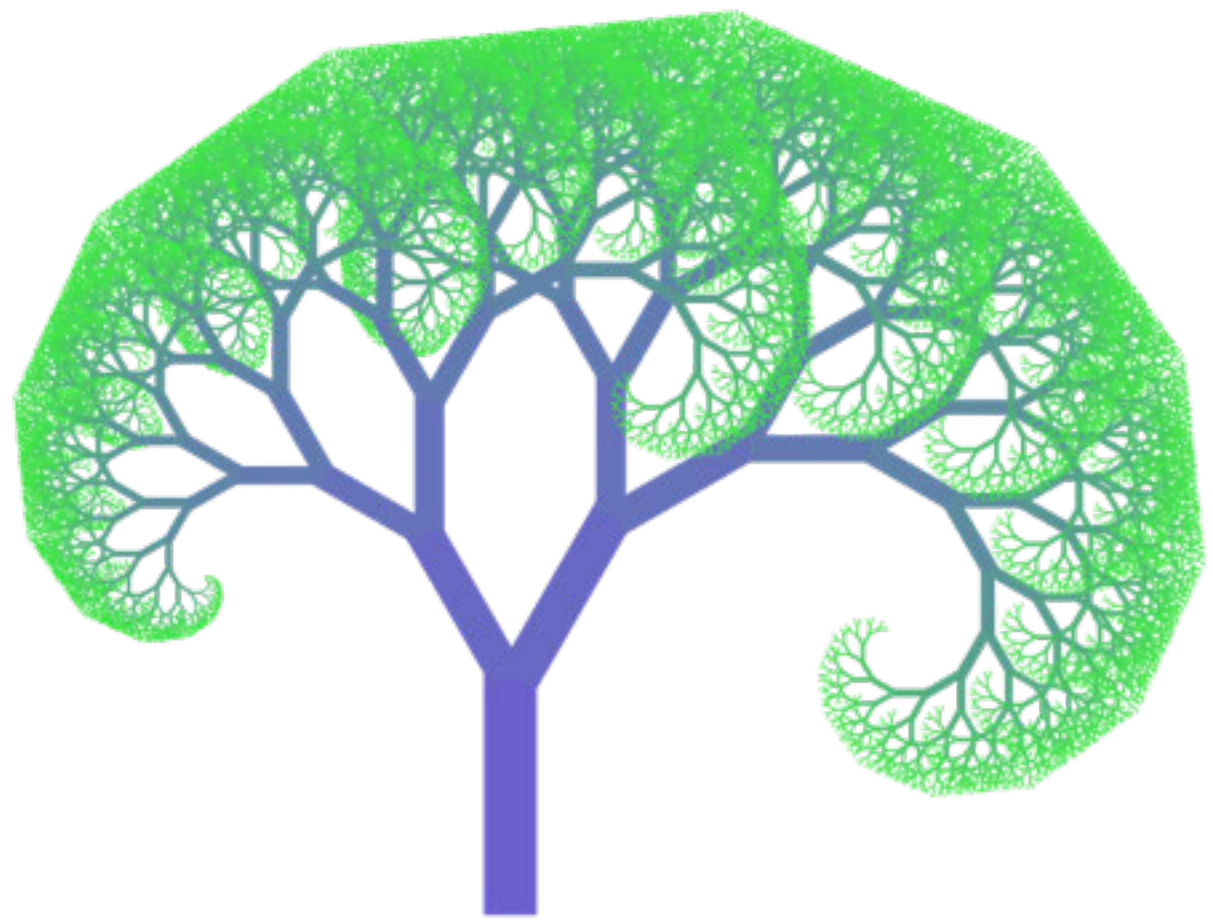
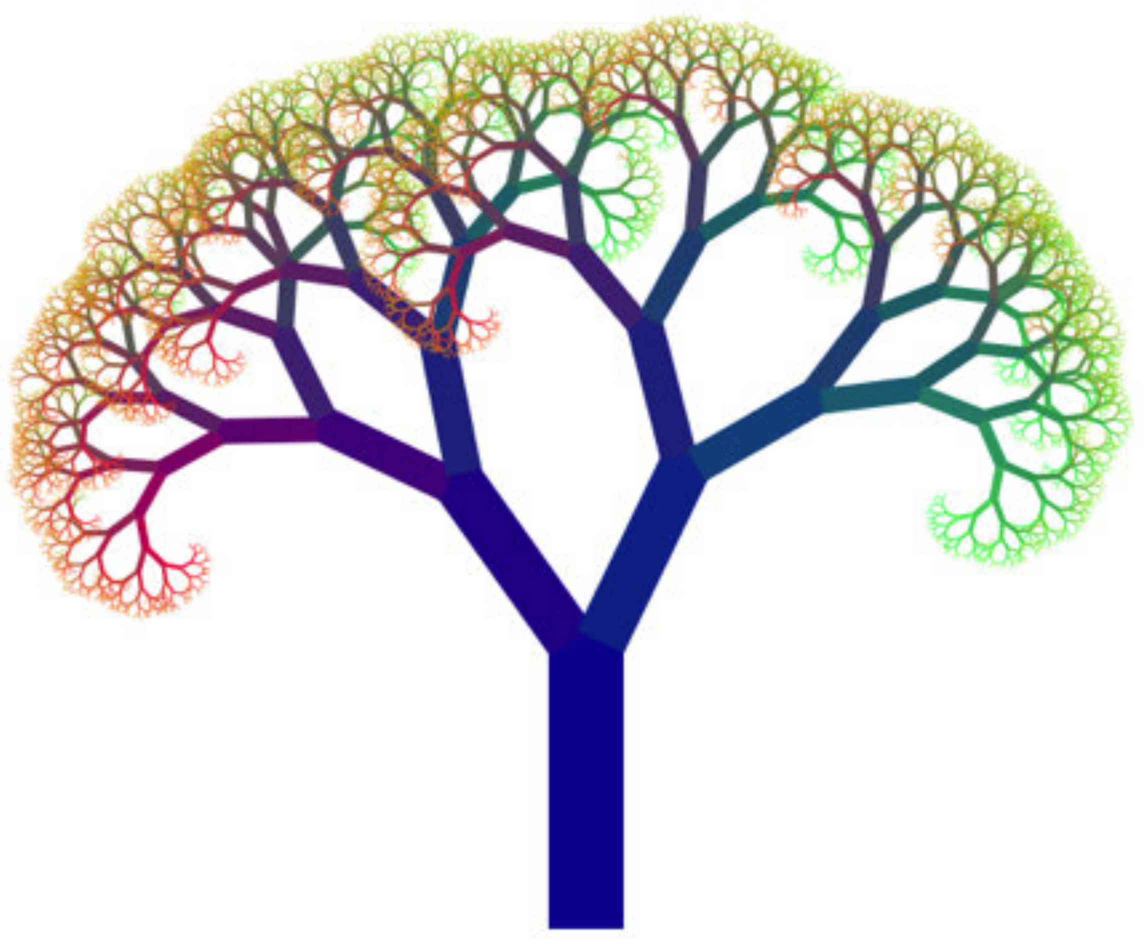
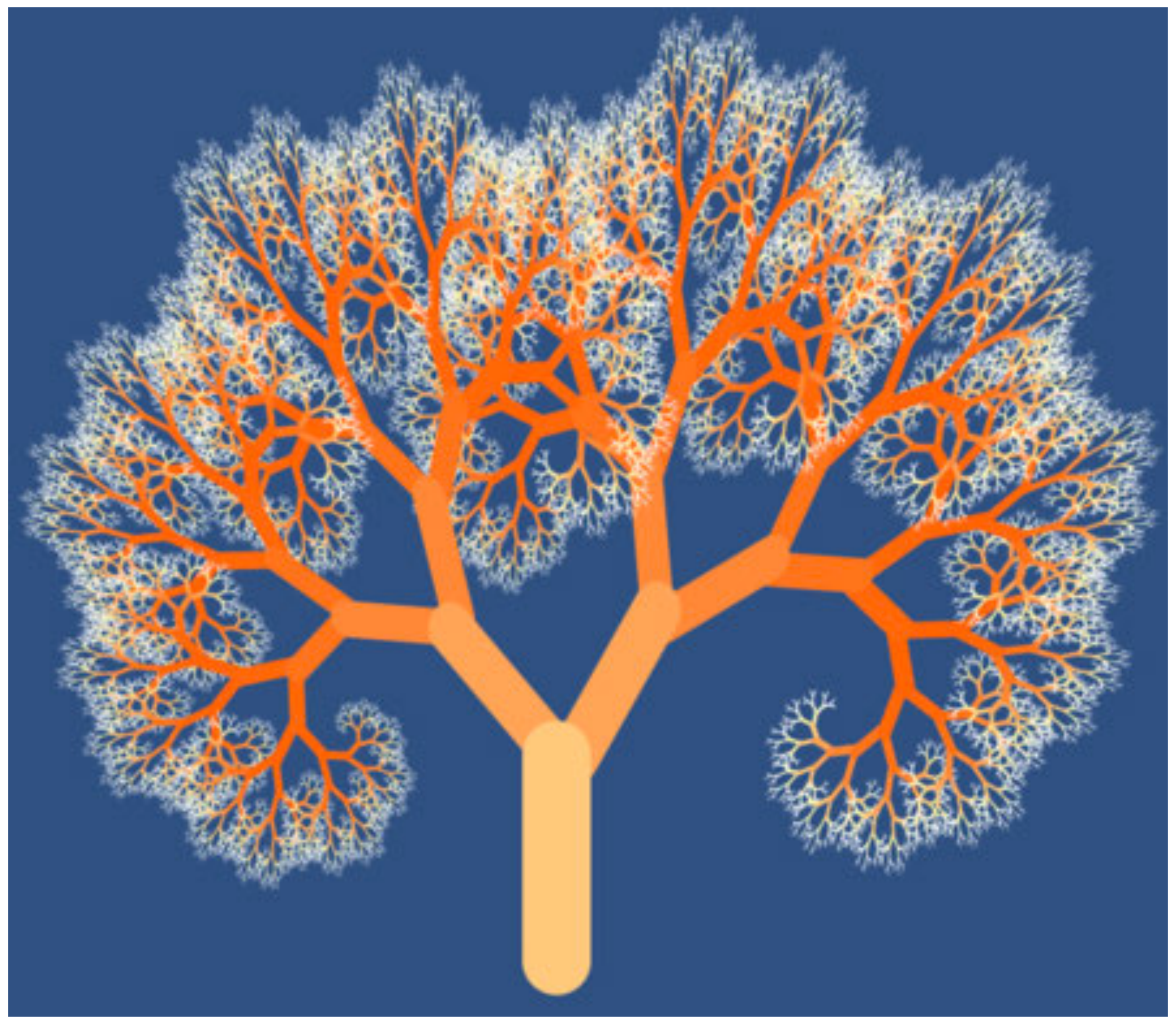
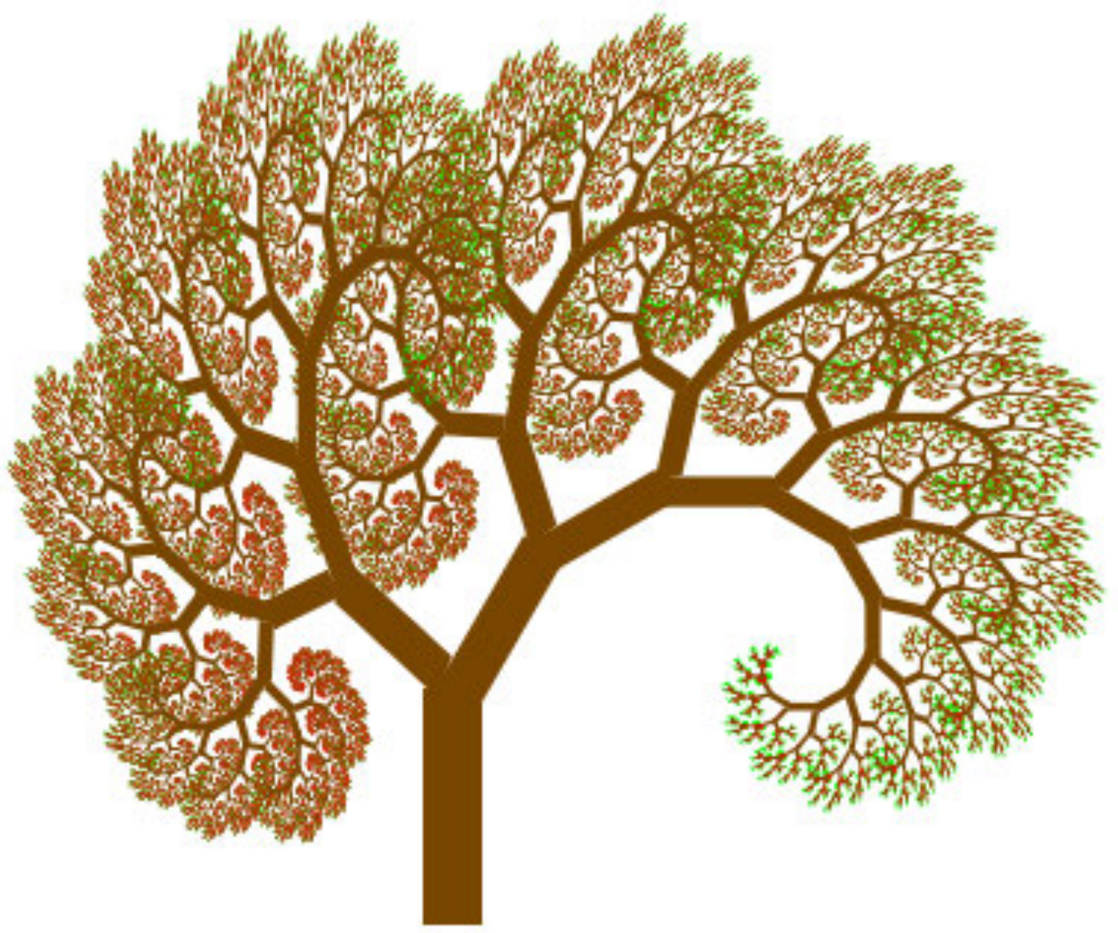
3D Transformations and Complex Representations

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

Quiz 4: Trees and Transformations

Student solutions (beautiful!):

Moving to 3D (and 3D-H)

Represent 3D transforms as 3x3 matrices and 3D-H transforms as 4x4 matrices

Scale:

3D **3D-H**

$$\mathbf{S_s} = \begin{bmatrix} \mathbf{S}_x & 0 & 0 \\ 0 & \mathbf{S}_y & 0 \\ 0 & 0 & \mathbf{S}_z \end{bmatrix} \quad \mathbf{S_s} = \begin{bmatrix} \mathbf{S}_x & 0 & 0 & 0 \\ 0 & \mathbf{S}_y & 0 & 0 \\ 0 & 0 & \mathbf{S}_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Shear (in x, based on y,z position):

$$\mathbf{H}_{x,\mathbf{d}} = \begin{bmatrix} 1 & \mathbf{d}_y & \mathbf{d}_z \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \mathbf{H}_{x,\mathbf{d}} = \begin{bmatrix} 1 & \mathbf{d}_y & \mathbf{d}_z & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Translate:

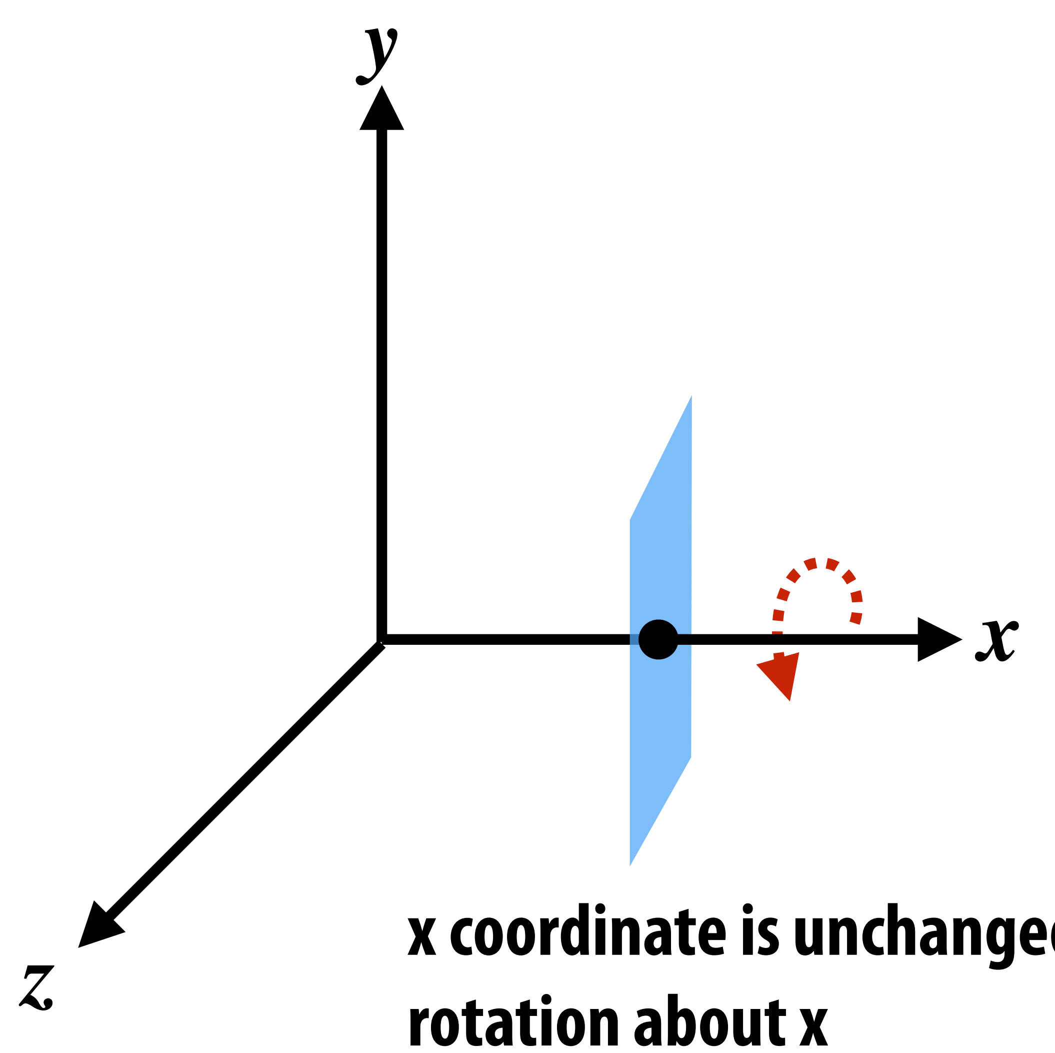
3D-H

$$\mathbf{T_b} = \begin{bmatrix} 1 & 0 & 0 & \mathbf{b}_x \\ 0 & 1 & 0 & \mathbf{b}_y \\ 0 & 0 & 1 & \mathbf{b}_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Rotations in 3D

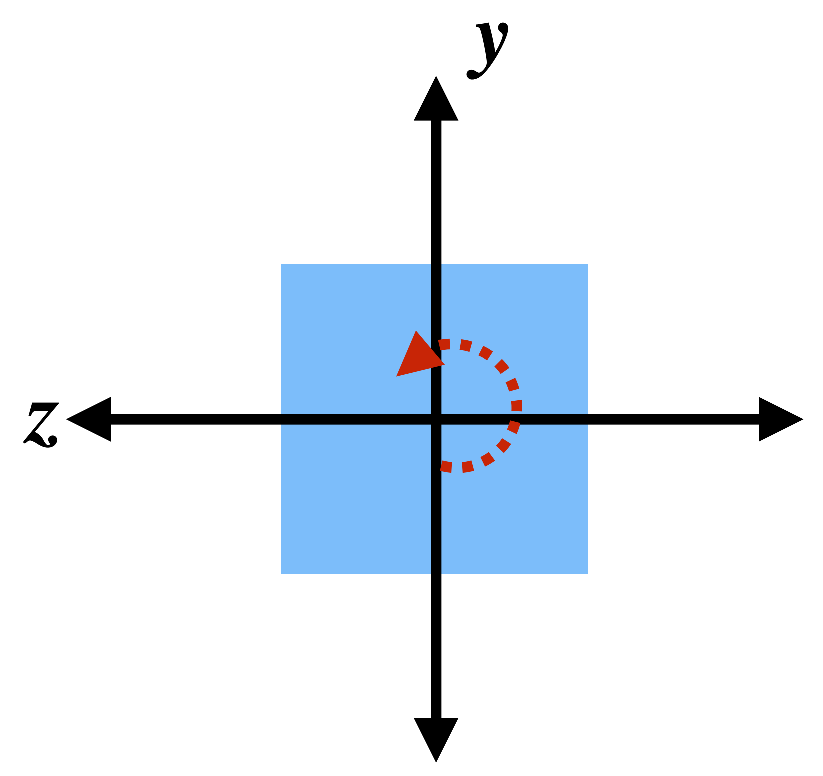
Rotation about x axis:

$$\mathbf{R}_{x,\theta} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{bmatrix}$$



x coordinate is unchanged by rotation about x

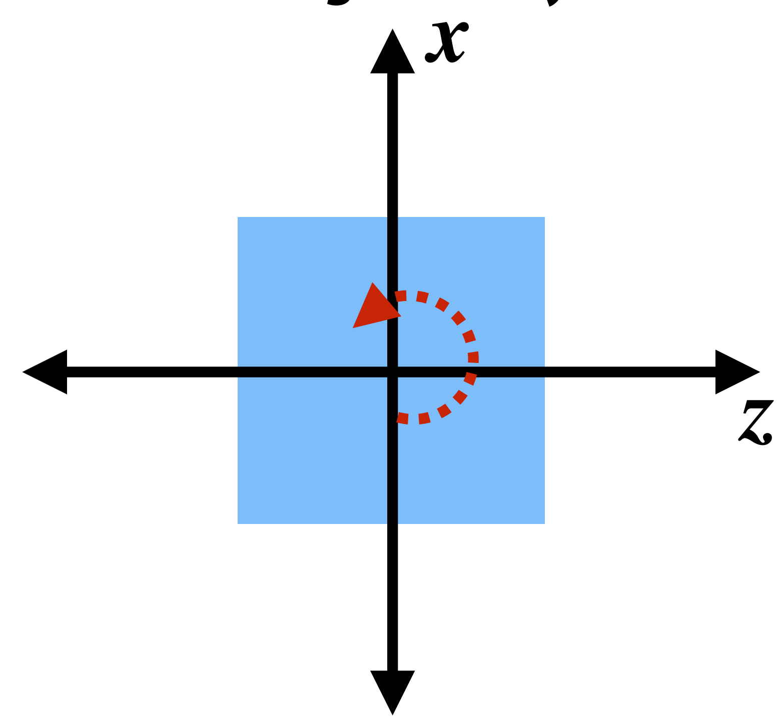
View looking down -x axis:



Rotation about y axis:

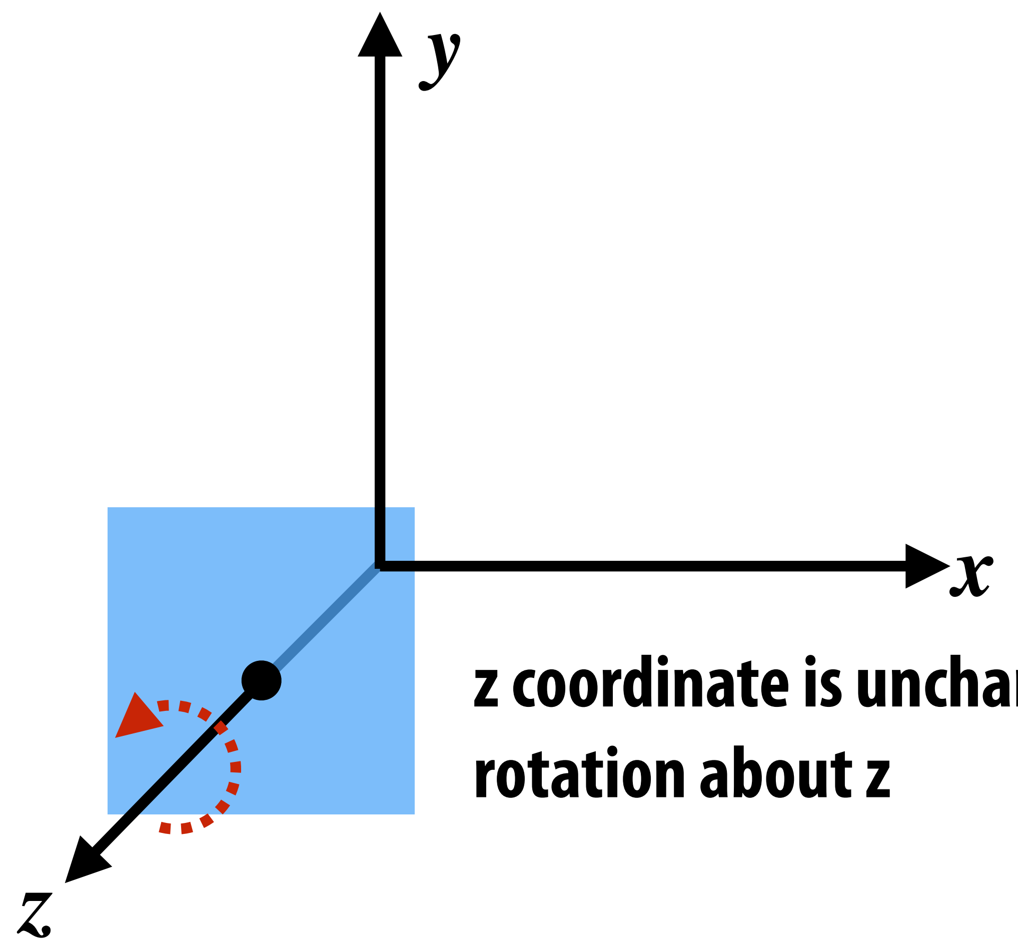
$$\mathbf{R}_{y,\theta} = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix}$$

View looking down -y axis:



Rotation about z axis:

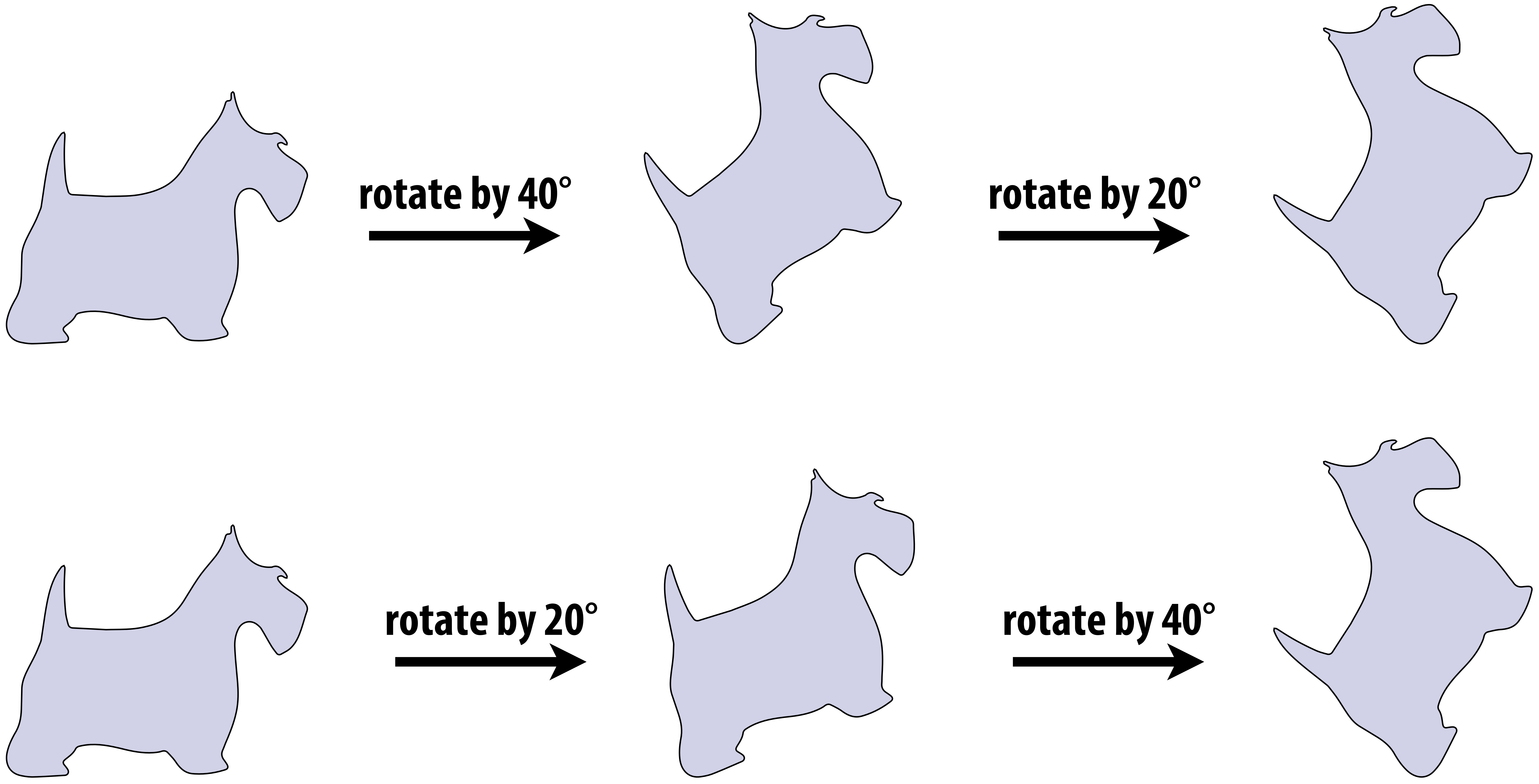
$$\mathbf{R}_{z,\theta} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$



z coordinate is unchanged by rotation about z

Commutativity of Rotations—2D

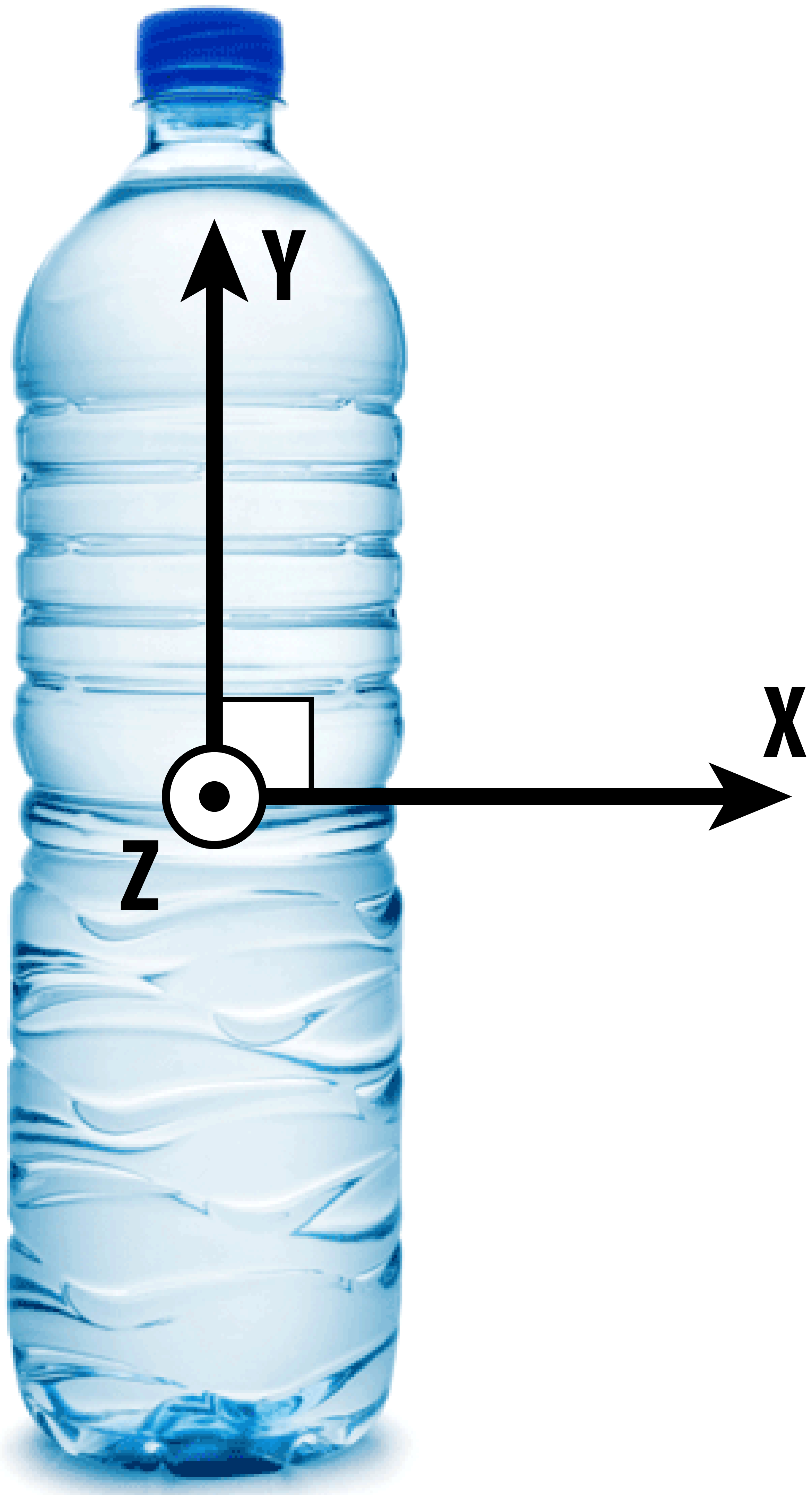
- In 2D, order of rotations doesn't matter:



Why not?

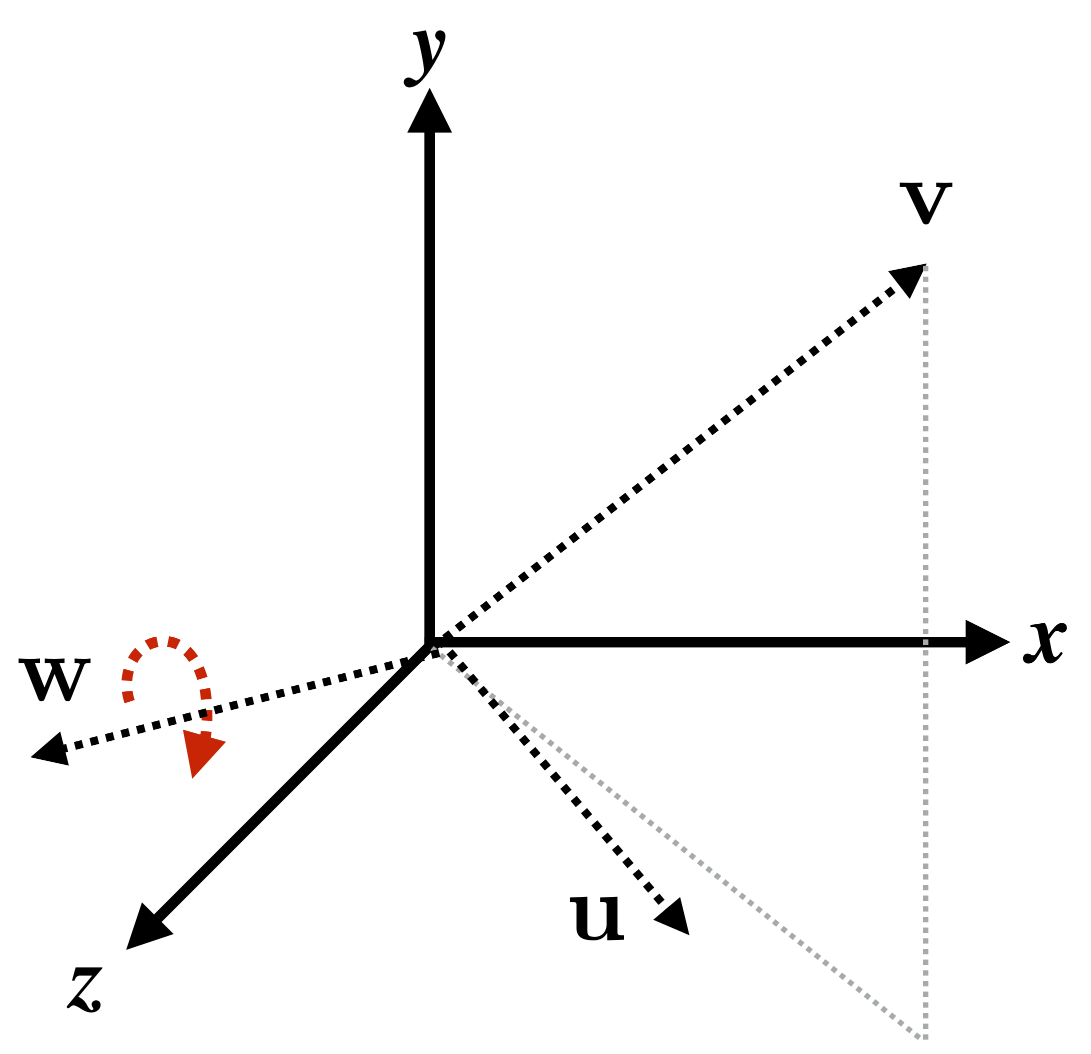
Commutativity of Rotations—3D

- What about in 3D?
- IN-CLASS ACTIVITY:
  - Rotate 90° around Y, then 90° around Z, then 90° around X
  - Rotate 90° around Z, then 90° around Y, then 90° around X
  - (Was there any difference?)



CONCLUSION: bad things can happen if we're not careful about the order in which we apply rotations!

Rotation about an arbitrary axis



To rotate by θ about $\mathbf{w}$:

1. Form orthonormal basis around $\mathbf{w}$ (see $\mathbf{u}$ and $\mathbf{v}$ in figure)
2. Rotate to map $\mathbf{w}$ to [0 0 1] (change in coordinate space)

$$\mathbf{R}_{uvw} = \begin{bmatrix} \mathbf{u}_x & \mathbf{u}_y & \mathbf{u}_z \\ \mathbf{v}_x & \mathbf{v}_y & \mathbf{v}_z \\ \mathbf{w}_x & \mathbf{w}_y & \mathbf{w}_z \end{bmatrix}$$

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

$$\mathbf{R}_{uvw}\mathbf{v} = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix}$$

$$\mathbf{R}_{uvw}\mathbf{w} = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}$$

3. Perform rotation about z: $\mathbf{R}_{z,\theta}$
4. Rotate back to original coordinate space: $\mathbf{R}_{\mathbf{uvw}}^{\mathbf{T}}$

$$\mathbf{R}_{uvw}^{-1} = \mathbf{R}_{uvw}^{T} = \begin{bmatrix} \mathbf{u}_x & \mathbf{v}_x & \mathbf{w}_x \\ \mathbf{u}_y & \mathbf{v}_y & \mathbf{w}_y \\ \mathbf{u}_z & \mathbf{v}_x & \mathbf{w}_z \end{bmatrix}$$

$$\mathbf{R}_{\mathbf{w},\theta} = \mathbf{R}_{\mathbf{uvw}}^{\mathbf{T}}\mathbf{R}_{z,\theta}\mathbf{R}_{\mathbf{uvw}}$$

Rotation from Axis/Angle

- **Alternatively, there is a general expression for a matrix that performs a rotation around a given axis u by a given angle θ:**

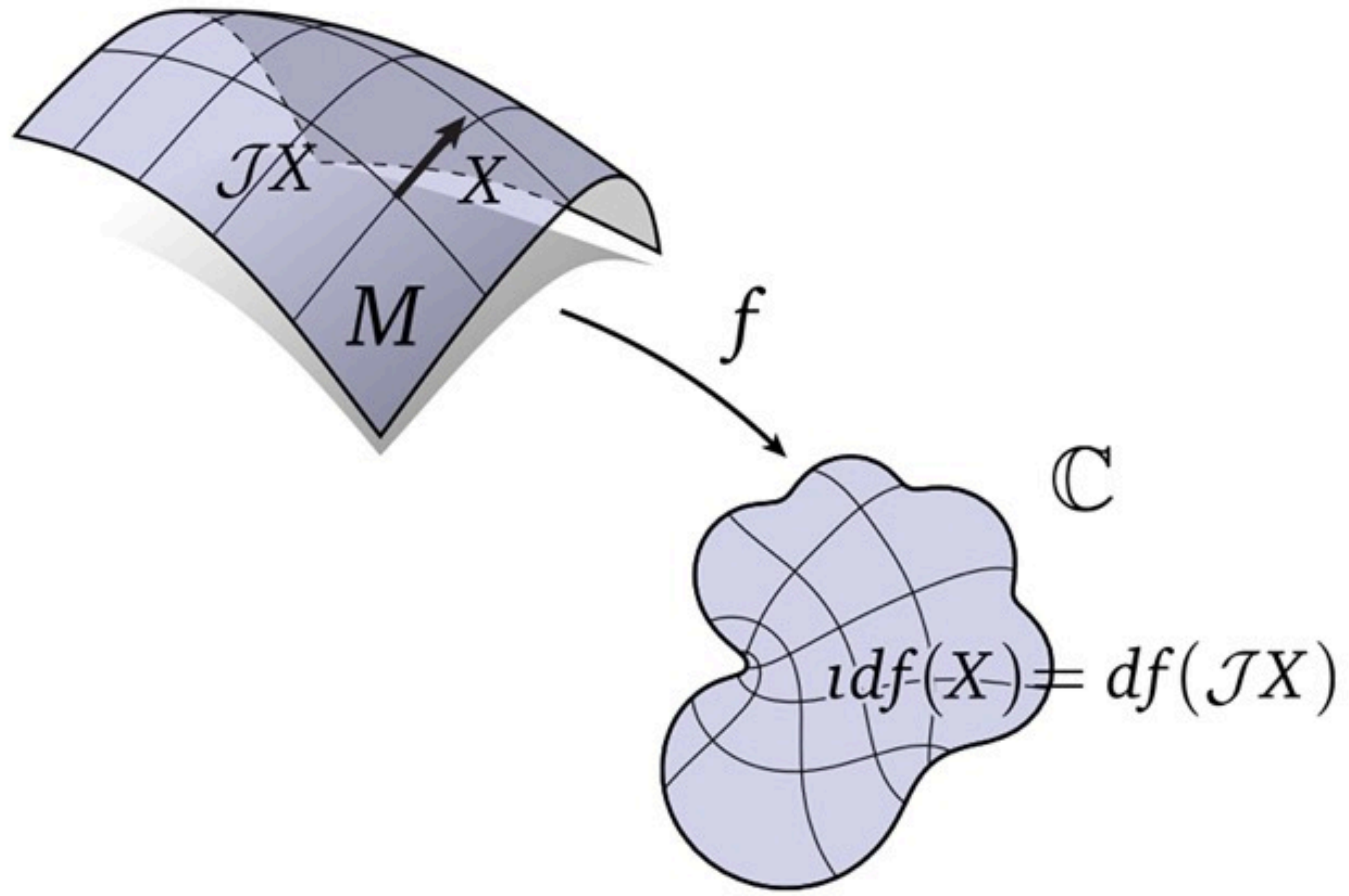
$$\begin{bmatrix} \cos\theta + u_x^2\left(1-\cos\theta\right) & u_x u_y\left(1-\cos\theta\right) - u_z\sin\theta & u_x u_z\left(1-\cos\theta\right) + u_y\sin\theta \\ u_y u_x\left(1-\cos\theta\right) + u_z\sin\theta & \cos\theta + u_y^2\left(1-\cos\theta\right) & u_y u_z\left(1-\cos\theta\right) - u_x\sin\theta \\ u_z u_x\left(1-\cos\theta\right) - u_y\sin\theta & u_z u_y\left(1-\cos\theta\right) + u_x\sin\theta & \cos\theta + u_z^2\left(1-\cos\theta\right) \end{bmatrix}$$

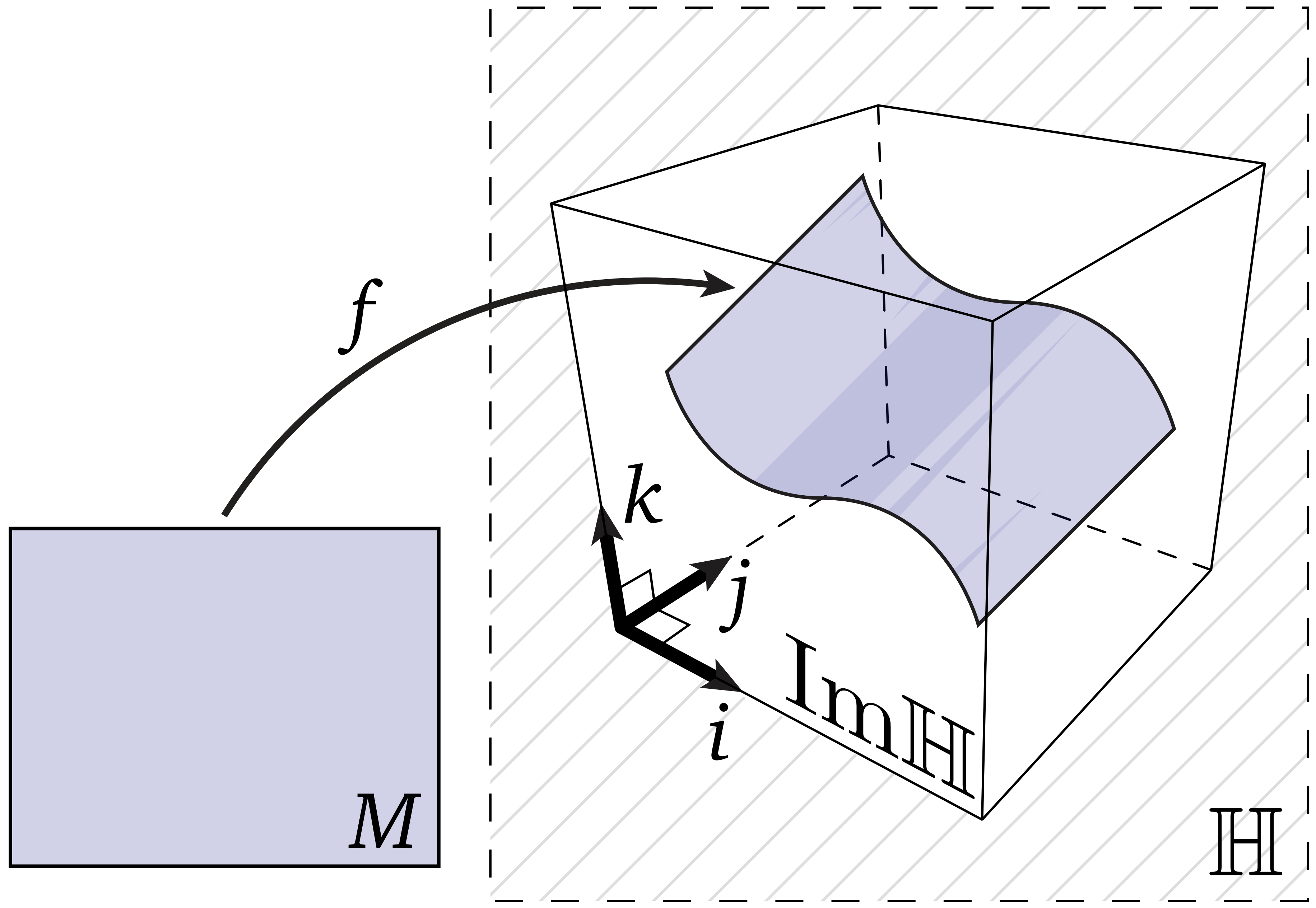
Just memorize this matrix! :-)

…we'll see a different way, later on.

Complex Analysis—Motivation

- Natural way to encode geometric transformations in 2D, 3D
- Simplifies notation / thinking / debugging
- *Moderate* reduction in computational cost/bandwidth/storage
- Fluency with complex analysis can lead into deeper/novel solutions to problems…





DON'T: Think of these numbers as "complex."

DO: Imagine we're simply defining additional operations (like dot and cross).

*A bit of an oversimplification, but go with it for now!

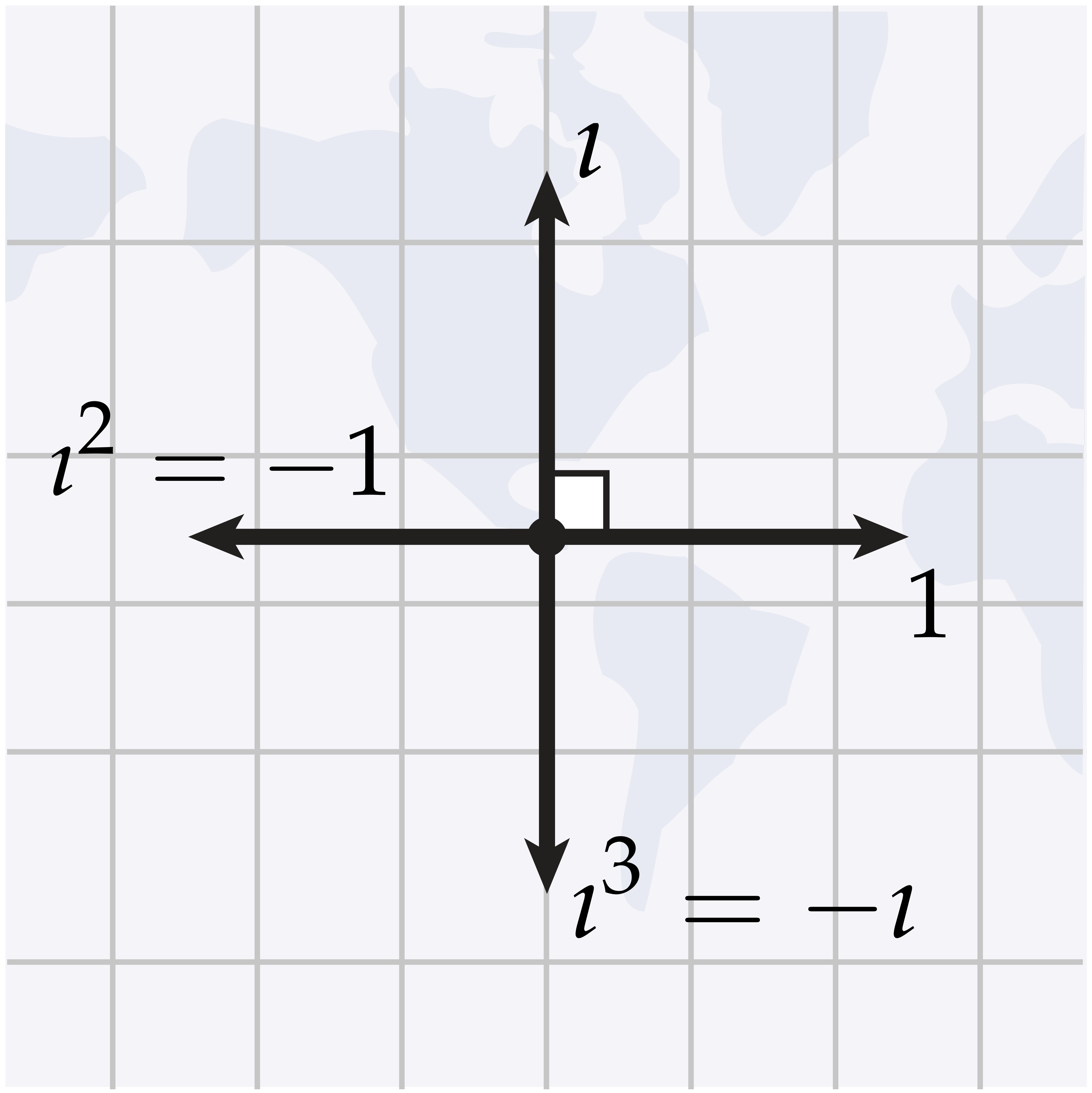
Imaginary Unit

$$\cancel{\imath := \sqrt{-1}}$$

nonsense!

More importantly: obscures geometric meaning.

Imaginary Unit—Geometric Description



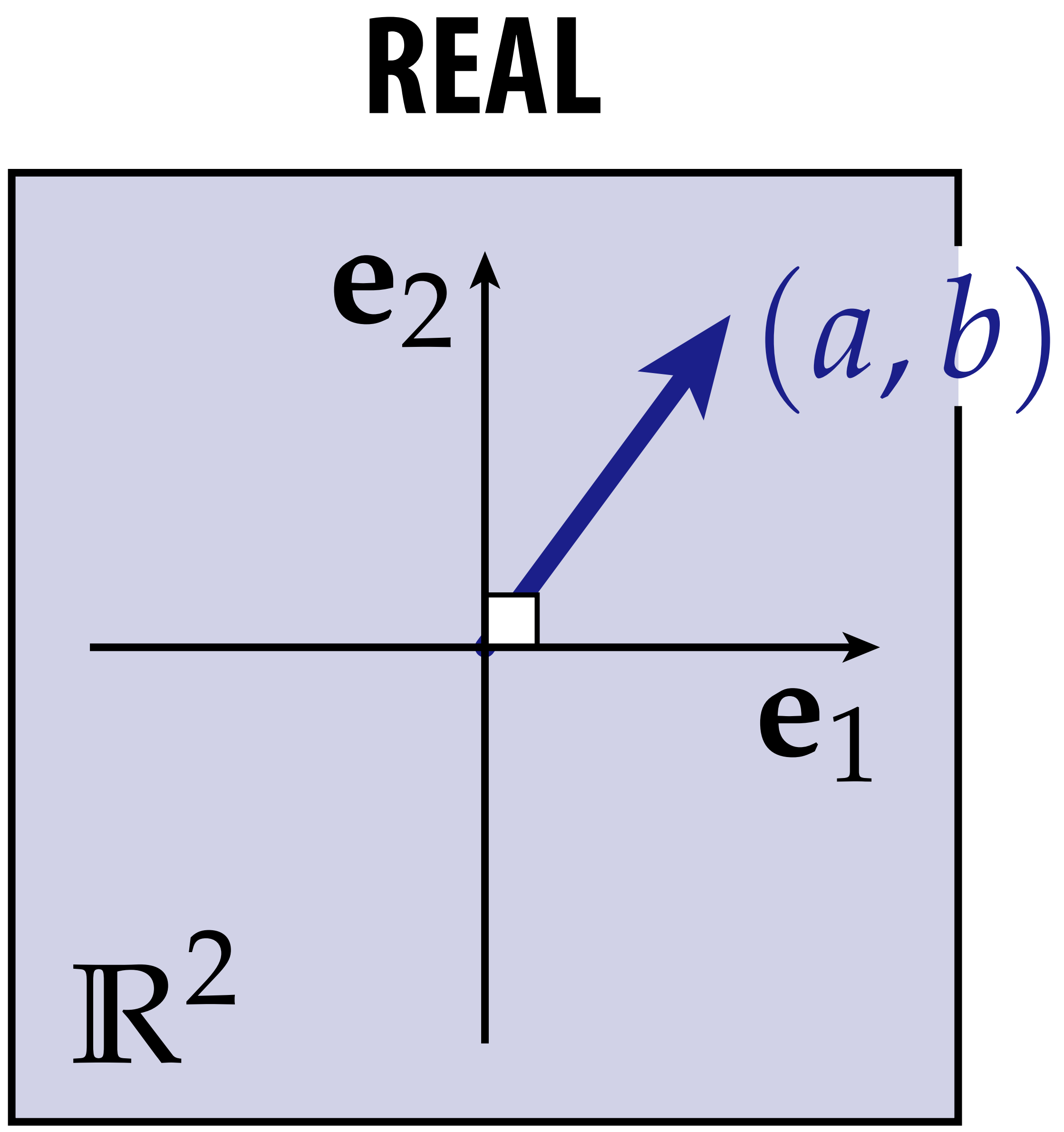
"iota*"

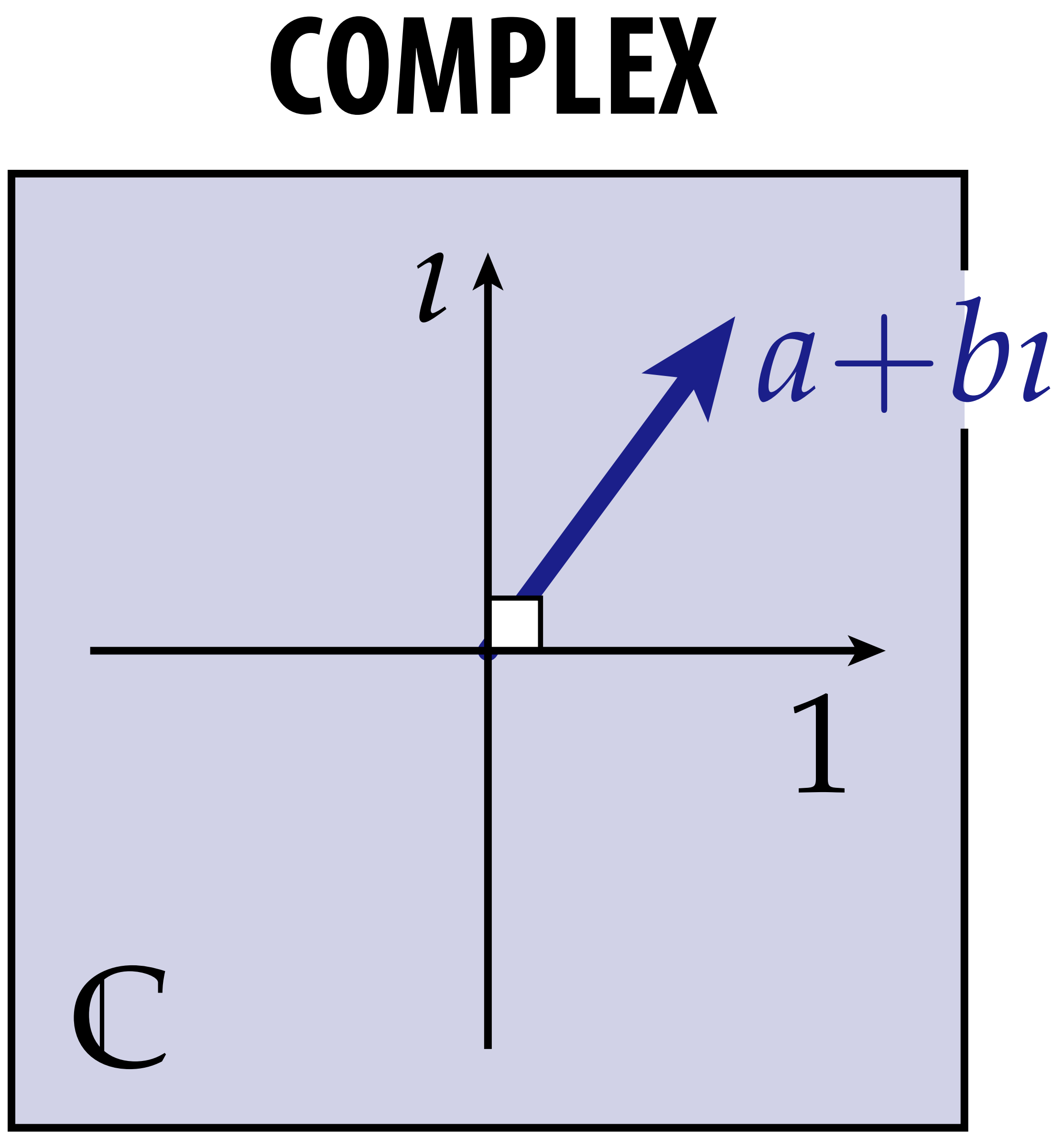
Symbol ι denotes quarter-turn in the counter-clockwise direction.

*Use ι instead of i to avoid confusion w/ indices i. (\imath in LaTeX)

Complex Numbers

- Complex numbers are then just 2-vectors
- Instead of e_1,e_1, use "1" and "$\imath$" to denote the two bases
- Otherwise, behaves exactly like a real 2-dimensional space

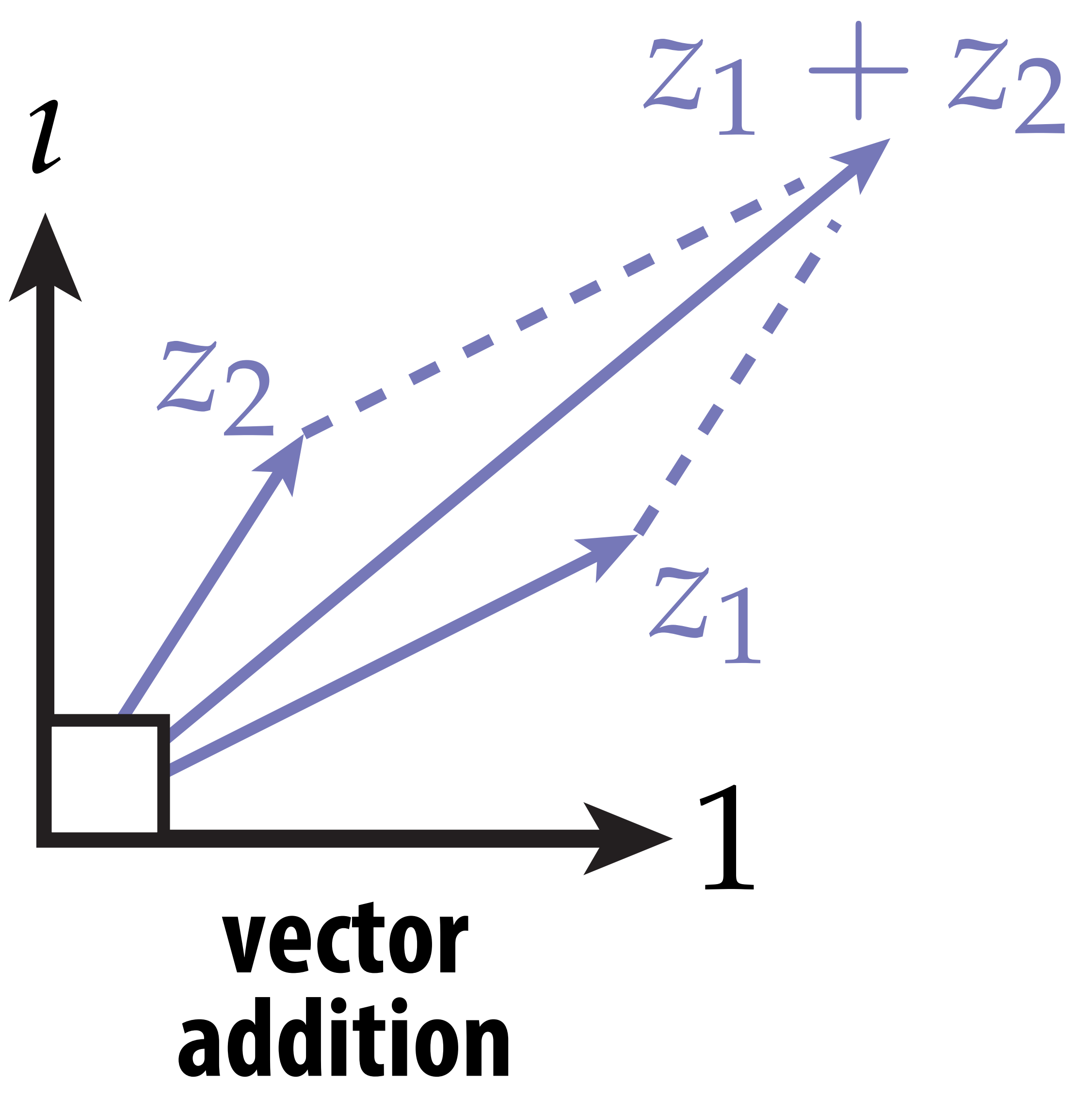




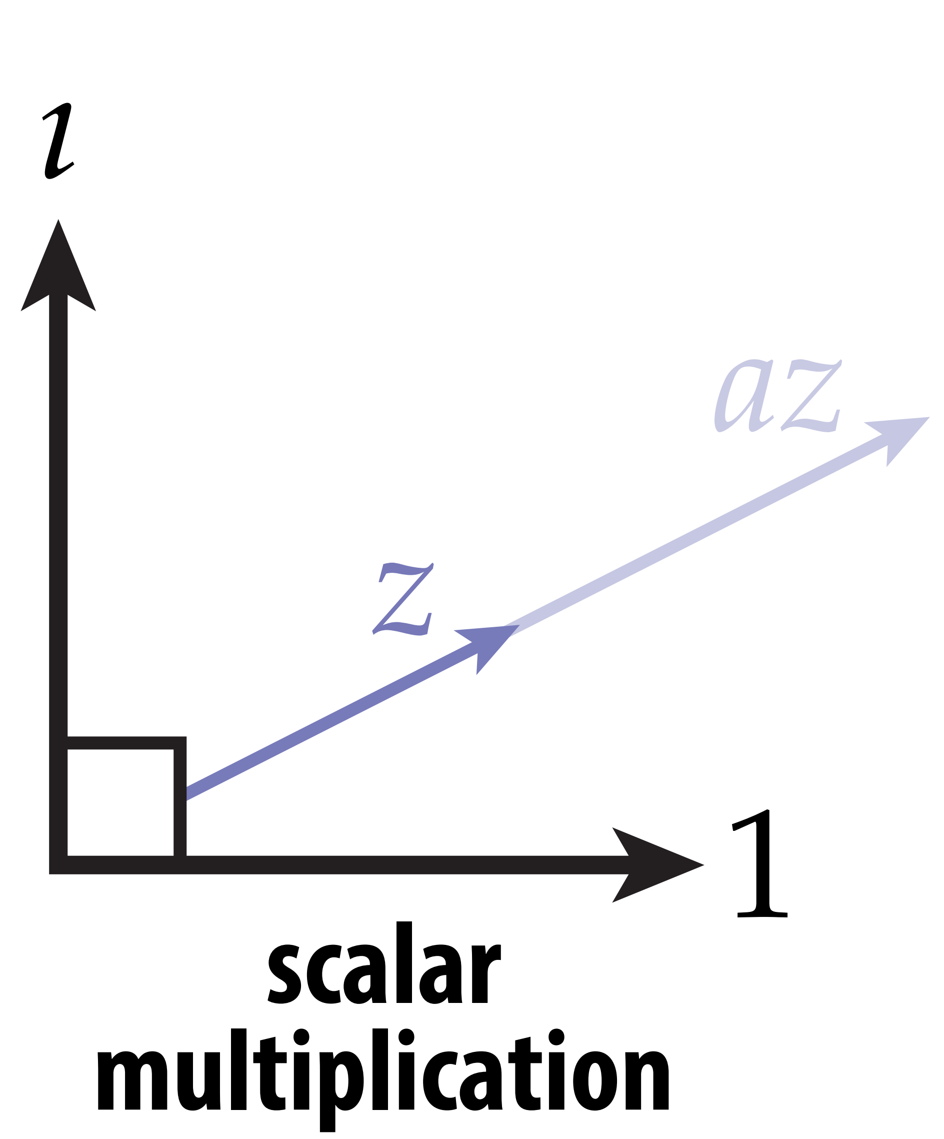
- …except that we're going to define a useful new notion of the product between two vectors.

Complex Arithmetic

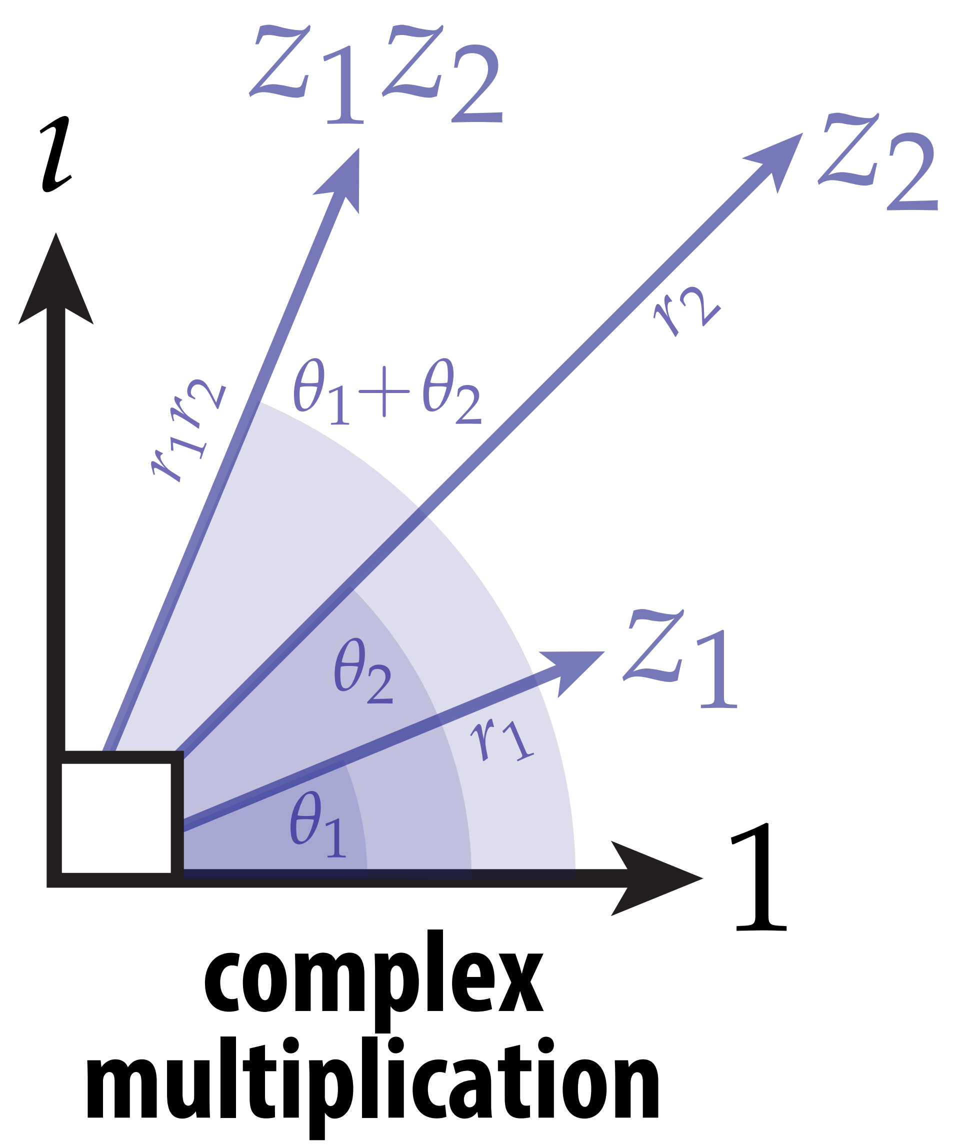
- Same operations as before, plus one more:



vector addition



scalar multiplication



complex multiplication

- Complex multiplication:
  - angles *add*
  - magnitudes *multiply*

"POLAR FORM"*:

$$z_1 := (r_1, \theta_1)$$
$$z_2 := (r_2, \theta_2)$$
$$z_1 z_2 = (r_1 r_2, \theta_1 + \theta_2)$$

have to be more careful here!

*Not *really* how it works, but useful geometric intuition.

Complex Product—Rectangular Form

- Complex product in "rectangular" coordinates (1, ı):

$$z_1 = (a + b\imath)$$

$$z_2 = (c + d\imath)$$

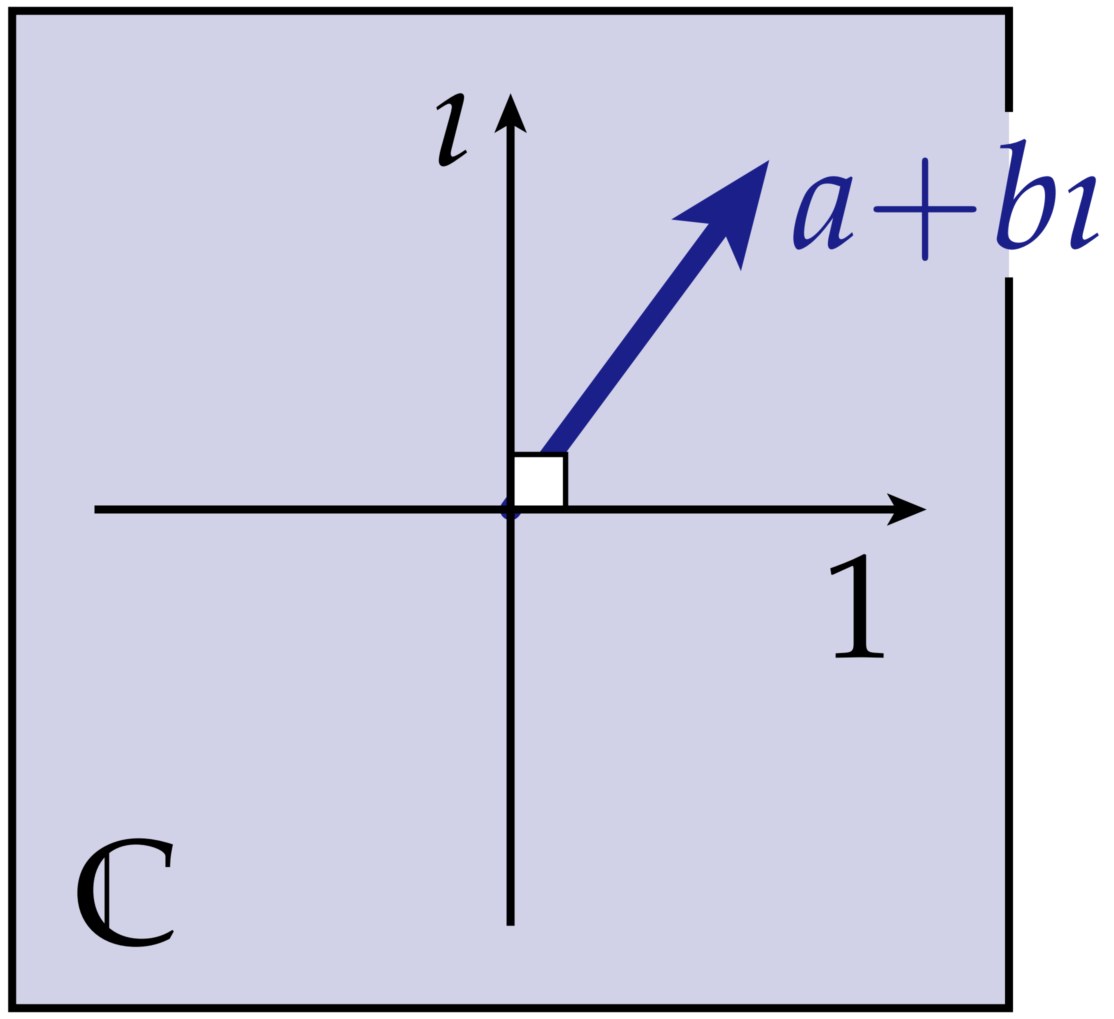
two quarter turns—same as -1

$$z_1 z_2 = ac + ad\imath + bc\imath + bd\imath^2 =$$

$$\boxed{(ac - bd) + (ad + bc)\imath.}$$

"real part" $\mathrm{Re}(z_1 z_2)$

"imaginary part" $\mathrm{Im}(z_1 z_2)$



- **We used a lot of "rules" here. Can you justify them geometrically?**
- **Does this product agree with our geometric description (last slide)?**

Complex Product—Polar Form

- Perhaps most beautiful identity in math:

$$e^{\imath\pi} + 1 = 0$$

- Specialization of *Euler's formula*:

$$e^{\imath\theta} = \cos(\theta) + \imath\sin(\theta)$$

- Can use to "implement" complex product:

$$z_1 = ae^{\imath\theta}, \quad z_2 = be^{\imath\phi}$$

$$z_1 z_2 = abe^{\imath(\theta+\phi)}$$

(as with real exponentiation, exponents *add*)

Leonhard Euler (1707–1783)

- Most prolific mathematician of all time
- Opera Omnia—1 vol./yr. starting 1911
- Still going! Now ~75 vols., 25k pages
- 228 papers posthumously
- Many later works while blind
- (Work was also *good*...)

[source: William Dunham]

Q: How does this operation differ from our earlier, "fake" polar multiplication?

2D Rotations: Matrices vs. Complex

- **Suppose we want to rotate a vector u by an angle θ, then by an angle ϕ.**

REAL / RECTANGULAR

$$\mathbf{u} = (x, y) \qquad \mathbf{A} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

$$\mathbf{B} = \begin{bmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{bmatrix}$$

$$\mathbf{Au} = \begin{bmatrix} x\cos\theta - y\sin\theta \\ x\sin\theta + y\cos\theta \end{bmatrix}$$

$$\mathbf{BAu} = \begin{bmatrix} (x\cos\theta - y\sin\theta)\cos\phi - (x\sin\theta + y\cos\theta)\sin\phi \\ (x\cos\theta - y\sin\theta)\sin\phi + (x\sin\theta + y\cos\theta)\cos\phi \end{bmatrix}$$

$$= \cdots \text{some trigonometry} \cdots =$$

$$\mathbf{BAu} = \begin{bmatrix} x\cos(\theta+\phi) - y\sin(\theta+\phi) \\ x\sin(\theta+\phi) + y\cos(\theta+\phi) \end{bmatrix}.$$

(…and simplification is not always this obvious.)

COMPLEX / POLAR

$$u = re^{\imath\alpha}$$
$$a = e^{\imath\theta}$$
$$b = e^{\imath\phi}$$

$$abu = re^{\imath(\alpha+\theta+\phi)}.$$

Or if we want rectangular coords:

$$= r\begin{bmatrix} \cos(\alpha+\theta+\phi) \\ \sin(\alpha+\theta+\phi) \end{bmatrix}$$

Pervasive theme in graphics:

Sure, there are often many "equivalent" representations.

...But why not choose the one that makes life easiest*?

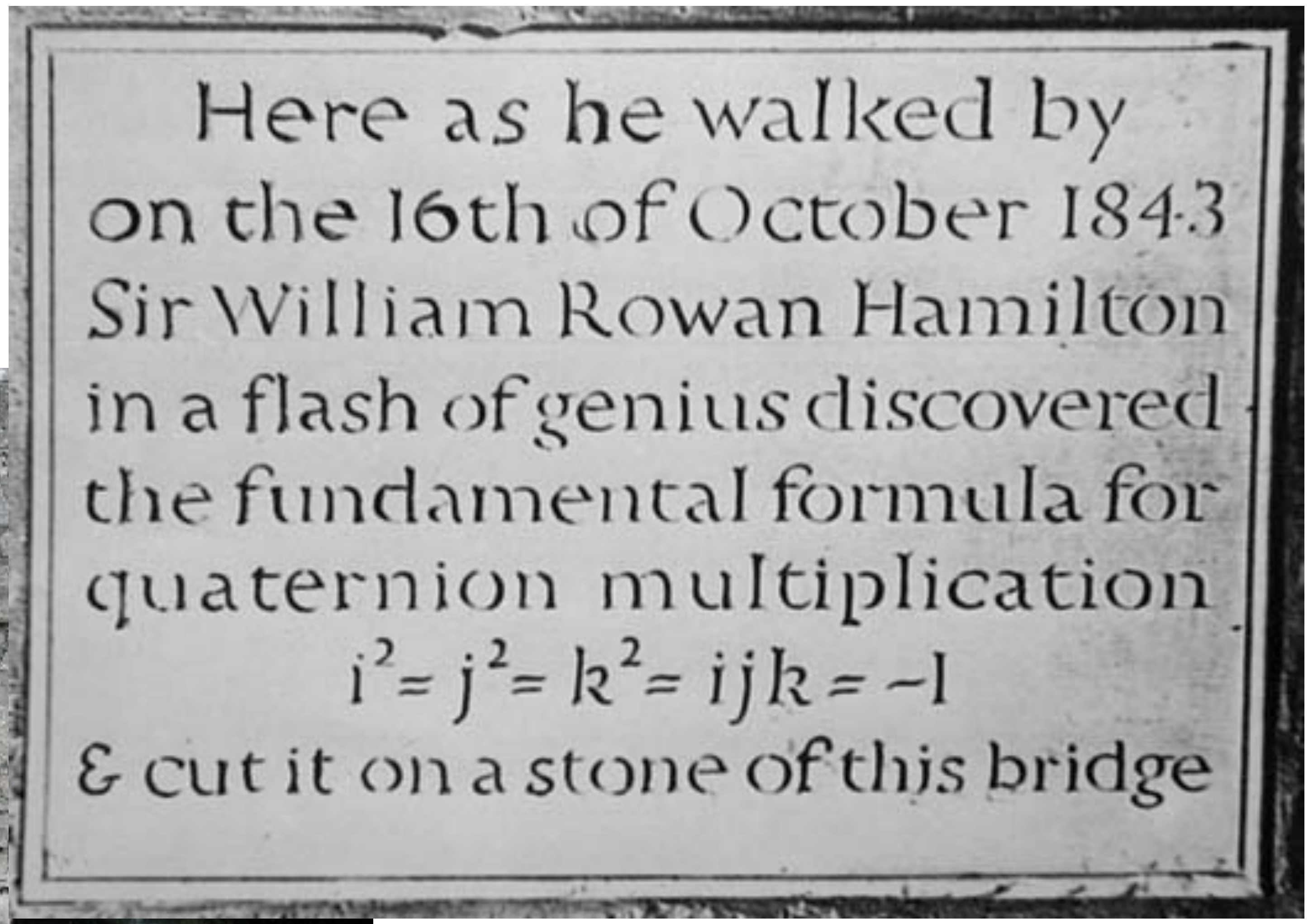
*Or most efficient, or most accurate...

Quaternions

- **TLDR: Kind of like complex numbers but for 3D rotations**
- **Weird situation: can't do 3D rotations w/ only 3 components!**

William Rowan Hamilton (1805-1865)

(Not Hamilton)



Here as he walked by
on the 16th of October 1843
Sir William Rowan Hamilton
in a flash of genius discovered
the fundamental formula for
quaternion multiplication
$i^2 = j^2 = k^2 = ijk = -1$
& cut it on a stone of this bridge

Quaternions in Coordinates

- **Hamilton's insight: in order to do 3D rotations in a way that mimics complex numbers for 2D, actually need FOUR coords.**
- **One real, *three* imaginary:**

$$\mathbb{H} := \text{span}(\{1, \imath, \jmath, k\})$$

"H" is for *Hamilton!*

$$q = a + b\imath + c\jmath + dk \in \mathbb{H}$$

- **Quaternion product determined by**

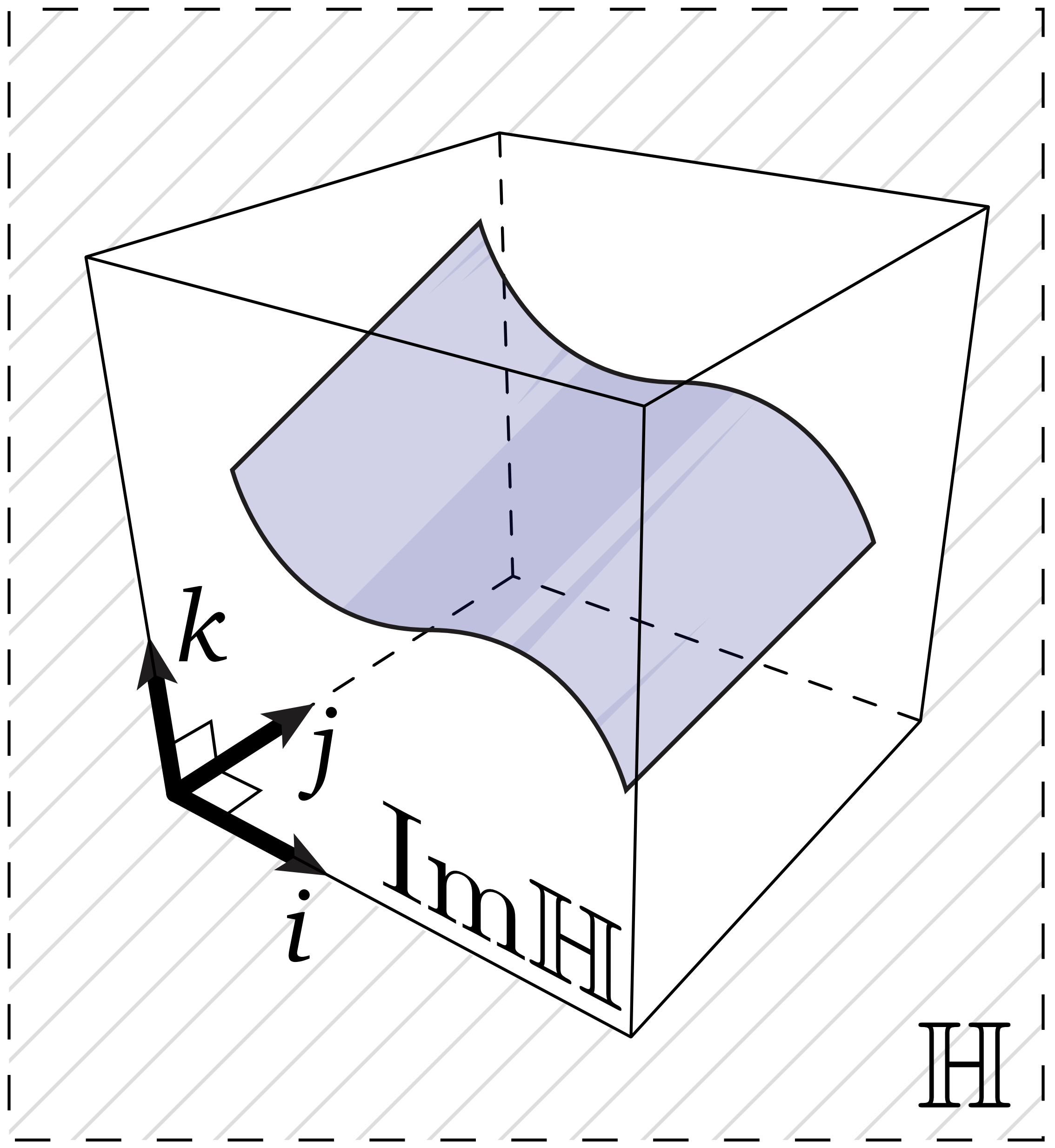
$$\imath^2 = \jmath^2 = k^2 = \imath\jmath k = -1$$

together w/ "natural" rules (distributivity, associativity, etc.)

- **WARNING: product no longer commutes!**

$$\text{For } q, p \in \mathbb{H}, \qquad qp \neq pq$$

(Will understand this *a lot* better when we study transformations.)



Quaternion Product in Components

- **Given two quaternions**

$$q = a_1 + b_1\imath + c_1\jmath + d_1k$$
$$p = a_2 + b_2\imath + c_2\jmath + d_2k$$

- **Can express their product as**

$$\begin{aligned} qp = a_1a_2 - b_1b_2 - c_1c_2 - d_1d_2 \\ +(a_1b_2 + b_1a_2 + c_1d_2 - d_1c_2)\imath \\ +(a_1c_2 - b_1d_2 + c_1a_2 + d_1b_2)\jmath \\ +(a_1d_2 + b_1c_2 - c_1b_2 + d_1a_2)k \end{aligned}$$

…fortunately there is a (much) nicer expression.

Quaternions—Scalar + Vector Form

- If we have *four* components, how do we talk about pts in 3D?
- Natural idea: we have three imaginary parts—why not use these to encode 3D vectors?

$$(x, y, z) \mapsto 0 + x\imath + y\jmath + zj$$

- Alternatively, can think of a quaternion as a pair

$$(\,\underset{\mathbb{R}}{\underset{\cap}{\text{scalar}}},\ \underset{\mathbb{R}^3}{\underset{\cap}{\text{vector}}}\,) \in \mathbb{H}$$

- Quaternion product then has simple(r) form:

$$(a, \mathbf{u})(b, \mathbf{v}) = (ab - \mathbf{u} \cdot \mathbf{v}, a\mathbf{v} + b\mathbf{u} + \mathbf{u} \times \mathbf{v})$$

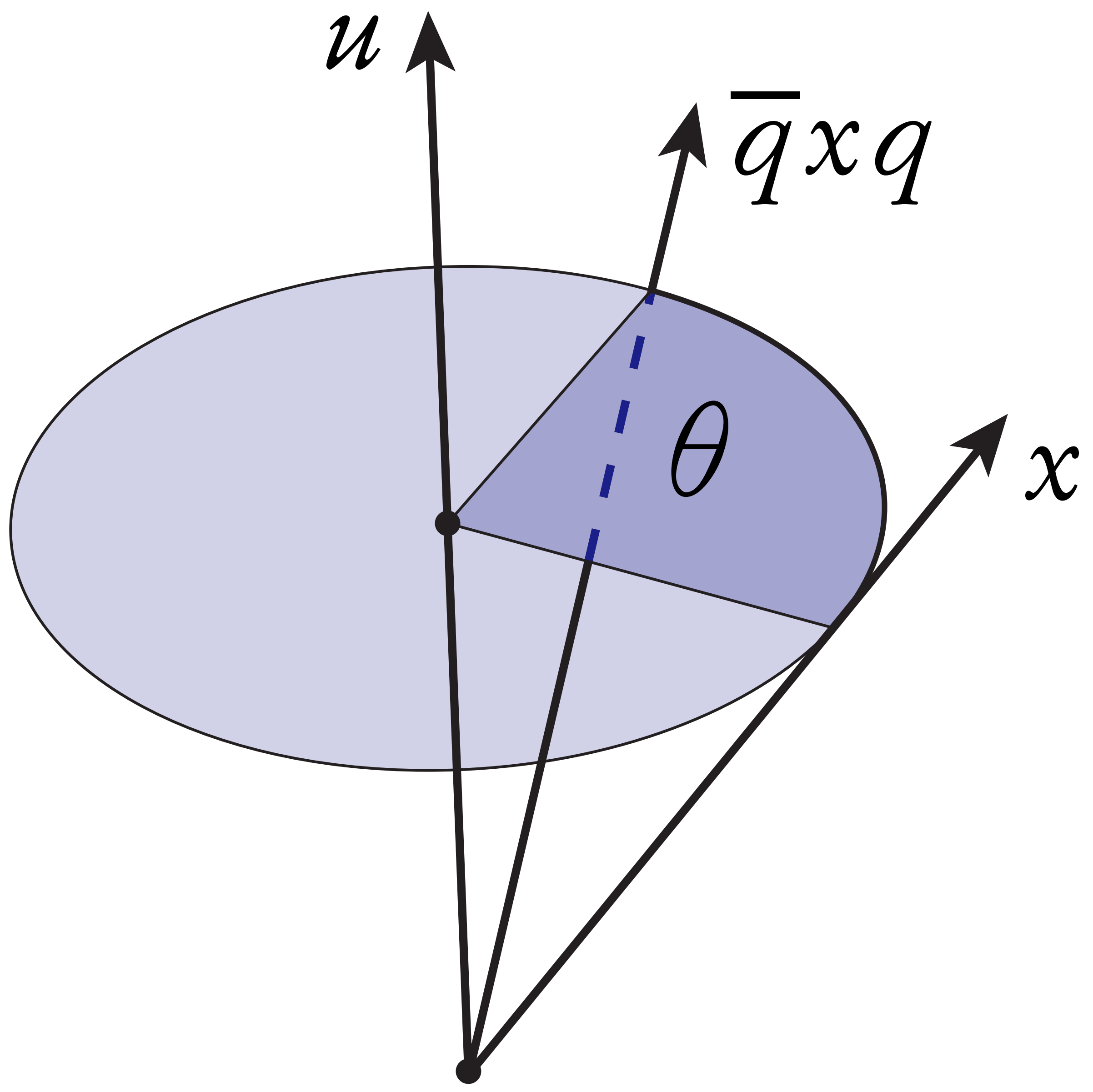
- For vectors in R3, gets even simpler:

$$\mathbf{uv} = \mathbf{u} \times \mathbf{v} - \mathbf{u} \cdot \mathbf{v}$$

3D Transformations via Quaternions

- **Main use for quaternions in graphics?** ***Rotations.***
- **Consider vector x ("pure imaginary") and *unit* quaternion q:**

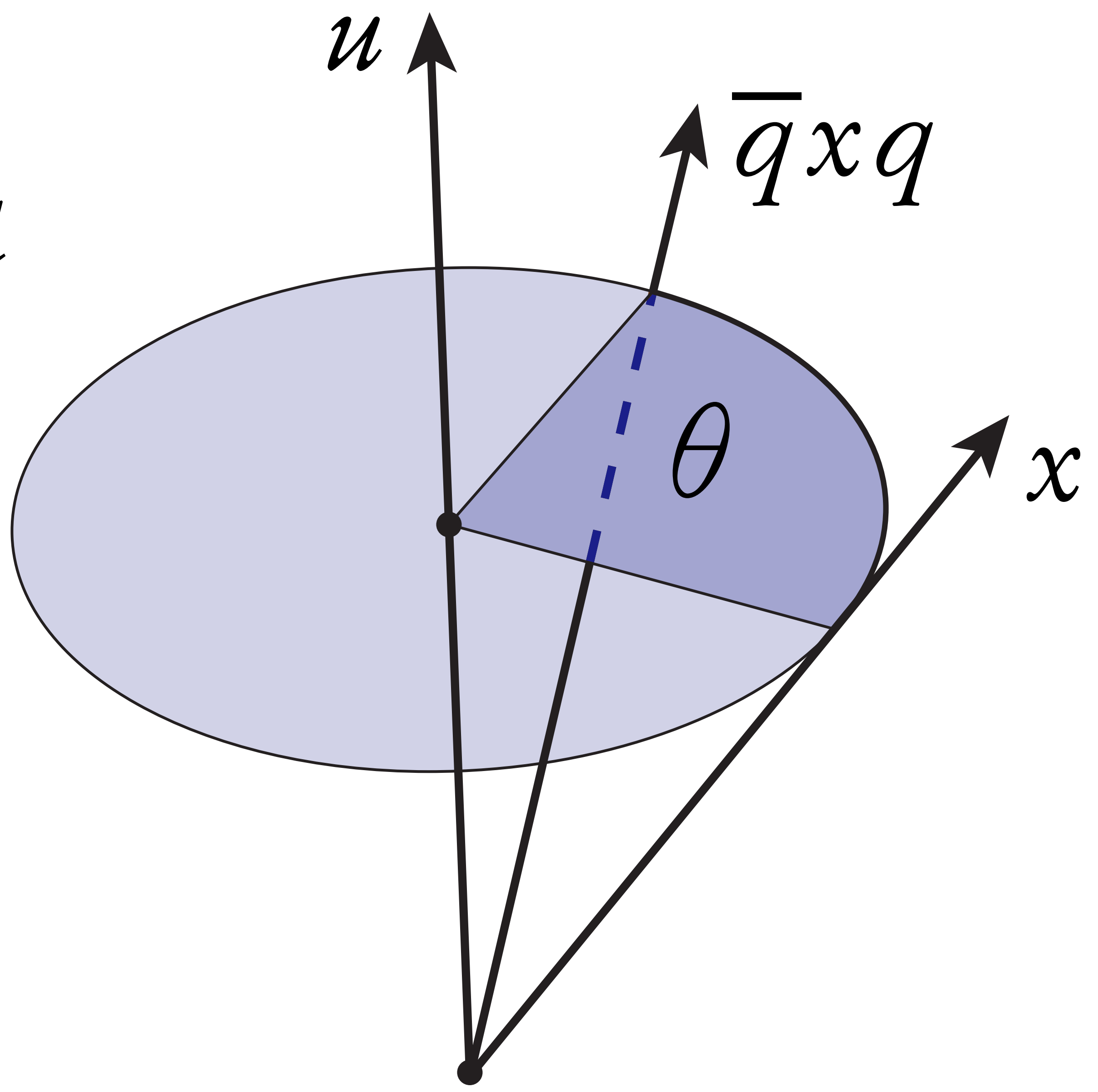
$$x \in \mathrm{Im}(\mathbb{H})$$
$$q \in \mathbb{H}, \quad |q|^2 = 1$$



Rotation from Axis/Angle, Revisited

- **Given axis u, angle θ, quaternion q representing rotation is**

$$q = \cos(\theta/2) + \sin(\theta/2)u$$

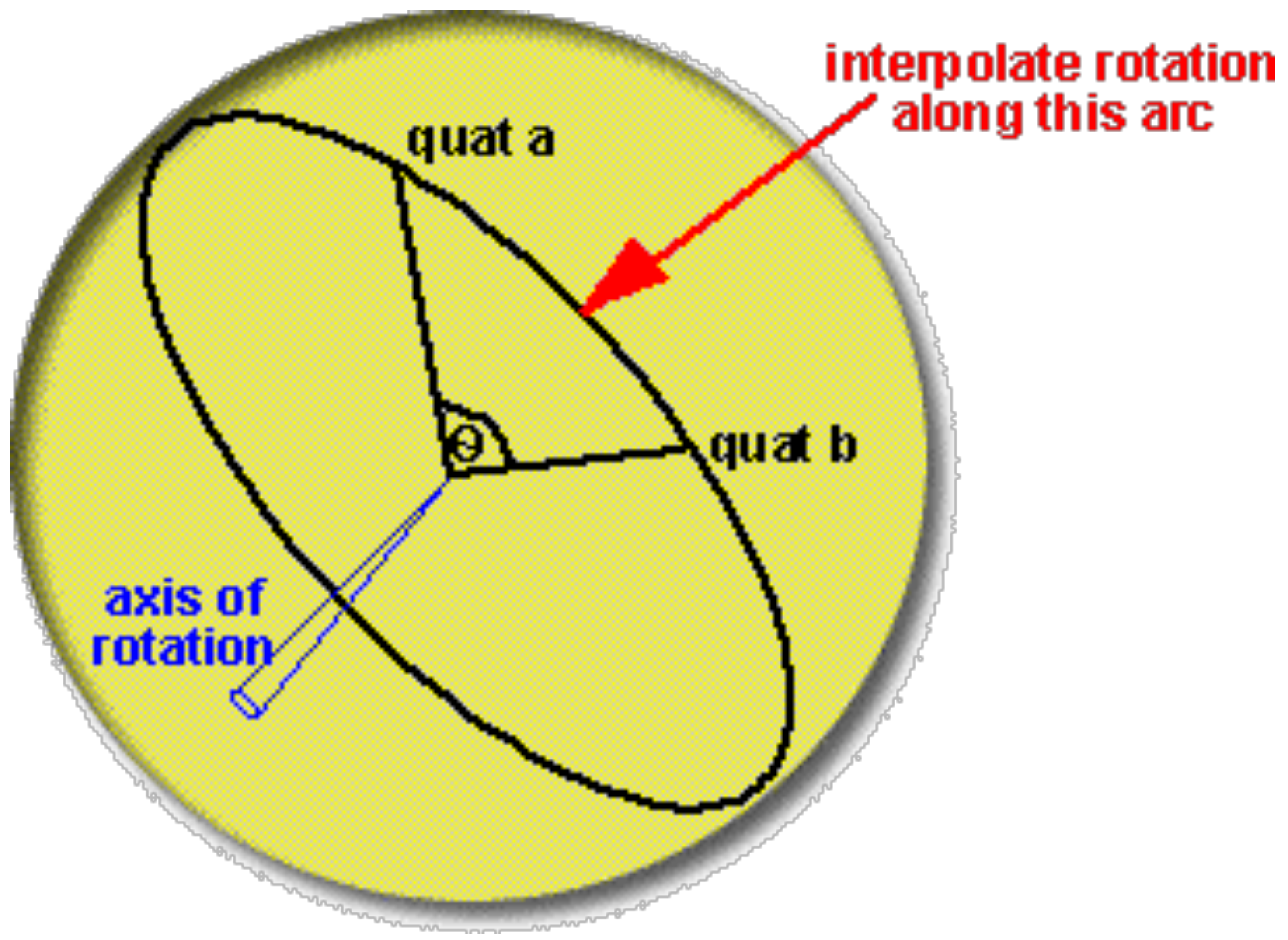


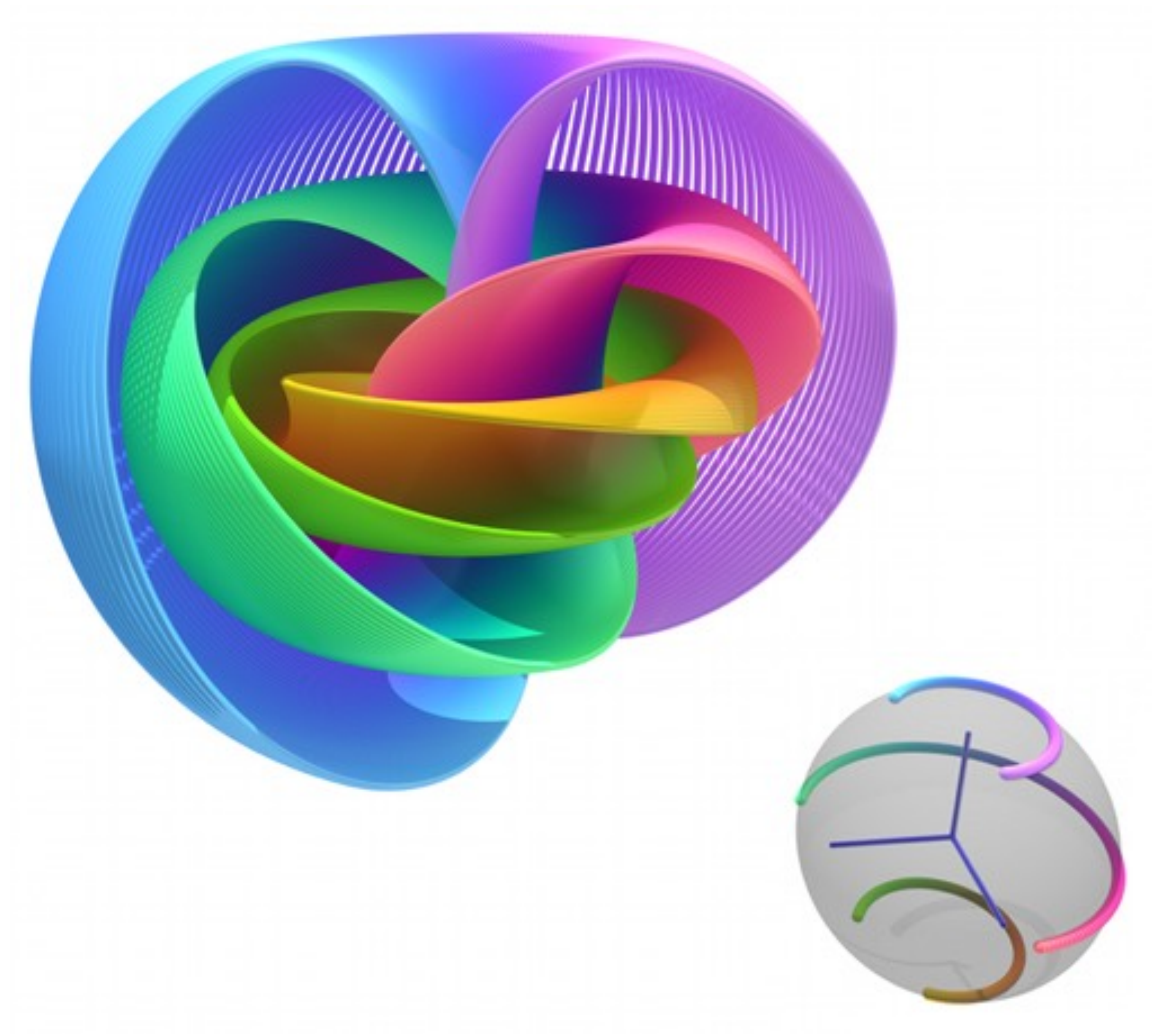
- **Slightly easier to remember (and manipulate) than matrix:**

$$\begin{bmatrix} \cos\theta + u_x^2(1-\cos\theta) & u_xu_y(1-\cos\theta) - u_z\sin\theta & u_xu_z(1-\cos\theta) + u_y\sin\theta \\ u_yu_x(1-\cos\theta) + u_z\sin\theta & \cos\theta + u_y^2(1-\cos\theta) & u_yu_z(1-\cos\theta) - u_x\sin\theta \\ u_zu_x(1-\cos\theta) - u_y\sin\theta & u_zu_y(1-\cos\theta) + u_x\sin\theta & \cos\theta + u_z^2(1-\cos\theta) \end{bmatrix}$$

More Quaternions and Rotation

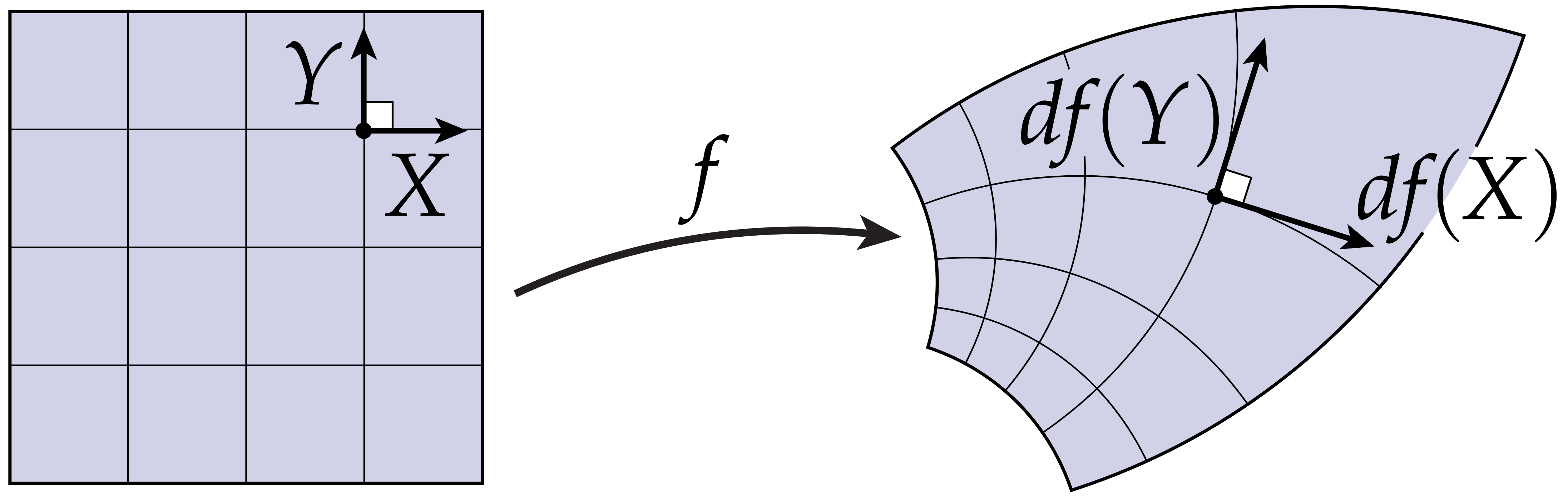
- Don't have time to cover everything, but…
- Quaternions provide some very nice utility/perspective when it comes to rotations:
  - Spherical linear interpolation ("slerp")
  - *Hopf fibration / "belt trick"*
  - ...

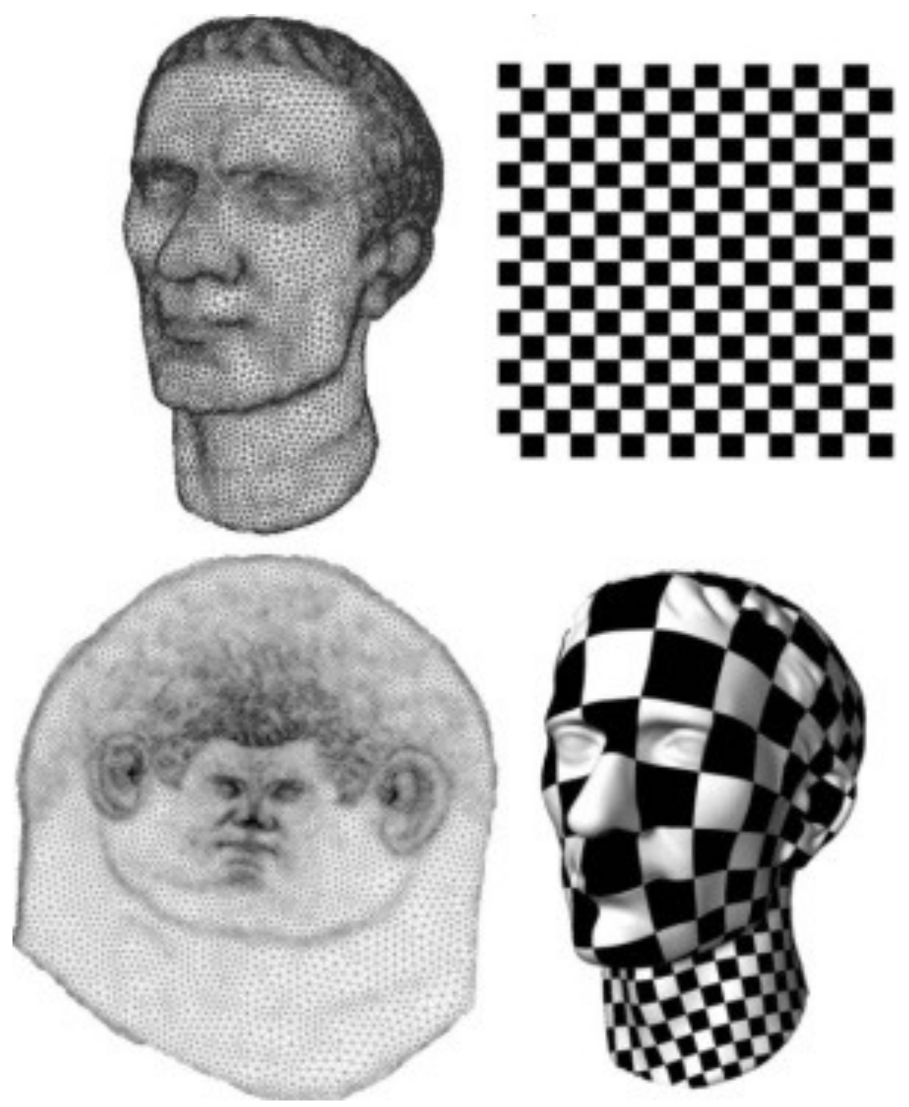


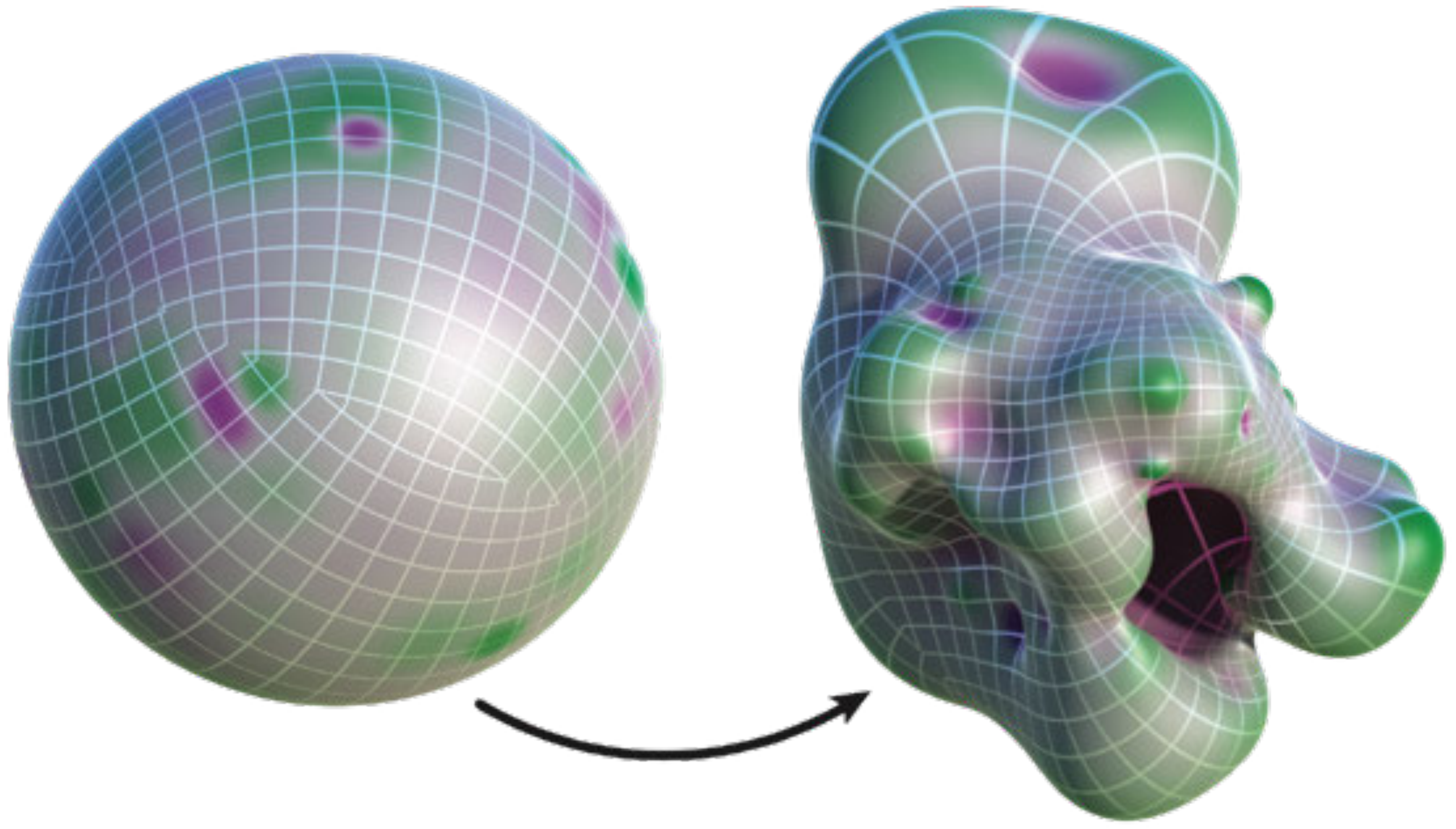


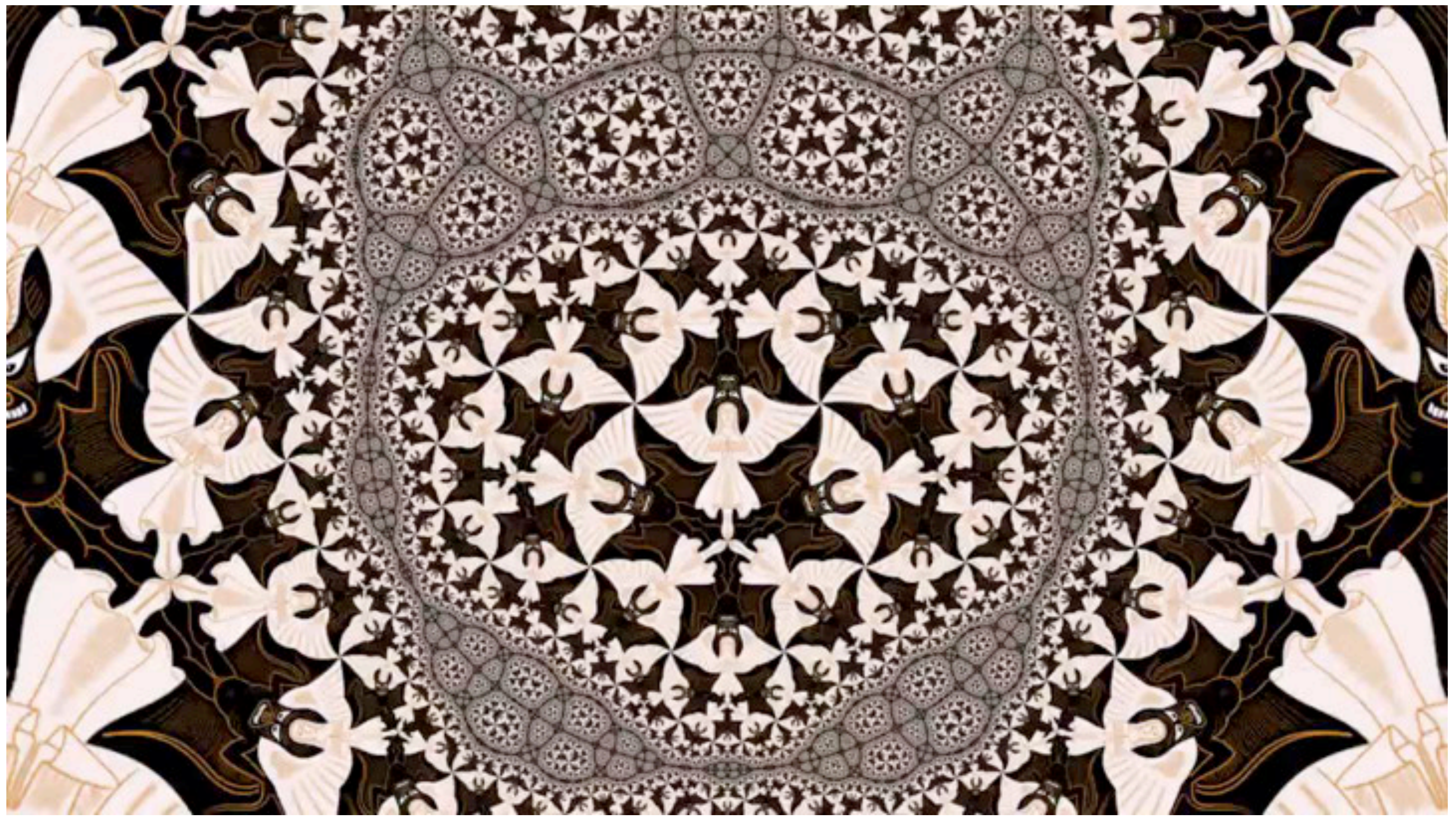
Where else are (hyper-)complex numbers useful in computer graphics?

Complex #s: Language of *Conformal Maps*



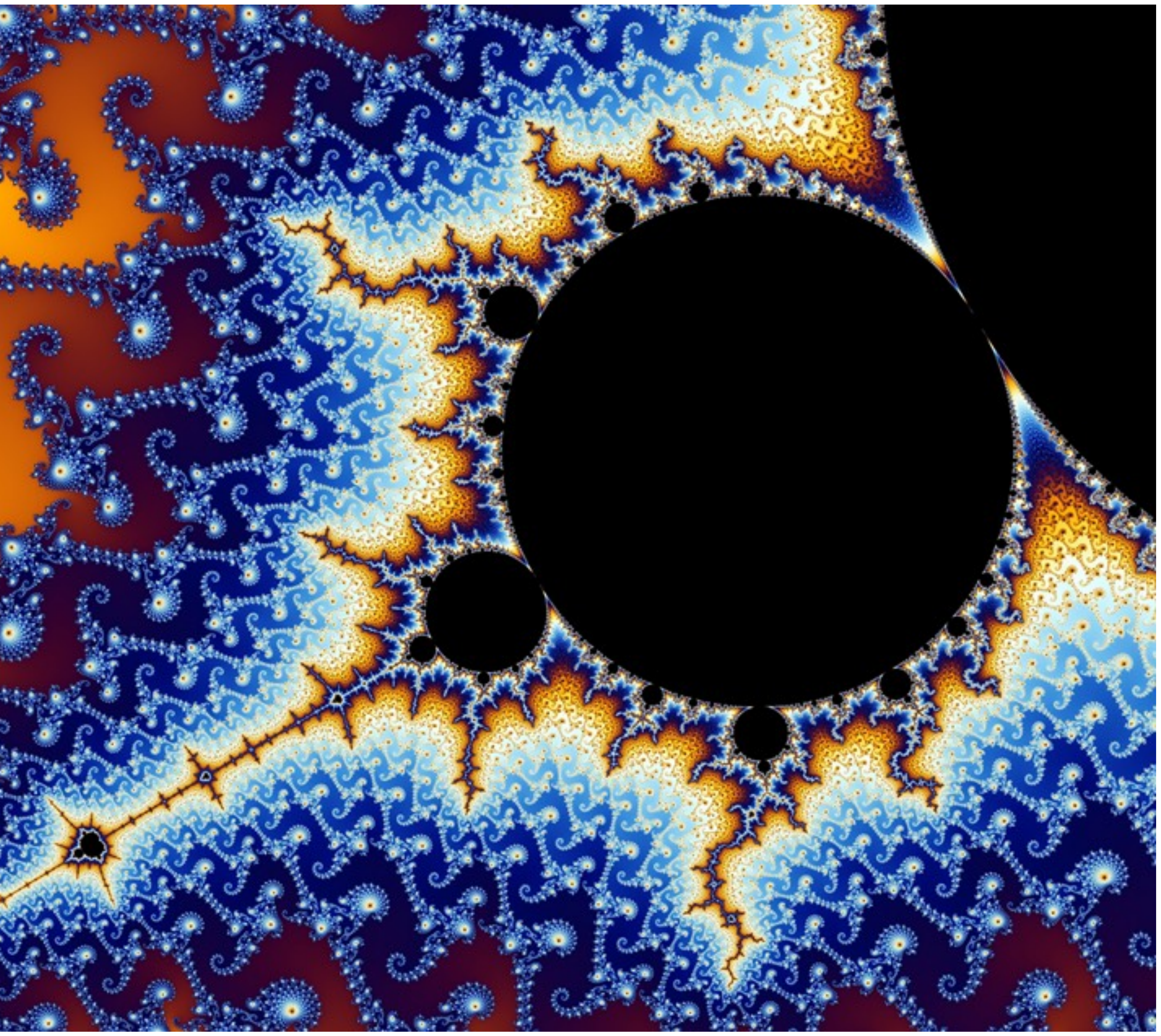


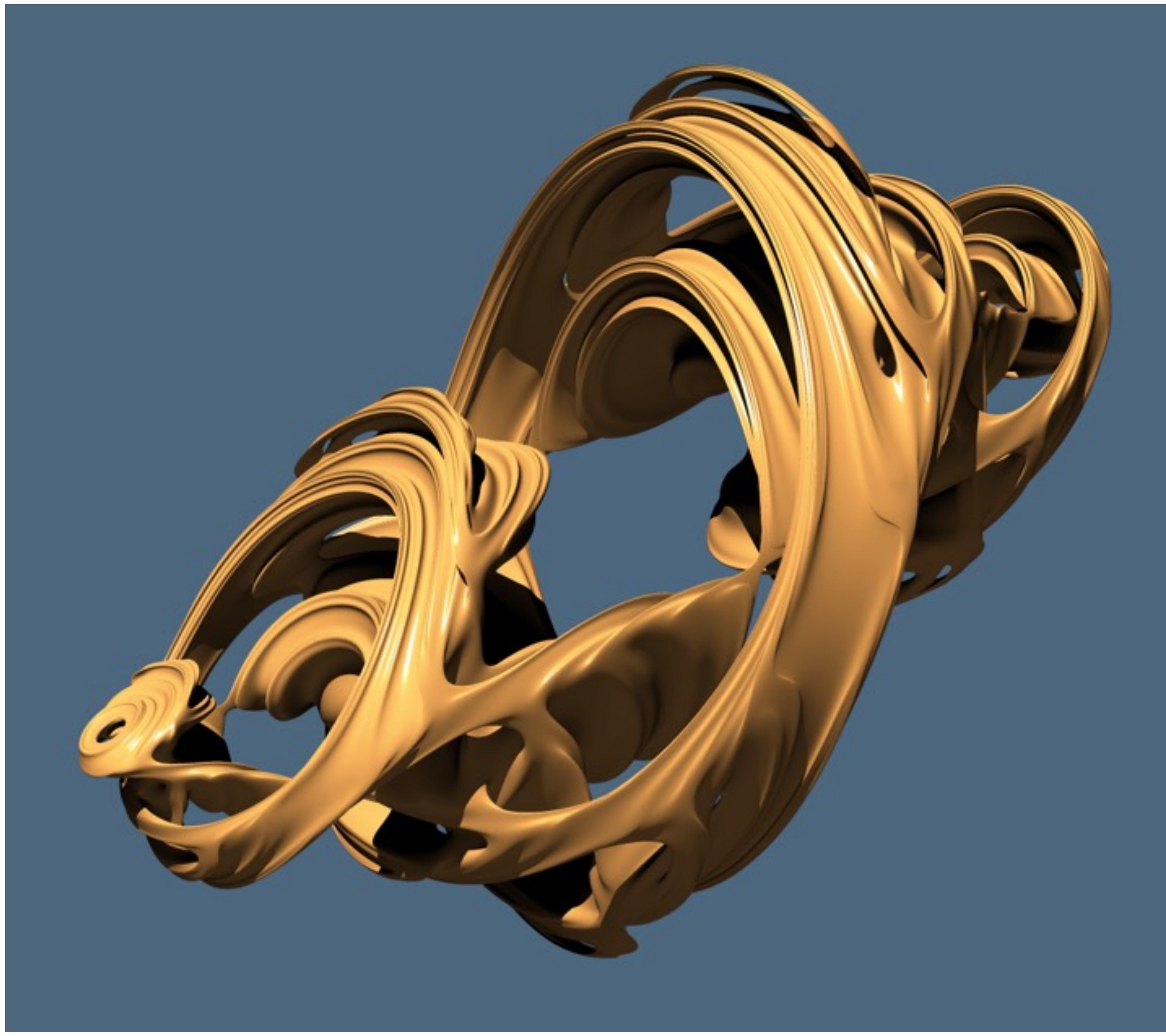




Useless-But-Beautiful Example: Fractals

- Defined in terms of iteration on (hyper)complex numbers:





(Will see exactly how this works later in class.)