Computer Graphics

Lecture 6

Affine Transformations

Translation

- Using the mathematics we've described so far, what about moving objects in space.

- Problem (in 2D), we have:

$$
\begin{aligned}
x' &= m_{11}x + m_{12}y \\
y' &= m_{21}x + m_{22}y
\end{aligned}
$$

- But we want:

$$
\begin{aligned}
x' &= m_{11}x + m_{12}y + x_t \\
y' &= m_{21}x + m_{22}y + y_t
\end{aligned}
$$

Homogeneous Coordinates

- To put this into one system of linear equations, we **promote** (increase the dimensionality) by adding a component w = 1 for vectors

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} m_{11} & m_{12} & x_t \\ m_{21} & m_{22} & y_t \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} m_{11}x + m_{12}y + x_t \\ m_{21}x + m_{22}y + y_t \\ 1 \end{bmatrix}$$

- Implements a linear transformation followed by a translation (x_t, y_t)
- These transformations are called **affine transformations**:
  - Like linear transformations, they keep straight lines straight and parallel lines parallel, but they do not preserve the origin

Homogeneous Coordinates

- We promote all points (x,y) to (x,y,w=1), and similarly in 3D we promote (x,y,z) to (x,y,z,w=1)
- These new coordinates are called **homogeneous coordinates**
- Can be thought of as a clever bookkeeping scheme, but also have a geometric interpretation, compare the following matrix with a standard shear:

$$\begin{bmatrix} 1 & 0 & x_t \\ 0 & 1 & y_t \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} x + x_t z \\ y + y_t z \\ z \end{bmatrix}$$

Homogeneous Coordinates

- Composition works just as before, but using 3x3 multiplication instead of 2x2

- This approach is easier than keeping the linear transform and the translate separately stored

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

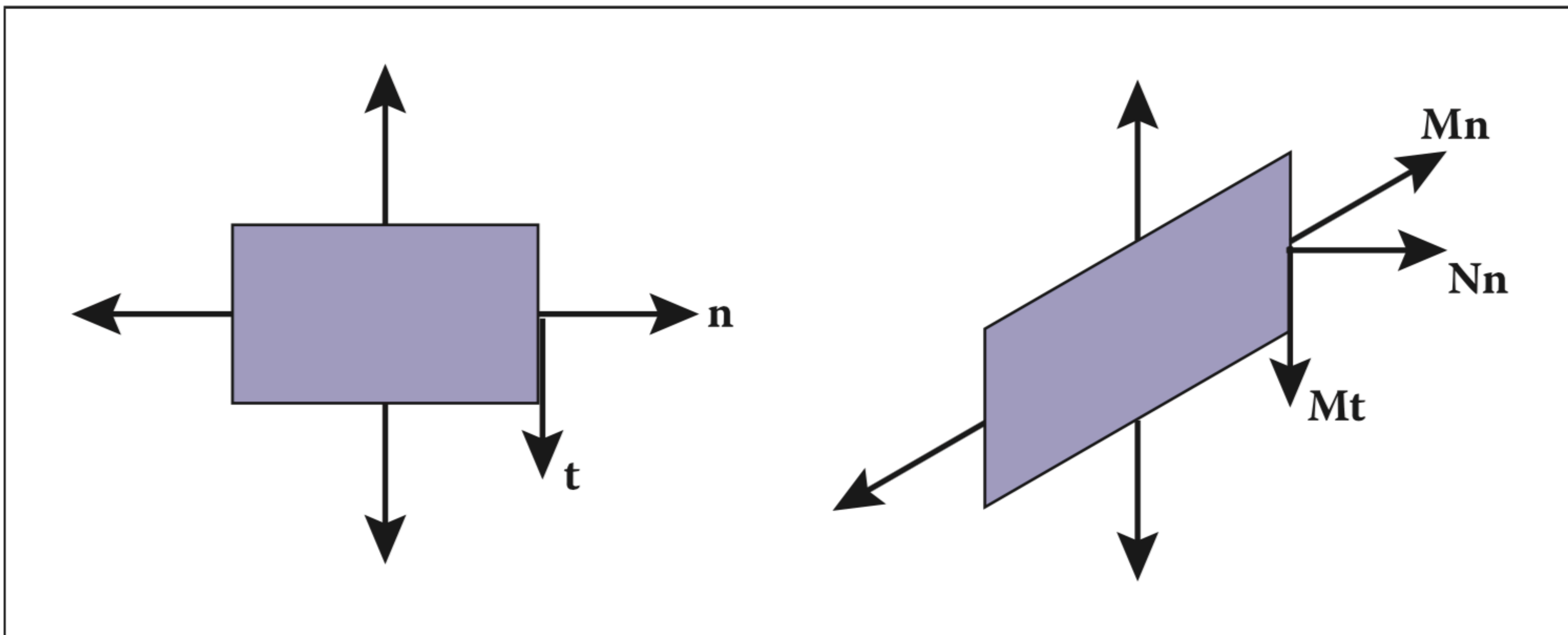
Transforming Points vs. Transforming Vectors

- Using homogeneous coordinates, we can differentiate between points and vectors
- Recall:
  - Vectors are just offsets (differences between points), and thus should be not affected by translation
  - Whereas points are represented by vectors offset from the origin
- Rule: vectors have w=0, whereas points have w=1:

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

Transforming Normals

- While differences between points transform OK, normals do not necessarily behave the same



Transforming Normals

- The problem is that the orthogonality constraint, that normals always point orthogonal to the surface, is not always preserved.

- One can solve for the correct transformation by observing that tangent vectors, $\mathbf{t}$, transform correctly and $\mathbf{t} \cdot \mathbf{n} = 0$.

$$\mathbf{n}^{\mathrm{T}}\mathbf{t} = \mathbf{0}$$

$$\mathbf{n}^{\mathrm{T}}\mathbf{t} = \mathbf{n}^{\mathrm{T}}\mathbf{I}\mathbf{t} = \mathbf{n}^{\mathrm{T}}\mathbf{M}^{-1}\mathbf{M}\mathbf{t} = 0$$

- So, we can transform normals using the inverse matrix

$$\left(\mathbf{M}^{-1}\right)^{\mathrm{T}}\mathbf{n}$$

Coordinate Transformations

Coordinate Systems

- Points in space can be represented using an origin position and a set of orthogonal basis vectors:

Coordinate Systems

- Points in space can be represented using an origin position and a set of orthogonal basis vectors:

$$\mathbf{p} = \left(x_p, y_p\right) \equiv \mathbf{0} + x_p\mathbf{x} + y_p\mathbf{y}$$

Coordinate Systems

- Points in space can be represented using an origin position and a set of orthogonal basis vectors:

$$\mathbf{p} = \left(x_p, y_p\right) \equiv \mathbf{0} + x_p\mathbf{x} + y_p\mathbf{y} \qquad \mathbf{p} = \left(u_p, v_p\right) \equiv \mathbf{e} + u_p\mathbf{u} + v_p\mathbf{v}$$

Coordinate Systems

- Points in space can be represented using an origin position and a set of orthogonal basis vectors:

$$\mathbf{p} = (x_p, y_p) \equiv \mathbf{0} + x_p\mathbf{x} + y_p\mathbf{y} \qquad \mathbf{p} = (u_p, v_p) \equiv \mathbf{e} + u_p\mathbf{u} + v_p\mathbf{v}$$

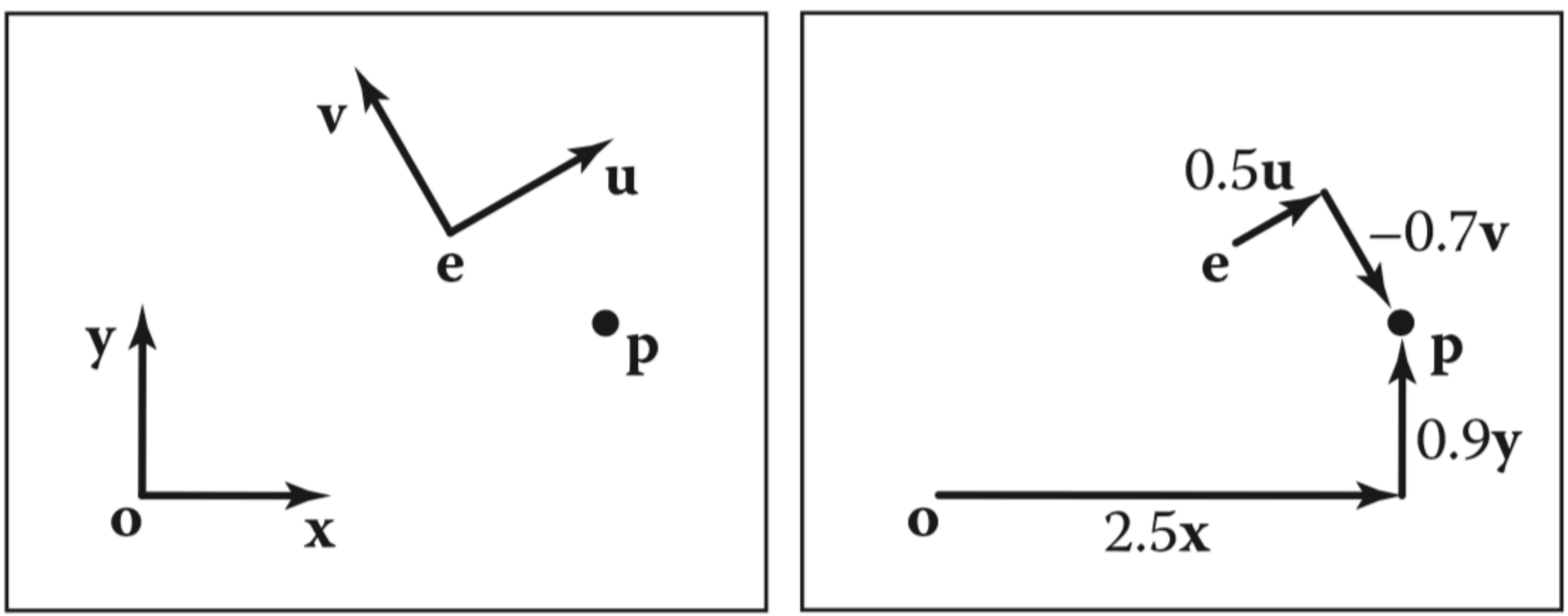
- Any point can be described in either coordinate system

Coordinate Systems

- Points in space can be represented using an origin position and a set of orthogonal basis vectors:

$$\mathbf{p} = (x_p, y_p) \equiv \mathbf{0} + x_p\mathbf{x} + y_p\mathbf{y} \qquad \mathbf{p} = (u_p, v_p) \equiv \mathbf{e} + u_p\mathbf{u} + v_p\mathbf{v}$$

- Any point can be described in either coordinate system



Matrices for Converting Coordinate Systems

- Using homogenous coordinates and affine transformations, we can convert between coordinate systems:

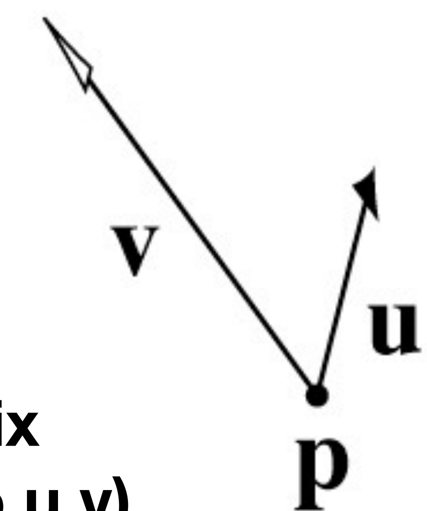
$$\begin{bmatrix} x_p \\ y_p \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & x_e \\ 0 & 1 & y_e \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_u & x_v & 0 \\ y_u & y_v & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u_p \\ v_p \\ 1 \end{bmatrix} = \begin{bmatrix} x_u & x_v & x_e \\ y_u & y_v & y_e \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u_p \\ v_p \\ 1 \end{bmatrix}$$

- More generally, any arbitrary coordinate system transform:

$$\mathbf{P}_{uv} = \begin{bmatrix} \mathbf{x}_{uv} & \mathbf{y}_{uv} & \mathbf{o}_{uv} \\ 0 & 0 & 1 \end{bmatrix} \mathbf{p}_{xy}$$

Affine Change of Coordinates

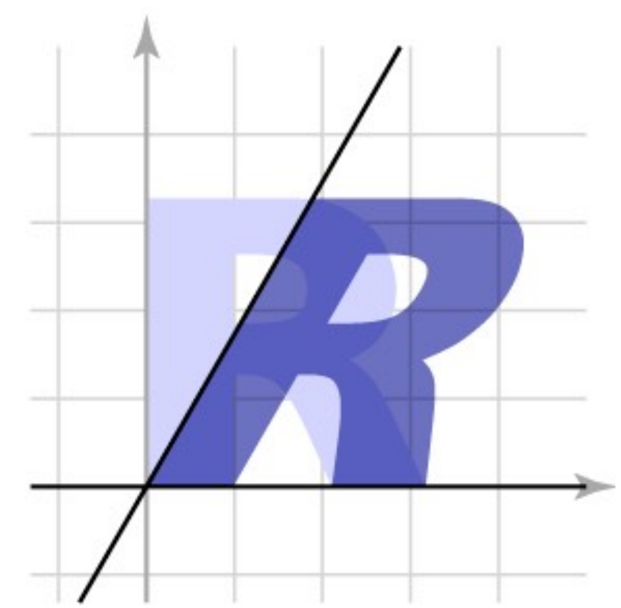
- It turns out this works even if u, v are not orthogonal.
- It also provides another way to interpret and construct transformation matrices



Frame matrix
(converts x,y to u,v)

$$\begin{bmatrix} \mathbf{u} & \mathbf{v} & \mathbf{p} \\ 0 & 0 & 1 \end{bmatrix}$$

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



Required Reading

- FOCG, Ch. 7