Scale
\[
\begin{bmatrix}
S_x & 0 \\
0 & S_y
\end{bmatrix}
\begin{bmatrix}
X \\
Y
\end{bmatrix}
\]

conceptually: transform each point of an object by \( \begin{bmatrix} S_x & 0 \\ 0 & S_y \end{bmatrix} \) to get new, scaled version

actually: transform each vertex

Non uniform scale
\[
\begin{bmatrix}
X \\
2Y
\end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix} \begin{bmatrix}
X \\
Y
\end{bmatrix}
\]

Shear
shear in \( X \)
\[
\begin{bmatrix}
1 & s \\
0 & 1
\end{bmatrix}
\]
shear in \( Y \)
\[
\begin{bmatrix}
1 & 0 \\
6 & 1
\end{bmatrix}
\]
Rotation:

A vector makes an angle $\alpha$ with the x-axis, length $r$.

$$x_a = r \cos \alpha$$

$$y_a = r \sin \alpha$$

$$r = x_a^2 + y_a^2$$

Want to rotate vector $a$ by angle $\phi$ counter-clockwise to get vector $b$.

$$x_b = r \cos(x + \phi)$$

(from basic trig)

$$y_b = r \sin(x + \phi)$$

In matrix form,

$$\text{rotate}(\phi) = \begin{bmatrix} \cos \phi & -\sin \phi \\ \sin \phi & \cos \phi \end{bmatrix}$$
Composition of 2D Transformations

Example: first scale by \( S \),
then rotate by \( R \).

\[
v_2 = Sv_1, \quad v_3 = Rv_2.
\]

\[
\Rightarrow v_3 = R(Sv_1).
\]

\[
v_3 = (RS)v_1,
\]

\[
M = RS.
\]

\[
v_3 = Mv_1.
\]

Order matters!
Matrix multiplication is not commutative,
\( RS \neq SR \).

\( M = RS \) will first scale and then rotate.

\[\begin{array}{c}
\text{scale} \\
\downarrow \\
\text{rotate}
\end{array}\quad \neq \quad \begin{array}{c}
\text{rotate} \\
\downarrow \\
\text{scale}
\end{array}\]
Basic 3D Transforms

Scale \((s_x, s_y, s_z)\):

\[
\begin{bmatrix}
  s_x & 0 & 0 \\
  0 & s_y & 0 \\
  0 & 0 & s_z
\end{bmatrix}
\]

Shear \((dx, dy)\):

\[
\begin{bmatrix}
  1 & dy & dx \\
  0 & 1 & 0 \\
  0 & 0 & 1
\end{bmatrix}
\]

Rotation \(\Rightarrow\) more complicated.
What axis do you want to rotate about?

\[
\begin{bmatrix}
  \cos \phi & -\sin \phi & 0 \\
  \sin \phi & \cos \phi & 0 \\
  0 & 0 & 1
\end{bmatrix}
\]

\[
\begin{bmatrix}
  1 & 0 & 0 \\
  0 & \cos \phi & -\sin \phi \\
  0 & \sin \phi & \cos \phi
\end{bmatrix}
\]

\[
\begin{bmatrix}
  \cos \phi & 0 & \sin \phi \\
  0 & 1 & 0 \\
  -\sin \phi & 0 & \cos \phi
\end{bmatrix}
\]

No change in value for the coordinate

being rotated about \(y\).
Rotation Matrices are orthonormal.
3 rows of matrix are mutually orthogonal unit vectors.

For 3 columns are also.

The inverse of an orthonormal matrix is its transpose:

\[ M^{-1} = M^T \]

Also geometric inverse under actions of \( M \)

\[ y = M^{-1} M x \]

Very handy; can compute inverse very cheaply!

See 2.4.5 for definition of orthonormal:

\[ u \cdot v = 0 \]

\[ ||u|| = ||v|| = 1 \]

\[ u \cdot v = ||u|| ||v|| \cos \phi \]

\[ \phi = 90^\circ \]
Translation:

What we have seen so far has the form:

$x' = m_{11}x + m_{12}y$

$y' = m_{21}x + m_{22}y$

No way to represent translation?

$x' = x + x_t$

$y' = y + y_t$

⇒ add a dimension to the transformation matrix

2D beamers

\[
\begin{bmatrix}
1 & 0 & x_t \\
0 & 1 & y_t \\
0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
x \\
y \\
1
\end{bmatrix} =
\begin{bmatrix}
x + x_t \\
y + y_t \\
1
\end{bmatrix}
\]

Note: this is a 3D shear with $z$ set = 1

In 3D

\[
\begin{bmatrix}
1 & 0 & x_t \\
0 & 1 & y_t \\
0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
x \\
y \\
2t
\end{bmatrix} =
\begin{bmatrix}
x + x_t \\
y + y_t \\
2 + 2t
\end{bmatrix}
\]

Does break the $M^T = M^{-1}$ property that we had for rotation matrices.

If $M = M_3^{-1} M_2^{-1} M_3$ then $M' = M_3^{-1} M_2^{-1} M_3^{-1}$.
Transforming Normal Vectors

Normals don’t work
Tangents do
Normals + Tangents should be \( \perp \Rightarrow \) dot product = 0

\[ n^T t = 0 \]

We want
\[ t = Mt \quad n = Mn \]

need to find \( N \) such that
\[ n^T n = 0 \quad \text{(normal + tangent \( \perp \))} \]

Algebraic trick:
\[ n^T t = n^T M^T t = n^T M^{-1} Mt = 0 \]

\[ m^{-1} m = I \]

\[ (n^T m^{-1}) (Mt) = (n^T m^{-1}) t_m = 0 \]

\[ t_m = Mt \]

\[ n^T = n^T M^{-1} \]

take the transpose to get
\[ n^T = (n^T m^{-1})^T = (M^{-1})^T n \]

\[ N = (M^{-1})^T \leq \text{what we wanted} \]
see p. 150 + Section 5.2.3
for a nice way to compute \( N \) from the elements of \( N \) without expensive operations