Computer Graphics

- Transformations -

Philipp Slusallek
Vector Space

- **Math recap**
  - 3D vector space over the real numbers
    - \( \mathbf{v} = \begin{pmatrix} v_1 \\ v_2 \\ v_3 \end{pmatrix} \in \mathbb{V}^3 = \mathbb{R}^3 \)
  - Vectors written as \( n \times 1 \) matrices
  - Vectors describe directions – **not positions**!
    - All vectors conceptually start from the origin of the coordinate system
  - 3 linear independent vectors create a basis
    - Standard basis
      \[ \{ \mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3 \} = \left\{ \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \right\} \]
  - Any 3D vector can be represented uniquely with coordinates \( v_i \) with respect to a basis
    - \( \mathbf{v} = v_1 \mathbf{e}_1 + v_2 \mathbf{e}_2 + v_3 \mathbf{e}_3 \quad v_1, v_2, v_3 \in \mathbb{R} \)
Vector Space - Metric

- **Standard scalar product, a.k.a. dot or inner product**
  - \( u \cdot v = u_1 v_1 + u_2 v_2 + u_3 v_3 \)
  - Used to measure lengths
    - \( |v|^2 = v \cdot v = v_1^2 + v_2^2 + v_3^2 \)
  - Used to compute angles
    - \( u \cdot v = |u||v|\cos(u, v) \)
  - Projection of vectors onto other vectors
    - \( |u|\cos(\theta) = \frac{u \cdot v}{|v|} = \frac{u \cdot v}{\sqrt{v \cdot v}} \)
Vector Space - Basis

- **Orthonormal basis**
  - Unit length vectors
    - $|e_1| = |e_1| = |e_1| = 1$
  - Orthogonal to each other
    - $e_i \cdot e_j = \delta_{ij}$

- **Handedness of a coordinate system**
  - Two options: $e_1 \times e_2 = \pm e_3$
    - Positive: Right-handed (RHS)
    - Negative: Left-handed (LHS)
  - Example: Screen Space
    - Typical: X goes right, Y goes up (thumb & index finger, respectively)
    - In a RHS: Z goes **out** of the screen (middle finger)
  - Be careful:
    - Most systems nowadays use a right handed coordinate system
    - But some are not (e.g. RenderMan) $\rightarrow$ can cause lots of confusion
Affine Space

- **Basic mathematical concept**
  - Denoted as $A^3$
    - Elements are positions (not directions!)
  - Defined via its associated vector space $V^3$
    - $a, b \in A^3 \iff \exists! v \in V^3: v = b - a$
    - $\rightarrow$: unique, $\leftarrow$: ambiguous
  - Operations on $A^3$
    - Subtraction of two elements yields a vector
    - No addition of affine elements
      - It's not clear what *sum of two points* would even mean
    - But: Addition of points and vectors:
      - $a + v = b \in A^3$
    - Distance
      - $dist(a, b) = |a - b|$
Affine Space - Basis

• **Affine Basis**
  – Given by its origin \( o \) (a point) and the basis of an associated vector space
    • \( \{ e_1, e_2, e_3, o \} \): \( e_1, e_2, e_3 \in V^3; o \in A^3 \)

• **Position vector of point \( p \)**
  – \((p - o)\) is in \( V^3 \)
Affine Coordinates

- **Affine Combination**
  - Linear combination of \((n+1)\) points
    - \(p_0, ..., p_n \in A^n\)
  - With weights forming a partition of unity
    - \(\alpha_0, ..., \alpha_n \in \mathbb{R}\) with \(\sum_i \alpha_i = 1\)
  - \(p = \sum_{i=0}^{n} \alpha_i p_i = p_0 + \sum_{i=1}^{n} \alpha_i (p_i - p_0) = o + \sum_{i=1}^{n} \alpha_i v_i\)

- **Basis**
  - \((n + 1)\) points form an **affine basis** of \(A^n\)
    - If none of these points can be expressed as an affine combination of the other points
    - Any point in \(A^n\) can then be uniquely represented as an affine combination of the affine basis \(p_0, ..., p_n \in A^n\)
    - Any point in another basis can also be expressed as a linear combination of the \(p_i\), yielding a matrix for the basis transform
Affine Coordinates

- Closely related to “Barycentric Coordinates”
  - Center of mass of \( (n + 1) \) points with arbitrary masses (weights) \( m_i \) is given as
    \[
    p = \frac{\sum m_i p_i}{\sum m_i} = \frac{1}{\sum m_i} \sum m_i p_i = \sum \alpha_i p_i
    \]
- Convex / Affine Hull
  - If all \( \alpha_i \) are non-negative than \( p \) is in the convex hull of the other points
- In 1D
  - Point is defined by the splitting ratio \( \alpha_1 : \alpha_2 \)
    \[
    p = \alpha_1 p_1 + \alpha_2 p_2 = \frac{|p-p_2|}{|p_2-p_1|} p_1 + \frac{|p-p_1|}{|p_2-p_1|} p_2
    \]
- In 2D
  - Weights are the relative areas in \( \Delta(A_1, A_2, A_3) \)
    \[
    t_i = \alpha_i = \frac{\Delta(P,A_{(i+1)} \% 3,A_{(i+2)} \% 3)}{\Delta(A_1,A_2,A_3)}
    \]
    \[
    p = \alpha_1 A_1 + \alpha_2 A_2 + \alpha_3 A_3
    \]

Note: Length and area measures are signed here.
Affine Mappings

**Properties**
- Affine mapping/transformations (continuous, bijective, invertible)
  - $T: \mathbb{A}^3 \rightarrow \mathbb{A}^3$
- Defined by two non-degenerated simplicies (that define a basis)
  - 2D: Triangle, 3D: Tetrahedron, ...
- Invariants under affine transformations:
  - Barycentric/affine coordinates
  - Straight lines, parallelism, splitting ratios, surface/volume ratios
- Characterization via fixed points and lines
  - Given as eigenvalues and eigenvectors of the mapping

**Representation**
- Matrix product and a translation vector:
  - $T_p = A p + t$ with $A \in \mathbb{R}^{n \times n}$, $t \in \mathbb{R}^n$
- Invariance of affine coordinates
  - $T_p = T(\sum \alpha_i p_i) = A(\sum \alpha_i p_i) + t = \sum \alpha_i (A p_i) + \sum \alpha_i t = \sum \alpha_i (T p_i)$
Homogeneous Coordinates for 3D

- Homogeneous embedding of $\mathbb{R}^3$ into the projective 4D space $\mathbb{P}(\mathbb{R}^4)$
  - Mapping into homogeneous space
    - $\mathbb{R}^3 \ni \begin{pmatrix} x \\ y \\ z \end{pmatrix} \mapsto \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} \in \mathbb{P}(\mathbb{R}^4)$
  - Mapping back by dividing through fourth component
    - $\begin{pmatrix} X \\ Y \\ Z \\ W \end{pmatrix} \mapsto \begin{pmatrix} X/W \\ Y/W \\ Z/W \end{pmatrix}$

- Consequence
  - This allows to represent affine transformations as 4x4 matrices
  - Mathematical trick
    - Convenient representation to express rotations and translations as matrix multiplications
    - Easy to find line through points, point-line/line-line intersections
  - Also allows to define projections (later)
Point Representation in 2D or P(3D)

- **Point in homogeneous coordinates**
  - All points along a line through the origin map to the same point in 2D

\[
p = (X, Y, W)\]

\[
x = \frac{X}{W} \quad y = \frac{Y}{W}
\]
Homogeneous Coordinates in 2D

- **Some tricks (work only in \( P(\mathbb{R}^3), \) i.e. only in 2D)**
  - Point representation
    - \((X) = \begin{pmatrix} X \\ Y \\ W \end{pmatrix} \in P(\mathbb{R}^3), \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} X/W \\ Y/W \end{pmatrix}\)
  - Representation of a line \( l \in \mathbb{R}^2 \)
    - Dot product of \( l \) vector with point in plane must be zero:
      - \( l = \begin{pmatrix} a \\ b \\ c \end{pmatrix} \cdot \begin{pmatrix} x \\ y \end{pmatrix} = 0 \Rightarrow X \in P(\mathbb{R}^3) | X \cdot l = 0, l = \begin{pmatrix} a \\ b \\ c \end{pmatrix} \)
    - Line \( l \) is normal vector of the plane through origin and points on line
  - Line through 2 points \( p \) and \( p' \)
    - Line must be orthogonal to both points
      - \( p \in l \land p' \in l \iff l = p \times p' \)
  - Intersection of lines \( l \) and \( l' \):
    - Point on both lines \( \Rightarrow \) point must be orthogonal to both line vectors
      - \( X \in l \cap l' \iff X = l \times l' \)
Line Representation

- **Definition of a 2D Line in $P(\mathbb{R}^3)$**
  - Set of all point $P$ where the dot product with $l$ is zero

\[
p \cdot l = 0
\]
Line Representation

- **Line**
  - Represented by normal vector to plane through line and origin

\[ l = (a, b, c) \]

\[ ax + by + c \cdot 1 = 0 \]
Line through 2 Points

- Construct line through two points
  - Line vector must be orthogonal to both points
  - Compute through cross product of point coordinates

\[ l = \mathbf{p} \times \mathbf{p}' \]
Intersection of Lines

- **Construct intersection of two lines**
  - A point that is on both lines and thus orthogonal to both lines
  - Computed by cross product of both line vectors

\[ p = l \times l' \]
Orthonormal Matrices

- **Columns are orthogonal vectors of unit length**
  - An example
    
    \[
    \begin{pmatrix}
    0 & 0 & 1 \\
    1 & 0 & 0 \\
    0 & 1 & 0
    \end{pmatrix}
    \]
  - Directly derived from the definition of the matrix product:
    - \( M^T M = 1 \)
  - In this case the transpose must be identical to the inverse:
    - \( M^{-1} := M^T \)
Linear Transformation: Matrix

- **Transformations in a Vector space: Multiplication by a Matrix**
  - Action of a linear transformation on a vector
    - Multiplication of matrix with column vectors (e.g. in 3D)

\[
p' = \begin{pmatrix} X' \\ Y' \\ Z' \end{pmatrix} = Tp = \begin{pmatrix} T_{xx} & T_{xy} & T_{xz} \\ T_{yx} & T_{yy} & T_{yz} \\ T_{zx} & T_{zy} & T_{zz} \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \end{pmatrix}
\]

- **Composition of transformations**
  - Simple matrix multiplication \((T_1, \text{then} \ T_2)\)
    - \(T_2T_1p = T_2(T_1p) = (T_2T_1)p = Tp\)
    - Note: matrix multiplication is associative but not commutative!
      - \(T_2T_1\) is not the same as \(T_1T_2\) (in general)
Affine Transformation

• **Remember:**
  – Affine map: Linear mapping and a translation
    • $T_p = A p + t$

• **For 3D: Combining it into a single matrix**
  – Using homogeneous 4D coordinates
  – Multiplication by 4x4 matrix in $\mathbb{P}(\mathbb{R}^4)$ space

\[
\begin{pmatrix}
X' \\
Y' \\
Z' \\
W'
\end{pmatrix} = T_p = 
\begin{pmatrix}
T_{xx} & T_{xy} & T_{xz} & T_{xw} \\
T_{yx} & T_{yy} & T_{yz} & T_{yw} \\
T_{zx} & T_{zy} & T_{zz} & T_{zw} \\
T_{wx} & T_{wy} & T_{wz} & T_{ww}
\end{pmatrix}
\begin{pmatrix}
X \\
Y \\
Z \\
W
\end{pmatrix}
\]
  – Allows for combining (concatenating) multiple transforms into one using normal (4x4) matrix products

• **Let’s go through the different transforms we need:**
Transformations: Translation

- **Translation (T)**

\[ T(t_x, t_y, t_z)p = \begin{pmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} x + t_x \\ y + t_y \\ z + t_z \\ 1 \end{pmatrix} \]
Translation of Vectors

- So far: only translated points
- Vectors: Difference between 2 points
  \[ \mathbf{v} = \mathbf{p} - \mathbf{q} = \begin{pmatrix} p_x \\ p_y \\ p_z \\ 1 \end{pmatrix} - \begin{pmatrix} q_x \\ q_y \\ q_z \\ 1 \end{pmatrix} = \begin{pmatrix} p_x - q_x \\ p_y - q_y \\ p_z - q_z \\ 0 \end{pmatrix} \]
  - Fourth component is zero

- Consequently: Translations do not affect vectors!

  \[ \mathbf{T}(t_x, t_y, t_z) \mathbf{v} = \begin{pmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} v_x \\ v_y \\ v_z \\ 0 \end{pmatrix} = \begin{pmatrix} v_x \\ v_y \\ v_z \\ 0 \end{pmatrix} \]
Translation: Properties

- **Properties**
  - Identity
    - \( T(0,0,0) = 1 \) (Identity Matrix)
  - Commutative (special case)
    - \( T(t_x, t_y, t_z)T(t'_x, t'_y, t'_z) = T(t'_x, t'_y, t'_z)T(t_x, t_y, t_z) = T(t_x + t'_x, t_y + t'_y, t_z + t'_z) \)
  - Inverse
    - \( T^{-1}(t_x, t_y, t_z) = T(-t'_x, -t'_y, -t'_z) \)
Basic Transformations (2)

- Scaling (S)

- \( S(s_x, s_y, s_z) = \begin{pmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \)

- Note: \( s_x, s_y, s_z \geq 0 \) (otherwise see mirror transformation)

- Uniform Scaling \( s = s_x = x_y = s_z \)
Basic Transformations

- **Reflection/Mirror Transformation (M)**
  - Reflection at plane \((x=0)\)
    \[
    M_x = \begin{pmatrix}
    -1 & 0 & 0 & 0 \\
    0 & 1 & 0 & 0 \\
    0 & 0 & 1 & 0 \\
    0 & 0 & 0 & 1
    \end{pmatrix}
    \begin{pmatrix}
    x \\
    y \\
    z \\
    1
    \end{pmatrix}
    = \begin{pmatrix}
    -x \\
    y \\
    z \\
    1
    \end{pmatrix}
    \]
  - Analogously for other axis
  - Note: changes orientation
    - Right-handed rotation becomes left-handed and v.v.
    - Indicated by \(\text{det}(M_i) < 0\)
  - Reflection at origin
    \[
    M_o = \begin{pmatrix}
    -1 & 0 & 0 & 0 \\
    0 & -1 & 0 & 0 \\
    0 & 0 & -1 & 0 \\
    0 & 0 & 0 & 1
    \end{pmatrix}
    \begin{pmatrix}
    x \\
    y \\
    z \\
    1
    \end{pmatrix}
    = \begin{pmatrix}
    -x \\
    -y \\
    -z \\
    1
    \end{pmatrix}
    \]
  - Note: changes orientation in 3D
    - But not in 2D (!!!): Just two scale factors
    - Each scale factor reverses orientation once
Basic Transformations (4)

- **Shear (H)**
  
  \[
  H(h_{xy}, h_{xz}, h_{yz}, h_{yx}, h_{zx}, h_{zy}) = 
  \begin{pmatrix}
  1 & h_{xy} & h_{xz} & 0 \\
  h_{yx} & 1 & h_{yz} & 0 \\
  h_{zx} & h_{zy} & 1 & 0 \\
  0 & 0 & 0 & 1
  \end{pmatrix}
  \begin{pmatrix}
  x \\
  y \\
  z \\
  1
  \end{pmatrix}
  = 
  \begin{pmatrix}
  x + h_{xy}y + h_{xz}z \\
  y + h_{yx}x + h_{yz}z \\
  z + h_{zx}x + h_{zy}y \\
  1
  \end{pmatrix}
  \]

  - Determinant is 1
    
    - Volume preserving (as volume is just shifted in some direction)
Rotation in 2D

• In 2D: Rotation around origin
  – Representation in spherical coordinates
    \[ x = r \cos \theta \rightarrow x' = r \cos(\theta + \phi) \]
    \[ y = r \sin \theta \rightarrow y' = r \sin(\theta + \phi) \]
  – Well know property
    \[ \cos(\theta + \phi) = \cos \theta \cos \phi - \sin \theta \sin \phi \]
    \[ \sin(\theta + \phi) = \cos \theta \sin \phi + \sin \theta \cos \phi \]
  – Gives
    \[ x' = (r \cos \theta) \cos \phi - (r \sin \theta) \sin \phi = x \cos \phi - y \sin \phi \]
    \[ y' = (r \cos \theta) \sin \phi + (r \sin \theta) \cos \phi = x \sin \phi + y \cos \phi \]
  – Or in matrix form
    \[ R_z(\phi) = \begin{pmatrix} \cos \phi & -\sin \phi \\ \sin \phi & \cos \phi \end{pmatrix} \]
Rotation in 3D

- **Rotation around major axes**
  
  - $R_x(\phi) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos \phi & -\sin \phi & 0 \\ 0 & \sin \phi & \cos \phi & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$
  
  - $R_y(\phi) = \begin{pmatrix} \cos \phi & 0 & \sin \phi & 0 \\ 0 & 1 & 0 & 0 \\ -\sin \phi & 0 & \cos \phi & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$
  
  - $R_z(\phi) = \begin{pmatrix} \cos \phi & -\sin \phi & 0 & 0 \\ \sin \phi & \cos \phi & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$

- 2D rotation around the respective axis
  
  - Assumes right-handed system, mathematically positive direction
  
  - Be aware of change in sign on sines in $R_y$ (off diagonal elements)

- Due to relative orientation of other axis
Rotation in 3D (2)

- **Properties**
  
  - \( R_a(0) = 1 \)
  
  - \( R_a(\theta)R_a(\phi) = R_a(\theta + \phi) = R_a(\phi)R_a(\theta) \)
    
    - Rotations around the same axis are commutative (special case)
  
  - In general: Not commutative
    
    - \( R_a(\theta)R_b(\phi) \neq R_b(\phi)R_a(\theta) \)
    
    - Order **does** matter for rotations around different axes
  
  - \( R_a^{-1}(\theta) = R_a(-\theta) = R_a^T(\theta) \)
    
    - Orthonormal matrix: Inverse is equal to the transpose
  
  - Determinant is 1
    
    - Volume preserving
Rotation Around Point

- **Rotate object around a point p and axis a**
  - Translate p to origin, rotate around axis a, translate back to p
  - \( R_a(p, \theta) = T(p)R_a(\theta)T(-p) \)

\[
B' = T(-p)B
\]
\[
B'' = R_z(\theta)B'
\]
\[
T(p)B''
\]
Rotation Around Some Axis

- Rotate around a given point \( p \) and vector \( r \) (\(|r|=1\))
  - Translate so that \( p \) is in the origin
  - Transform with rotation \( R=M^T \)
    - \( M \) given by orthonormal basis \((r,s,t)\) such that \( r \) becomes the \( x \) axis
    - Requires construction of a orthonormal basis \((r,s,t)\), see next slide
  - Rotate around \( x \) axis
  - Transform back with \( R^{-1} \)
  - Translate back to point \( p \)

\[
R(p, r, \phi) = T(p)M(r)R_x(\phi)M^T(r)T(-p)
\]
Compute orthonormal basis given a 3D vector \( r \)

- Using a numerically stable method
- Construct \( s \) such that it is normal to \( r \) (\( r \) being normalized)
  - Use fact that in 2D, orthogonal vector to \((x, y)\) is \((-y, x)\)
    - Do this in coordinate plane that has largest components
      \[
      \begin{align*}
      (0, -r_z, r_y), & \text{ if } x = \arg\min_{x, y, z}\{ |r_x|, |r_y|, |r_z| \} \\
      (-r_z, 0, r_x), & \text{ if } y = \arg\min_{x, y, z}\{ |r_x|, |r_y|, |r_z| \} \\
      (-r_y, r_x, 0), & \text{ if } z = \arg\min_{x, y, z}\{ |r_x|, |r_y|, |r_z| \}
      \end{align*}
      \]  
  - Normalize
    \[
    s = s'/|s'|
    \]  
- Compute \( t \) as cross product
  - \( t = r \times s \)
- \( r, s, t \) forms orthonormal basis, thus \( M \) transforms into this basis

\[
M(r) = \begin{pmatrix}
  r_x & s_x & t_x & 0 \\
  r_y & s_y & t_y & 0 \\
  r_z & s_z & t_z & 0 \\
  0 & 0 & 0 & 1
\end{pmatrix}, \text{ inverse is given as its transpose: } M^{-1} = M^T
\]
Concatenation of Transforms

- Multiply matrices to concatenate
  - Matrix-matrix multiplication is not commutative (in general)
  - Order of transformations matters!

\[ \begin{align*}
  \text{B} & \xrightarrow{T(1,1)} \text{R}_z(45^\circ)B \\
  \text{R}_z(45^\circ)B & \xrightarrow{T(1,1)} \text{R}_z(45^\circ)T(1,1)B
\end{align*} \]
Transformations

- **Line**
  - Transform end points
- **Plane**
  - Transform three points
- **Vector**
  - Translations to not act on vectors
- **Normal vectors (e.g. plane in Hesse form)**
  - Problem: e.g. with non-uniform scaling
Transforming Normals

• Dot product as matrix multiplication

\[ n \cdot v = n^T v = (n_x \ n_y \ n_z) \begin{pmatrix} v_x \\ v_y \\ v_z \end{pmatrix} \]

• Normal N on a plane
  – For any vector \( v \) in the plane: \( n^T v = 0 \)
  – Find transformation \( M' \) for normal vector, such that:

\[
(M' n)^T (M v) = 0 \quad \quad \quad M'^T M M^{-1} = 1 M^{-1}
\]

\[ n^T (M'^T M) v = 0 \quad \quad \quad \text{and thus} \quad \quad \quad M'^T = M^{-1} \quad \quad \quad M' = (M^{-1})^T \]

– \( M' \) is the adjoint of \( M \)
  • Exists even for non-invertible matrices
  • For \( M \) invertible and orthogonal \( M' = (M^{-1})^T = (M^T)^T = M \)

• Remember:

– Normals are transformed by the transpose of the inverse of the 4x4 transformation matrix of points and vectors