Lecture 9: Reference Frames

Changing Coordinates and the Scene Graph

CS 453 — Computer Graphics

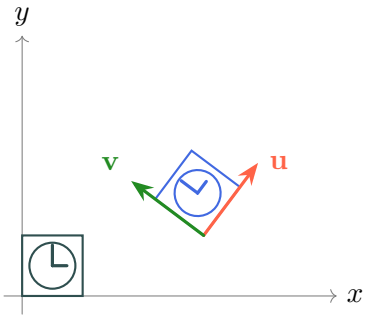
Fakhir Shaheen

Forman Christian University

Today: give the clock its own axes

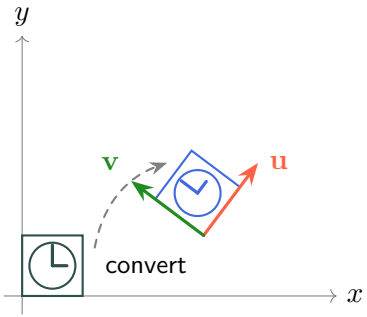
- Same clock as this week

Today: give the clock its own axes



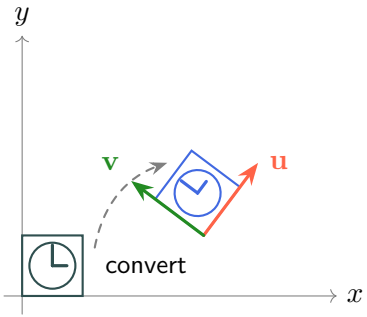
- Same clock as this week
- Give it its own axes

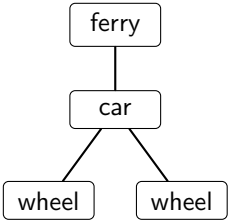
Today: give the clock its own axes



- Same clock as this week
- Give it its own axes
- Convert between its axes and world

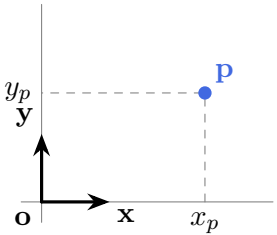
Today: give the clock its own axes





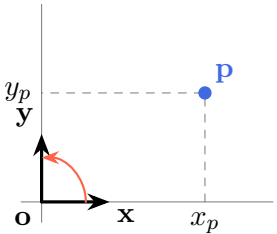
- Same clock as this week
- Give it its own axes
- Convert between its axes and world
- Hang parts on parts: a tree

Our conventions in one picture



- Bold: points and arrows

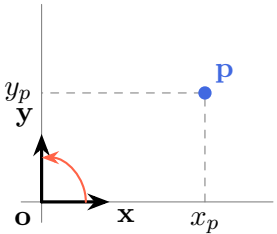
Our conventions in one picture



y is counterclockwise from x

- Bold: points and arrows
- y counterclockwise from x

Our conventions in one picture

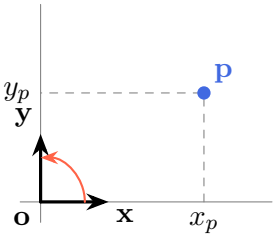


$$\mathbf{p}' = \mathbf{M} \begin{bmatrix} x_p \\ y_p \\ 1 \end{bmatrix}$$

- Bold: points and arrows
- $\mathbf{y}$ counterclockwise from $\mathbf{x}$
- Column vector, matrix on its left

$\mathbf{y}$ is counterclockwise from $\mathbf{x}$

Our conventions in one picture



$$\mathbf{p}' = \mathbf{M} \begin{bmatrix} x_p \\ y_p \\ 1 \end{bmatrix}$$

$\mathbf{y}$ is counterclockwise from $\mathbf{x}$

$\mathbf{M}_2 \quad \mathbf{M}_1 \quad \mathbf{p}$

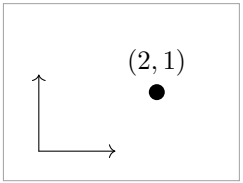
$\mathbf{M}_1$ first

- Bold: points and arrows
- $\mathbf{y}$ counterclockwise from $\mathbf{x}$
- Column vector, matrix on its left
- Read products right to left

Coordinate frames

Every object gets its own axes

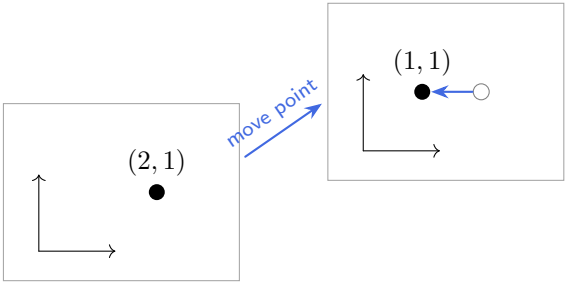
Move the point, or move the axes



translate by $(-1, 0)$

- Start: the point at $(2, 1)$

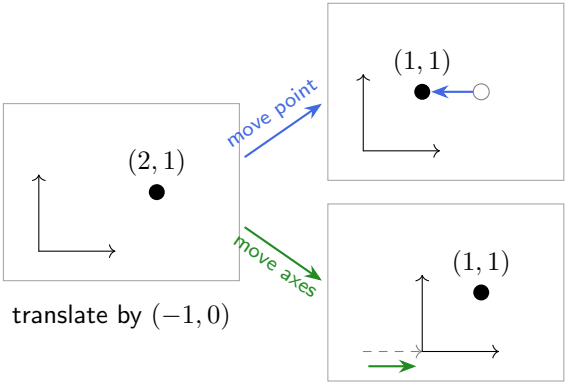
Move the point, or move the axes



translate by $(-1, 0)$

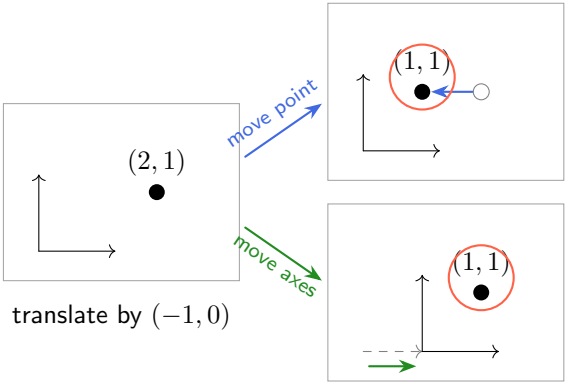
- Start: the point at $(2, 1)$
- Move the point one step left

Move the point, or move the axes



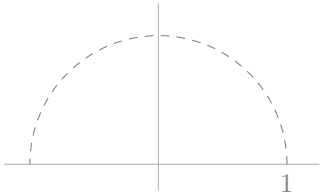
- Start: the point at $(2, 1)$
- Move the point one step left
- Or move the axes one step right

Move the point, or move the axes



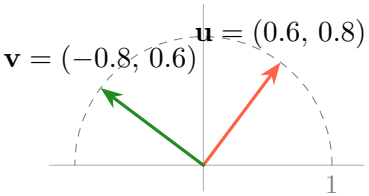
- Start: the point at $(2, 1)$
- Move the point one step left
- Or move the axes one step right
- Both give $(1, 1)$

Orthonormal basis: unit length, right angles



- Movable axes need two arrows to measure along: a *basis*

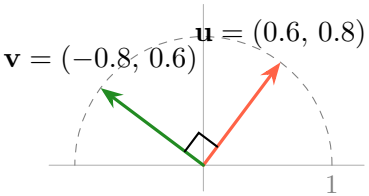
Orthonormal basis: unit length, right angles



- Movable axes need two arrows to measure along: a *basis*
- Unit length: $\|\mathbf{u}\| = \|\mathbf{v}\| = 1$

This $\mathbf{u}, \mathbf{v}$ is the clock's basis for the rest of today

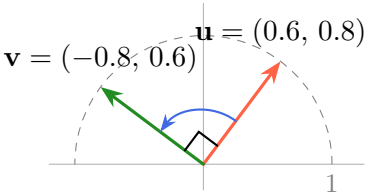
Orthonormal basis: unit length, right angles



- Movable axes need two arrows to measure along: a *basis*
- Unit length: $\|\mathbf{u}\| = \|\mathbf{v}\| = 1$
- Right angle: $\mathbf{u} \cdot \mathbf{v} = 0$

This $\mathbf{u}, \mathbf{v}$ is the clock's basis for the rest of today

Orthonormal basis: unit length, right angles

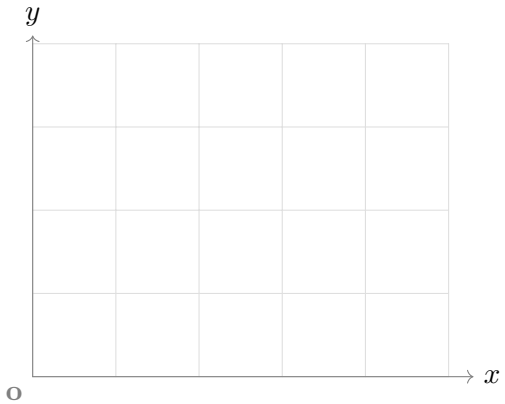


- Movable axes need two arrows to measure along: a *basis*
- Unit length: $\|\mathbf{u}\| = \|\mathbf{v}\| = 1$
- Right angle: $\mathbf{u} \cdot \mathbf{v} = 0$
- Right-handed: $\mathbf{v}$ a quarter turn counterclockwise from $\mathbf{u}$

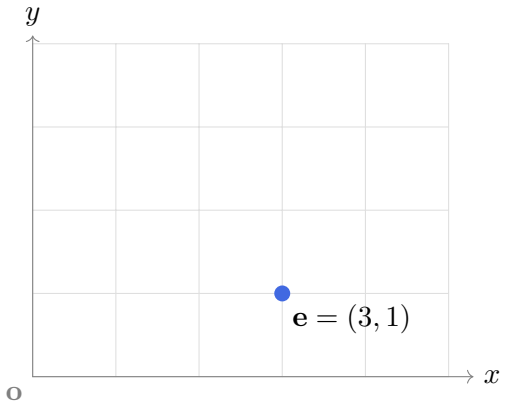
This $\mathbf{u}, \mathbf{v}$ is the clock's basis for the rest of today

Why: coordinates become dot products

A frame is an origin plus a basis

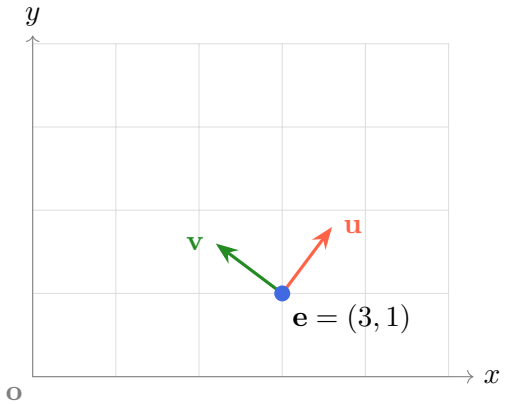


A frame is an origin plus a basis



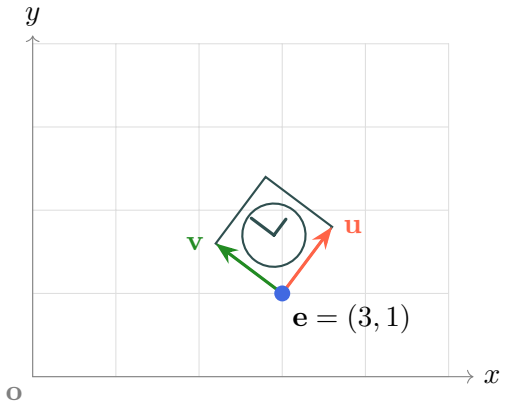
- Origin $\mathbf{e}$: where it sits

A frame is an origin plus a basis



- Origin $\mathbf{e}$: where it sits
- Basis $\mathbf{u}, \mathbf{v}$: which way it faces

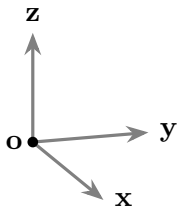
A frame is an origin plus a basis



- Origin $\mathbf{e}$: where it sits
- Basis $\mathbf{u}, \mathbf{v}$: which way it faces
- Draw the clock in its own frame

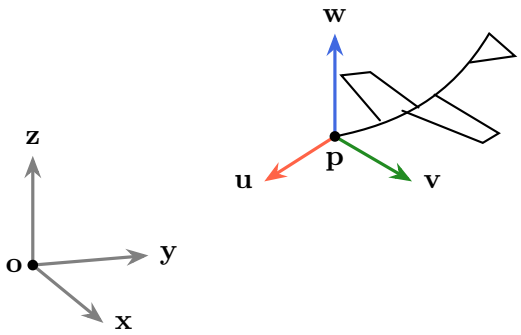
$$\mathbf{e} + u\,\mathbf{u} + v\,\mathbf{v}$$

The world frame is never stored



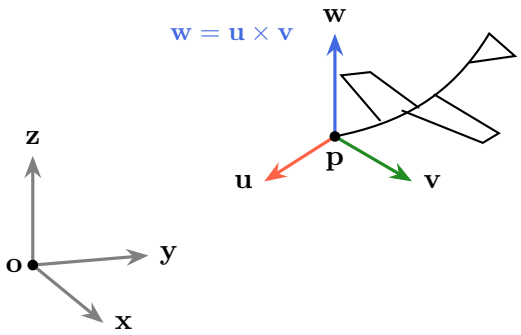
- World axes: assumed, never stored

The world frame is never stored



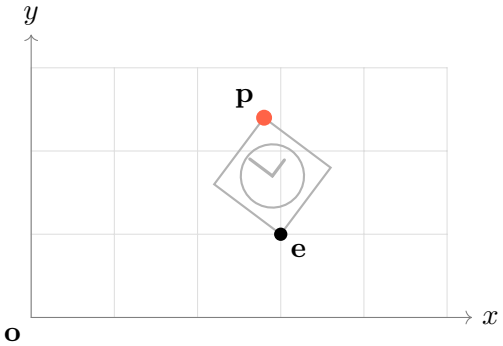
- World axes: assumed, never stored
- Plane's frame: stored in world numbers

The world frame is never stored



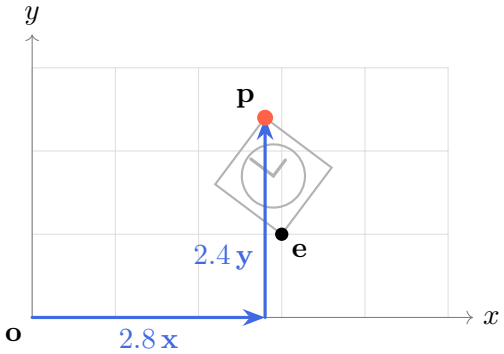
- World axes: assumed, never stored
- Plane's frame: stored in world numbers
- Right-handed when $\mathbf{w} = \mathbf{u} \times \mathbf{v}$

One point, two addresses



- Same point $\mathbf{p}$, two frames

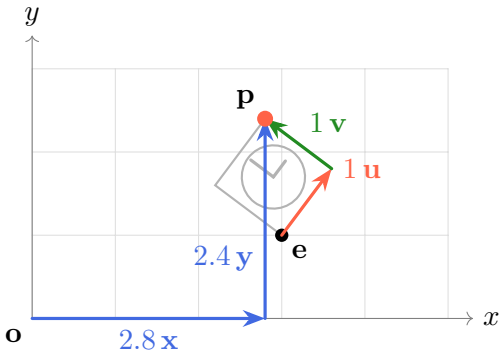
One point, two addresses



- Same point $\mathbf{p}$, two frames
- World address: $(2.8, 2.4)$

  $\mathbf{p} = \mathbf{o} + x_p\,\mathbf{x} + y_p\,\mathbf{y}$

One point, two addresses



- Same point $\mathbf{p}$, two frames
- World address: $(2.8, 2.4)$

  $\mathbf{p} = \mathbf{o} + x_p\,\mathbf{x} + y_p\,\mathbf{y}$
- Clock address: $(1, 1)$

  $\mathbf{p} = \mathbf{e} + u_p\,\mathbf{u} + v_p\,\mathbf{v}$

Recap: coordinate frames

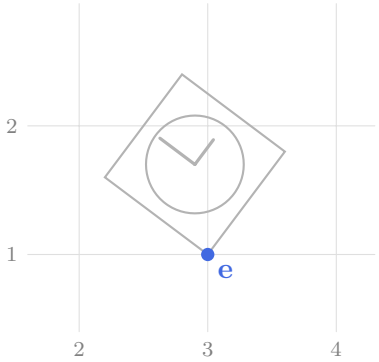
$$\mathbf{p} = \mathbf{e} + u_p \, \mathbf{u} + v_p \, \mathbf{v}$$

Frame coordinates (u_p, v_p) in, world point $\mathbf{p}$ out

Changing coordinates

Convert between world and frame

Frame to world: add up the arrows

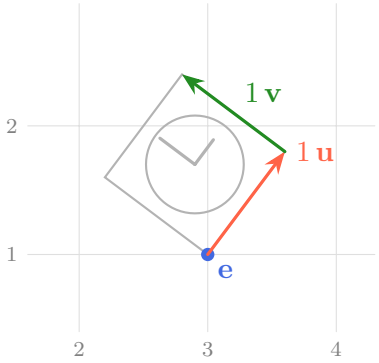


$$\mathbf{p} = \mathbf{e} + 1\,\mathbf{u} + 1\,\mathbf{v}$$

address $(1, 1)$

- Start at $\mathbf{e}$, add the arrows

Frame to world: add up the arrows

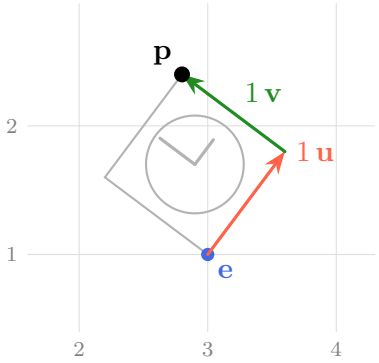


$$\mathbf{p} = \mathbf{e} + 1\,\mathbf{u} + 1\,\mathbf{v}$$ address $(1, 1)$

$$= (3, 1) + (0.6, 0.8) + (-0.8, 0.6)$$ world numbers in

- Start at $\mathbf{e}$, add the arrows

Frame to world: add up the arrows



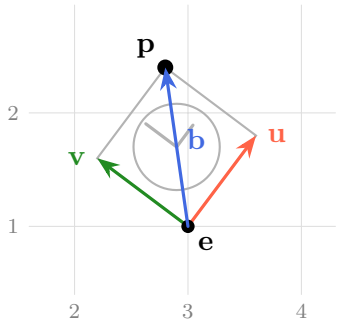
$$\mathbf{p} = \mathbf{e} + 1\,\mathbf{u} + 1\,\mathbf{v}$$ address $(1, 1)$

$$= (3, 1) + (0.6, 0.8) + (-0.8, 0.6)$$ world numbers in

$$= (2.8,\ 2.4)$$ world numbers out

- Start at $\mathbf{e}$, add the arrows
- Answer is already in world numbers

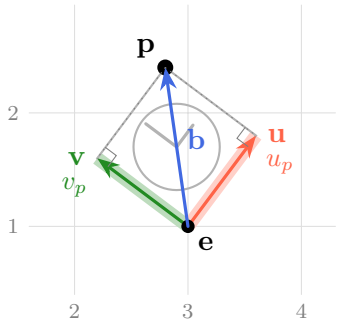
World to frame: subtract $\mathbf{e}$, then dot with each axis



$$\mathbf{b} = \mathbf{p} - \mathbf{e}$$

arrow from the frame's origin

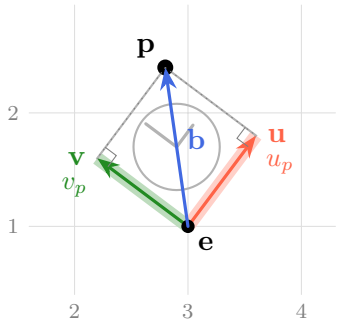
World to frame: subtract $\mathbf{e}$, then dot with each axis



$$\mathbf{b} = \mathbf{p} - \mathbf{e}$$ arrow from the frame's origin

$$u_p = \mathbf{u} \cdot \mathbf{b}, \qquad v_p = \mathbf{v} \cdot \mathbf{b}$$ its shadow on each axis

World to frame: subtract e, then dot with each axis

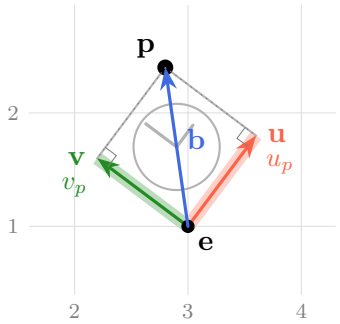


$\mathbf{b} = \mathbf{p} - \mathbf{e}$ arrow from the frame's origin

$u_p = \mathbf{u} \cdot \mathbf{b}, \qquad v_p = \mathbf{v} \cdot \mathbf{b}$ its shadow on each axis

$\mathbf{b} = (2.8,\ 2.4) - (3,\ 1) = (-0.2,\ 1.4)$

World to frame: subtract e, then dot with each axis



$\mathbf{b} = \mathbf{p} - \mathbf{e}$ arrow from the frame's origin

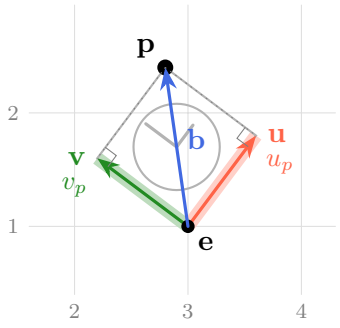
$u_p = \mathbf{u} \cdot \mathbf{b}, \qquad v_p = \mathbf{v} \cdot \mathbf{b}$ its shadow on each axis

$\mathbf{b} = (2.8,\ 2.4) - (3,\ 1) = (-0.2,\ 1.4)$

$u_p = \mathbf{u} \cdot \mathbf{b} = 1, \qquad v_p = \mathbf{v} \cdot \mathbf{b} = 1$

- Clock address $(1, 1)$: the corner, as it should be

World to frame: subtract e, then dot with each axis



$\mathbf{b} = \mathbf{p} - \mathbf{e}$ arrow from the frame's origin

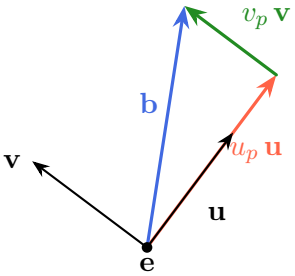
$u_p = \mathbf{u} \cdot \mathbf{b}, \qquad v_p = \mathbf{v} \cdot \mathbf{b}$ its shadow on each axis

$\mathbf{b} = (2.8,\ 2.4) - (3,\ 1) = (-0.2,\ 1.4)$

$u_p = \mathbf{u} \cdot \mathbf{b} = 1, \qquad v_p = \mathbf{v} \cdot \mathbf{b} = 1$

- Clock address $(1, 1)$: the corner, as it should be
- Dot products take arrows: subtract $\mathbf{e}$ first

Why $\mathbf{u} \cdot \mathbf{b}$ is exactly u_p: the basis is orthonormal

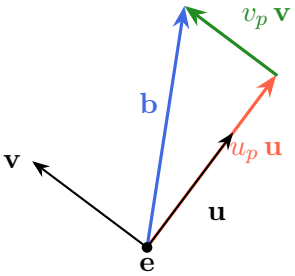


$$\mathbf{b} = u_p\,\mathbf{u} + v_p\,\mathbf{v}$$

u_p, v_p unknown

Any arrow $\mathbf{b}$: some $\mathbf{u}$ plus some $\mathbf{v}$

Why $\mathbf{u} \cdot \mathbf{b}$ is exactly u_p: the basis is orthonormal

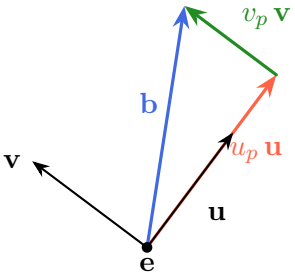


$$\mathbf{b} = u_p\,\mathbf{u} + v_p\,\mathbf{v} \qquad u_p, v_p \text{ unknown}$$

$$\mathbf{u} \cdot \mathbf{b} = \mathbf{u} \cdot (u_p\,\mathbf{u} + v_p\,\mathbf{v}) \qquad \text{dot both sides with } \mathbf{u}$$

Any arrow $\mathbf{b}$: some $\mathbf{u}$ plus some $\mathbf{v}$

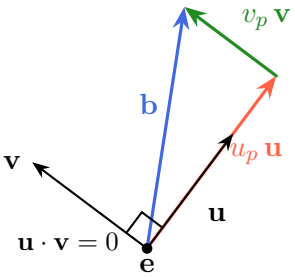
Why $\mathbf{u} \cdot \mathbf{b}$ is exactly u_p: the basis is orthonormal



$$\begin{aligned} \mathbf{b} &= u_p\,\mathbf{u} + v_p\,\mathbf{v} && u_p, v_p \text{ unknown} \\ \mathbf{u} \cdot \mathbf{b} &= \mathbf{u} \cdot (u_p\,\mathbf{u} + v_p\,\mathbf{v}) && \text{dot both sides with } \mathbf{u} \\ &= u_p\,(\mathbf{u} \cdot \mathbf{u}) + v_p\,(\mathbf{u} \cdot \mathbf{v}) && \text{dot distributes} \end{aligned}$$

Any arrow $\mathbf{b}$: some $\mathbf{u}$ plus some $\mathbf{v}$

Why $\mathbf{u} \cdot \mathbf{b}$ is exactly u_p: the basis is orthonormal

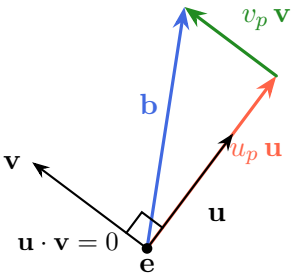


Any arrow $\mathbf{b}$: some $\mathbf{u}$ plus some $\mathbf{v}$

$$\begin{aligned}
\mathbf{b} &= u_p\,\mathbf{u} + v_p\,\mathbf{v} && u_p, v_p \text{ unknown} \\
\mathbf{u} \cdot \mathbf{b} &= \mathbf{u} \cdot (u_p\,\mathbf{u} + v_p\,\mathbf{v}) && \text{dot both sides with } \mathbf{u} \\
&= u_p\,(\mathbf{u} \cdot \mathbf{u}) + v_p\,(\mathbf{u} \cdot \mathbf{v}) && \text{dot distributes} \\
&= u_p \cdot 1 + v_p \cdot 0 = u_p && \|\mathbf{u}\| = 1,\ \mathbf{u} \cdot \mathbf{v} = 0
\end{aligned}$$

- $\mathbf{u} \cdot \mathbf{u} = 1$ keeps u_p; $\mathbf{u} \cdot \mathbf{v} = 0$ removes v_p

Why $\mathbf{u} \cdot \mathbf{b}$ is exactly u_p: the basis is orthonormal

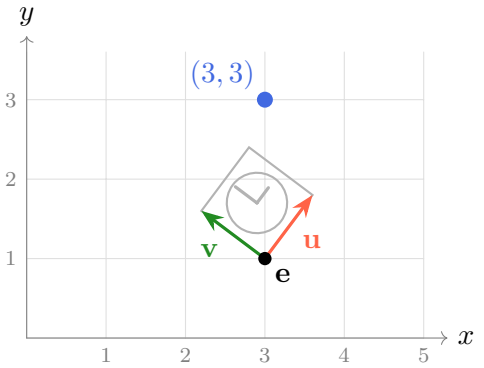


Any arrow $\mathbf{b}$: some $\mathbf{u}$ plus some $\mathbf{v}$

$$\begin{aligned}
\mathbf{b} &= u_p\,\mathbf{u} + v_p\,\mathbf{v} && u_p, v_p \text{ unknown} \\
\mathbf{u} \cdot \mathbf{b} &= \mathbf{u} \cdot (u_p\,\mathbf{u} + v_p\,\mathbf{v}) && \text{dot both sides with } \mathbf{u} \\
&= u_p\,(\mathbf{u} \cdot \mathbf{u}) + v_p\,(\mathbf{u} \cdot \mathbf{v}) && \text{dot distributes} \\
&= u_p \cdot 1 + v_p \cdot 0 = u_p && \|\mathbf{u}\| = 1,\ \mathbf{u} \cdot \mathbf{v} = 0
\end{aligned}$$

- $\mathbf{u} \cdot \mathbf{u} = 1$ keeps u_p; $\mathbf{u} \cdot \mathbf{v} = 0$ removes v_p
- Not orthonormal: $\mathbf{u} \cdot \mathbf{b} \neq u_p$, the recipe breaks

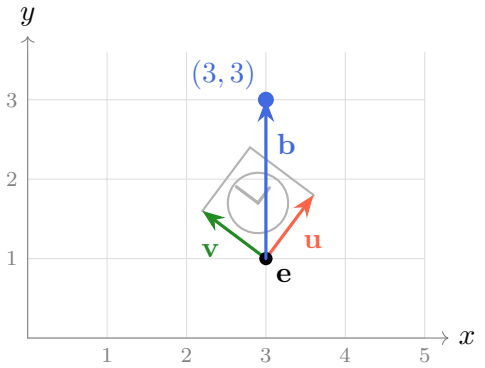
Quick check 1: find $(3, 3)$ in clock coordinates



$\mathbf{e} = (3, 1), \quad \mathbf{u} = (0.6,\ 0.8), \quad \mathbf{v} = (-0.8,\ 0.6)$

- Same clock frame as before
- What is its clock address (u_p, v_p)?

Quick check 1: find $(3, 3)$ in clock coordinates

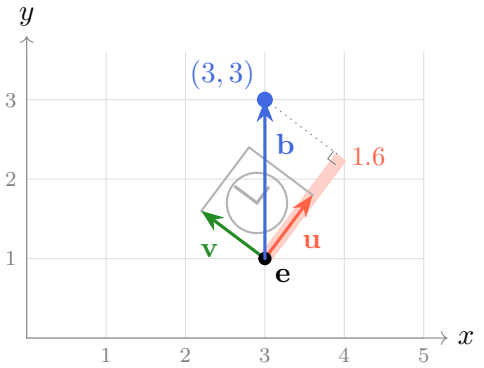


$\mathbf{e} = (3, 1), \quad \mathbf{u} = (0.6,\ 0.8), \quad \mathbf{v} = (-0.8,\ 0.6)$

- Same clock frame as before
- What is its clock address (u_p, v_p)?

$\mathbf{b} = (3, 3) - (3, 1) = (0,\ 2)$

Quick check 1: find $(3, 3)$ in clock coordinates



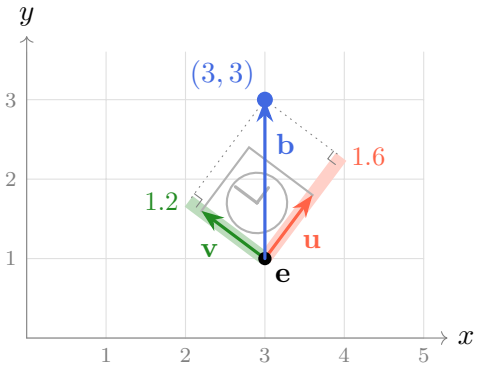
$\mathbf{e} = (3, 1), \quad \mathbf{u} = (0.6,\ 0.8), \quad \mathbf{v} = (-0.8,\ 0.6)$

- Same clock frame as before
- What is its clock address (u_p, v_p)?

$$\mathbf{b} = (3,3) - (3,1) = (0,\ 2)$$

$$\mathbf{u} \cdot \mathbf{b} = 1.6$$ shadow of $\mathbf{b}$ on the $\mathbf{u}$ line

Quick check 1: find $(3, 3)$ in clock coordinates



$\mathbf{e} = (3, 1), \quad \mathbf{u} = (0.6,\ 0.8), \quad \mathbf{v} = (-0.8,\ 0.6)$

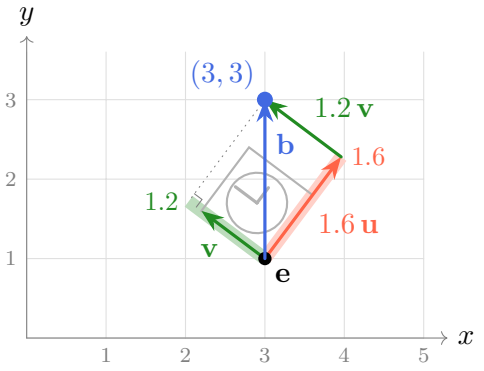
- Same clock frame as before
- What is its clock address (u_p, v_p)?

$$\mathbf{b} = (3, 3) - (3, 1) = (0,\ 2)$$

$$\mathbf{u} \cdot \mathbf{b} = 1.6 \quad \text{shadow of } \mathbf{b} \text{ on the } \mathbf{u} \text{ line}$$

$$\mathbf{v} \cdot \mathbf{b} = 1.2 \quad \text{shadow of } \mathbf{b} \text{ on the } \mathbf{v} \text{ line}$$

Quick check 1: find $(3, 3)$ in clock coordinates



$\mathbf{e} = (3, 1), \quad \mathbf{u} = (0.6,\ 0.8), \quad \mathbf{v} = (-0.8,\ 0.6)$

- Same clock frame as before
- What is its clock address (u_p, v_p)?

$$\mathbf{b} = (3, 3) - (3, 1) = (0,\ 2)$$

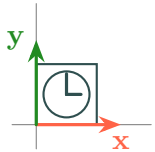
$$\mathbf{u} \cdot \mathbf{b} = 1.6 \qquad \text{shadow of } \mathbf{b} \text{ on the } \mathbf{u} \text{ line}$$

$$\mathbf{v} \cdot \mathbf{b} = 1.2 \qquad \text{shadow of } \mathbf{b} \text{ on the } \mathbf{v} \text{ line}$$

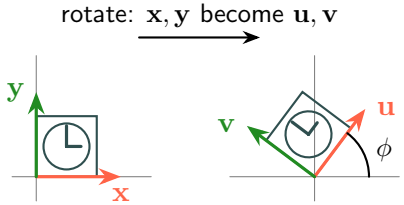
$$(u_p, v_p) = (1.6,\ 1.2)$$

- Check: walk $1.6\,\mathbf{u}$, then $1.2\,\mathbf{v}$
- Both > 1: outside the clock

Frame to world: rotate, then translate



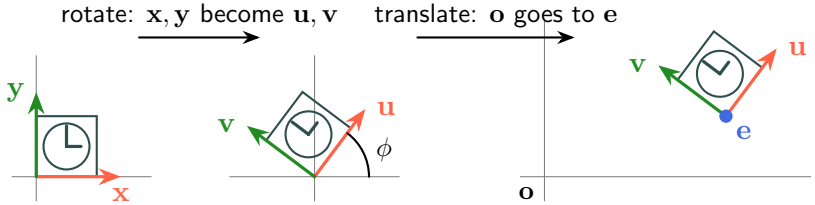
Frame to world: rotate, then translate



$$\underbrace{\begin{bmatrix} x_u & x_v & 0 \\ y_u & y_v & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{\text{rotate}}$$

- Rotate by ϕ: columns $\mathbf{u} = (\cos\phi, \sin\phi)$, $\mathbf{v} = (-\sin\phi, \cos\phi)$, as in lecture 7

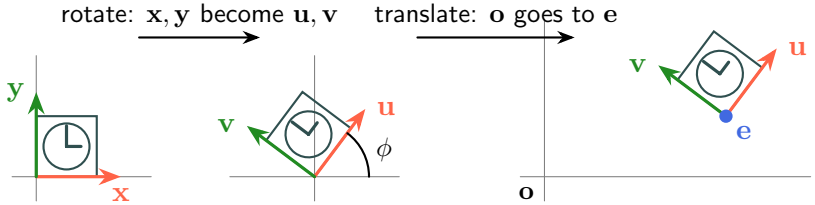
Frame to world: rotate, then translate



$$\underbrace{\begin{bmatrix} 1 & 0 & x_e \\ 0 & 1 & y_e \\ 0 & 0 & 1 \end{bmatrix}}_{\text{translate}} \underbrace{\begin{bmatrix} x_u & x_v & 0 \\ y_u & y_v & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{\text{rotate}}$$

- Rotate by ϕ: columns $\mathbf{u} = (\cos\phi, \sin\phi)$, $\mathbf{v} = (-\sin\phi, \cos\phi)$, as in lecture 7
- Translate: $\mathbf{o}$ goes to $\mathbf{e} = (x_e, y_e)$: the last column

Frame to world: rotate, then translate

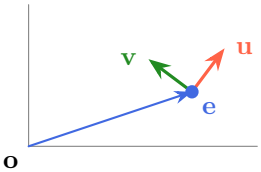


$$\underbrace{\begin{bmatrix} x_p \\ y_p \\ 1 \end{bmatrix}}_{\mathbf{p}_{xy}} = \underbrace{\begin{bmatrix} 1 & 0 & x_e \\ 0 & 1 & y_e \\ 0 & 0 & 1 \end{bmatrix}}_{\text{translate}} \underbrace{\begin{bmatrix} x_u & x_v & 0 \\ y_u & y_v & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{\text{rotate}} \underbrace{\begin{bmatrix} u_p \\ v_p \\ 1 \end{bmatrix}}_{\mathbf{p}_{uv}}$$

- Rotate by ϕ: columns $\mathbf{u} = (\cos\phi, \sin\phi)$, $\mathbf{v} = (-\sin\phi, \cos\phi)$, as in lecture 7
- Translate: $\mathbf{o}$ goes to $\mathbf{e} = (x_e, y_e)$: the last column
- Rightmost matrix acts first: clock numbers in, world numbers out

Multiply: $\mathbf{u}$, $\mathbf{v}$, $\mathbf{e}$ land in the columns

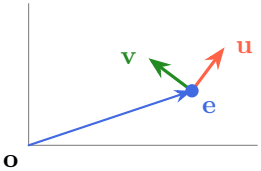
$$\begin{bmatrix} 1 & 0 & x_e \\ 0 & 1 & y_e \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_u & x_v & 0 \\ y_u & y_v & 0 \\ 0 & 0 & 1 \end{bmatrix}$$



$\mathbf{u}, \mathbf{v}, \mathbf{e}$ stored in world numbers

Multiply: $\mathbf{u}$, $\mathbf{v}$, $\mathbf{e}$ land in the columns

$$\begin{bmatrix} 1 & 0 & x_e \\ 0 & 1 & y_e \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_u & x_v & 0 \\ y_u & y_v & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} x_u & x_v & x_e \\ y_u & y_v & y_e \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{u} & \mathbf{v} & \mathbf{e} \\ 0 & 0 & 1 \end{bmatrix}$$



$\mathbf{u}, \mathbf{v}, \mathbf{e}$ stored in world numbers

- Translation on the left only fills the last column (lecture 8)

Multiply: $\mathbf{u}$, $\mathbf{v}$, $\mathbf{e}$ land in the columns

$$\begin{bmatrix} 1 & 0 & x_e \\ 0 & 1 & y_e \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_u & x_v & 0 \\ y_u & y_v & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} x_u & x_v & x_e \\ y_u & y_v & y_e \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{u} & \mathbf{v} & \mathbf{e} \\ 0 & 0 & 1 \end{bmatrix}$$

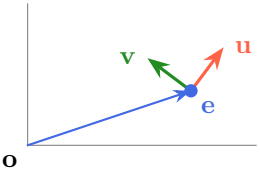
$$\mathbf{p}_{xy} = \begin{bmatrix} \mathbf{u} & \mathbf{v} & \mathbf{e} \\ 0 & 0 & 1 \end{bmatrix} \mathbf{p}_{uv}$$

frame-to-canonical matrix

$\mathbf{p}_{uv}$: point in clock numbers

$\mathbf{p}_{xy}$: same point in world numbers

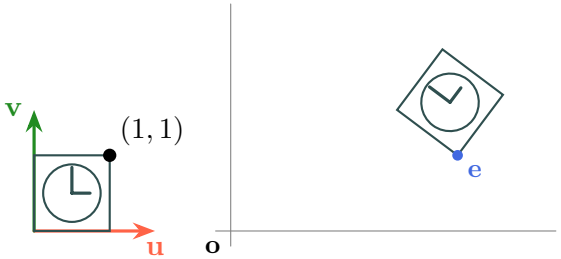
canonical = the world frame



$\mathbf{u}, \mathbf{v}, \mathbf{e}$ stored in world numbers

- Translation on the left only fills the last column (lecture 8)
- Columns straight from the frame: no trigonometry, no angles

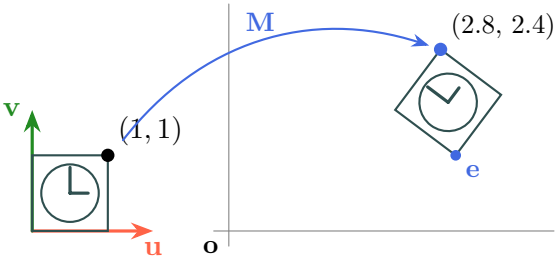
Worked: clock corner to world



$$\mathbf{M} = \begin{bmatrix} 0.6 & -0.8 & 3 \\ 0.8 & 0.6 & 1 \\ 0 & 0 & 1 \end{bmatrix}$$

- Columns straight from $\mathbf{u}, \mathbf{v}, \mathbf{e}$

Worked: clock corner to world



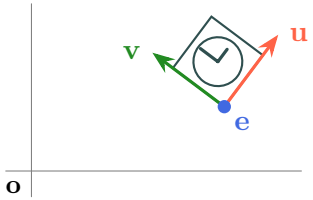
$$\mathbf{M} = \begin{bmatrix} 0.6 & -0.8 & 3 \\ 0.8 & 0.6 & 1 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{M} \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 2.8 \\ 2.4 \\ 1 \end{bmatrix}$$

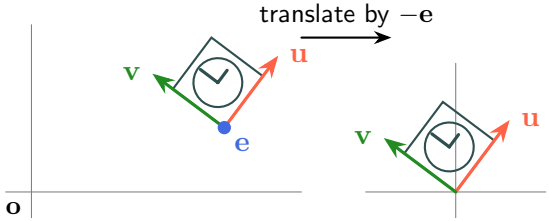
same as $\mathbf{e} + 1\mathbf{u} + 1\mathbf{v}$

- Columns straight from $\mathbf{u}, \mathbf{v}, \mathbf{e}$
- Corner lands at $(2.8,\ 2.4)$

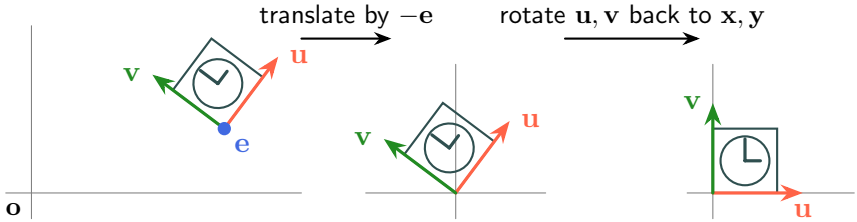
World to frame: translate, then rotate



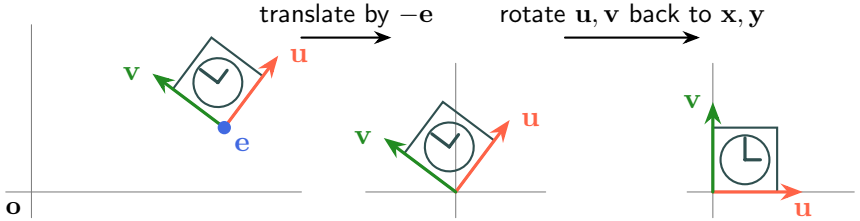
World to frame: translate, then rotate



World to frame: translate, then rotate



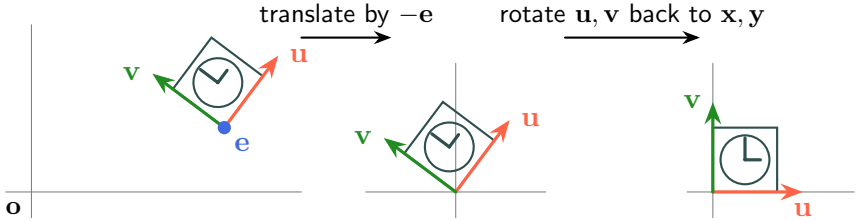
World to frame: translate, then rotate



$$\begin{bmatrix} u_p \\ v_p \\ 1 \end{bmatrix} = \underbrace{\begin{bmatrix} x_u & y_u & 0 \\ x_v & y_v & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{\text{rotate back}} \underbrace{\begin{bmatrix} 1 & 0 & -x_e \\ 0 & 1 & -y_e \\ 0 & 0 & 1 \end{bmatrix}}_{\text{translate back}} \begin{bmatrix} x_p \\ y_p \\ 1 \end{bmatrix}$$

- Undo both steps, in reverse order

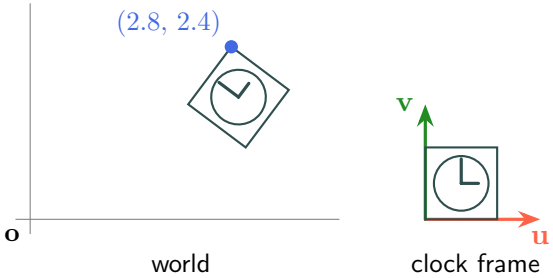
World to frame: translate, then rotate



$$\begin{bmatrix} u_p \\ v_p \\ 1 \end{bmatrix} = \underbrace{\begin{bmatrix} x_u & y_u & 0 \\ x_v & y_v & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{\text{rotate back}} \underbrace{\begin{bmatrix} 1 & 0 & -x_e \\ 0 & 1 & -y_e \\ 0 & 0 & 1 \end{bmatrix}}_{\text{translate back}} \begin{bmatrix} x_p \\ y_p \\ 1 \end{bmatrix}$$

- Undo both steps, in reverse order
- $\mathbf{p}_{uv} = \begin{bmatrix} \mathbf{u} & \mathbf{v} & \mathbf{e} \\ 0 & 0 & 1 \end{bmatrix}^{-1} \mathbf{p}_{xy}$: canonical-to-frame

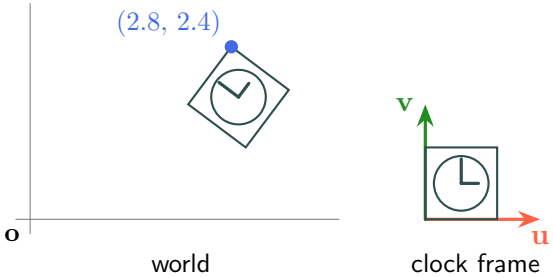
Worked: the corner comes back to $(1, 1)$



$$\mathbf{M}^{-1} = \begin{bmatrix} 0.6 & 0.8 & -2.6 \\ -0.8 & 0.6 & 1.8 \\ 0 & 0 & 1 \end{bmatrix}$$

- World numbers in, frame numbers out

Worked: the corner comes back to $(1, 1)$

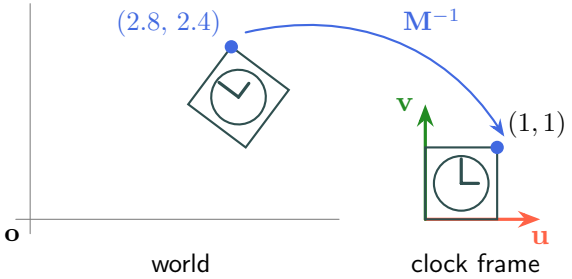


$$\mathbf{M}^{-1} = \begin{bmatrix} 0.6 & 0.8 & -2.6 \\ -0.8 & 0.6 & 1.8 \\ 0 & 0 & 1 \end{bmatrix}$$

rows: $\mathbf{u}$ and $\mathbf{v}$
last column: $-\mathbf{u} \cdot \mathbf{e} = -2.6$, $-\mathbf{v} \cdot \mathbf{e} = 1.8$

- World numbers in, frame numbers out

Worked: the corner comes back to $(1, 1)$



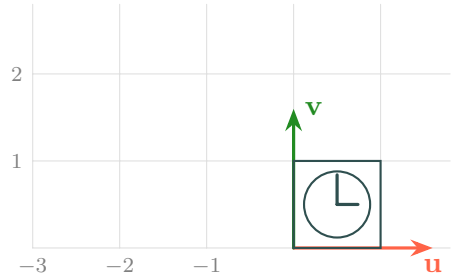
$$\mathbf{M}^{-1} = \begin{bmatrix} 0.6 & 0.8 & -2.6 \\ -0.8 & 0.6 & 1.8 \\ 0 & 0 & 1 \end{bmatrix}$$

rows: $\mathbf{u}$ and $\mathbf{v}$
last column: $-\mathbf{u} \cdot \mathbf{e} = -2.6$, $-\mathbf{v} \cdot \mathbf{e} = 1.8$

$$\mathbf{M}^{-1} \begin{bmatrix} 2.8 \\ 2.4 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

- World numbers in, frame numbers out
- Back to $(1, 1)$: the inverse works

Its columns: the world seen from the clock

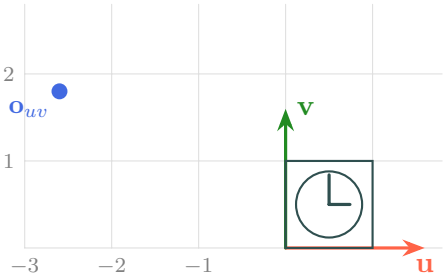


$\mathbf{M}^{-1}$ from the last slide:

$$\mathbf{M}^{-1} = \begin{bmatrix} \mathbf{x}_{uv} & \mathbf{y}_{uv} & \mathbf{o}_{uv} \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0.6 & 0.8 & -2.6 \\ -0.8 & 0.6 & 1.8 \\ 0 & 0 & 1 \end{bmatrix}$$

- Stand in the clock's frame: now the world is tilted

Its columns: the world seen from the clock

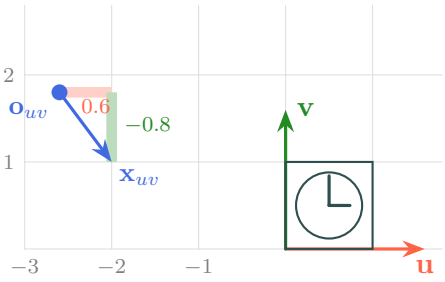


$\mathbf{M}^{-1}$ from the last slide:

$$\mathbf{M}^{-1} = \begin{bmatrix} \mathbf{x}_{uv} & \mathbf{y}_{uv} & \mathbf{o}_{uv} \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0.6 & 0.8 & -2.6 \\ -0.8 & 0.6 & 1.8 \\ 0 & 0 & 1 \end{bmatrix}$$

- Stand in the clock's frame: now the world is tilted
- Each column: a world arrow or origin, in clock numbers

Its columns: the world seen from the clock

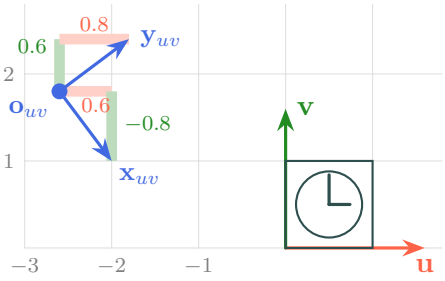


$\mathbf{M}^{-1}$ from the last slide:

$$\mathbf{M}^{-1} = \begin{bmatrix} \mathbf{x}_{uv} & \mathbf{y}_{uv} & \mathbf{o}_{uv} \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0.6 & 0.8 & -2.6 \\ -0.8 & 0.6 & 1.8 \\ 0 & 0 & 1 \end{bmatrix}$$

- Stand in the clock's frame: now the world is tilted
- Each column: a world arrow or origin, in clock numbers

Its columns: the world seen from the clock

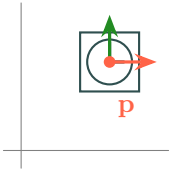


$\mathbf{M}^{-1}$ from the last slide:

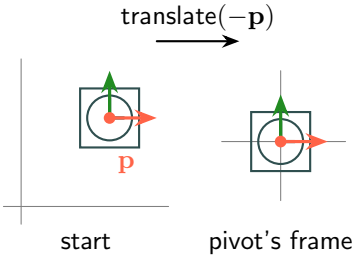
$$\mathbf{M}^{-1} = \begin{bmatrix} \mathbf{x}_{uv} & \mathbf{y}_{uv} & \mathbf{o}_{uv} \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0.6 & 0.8 & -2.6 \\ -0.8 & 0.6 & 1.8 \\ 0 & 0 & 1 \end{bmatrix}$$

- Stand in the clock's frame: now the world is tilted
- Each column: a world arrow or origin, in clock numbers
- Same rule as $\mathbf{M}$, viewpoint swapped

Rotate about a point: move, turn, move back

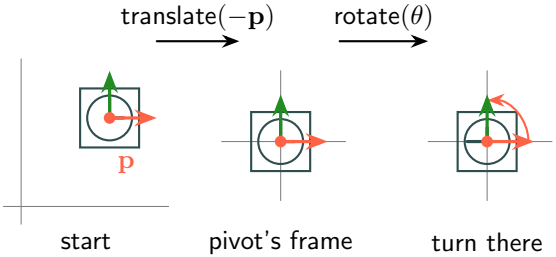


Rotate about a point: move, turn, move back



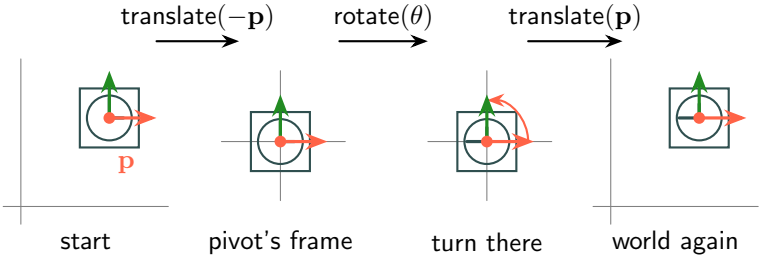
- A frame at the pivot: $\mathbf{e} = \mathbf{p},\ \mathbf{u}, \mathbf{v} = \mathbf{x}, \mathbf{y}$; into it with translate$(-\mathbf{p})$

Rotate about a point: move, turn, move back



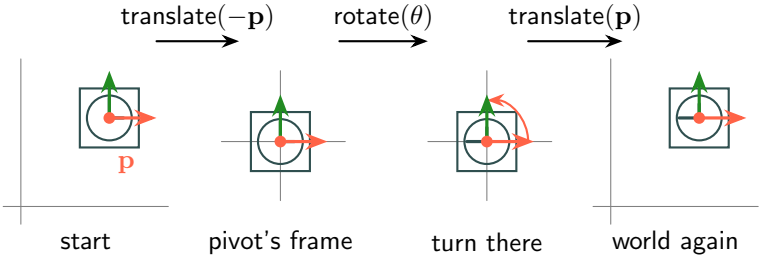
- A frame at the pivot: $\mathbf{e} = \mathbf{p}$, $\mathbf{u}, \mathbf{v} = \mathbf{x}, \mathbf{y}$; into it with translate$(-\mathbf{p})$
- rotate(θ) turns about the origin, and in this frame the origin is $\mathbf{p}$

Rotate about a point: move, turn, move back



- A frame at the pivot: $\mathbf{e} = \mathbf{p}$, $\mathbf{u}, \mathbf{v} = \mathbf{x}, \mathbf{y}$; into it with translate$(-\mathbf{p})$
- rotate(θ) turns about the origin, and in this frame the origin is $\mathbf{p}$

Rotate about a point: move, turn, move back



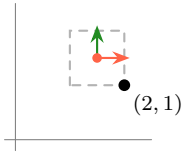
$$\mathbf{M} = \underbrace{\text{translate}(\mathbf{p})}_{\text{frame-to-canonical}} \; \text{rotate}(\theta) \; \underbrace{\text{translate}(-\mathbf{p})}_{\text{canonical-to-frame}}$$

- A frame at the pivot: $\mathbf{e} = \mathbf{p}$, $\mathbf{u}, \mathbf{v} = \mathbf{x}, \mathbf{y}$; into it with translate$(-\mathbf{p})$
- rotate(θ) turns about the origin, and in this frame the origin is $\mathbf{p}$
- Back to world numbers with translate$(\mathbf{p})$: the pivot stays put

Worked: a quarter turn about the clock's centre

$\mathbf{M} = \text{translate}(\mathbf{p})\ \text{rotate}(90°)\ \text{translate}(-\mathbf{p})$ frame at the pivot $\mathbf{p} = (1.5,\ 1.5)$; $\cos 90° = 0,\ \sin 90° = 1$

$$= \underbrace{\begin{bmatrix} 1 & 0 & 1.5 \\ 0 & 1 & 1.5 \\ 0 & 0 & 1 \end{bmatrix}}_{\text{frame-to-canonical}} \underbrace{\begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{\text{turn at the origin}} \underbrace{\begin{bmatrix} 1 & 0 & -1.5 \\ 0 & 1 & -1.5 \\ 0 & 0 & 1 \end{bmatrix}}_{\text{canonical-to-frame}}$$



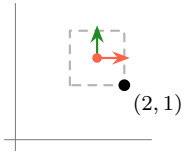
Worked: a quarter turn about the clock's centre

$\mathbf{M} = \text{translate}(\mathbf{p})\ \text{rotate}(90°)\ \text{translate}(-\mathbf{p})$ frame at the pivot $\mathbf{p} = (1.5,\ 1.5)$; $\cos 90° = 0,\ \sin 90° = 1$

$$= \underbrace{\begin{bmatrix} 1 & 0 & 1.5 \\ 0 & 1 & 1.5 \\ 0 & 0 & 1 \end{bmatrix}}_{\text{frame-to-canonical}} \underbrace{\begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{\text{turn at the origin}} \underbrace{\begin{bmatrix} 1 & 0 & -1.5 \\ 0 & 1 & -1.5 \\ 0 & 0 & 1 \end{bmatrix}}_{\text{canonical-to-frame}}$$

$$= \begin{bmatrix} 0 & -1 & 3 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

turn block unchanged; the last column is what keeps $\mathbf{p}$ in place



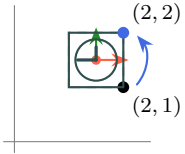
Worked: a quarter turn about the clock's centre

$\mathbf{M} = \text{translate}(\mathbf{p})\ \text{rotate}(90°)\ \text{translate}(-\mathbf{p})$ frame at the pivot $\mathbf{p} = (1.5,\ 1.5)$; $\cos 90° = 0,\ \sin 90° = 1$

$$= \underbrace{\begin{bmatrix} 1 & 0 & 1.5 \\ 0 & 1 & 1.5 \\ 0 & 0 & 1 \end{bmatrix}}_{\text{frame-to-canonical}} \underbrace{\begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{\text{turn at the origin}} \underbrace{\begin{bmatrix} 1 & 0 & -1.5 \\ 0 & 1 & -1.5 \\ 0 & 0 & 1 \end{bmatrix}}_{\text{canonical-to-frame}}$$

$$= \begin{bmatrix} 0 & -1 & 3 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$ turn block unchanged; the last column is what keeps $\mathbf{p}$ in place

$$\mathbf{M} \begin{bmatrix} 2 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 2 \\ 2 \\ 1 \end{bmatrix}, \qquad \mathbf{M} \begin{bmatrix} 1.5 \\ 1.5 \\ 1 \end{bmatrix} = \begin{bmatrix} 1.5 \\ 1.5 \\ 1 \end{bmatrix}$$



- Corner $(2, 1)$ turns to $(2, 2)$, pivot stays put; rotate$(90°)$ alone sends $(2, 1)$ to $(-1, 2)$

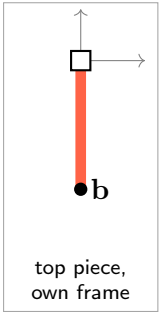
Recap: changing coordinates

$$\mathbf{p}_{xy} = \begin{bmatrix} \mathbf{u} & \mathbf{v} & \mathbf{e} \\ 0 & 0 & 1 \end{bmatrix} \mathbf{p}_{uv}$$

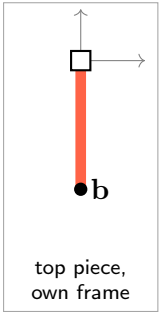
Scene graphs

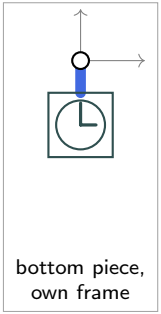
Parts hang on parts

A pendulum: two pieces, two frames



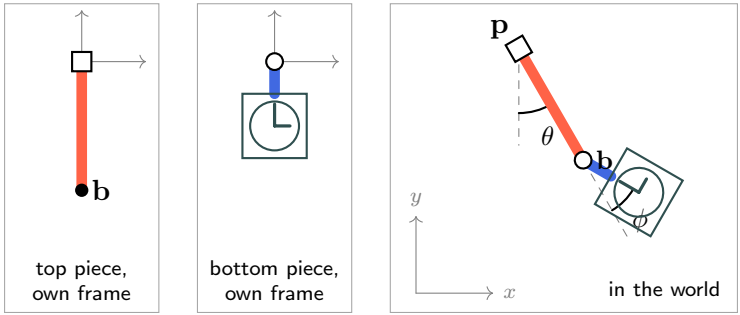
A pendulum: two pieces, two frames





- Each piece drawn once, in its own frame, hinge at the origin

A pendulum: two pieces, two frames



- Each piece drawn once, in its own frame, hinge at the origin
- Free choices θ, ϕ, $\mathbf{p}$; wanted: one local-to-world matrix per piece

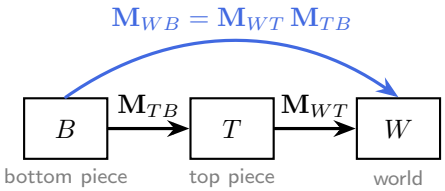
Naming a matrix by its two frames

- Frames get letters: W world, T top piece, B bottom piece

Naming a matrix by its two frames

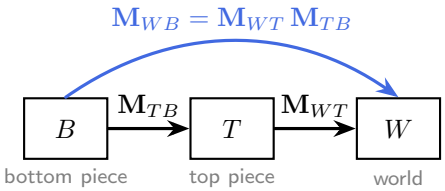
- Frames get letters: W world, T top piece, B bottom piece
- Subscript reads *to, from*: $\mathbf{p}_T = \mathbf{M}_{TB}\,\mathbf{p}_B$, bottom numbers in, top numbers out

Naming a matrix by its two frames



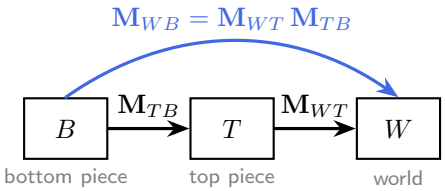
- Frames get letters: W world, T top piece, B bottom piece
- Subscript reads *to, from*: $\mathbf{p}_T = \mathbf{M}_{TB}\,\mathbf{p}_B$, bottom numbers in, top numbers out
- Chains cancel inner letters: $\mathbf{p}_W = \mathbf{M}_{WT}\mathbf{M}_{TB}\,\mathbf{p}_B$, so $\mathbf{M}_{WB} = \mathbf{M}_{WT}\mathbf{M}_{TB}$

Naming a matrix by its two frames



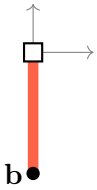
- Frames get letters: W world, T top piece, B bottom piece
- Subscript reads *to, from*: $\mathbf{p}_T = \mathbf{M}_{TB}\,\mathbf{p}_B$, bottom numbers in, top numbers out
- Chains cancel inner letters: $\mathbf{p}_W = \mathbf{M}_{WT}\mathbf{M}_{TB}\,\mathbf{p}_B$, so $\mathbf{M}_{WB} = \mathbf{M}_{WT}\mathbf{M}_{TB}$
- Inverse swaps the letters: $\mathbf{M}_{BT} = \mathbf{M}_{TB}^{-1}$

Naming a matrix by its two frames



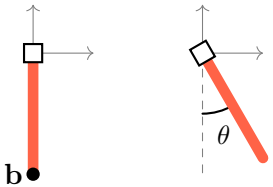
- Frames get letters: W world, T top piece, B bottom piece
- Subscript reads *to, from*: $\mathbf{p}_T = \mathbf{M}_{TB}\,\mathbf{p}_B$, bottom numbers in, top numbers out
- Chains cancel inner letters: $\mathbf{p}_W = \mathbf{M}_{WT}\mathbf{M}_{TB}\,\mathbf{p}_B$, so $\mathbf{M}_{WB} = \mathbf{M}_{WT}\mathbf{M}_{TB}$
- Inverse swaps the letters: $\mathbf{M}_{BT} = \mathbf{M}_{TB}^{-1}$
- Section 2, clock frame C: $\mathbf{M}_{WC} = \begin{bmatrix} \mathbf{u} & \mathbf{v} & \mathbf{e} \\ 0 & 0 & 1 \end{bmatrix}$ is frame-to-canonical, $\mathbf{M}_{CW}$ canonical-to-frame

Top piece: turn by θ, then move to $\mathbf{p}$



own frame,
hinge at origin

Top piece: turn by θ, then move to $\mathbf{p}$



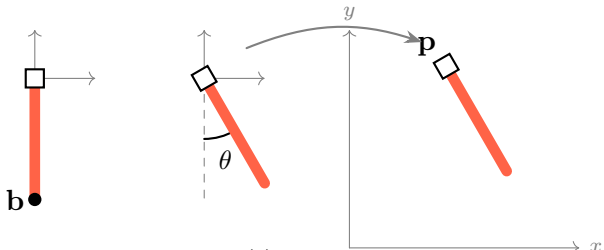
own frame, hinge at origin

rotate(θ): about the hinge

$\mathbf{R}_\theta = \text{rotate}(\theta)$

- Hinge at the origin, so rotate(θ) turns about the hinge: no sandwich

Top piece: turn by θ, then move to $\mathbf{p}$



own frame, hinge at origin

rotate(θ): about the hinge

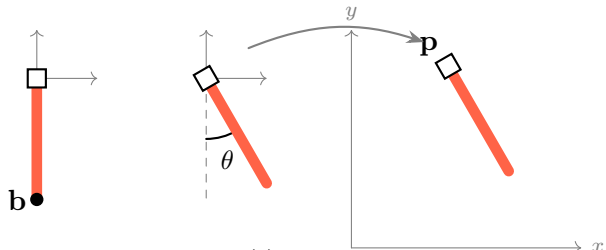
translate($\mathbf{p}$): hinge to $\mathbf{p}$, in the world

$\mathbf{R}_\theta = \text{rotate}(\theta)$

$\mathbf{T}_\mathbf{p} = \text{translate}(\mathbf{p})$

- Hinge at the origin, so rotate(θ) turns about the hinge: no sandwich

Top piece: turn by θ, then move to $\mathbf{p}$



own frame, hinge at origin

rotate(θ): about the hinge

translate($\mathbf{p}$): hinge to $\mathbf{p}$, in the world

$\mathbf{R}_\theta = \text{rotate}(\theta)$

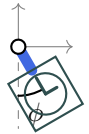
$\mathbf{T}_\mathbf{p} = \text{translate}(\mathbf{p})$

$\mathbf{M}_{WT} = \mathbf{T}_\mathbf{p}\,\mathbf{R}_\theta$ top to world

the top piece's frame-to-canonical matrix

- Hinge at the origin, so rotate(θ) turns about the hinge: no sandwich

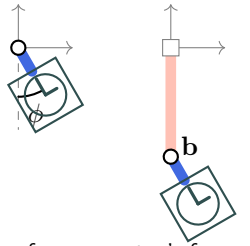
Bottom piece rides along with the top



own frame:
rotate(ϕ)

$$\mathbf{R}_\phi = \text{rotate}(\phi)$$

Bottom piece rides along with the top



own frame:
rotate(ϕ)

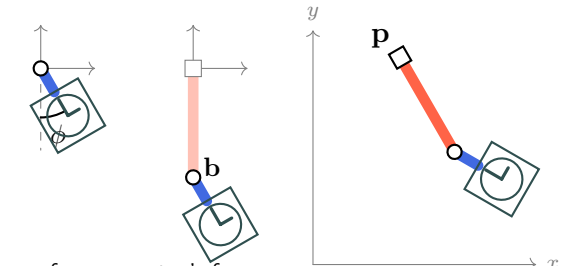
top's frame:
translate($\mathbf{b}$)

$$\mathbf{R}_\phi = \text{rotate}(\phi)$$

$$\mathbf{T}_\mathbf{b} = \text{translate}(\mathbf{b})$$

$$\mathbf{M}_{TB} = \mathbf{T}_\mathbf{b}\,\mathbf{R}_\phi$$ bottom to top

Bottom piece rides along with the top



$\mathbf{R}_\phi = \text{rotate}(\phi)$

$\mathbf{T}_\mathbf{b} = \text{translate}(\mathbf{b})$

$\mathbf{M}_{TB} = \mathbf{T}_\mathbf{b}\,\mathbf{R}_\phi$ bottom to top

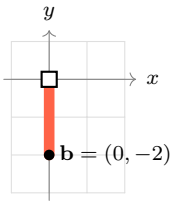
$\mathbf{M}_{WB} = \mathbf{M}_{WT}\,\mathbf{M}_{TB}$ bottom to world

- $\mathbf{b}$ lives in the top piece's frame; $\mathbf{M}_{WT}$ carries that frame, and the bottom with it

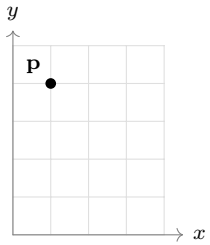
Worked: the top piece's matrix

Givens: $\theta = 90°$, hinge $\mathbf{p} = (1, 4)$, rod length 2, so $\mathbf{b} = (0, -2)$ in the top piece's frame

$$\mathbf{R}_\theta = \text{rotate}(90°) = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$



top piece's own frame



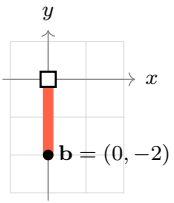
world

Worked: the top piece's matrix

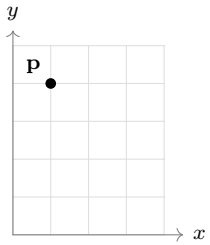
Givens: $\theta = 90°$, hinge $\mathbf{p} = (1, 4)$, rod length 2, so $\mathbf{b} = (0, -2)$ in the top piece's frame

$$\mathbf{R}_\theta = \text{rotate}(90°) = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{T}_\mathbf{p} = \text{translate}(1, 4) = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 4 \\ 0 & 0 & 1 \end{bmatrix}$$



top piece's own frame



world

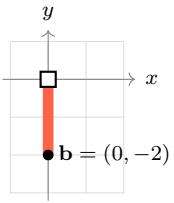
Worked: the top piece's matrix

Givens: $\theta = 90°$, hinge $\mathbf{p} = (1, 4)$, rod length 2, so $\mathbf{b} = (0, -2)$ in the top piece's frame

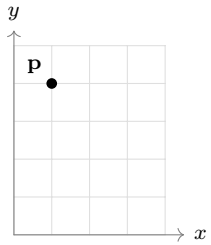
$$\mathbf{R}_\theta = \text{rotate}(90°) = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{T}_\mathbf{p} = \text{translate}(1, 4) = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 4 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{M}_{WT} = \mathbf{T}_\mathbf{p}\mathbf{R}_\theta = \begin{bmatrix} 0 & -1 & 1 \\ 1 & 0 & 4 \\ 0 & 0 & 1 \end{bmatrix}$$



top piece's own frame



world

Worked: the top piece's matrix

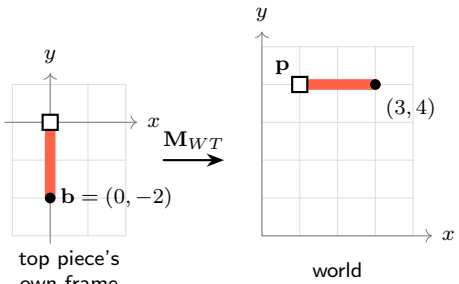
Givens: $\theta = 90°$, hinge $\mathbf{p} = (1, 4)$, rod length 2, so $\mathbf{b} = (0, -2)$ in the top piece's frame

$$\mathbf{R}_\theta = \text{rotate}(90°) = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{T}_\mathbf{p} = \text{translate}(1, 4) = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 4 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{M}_{WT} = \mathbf{T}_\mathbf{p}\mathbf{R}_\theta = \begin{bmatrix} 0 & -1 & 1 \\ 1 & 0 & 4 \\ 0 & 0 & 1 \end{bmatrix}$$

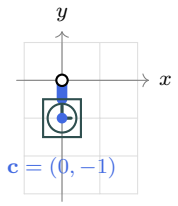
$$\mathbf{M}_{WT}\,\mathbf{b} = \mathbf{M}_{WT} \begin{bmatrix} 0 \\ -2 \\ 1 \end{bmatrix} = \begin{bmatrix} 3 \\ 4 \\ 1 \end{bmatrix}$$



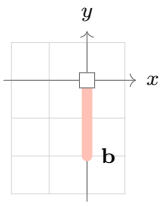
Worked: the bottom piece's local matrix

Givens: $\phi = -90°$, hinge at $\mathbf{b} = (0, -2)$, clock centre $\mathbf{c} = (0, -1)$ in the bottom piece's frame

$$\mathbf{R}_\phi = \text{rotate}(-90°) = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$



bottom piece's own frame



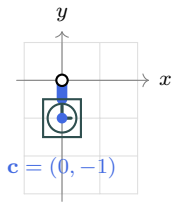
top piece's frame

Worked: the bottom piece's local matrix

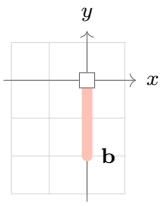
Givens: $\phi = -90°$, hinge at $\mathbf{b} = (0, -2)$, clock centre $\mathbf{c} = (0, -1)$ in the bottom piece's frame

$$\mathbf{R}_\phi = \text{rotate}(-90°) = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{T}_\mathbf{b} = \text{translate}(0, -2) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & -2 \\ 0 & 0 & 1 \end{bmatrix}$$



bottom piece's own frame



top piece's frame

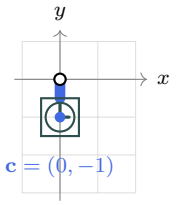
Worked: the bottom piece's local matrix

Givens: $\phi = -90°$, hinge at $\mathbf{b} = (0, -2)$, clock centre $\mathbf{c} = (0, -1)$ in the bottom piece's frame

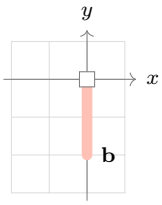
$$\mathbf{R}_\phi = \text{rotate}(-90°) = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{T}_\mathbf{b} = \text{translate}(0, -2) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & -2 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{M}_{TB} = \mathbf{T}_\mathbf{b}\mathbf{R}_\phi = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & -2 \\ 0 & 0 & 1 \end{bmatrix}$$



bottom piece's own frame



top piece's frame

Worked: the bottom piece's local matrix

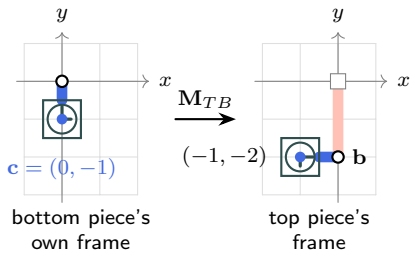
Givens: $\phi = -90°$, hinge at $\mathbf{b} = (0, -2)$, clock centre $\mathbf{c} = (0, -1)$ in the bottom piece's frame

$$\mathbf{R}_\phi = \text{rotate}(-90°) = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{T}_\mathbf{b} = \text{translate}(0, -2) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & -2 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{M}_{TB} = \mathbf{T}_\mathbf{b}\mathbf{R}_\phi = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & -2 \\ 0 & 0 & 1 \end{bmatrix}$$

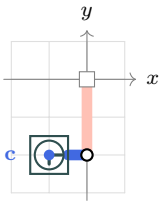
$$\mathbf{M}_{TB}\,\mathbf{c} = \mathbf{M}_{TB} \begin{bmatrix} 0 \\ -1 \\ 1 \end{bmatrix} = \begin{bmatrix} -1 \\ -2 \\ 1 \end{bmatrix}$$



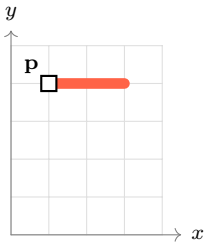
Worked: follow the clock to the world

$\mathbf{c} = (-1, -2)$ in the top piece's frame, from the last slide; $\mathbf{M}_{WT}$ takes that frame to the world

$$\mathbf{M}_{WB} = \mathbf{M}_{WT}\mathbf{M}_{TB} = \begin{bmatrix} 0 & -1 & 1 \\ 1 & 0 & 4 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & -2 \\ 0 & 0 & 1 \end{bmatrix}$$



top piece's frame



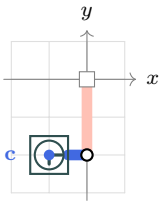
world

Worked: follow the clock to the world

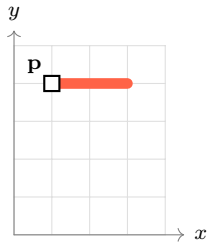
$\mathbf{c} = (-1, -2)$ in the top piece's frame, from the last slide; $\mathbf{M}_{WT}$ takes that frame to the world

$$\mathbf{M}_{WB} = \mathbf{M}_{WT}\mathbf{M}_{TB} = \begin{bmatrix} 0 & -1 & 1 \\ 1 & 0 & 4 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & -2 \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & 3 \\ 0 & 1 & 4 \\ 0 & 0 & 1 \end{bmatrix}$$ a pure shift: $+90°$ and $-90°$ cancel



top piece's frame



world

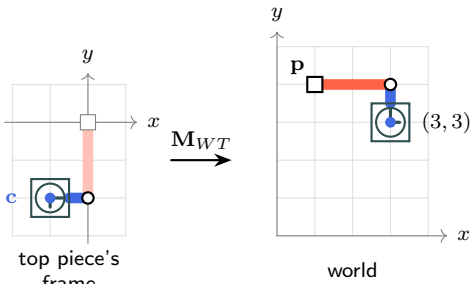
Worked: follow the clock to the world

$\mathbf{c} = (-1, -2)$ in the top piece's frame, from the last slide; $\mathbf{M}_{WT}$ takes that frame to the world

$$\mathbf{M}_{WB} = \mathbf{M}_{WT}\mathbf{M}_{TB} = \begin{bmatrix} 0 & -1 & 1 \\ 1 & 0 & 4 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & -2 \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & 3 \\ 0 & 1 & 4 \\ 0 & 0 & 1 \end{bmatrix}$$ a pure shift: $+90°$ and $-90°$ cancel

$$\mathbf{M}_{WB}\,\mathbf{c} = \mathbf{M}_{WB} \begin{bmatrix} 0 \\ -1 \\ 1 \end{bmatrix} = \begin{bmatrix} 3 \\ 3 \\ 1 \end{bmatrix}$$



Worked: follow the clock to the world

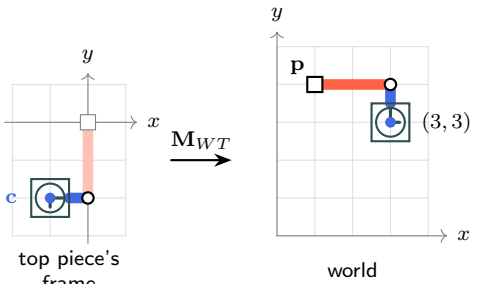
$\mathbf{c} = (-1, -2)$ in the top piece's frame, from the last slide; $\mathbf{M}_{WT}$ takes that frame to the world

$$\mathbf{M}_{WB} = \mathbf{M}_{WT}\mathbf{M}_{TB} = \begin{bmatrix} 0 & -1 & 1 \\ 1 & 0 & 4 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & -2 \\ 0 & 0 & 1 \end{bmatrix}$$

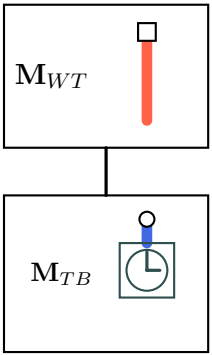
$$= \begin{bmatrix} 1 & 0 & 3 \\ 0 & 1 & 4 \\ 0 & 0 & 1 \end{bmatrix}$$ a pure shift: $+90°$ and $-90°$ cancel

$$\mathbf{M}_{WB}\,\mathbf{c} = \mathbf{M}_{WB} \begin{bmatrix} 0 \\ -1 \\ 1 \end{bmatrix} = \begin{bmatrix} 3 \\ 3 \\ 1 \end{bmatrix}$$

check: $\mathbf{M}_{WT} \begin{bmatrix} -1 \\ -2 \\ 1 \end{bmatrix} = \begin{bmatrix} 3 \\ 3 \\ 1 \end{bmatrix}$ too

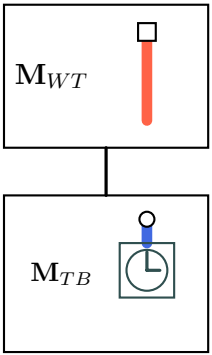


Scene graph: a tree of local matrices



- Node: a piece plus its local matrix

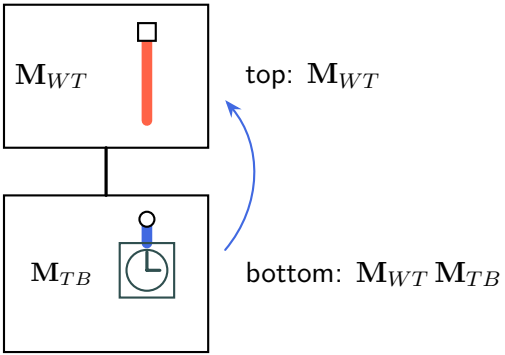
Scene graph: a tree of local matrices



top: $\mathbf{M}_{WT}$

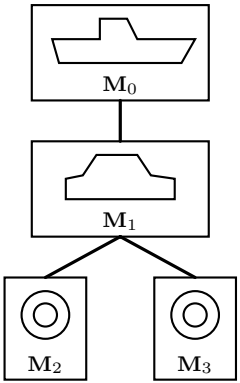
- Node: a piece plus its local matrix
- Top node: just its own matrix

Scene graph: a tree of local matrices

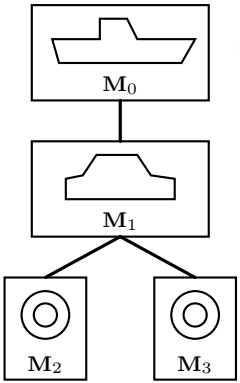


- Node: a piece plus its local matrix
- Top node: just its own matrix
- Multiply up the chain to the root

Multiply matrices along the path from root

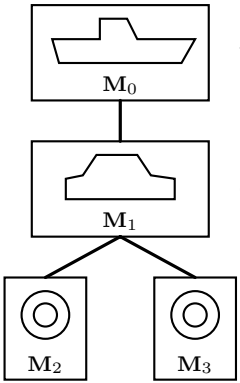


Multiply matrices along the path from root



ferry: $\mathbf{M}_0$

Multiply matrices along the path from root

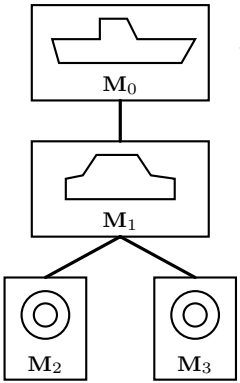


ferry: $\mathbf{M}_0$

car body: $\mathbf{M}_0\mathbf{M}_1$

- Car moves freely on the deck

Multiply matrices along the path from root



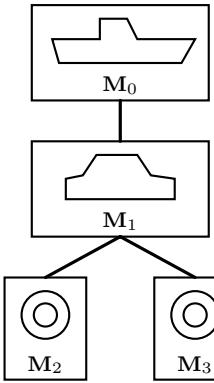
ferry: $\mathbf{M}_0$

car body: $\mathbf{M}_0\mathbf{M}_1$

left wheel: $\mathbf{M}_0\mathbf{M}_1\mathbf{M}_2$

- Car moves freely on the deck
- Wheels move relative to the car

Multiply matrices along the path from root



ferry: $\mathbf{M}_0$

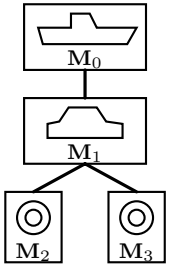
car body: $\mathbf{M}_0\mathbf{M}_1$

left wheel: $\mathbf{M}_0\mathbf{M}_1\mathbf{M}_2$

right wheel: $\mathbf{M}_0\mathbf{M}_1\mathbf{M}_3$

- Car moves freely on the deck
- Wheels move relative to the car
- Product along the path from root

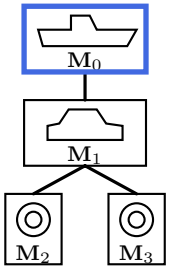
Traverse with a matrix stack: push, draw, children, pop



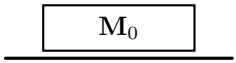
active product:

function traverse(node)
  push($\mathbf{M}_{\text{local}}$)
  draw node with the stack's product
  for each child: traverse(child)
  pop()

Traverse with a matrix stack: push, draw, children, pop



push $\mathbf{M}_0$, draw ferry



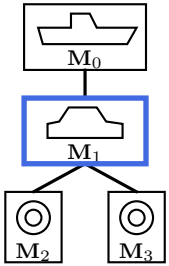
active product:

$\mathbf{M}_0$

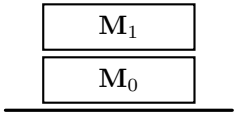
function traverse(node)
  push($\mathbf{M}_{\text{local}}$)
  draw node with the stack's product
  for each child: traverse(child)
  pop()

- push puts the new matrix on the right: most local acts first

Traverse with a matrix stack: push, draw, children, pop



push $\mathbf{M}_1$, draw car



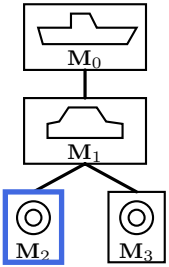
active product:

$\mathbf{M}_0\mathbf{M}_1$

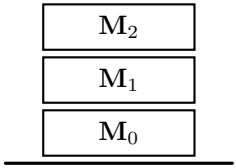
function traverse(node)
  push($\mathbf{M}_{\text{local}}$)
  draw node with the stack's product
  for each child: traverse(child)
  pop()

- push puts the new matrix on the right: most local acts first

Traverse with a matrix stack: push, draw, children, pop



push $\mathbf{M}_2$, draw left wheel



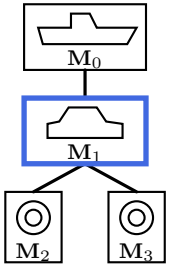
active product:

$\mathbf{M}_0\mathbf{M}_1\mathbf{M}_2$

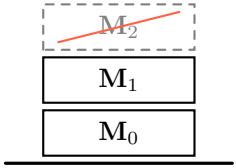
function traverse(node)
  push($\mathbf{M}_{\text{local}}$)
  draw node with the stack's product
  for each child: traverse(child)
  pop()

- push puts the new matrix on the right: most local acts first

Traverse with a matrix stack: push, draw, children, pop



pop: wheel done



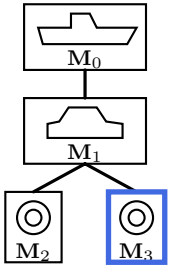
active product:

$\mathbf{M}_0\mathbf{M}_1$

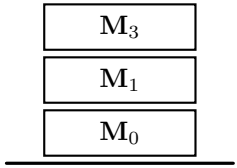
function traverse(node)
  push($\mathbf{M}_{\text{local}}$)
  draw node with the stack's product
  for each child: traverse(child)
  pop()

- push puts the new matrix on the right: most local acts first
- pop undoes the push: back to the parent's product

Traverse with a matrix stack: push, draw, children, pop



push $\mathbf{M}_3$, draw right wheel



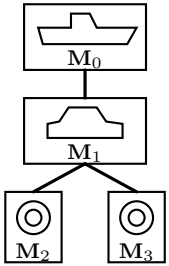
active product:

$\mathbf{M}_0\mathbf{M}_1\mathbf{M}_3$

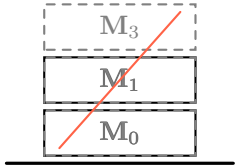
function traverse(node)
  push($\mathbf{M}_{\text{local}}$)
  draw node with the stack's product
  for each child: traverse(child)
  pop()

- push puts the new matrix on the right: most local acts first
- pop undoes the push: back to the parent's product

Traverse with a matrix stack: push, draw, children, pop



pop, pop, pop: empty



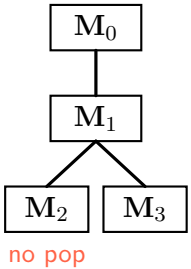
active product:

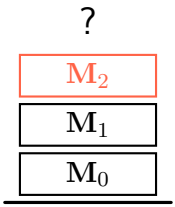
identity

function traverse(node)
  push($\mathbf{M}_{\text{local}}$)
  draw node with the stack's product
  for each child: traverse(child)
  pop()

- push puts the new matrix on the right: most local acts first
- pop undoes the push: back to the parent's product
- Each product computed once, on the way down

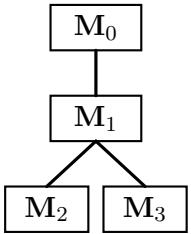
Quick check 2: a pop goes missing



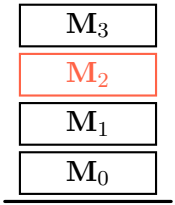


- Left wheel forgets its pop()
- Which matrix draws the right wheel?

Quick check 2: a pop goes missing

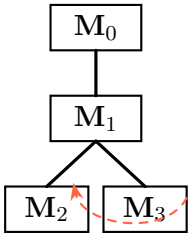


$$\mathbf{M} = \mathbf{M}_0\mathbf{M}_1\mathbf{M}_2\mathbf{M}_3$$

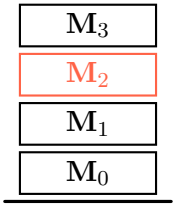


- Left wheel forgets its pop()
- Which matrix draws the right wheel?

Quick check 2: a pop goes missing

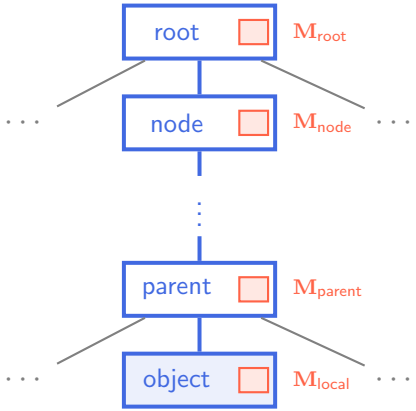


$$\mathbf{M} = \mathbf{M}_0\mathbf{M}_1\mathbf{M}_2\mathbf{M}_3$$



- Left wheel forgets its pop()
- Which matrix draws the right wheel?
- Right wheel rides on the left wheel

Recap: scene graphs



local matrix: node frame to parent frame;
the root's parent is the world

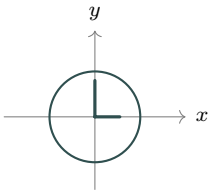
$$\mathbf{M}_{\text{object}} = \mathbf{M}_{\text{root}}\,\mathbf{M}_{\text{node}} \cdots \mathbf{M}_{\text{parent}}\,\mathbf{M}_{\text{local}}$$

each node stores one matrix, its own frame seen from its parent;
an object is drawn with the product down its path, root on the left

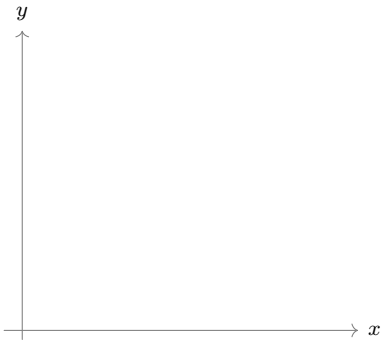
Instancing in ray tracing

Store the object once, plus a matrix

Instancing: one shape stored, placed many times

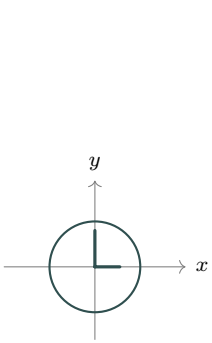


base object: unit circle
stored once

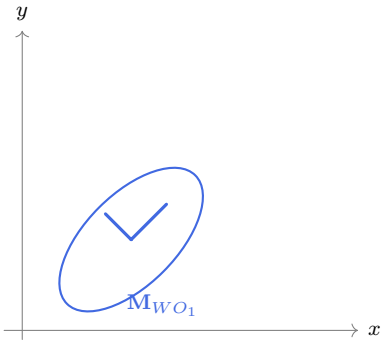


world

Instancing: one shape stored, placed many times



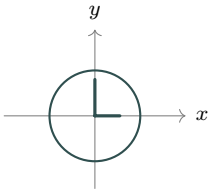
base object: unit circle
stored once



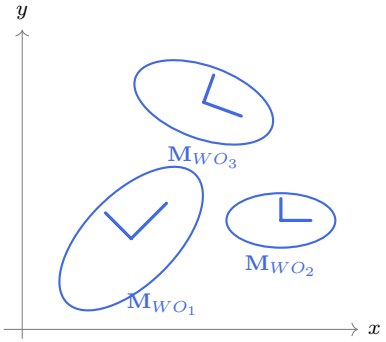
world

- An *instance* = a reference to the base shape + its own matrix $\mathbf{M}_{WO}$, object frame to world

Instancing: one shape stored, placed many times



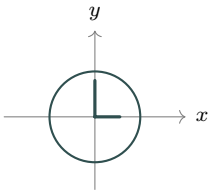
base object: unit circle stored once



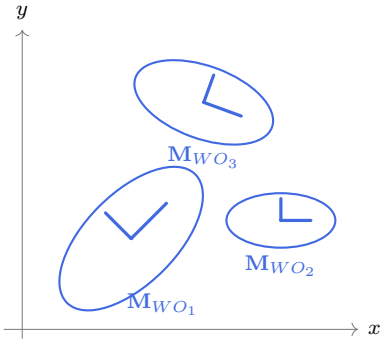
world

- An *instance* = a reference to the base shape + its own matrix $\mathbf{M}_{WO}$, object frame to world
- Instancing: the shape is stored once; each use in the scene is just a matrix

Instancing: one shape stored, placed many times



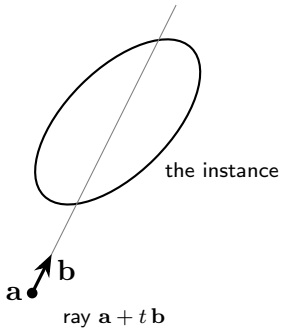
base object: unit circle
stored once



world

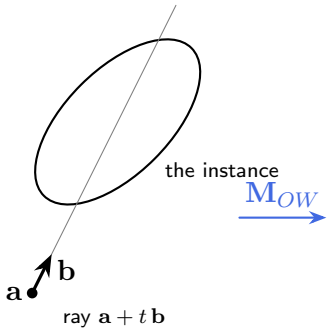
- An *instance* = a reference to the base shape + its own matrix $\mathbf{M}_{WO}$, object frame to world
- Instancing: the shape is stored once; each use in the scene is just a matrix
- $\mathbf{M}_{WO_1} =$ translate$(2.4,\ 2)$ rotate$(45°)$ scale$(2, 1)$
- The ellipses are never built: hit them through the circle

Hit test: bring the ray into the object's frame



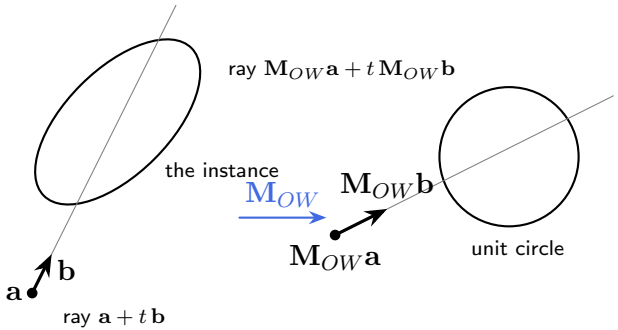
- The ray comes in world numbers, $\mathbf{a} + t\,\mathbf{b}$; no ellipse test

Hit test: bring the ray into the object's frame



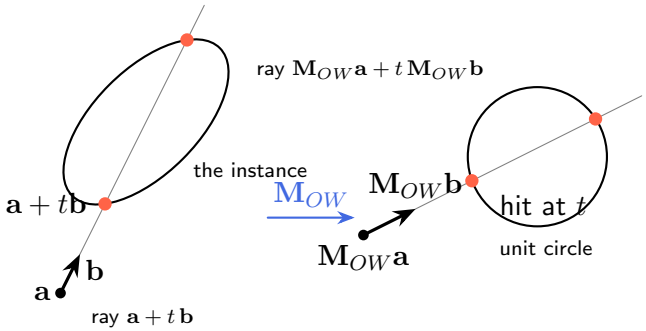
- The ray comes in world numbers, $\mathbf{a} + t\,\mathbf{b}$; no ellipse test
- So move the ray, not the shape: $\mathbf{M}_{OW}$ on $\mathbf{a}$ (a point) and on $\mathbf{b}$ (a direction)

Hit test: bring the ray into the object's frame



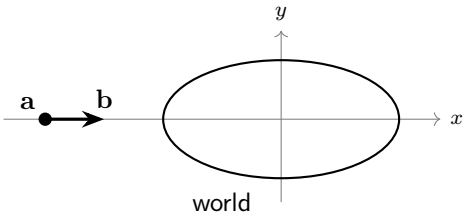
- The ray comes in world numbers, $\mathbf{a} + t\,\mathbf{b}$; no ellipse test
- So move the ray, not the shape: $\mathbf{M}_{OW}$ on $\mathbf{a}$ (a point) and on $\mathbf{b}$ (a direction)
- Hit the unit circle there and read off t

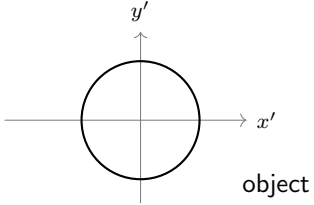
Hit test: bring the ray into the object's frame



- The ray comes in world numbers, $\mathbf{a} + t\,\mathbf{b}$; no ellipse test
- So move the ray, not the shape: $\mathbf{M}_{OW}$ on $\mathbf{a}$ (a point) and on $\mathbf{b}$ (a direction)
- Hit the unit circle there and read off t
- Same t in both frames: the world hit is $\mathbf{a} + t\,\mathbf{b}$

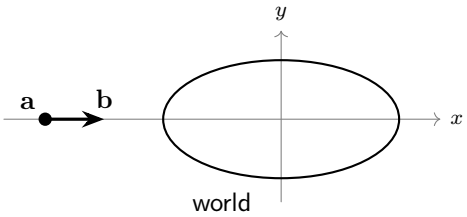
Worked: the same t in both spaces

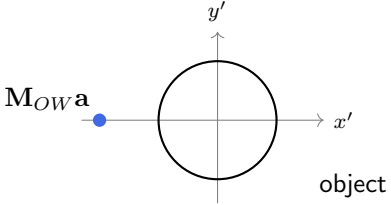




$\mathbf{M}_{WO} = \text{scale}(2, 1), \quad \mathbf{a} = (-4,\, 0), \quad \mathbf{b} = (1,\, 0)$

Worked: the same t in both spaces

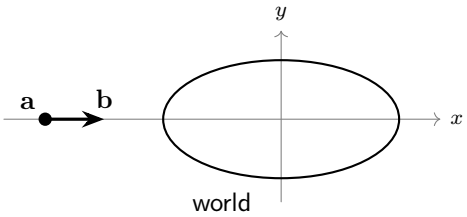


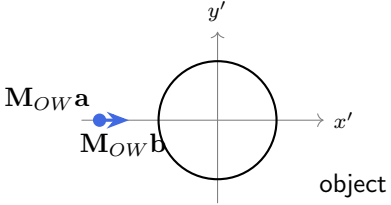


$\mathbf{M}_{WO} = \mathsf{scale}(2, 1), \quad \mathbf{a} = (-4,\ 0), \quad \mathbf{b} = (1,\ 0)$

$\mathbf{M}_{OW} = \mathsf{scale}(1/2,\ 1)$: $\quad \mathbf{M}_{OW}\mathbf{a} = (-2,\ 0)$

Worked: the same t in both spaces



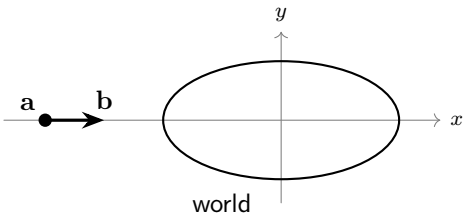


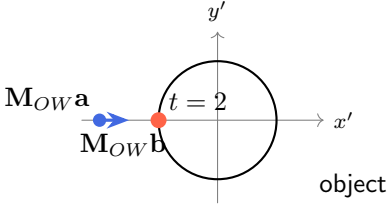
$\mathbf{M}_{WO} = \mathsf{scale}(2, 1), \quad \mathbf{a} = (-4,\ 0), \quad \mathbf{b} = (1,\ 0)$

$\mathbf{M}_{OW} = \mathsf{scale}(1/2,\ 1)$: $\quad \mathbf{M}_{OW}\mathbf{a} = (-2,\ 0)$

$\mathbf{M}_{OW}\mathbf{b} = (0.5,\ 0)$

Worked: the same t in both spaces





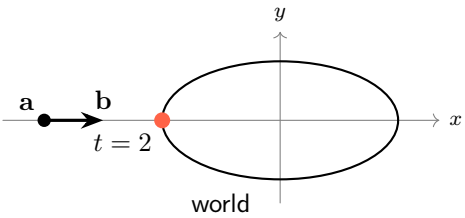
$\mathbf{M}_{WO} = \text{scale}(2, 1), \quad \mathbf{a} = (-4,\ 0), \quad \mathbf{b} = (1,\ 0)$

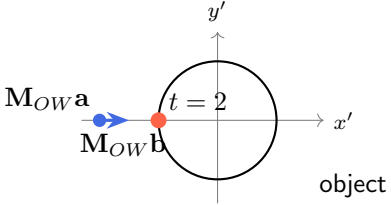
$\mathbf{M}_{OW} = \text{scale}(1/2,\ 1)$: $\quad \mathbf{M}_{OW}\mathbf{a} = (-2,\ 0)$

$\mathbf{M}_{OW}\mathbf{b} = (0.5,\ 0)$

circle hit at $x' = -1$: $\quad -2 + 0.5\,t = -1 \ \Rightarrow\ t = 2$

Worked: the same t in both spaces





$\mathbf{M}_{WO} = \text{scale}(2, 1), \quad \mathbf{a} = (-4,\ 0), \quad \mathbf{b} = (1,\ 0)$

$\mathbf{M}_{OW} = \text{scale}(1/2,\ 1)$: $\quad \mathbf{M}_{OW}\mathbf{a} = (-2,\ 0)$

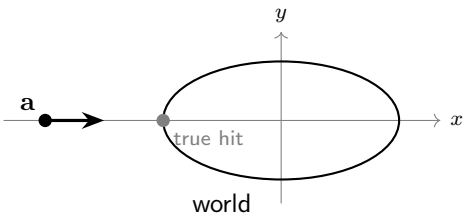
$\mathbf{M}_{OW}\mathbf{b} = (0.5,\ 0)$

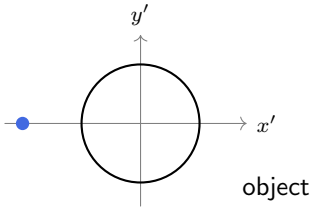
circle hit at $x' = -1$: $\quad -2 + 0.5\,t = -1 \ \Rightarrow\ t = 2$

world: $\mathbf{a} + 2\mathbf{b} = (-2,\ 0)$, the ellipse's left tip ✓

- Same t in both spaces

Never force ray directions to unit length

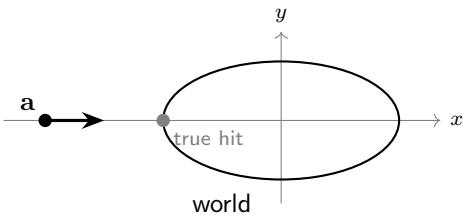


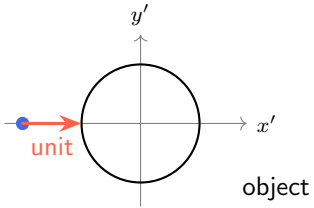


$\mathbf{M}_{OW}\mathbf{b} = (0.5,\ 0)$ has length 0.5

- Keep $\mathbf{M}_{OW}\mathbf{b}$ exactly as it comes

Never force ray directions to unit length



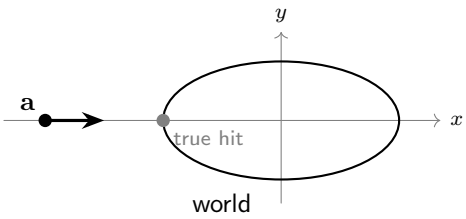


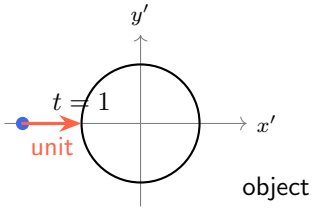
$\mathbf{M}_{OW}\mathbf{b} = (0.5,\ 0)$ has length 0.5

normalize it: $(1,\ 0)$

- Keep $\mathbf{M}_{OW}\mathbf{b}$ exactly as it comes

Never force ray directions to unit length





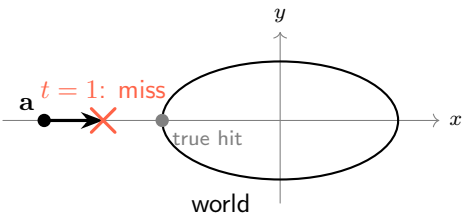
$\mathbf{M}_{OW}\mathbf{b} = (0.5,\ 0)$ has length 0.5

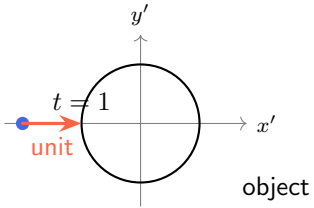
normalize it: $(1,\ 0)$

circle hit: $-2 + t = -1 \ \Rightarrow\ t = 1$

- Keep $\mathbf{M}_{OW}\mathbf{b}$ exactly as it comes

Never force ray directions to unit length





$\mathbf{M}_{OW}\mathbf{b} = (0.5,\ 0)$ has length 0.5

normalize it: $(1,\ 0)$

circle hit: $-2 + t = -1 \ \Rightarrow\ t = 1$

world: $\mathbf{a} + 1\,\mathbf{b} = (-3,\ 0)$, outside the ellipse

- Keep $\mathbf{M}_{OW}\mathbf{b}$ exactly as it comes
- Unit length changes t: wrong hit

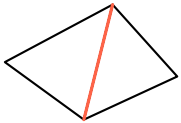
Recap: instancing

$$\mathbf{r}' = \mathbf{M}_{OW}\mathbf{a} + t\,\mathbf{M}_{OW}\mathbf{b}$$

Triangle meshes

Rules that keep a surface a surface

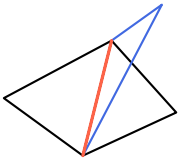
Manifold: two per edge, one loop per vertex



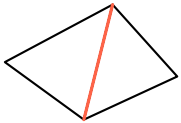
2 triangles: OK

- Mesh: triangles sharing edges and vertices

Manifold: two per edge, one loop per vertex



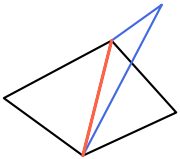
3 triangles: bad



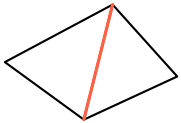
2 triangles: OK

- Mesh: triangles sharing edges and vertices
- Every edge: exactly two triangles

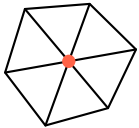
Manifold: two per edge, one loop per vertex



3 triangles: bad



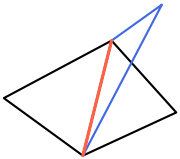
2 triangles: OK



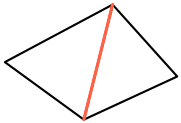
one loop: OK

- Mesh: triangles sharing edges and vertices
- Every edge: exactly two triangles
- Every vertex: one complete loop

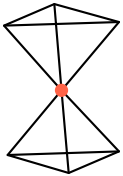
Manifold: two per edge, one loop per vertex



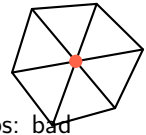
3 triangles: bad



2 triangles: OK



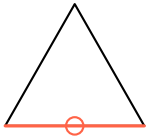
two loops: bad



one loop: OK

- Mesh: triangles sharing edges and vertices
- Every edge: exactly two triangles
- Every vertex: one complete loop

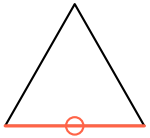
Edges allowed: a manifold with boundary



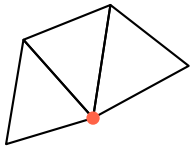
OK

- Edge: one or two triangles

Edges allowed: a manifold with boundary



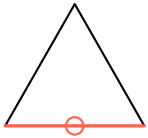
OK



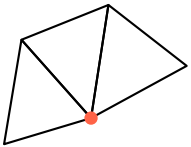
OK

- Edge: one or two triangles
- Vertex: one edge-connected set of triangles

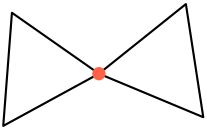
Edges allowed: a manifold with boundary



OK



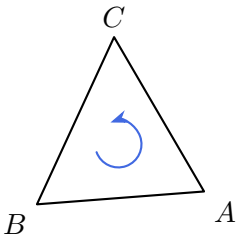
OK



bad

- Edge: one or two triangles
- Vertex: one edge-connected set of triangles
- Two separate sets at a vertex: bad

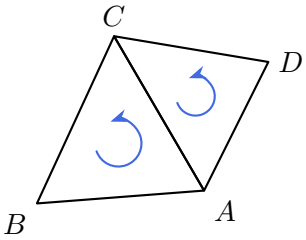
Orientation: neighbors agree on the front



(B, A, C) with (D, C, A)

- Front: corners listed counterclockwise

Orientation: neighbors agree on the front

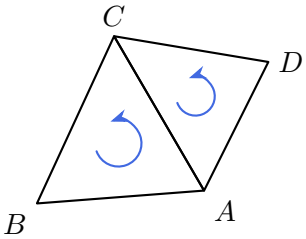


OK

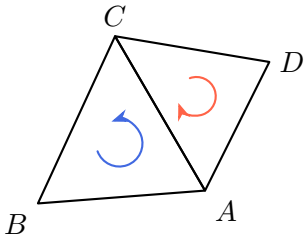
(B, A, C) with (D, C, A)

- Front: corners listed counterclockwise
- Shared edge: opposite order in neighbors

Orientation: neighbors agree on the front



OK

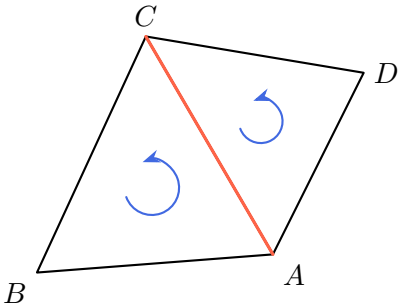


bad

(B, A, C) with (D, C, A) OK; (A, C, D) bad

- Front: corners listed counterclockwise
- Shared edge: opposite order in neighbors
- Same order: neighbors disagree

Recap: triangle meshes



Summary

$\mathbf{p} = \mathbf{e} + u_p\,\mathbf{u} + v_p\,\mathbf{v}$

a frame: origin plus orthonormal basis

$u_p = \mathbf{u} \cdot (\mathbf{p} - \mathbf{e}),\;\; v_p = \mathbf{v} \cdot (\mathbf{p} - \mathbf{e})$

world address to frame address

$$\mathbf{p}_{xy} = \begin{bmatrix} \mathbf{u} & \mathbf{v} & \mathbf{e} \\ 0 & 0 & 1 \end{bmatrix} \mathbf{p}_{uv}$$

frame-to-canonical: columns $\mathbf{u}, \mathbf{v}, \mathbf{e}$

$$\mathbf{p}_{uv} = \begin{bmatrix} \mathbf{u} & \mathbf{v} & \mathbf{e} \\ 0 & 0 & 1 \end{bmatrix}^{-1} \mathbf{p}_{xy}$$

canonical-to-frame: its inverse

$\mathsf{translate}(\mathbf{p})\,\mathsf{rotate}(\theta)\,\mathsf{translate}(-\mathbf{p})$

turn about a point

$\mathbf{M}_{WB} = \mathbf{M}_{WT}\,\mathbf{M}_{TB},\;\; \mathbf{M}_0\mathbf{M}_1\mathbf{M}_2$

multiply down the path from root; inner letters cancel

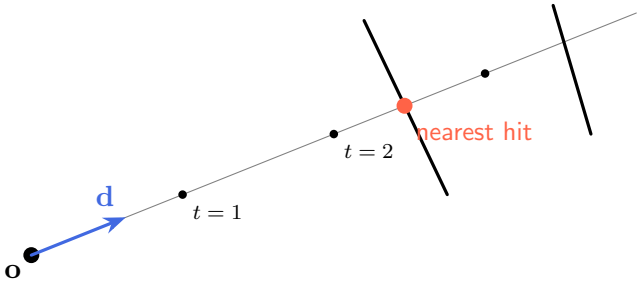
$\mathbf{M}_{OW}\mathbf{a} + t\,\mathbf{M}_{OW}\mathbf{b}$

hit in object space, same t

2 triangles per edge, 1 loop per vertex

manifold mesh

Next lecture: rays and the nearest hit



- A ray: $\mathbf{p} = \mathbf{o} + t\,\mathbf{d}$
- March along, keep the nearest hit

Shirley §2.7.6–2.7.7, §4.1