Week 8: 3D Geometry II
Part 2: Coordinate transforms

COMP270: Mathematics for 3D Worlds and Simulations

Objectives

- **Apply** matrix transformations to express points known in one coordinate space relative to another coordinate space

Recap: coordinate spaces

Transforming between coordinate spaces

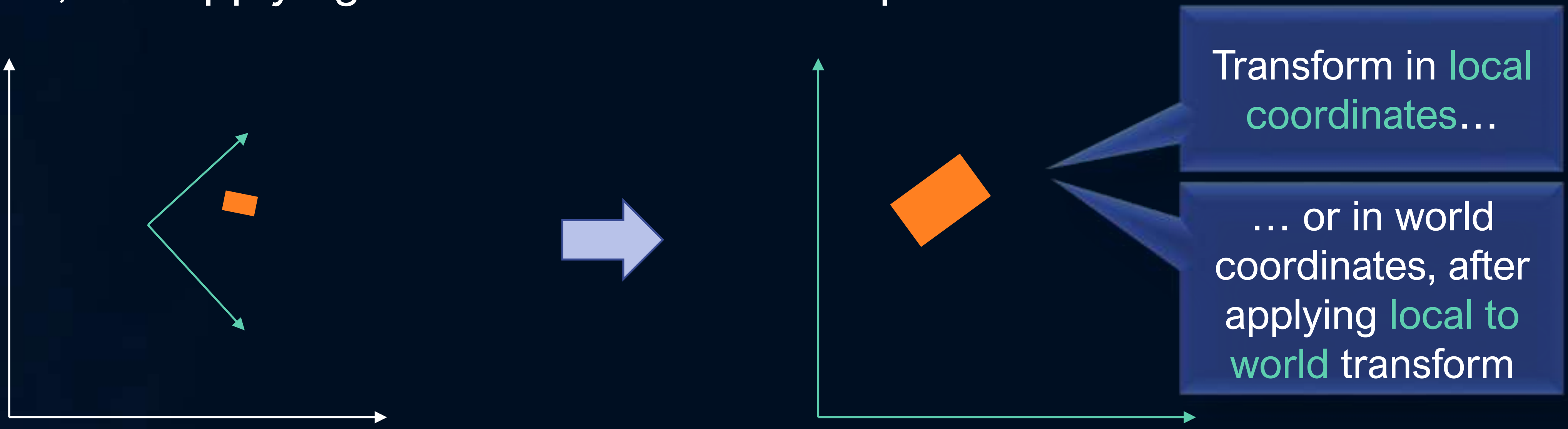
- Individual vertices of an object are probably stored in object space, with the object's overall transform specified in world space

Apply world transform to get vertices in world space

- To find collisions between two objects, we need both sets of vertices in the same space
  - Either transform both to world space, or one object to the other object's space (via the world) – or define a new 'collision space'
- To render the objects, we need to know their vertex positions in camera space (via world space).

Transforming objects vs. spaces

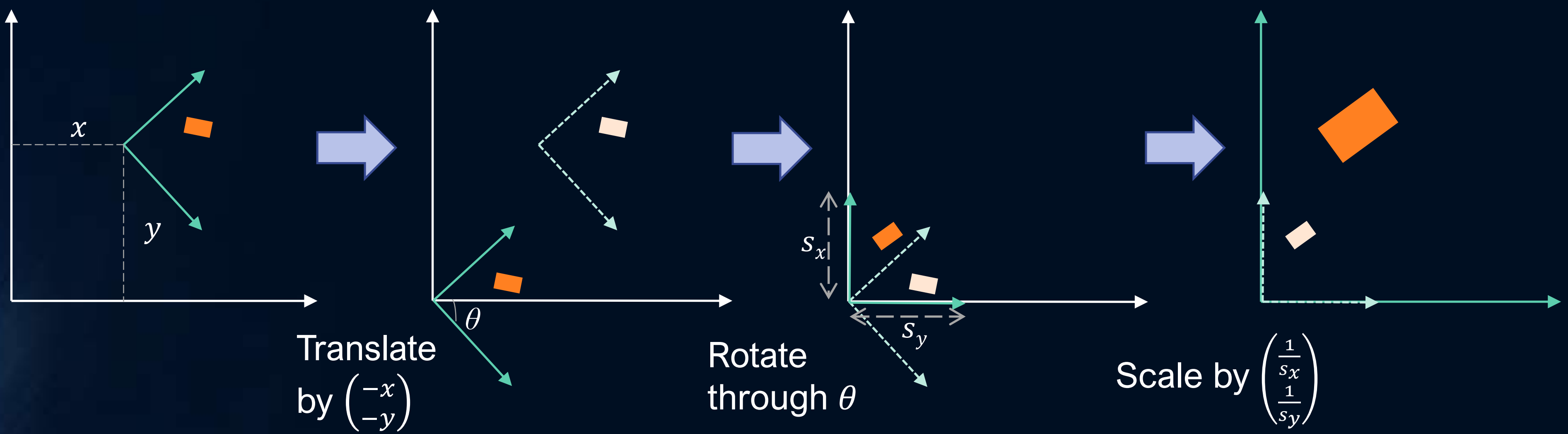
- Duality between describing a point in a different coordinate space, and applying a transformation to the point:



- Transforming a point to a new coordinate space = transforming the new space to the old
  - i.e. applying the inverse of the new space's transform in the old space

World to local space

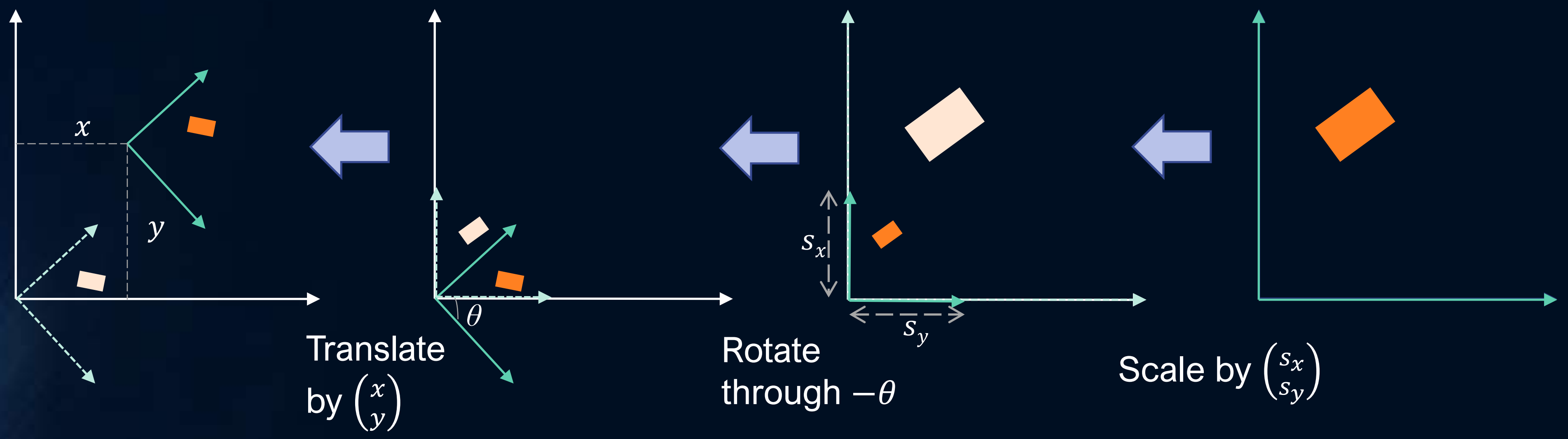
- Translate (to world space origin), rotate, scale:



- The opposite transformation to the one that describes the local space in world coordinates

Local to world space

- Scale, rotate, translate:



- the same transformation as the one that describes the local space in world coordinates… this is just how we move objects around the world!

Matrices and coordinate space transforms

- Remember that a matrix describes a linear mapping:

$$\begin{pmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \\ m_{31} & m_{32} & m_{33} \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} m_{11}x + m_{12}y + m_{13}z \\ m_{21}x + m_{22}y + m_{23}z \\ m_{31}x + m_{32}y + m_{33}z \end{pmatrix}$$

- Applied to the standard basis vectors:

$$\begin{pmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \\ m_{31} & m_{32} & m_{33} \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} m_{11} \\ m_{21} \\ m_{31} \end{pmatrix}$$

$$\begin{pmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \\ m_{31} & m_{32} & m_{33} \end{pmatrix} \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} = \begin{pmatrix} m_{12} \\ m_{22} \\ m_{32} \end{pmatrix}$$

$$\begin{pmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \\ m_{31} & m_{32} & m_{33} \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} m_{13} \\ m_{23} \\ m_{33} \end{pmatrix}$$

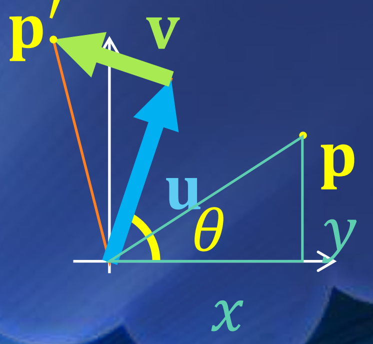
Matrices and basis vectors

- **Theorem**: the columns of a transformation matrix $\mathbf{M}$ can be interpreted as basis vectors of the space that $\mathbf{M}$ transforms to.
- Proof: since any vector $\mathbf{x}$ can be written as a linear combination of $\mathbf{i}$, $\mathbf{j}$ and $\mathbf{k}$:

$$\mathbf{x} = a\mathbf{i} + b\mathbf{j} + c\mathbf{k}$$
$$\mathbf{Mx} = \mathbf{M}(a\mathbf{i} + b\mathbf{j} + c\mathbf{k})$$
$$= \mathbf{M}(a\mathbf{i}) + \mathbf{M}(b\mathbf{j}) + \mathbf{M}(c\mathbf{k})$$
$$= a(\mathbf{Mi}) + b(\mathbf{Mj}) + c(\mathbf{Mk})$$

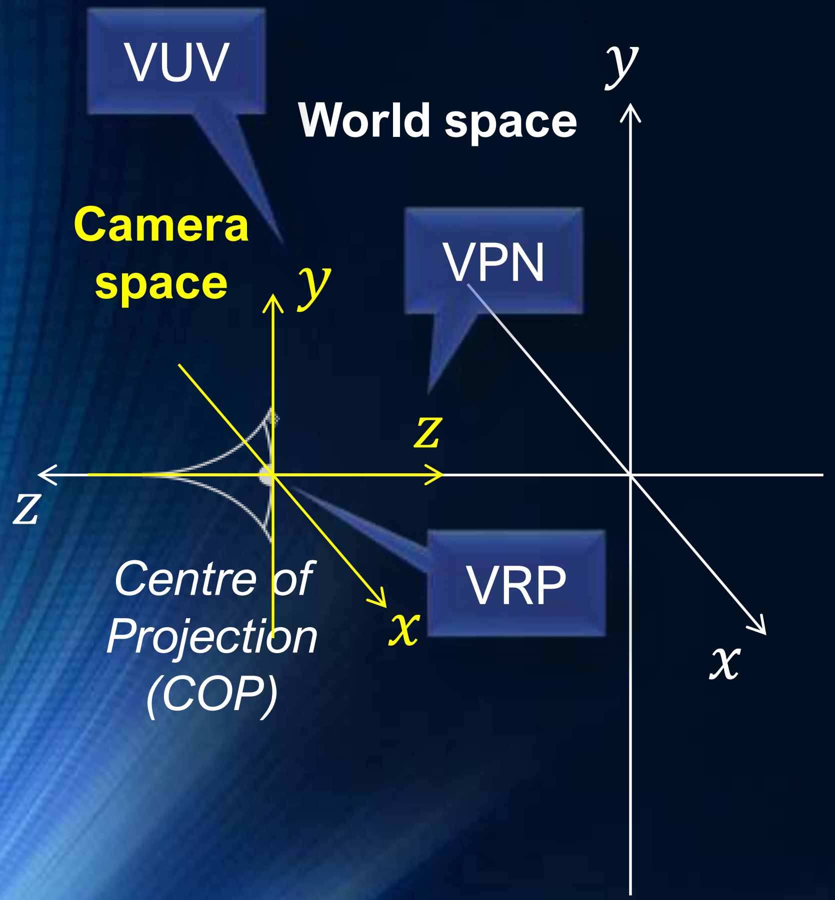
$$= a\begin{pmatrix} m_{11} \\ m_{21} \\ m_{31} \end{pmatrix} + b\begin{pmatrix} m_{12} \\ m_{22} \\ m_{32} \end{pmatrix} + c\begin{pmatrix} m_{13} \\ m_{23} \\ m_{33} \end{pmatrix}$$

We used this idea to create the 2D rotation matrix:



Tip: visualise a transformation by extracting the basis vectors and comparing them to the original axes.

Example: generalised camera coordinates



Viewing coordinate system (VC):

- **View reference point (VRP)**: the origin (point) of the VC system in world space
  - The point with respect to which the COP and view plane are defined
- **View plane normal (VPN)**: direction vector specifying the positive z-axis of the VC system in world space
  - Direction the camera is pointing
- **View up vector (VUV)**: direction vector used to define the positive y-axis of the VC system in world space
  - The VC y-axis is formed by projecting the VUV onto a plane perpendicular to the VPN, passing through the VRP

Generalised camera: viewing coordinates

- Let the x-, y- and z-axes of the viewing coordinates be referred to as u, v and n respectively:
- $\mathbf{n}$ is a unit vector in the direction of the VPN:

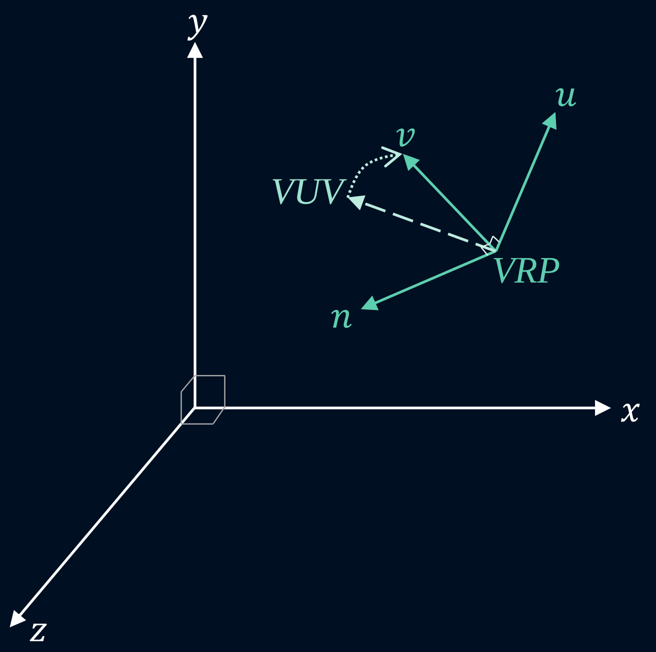
$$\mathbf{n} = \frac{\boldsymbol{VPN}}{\|\boldsymbol{VPN}\|}$$

- $\mathbf{u}$ is unit vector in the direction of the u-axis of the viewing coordinates. To form a left-handed system,

$$\mathbf{u} = \frac{\mathbf{n} \times \boldsymbol{VUV}}{\|\mathbf{n} \times \boldsymbol{VUV}\|}$$

- To obtain the unit vector $\mathbf{v}$ along the v-axis:

$$\mathbf{v} = \mathbf{u} \times \mathbf{n}$$



Generalised camera: rotation

Gives a world space point in camera space (= the inverse transform of the camera in world space)

- Let $\mathbf{M}$ be the 4×4 matrix that maps world coordinate space into viewing coordinate space, partitioned into a rotational part, $\mathbf{R}$, and translation vector $\mathbf{t}$:

$$\mathbf{M} = \begin{pmatrix} \mathbf{R} & \mathbf{t} \\ 0 & 1 \end{pmatrix}$$

$$\begin{pmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

- The vectors $\mathbf{u}$, $\mathbf{v}$, $\mathbf{n}$ (in world space) must be rotated by $\mathbf{R}$ into the unit basis vectors of VC space:

$$\mathbf{Ru} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \mathbf{Rv} = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}, \mathbf{Rn} = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

$\mathbf{R}$ is the inverse of the matrix with columns $\mathbf{u}$, $\mathbf{v}$, $\mathbf{n}$

- That is,

$$\mathbf{R}(\mathbf{u} \quad \mathbf{v} \quad \mathbf{n}) = \mathbf{I}$$

- Since $\mathbf{u}$, $\mathbf{v}$ and $\mathbf{n}$ are orthonormal, $\mathbf{R} = (\mathbf{u} \quad \mathbf{v} \quad \mathbf{n})^T = \begin{pmatrix} u_1 & u_2 & u_3 \\ v_1 & v_2 & v_3 \\ n_1 & n_2 & n_3 \end{pmatrix}$

(explanation here).

Generalised camera: translation

- Similarly, the VRP must be transformed into the origin of the VC space. If the position of the VRP in world space is given by $\mathbf{q}$, then

$$\mathbf{M}\begin{pmatrix}\mathbf{q}\\1\end{pmatrix}=\begin{pmatrix}0\\0\\0\\1\end{pmatrix}$$

- Substituting $\mathbf{M}$ for its partitioned form,

$$\begin{pmatrix}\mathbf{R} & \mathbf{t}\\0 & 1\end{pmatrix}\begin{pmatrix}\mathbf{q}\\1\end{pmatrix}=\begin{pmatrix}0\\0\\0\\1\end{pmatrix}\Rightarrow \mathbf{Rq}+\mathbf{t}=\mathbf{0}\Rightarrow \mathbf{t}=-\mathbf{Rq}$$

Generalised camera: full transform

- Putting everything together, we get

$$\mathbf{M} = \begin{pmatrix} \mathbf{R} & \mathbf{t} \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} \mathbf{u} & \\ \mathbf{v} & -\mathbf{Rq} \\ \mathbf{n} & \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} u_1 & u_2 & u_3 & -\mathbf{u}\cdot\mathbf{q} \\ v_1 & v_2 & v_3 & -\mathbf{v}\cdot\mathbf{q} \\ n_1 & n_2 & n_3 & -\mathbf{n}\cdot\mathbf{q} \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

- In addition, the inverse (which transforms from viewing coordinates back to world coordinates) can be written as:

$$\mathbf{M}^{-1} = \begin{pmatrix} \mathbf{R}^T & \mathbf{q} \\ 0 & 1 \end{pmatrix}$$