Geometric Spaces and Operations

Mathematical underpinnings of computer graphics

- Hierarchy of geometric spaces
  - Vector spaces
  - Affine spaces
  - Euclidean spaces
  - Cartesian spaces
  - Projective spaces
- Affine geometry and transformations
- Projective transformations and perspective
- Matrix formulations of transformations

Formally, a space is defined by

- A set of objects
• Operations on the objects
• Axioms defining invariant properties
Vector Spaces

Definition:

- Set of vectors $\mathcal{V}$
- Operations on $\vec{u}$, $\vec{v} \in \mathcal{V}$:
  - Addition: $\vec{u} + \vec{v} \in \mathcal{V}$
  - Scalar Multiplication: $\alpha \vec{u} \in \mathcal{V}$ where $\alpha \in$ some field $\mathcal{F}$
- Axioms
  - Unique zero element: $0 + \vec{u} = \vec{u}$
  - Field unit element: $1 \vec{u} = \vec{u}$
  - Addition commutative: $\vec{u} + \vec{v} = \vec{v} + \vec{u}$
  - Addition associative: $(\vec{u} + \vec{v}) + \vec{w} = \vec{u} + (\vec{v} + \vec{w})$
  - Distributive scalar multiplication: $\alpha(\vec{u} + \vec{v}) = \alpha \vec{u} + \alpha \vec{v}$
- Additional definitions
  - Let $\mathcal{B} = \{\vec{v}_1, \vec{v}_2, \cdots, \vec{v}_n\}$.
  - Then $\mathcal{B}$ spans $\mathcal{V}$ iff any $\vec{v} \in \mathcal{V}$ can be written as $\vec{v} = \sum_{i=1}^{n} \alpha_i \vec{v}_i$.
  - $\sum_{i=1}^{n} \alpha_i \vec{v}_i$ is called a linear combination of the vectors in $\mathcal{B}$.
  - $\mathcal{B}$ is called a basis of $\mathcal{V}$ if it is a minimal spanning set.
  - All bases of $\mathcal{V}$ contain the same number of vectors.
- The number of vectors in any basis of $\mathcal{V}$ is called the *dimension* of $\mathcal{V}$.

- **Comments:**
  - We are interested in 2 and 3 dimensional spaces.
  - No definition of distance (size) exists yet.
  - Angles and points have not been defined.
Affine Spaces

Definition:

- A set of vectors $\mathcal{V}$ and a set of points $\mathcal{P}$
- $\mathcal{V}$ is a vector space.
- Point-vector sum: $P + \vec{v} = Q$ with $P, Q \in \mathcal{P}$ and $\vec{v} \in \mathcal{V}$
- Additional definitions:
  - A frame $F = (\mathcal{B}, \mathcal{O})$ where $\mathcal{B} = \{\vec{v}_1, \vec{v}_2, \cdots, \vec{v}_n\}$ is a basis of $\mathcal{V}$ and the point $\mathcal{O}$ is called the origin of the frame.
  - The dimension of $F$ is the same as the dimension of $\mathcal{V}$.
- Comments:
  - Still no distances or angles
  - Closer to what we want for graphics
  - The space has no distinguished origin
Euclidean Spaces

Definition:

- A *metric space* is any space with a *distance metric* $d(P, Q)$ defined on its elements.
- Distance metric axioms:
  - $d(P, Q) \geq 0$
  - $d(P, Q) = 0$ iff $P = Q$
  - $d(P, Q) = d(Q, P)$
  - $d(P, Q) \leq d(P, R) + d(R, Q)$ (triangle inequality)
- *Euclidean* distance metric:
  $$d^2(P, Q) = (P - Q) \cdot (P - Q)$$

- Comments:
  - Euclidean metric based on dot product
  - Dot product defined on vectors
  - Distance metric defined on points
  - Distance is a property of the space, not a frame
• Dot product axioms:
  \[ (\vec{u} + \vec{v}) \cdot \vec{w} = \vec{u} \cdot \vec{w} + \vec{v} \cdot \vec{w} \]
  \[ \alpha (\vec{u} \cdot \vec{v}) = (\alpha \vec{u}) \cdot \vec{v} = \vec{u} \cdot (\alpha \vec{v}) \]
  \[ \vec{u} \cdot \vec{v} = \vec{v} \cdot \vec{u} \]

• Additional definitions:
  \[ \text{The norm of a vector } \vec{u} \text{ is given by } |\vec{u}| = \sqrt{\vec{u} \cdot \vec{u}}. \]
  \[ \text{Angles are defined by their cosines: } \cos(\angle \vec{u} \vec{v}) = \frac{\vec{u} \cdot \vec{v}}{|\vec{u}| |\vec{v}|} \]
  \[ \text{Orthogonal vectors: } \vec{u} \cdot \vec{v} = 0 \rightarrow \vec{u} \perp \vec{v} \]
**Cartesian Spaces**

*Definition:*

- A frame \((\vec{i}, \vec{j}, \vec{k}, \mathcal{O})\) is *orthonormal* iff
  - \(\vec{i}, \vec{j},\) and \(\vec{k}\) are *orthogonal*, i.e. \(\vec{i} \cdot \vec{j} = \vec{j} \cdot \vec{k} = \vec{k} \cdot \vec{i} = 0\) and
  - \(\vec{i}, \vec{j},\) and \(\vec{k}\) are *normal*, i.e. \(|\vec{i}| = |\vec{j}| = |\vec{k}| = 1\)

- Additional definitions:
  - The *standard frame* \(F_s = (\vec{i}, \vec{j}, \vec{k}, \mathcal{O})\)
  - Points can be distinguished from vectors using an extra coordinate
    - *0* for vectors: \(\vec{v} = (v_x, v_y, v_z, 0)\) means \(\vec{v} = v_x\vec{i} + v_y\vec{j} + v_z\vec{k}\)
    - *1* for points: \(P = (p_x, p_y, p_z, 1)\) means \(P = p_x\vec{i} + p_y\vec{j} + p_z\vec{k} + \mathcal{O}\)

- Comments
  - Coordinates have no meaning without an associated frame
  - There will be other ways to look at the extra coordinate
  - Sometimes we are sloppy and omit the extra coordinate
  - Assume standard frame unless specified otherwise
  - Points and vectors are different
  - Points and vectors have different operations
  - Points and vectors transform differently
Projective Space

Homogeneous Coordinates:

- Reminders
- Notation
- Imbedding

Projective Space:

- Division by the homogeneous coordinate
- Equivalence of affine points and homogeneous points
- Relationship with perspective
- More generally: rational splines
Homogeneous Coordinates

- Reminder about Spaces
  - Affine Space = Vector Space + Points
- Homogeneous notation
  - vector: \((x, y, z, 0)\)
  - point: \((x, y, z, 1)\)
- Imbedding of vectors and points in space of one higher dimension
Vector and Affine Algebra

- Difference of points

\[(x_1, 1) - (x_0, 1) = (x_1 - x_0, 0)\]
• Affine combination of points

\[(1 - t)(x_1, 1) + t(x_0, 1) = ((1 - t)x_1 + tx_0, 1)\]
Linear combinations of vectors

\[ a(v_0, 0) + b(v_1, 0) = (av_0 + bv_1, 0) \]
Homogeneous Coordinates

- Homogeneous coordinates represent $n$-space as a subspace of $n + 1$ space
- For instance, homogeneous 4-space embeds ordinary 3-space as the $w = 1$ hyperplane
- Thus, we can obtain the 3-d image of any homogeneous point $(wx, wy, wz, w)$, $w \neq 0$ as $(x, y, z, 1) = (wx/w, wy/w, wz/w, w/w)$, that is, by dividing all coordinates by $w$.
- Lines in homogeneous space which intersect the $w = 1$ hyperplane project to 3-space points.
- Notice that this is just a perspective projection from 4-d homogeneous space to 3-space, instead of dividing by $z$, we are dividing by $w$. 
Projective Space

- Divide through by $w$
  $$(x, w) \rightarrow \left(\frac{x}{w}, 1\right)$$
- All homogeneous points of the form $\alpha(x, 1), \alpha > 0$ are equivalent
- *Projects* homogeneous points *centrally* onto the affine plane
Relationship to Perspective:

- In rendering, the $w$ values we generate are proportional to $z$
  - Equivalence corresponds to perspective projection

More Generally: Rational splines

- Homogeneous spline curve
  - Spline curve of the form

\[
\begin{bmatrix}
  x(t) \\
y(t) \\
z(t) \\
w(t)
\end{bmatrix} = \sum_{i=0}^{n} \begin{bmatrix}
x_i \\
y_i \\
z_i \\
w_i
\end{bmatrix} B_i^d(t) = \begin{bmatrix}
\sum_{i=0}^{n} x_i B_i^d(t) \\
\sum_{i=0}^{n} y_i B_i^d(t) \\
\sum_{i=0}^{n} z_i B_i^d(t) \\
\sum_{i=0}^{n} w_i B_i^d(t)
\end{bmatrix}
\]
• Rational spline curve
  – Affine projective spline curve

\[
\begin{bmatrix}
\tilde{x}(t) \\
\tilde{y}(t) \\
\tilde{z}(t)
\end{bmatrix} = \left[ \begin{array}{c}
\sum_{i=0}^{n} x_i B_i^d(t) / \sum_{i=0}^{n} w_i B_i^d(t) \\
\sum_{i=0}^{n} y_i B_i^d(t) / \sum_{i=0}^{n} w_i B_i^d(t) \\
\sum_{i=0}^{n} z_i B_i^d(t) / \sum_{i=0}^{n} w_i B_i^d(t)
\end{array} \right]
\]

\[
= \frac{1}{\sum_{i=0}^{n} w_i B_i^d(t)} \sum_{i=0}^{n} \begin{bmatrix}
x_i \\
y_i \\
z_i
\end{bmatrix} B_i^d(t)
\]
Linear Transformations

Vector space $\mathcal{V}$

- Linear combinations of vectors in $\mathcal{V}$ are in $\mathcal{V}$
- For $\vec{u}, \vec{v} \in \mathcal{V}$
  - $\vec{u} + \vec{v} \in \mathcal{V}$
  - $\alpha \vec{u} \in \mathcal{V}$ for any scalar $\alpha$
  - In general, $\sum_i \alpha_i \vec{u}_i \in \mathcal{V}$ for any scalars $\alpha_i$
- Linear transformations
  - Let $T: \mathcal{V}_0 \mapsto \mathcal{V}_1$, where $\mathcal{V}_0$ and $\mathcal{V}_1$ are vector spaces
  - Then $T$ is linear iff
    * $T(\vec{u} + \vec{v}) = T(\vec{u}) + T(\vec{v})$
    * $T(\alpha \vec{u}) = \alpha T(\vec{u})$
    * In general, $T(\sum_i \alpha_i \vec{u}_i) = \sum_i \alpha_i T(\vec{u}_i)$
Affine Transformations

Affine space $\mathcal{A} = (\mathcal{V}, \mathcal{P})$

- For $\vec{u} \in \mathcal{V}$ and $P \in \mathcal{P}$
  \[ P + \vec{u} \in \mathcal{P} \]

- Define point subtraction:
  - For $P, Q \in \mathcal{P}$ and $\vec{u} \in \mathcal{V}$, if $P + \vec{u} = Q$, then $Q - P \equiv \vec{u}$
  - So in general we have $\sum_i \alpha_i P_i$ is a vector iff $\sum_i \alpha_i = 0$

- Define point blending:
  - For $P, P_1, P_2 \in \mathcal{P}$ and scalar $\alpha$, if $P = P_1 + \alpha (P_2 - P_1)$ then $P \equiv (1 - \alpha) P_1 + \alpha P_2$
  - This can also be written $P \equiv \alpha_1 P_1 + \alpha_2 P_2$ where $\alpha_1 + \alpha_2 = 1$
  - So in general we have $\sum_i \alpha_i P_i$ is a point iff $\sum_i \alpha_i = 1$

- Geometrically, we have $\frac{|P - P_0|}{|P - P_1|} = \frac{d_1}{d_2}$ or $P = \frac{d_1 P_1 + d_2 P_2}{d_1 + d_2}$

- Vectors can always be combined linearly $\sum_i \alpha_i \vec{u}_i$

- Points can be combined linearly $\sum_i \alpha_i P_i$ iff
  - The coefficients sum to 1, giving a point ("affine combination")
  - The coefficients sum to 0, giving a vector ("vector combination")
Example affine combination:

\[ P(t) = P_0 + t(P_1 - P_0) = (1 - t)P_0 + tP_1 \]

This says any point on the line is an affine combination of the line segment’s endpoints.

Affine transformations
- Let \( T : A_0 \mapsto A_1 \) where \( A_0 \) and \( A_1 \) are affine spaces
- \( T \) is said to be an affine transformation iff
  * \( T \) maps vectors to vectors and points to points
  * \( T \) is a linear transformation on the vectors
  * \( T(P + \vec{u}) = T(P) + T(\vec{u}) \)

Properties of affine transformations
- \( T \) preserves affine combinations:
  \[ T(\alpha_0 P_0 + \cdots + \alpha_n P_n) = \alpha_0 T(P_0) + \cdots + \alpha_n T(P_n) \]
  where \( \sum_i \alpha_i = 0 \) or \( \sum_i \alpha_i = 1 \)
- \( T \) maps lines to lines:
  \[ T((1 - t)P_0 + tP_1) = (1 - t)T(P_0) + tT(P_1) \]
* \( T \) is affine iff it preserves ratios of distance along a line:

\[
P = \frac{d_0 P_0 + d_1 P_1}{d_0 + d_1} \Rightarrow T(P) = \frac{d_0 T(P_0) + d_1 T(P_1)}{d_0 + d_1}
\]

* \( T \) maps parallel lines to parallel lines (can you prove this?)

- Example affine transformations
  * Rigid body motions (translations, rotations)
  * Scales, reflections
  * Shears
Matrix Representation of Transformations

- Let $\mathcal{A}_0$ and $\mathcal{A}_1$ be affine spaces.
  Let $T : \mathcal{A}_0 \mapsto \mathcal{A}_1$ be an affine transformation.
  Let $F_0 = (\vec{i}_0, \vec{j}_0, \mathcal{O}_0)$ be a frame for $\mathcal{A}_0$.
  Let $F_1 = (\vec{i}_1, \vec{j}_1, \mathcal{O}_1)$ be a frame for $\mathcal{A}_1$.

- Let $P = x\vec{i}_0 + y\vec{j}_0 + \mathcal{O}_0$ be a point in $\mathcal{A}_0$.
  The coordinates of $P$ relative to $\mathcal{A}_0$ are $(x, y, 1)$.

This can also be represented in vector form as $P = \begin{bmatrix} \vec{i}_0 & \vec{j}_0 & \mathcal{O}_0 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$
• What are the coordinates \((x', y', 1)\) of \(T(P)\) relative to \(F_1\)?
  - An affine transformation is characterized by the image of a frame in the domain.

  \[
  T(P) \quad = \quad T(x\vec{i}_0 + y\vec{j}_0 + O_0) \\
  = \quad xT(\vec{i}_0) + yT(\vec{j}_0) + T(O_0)
  \]

  - \(T(\vec{i}_0)\) must be a linear combination of \(\vec{i}_1\) and \(\vec{j}_1\),
    say \(T(\vec{i}_0) = t_{1,1}\vec{i}_1 + t_{2,1}\vec{j}_1\).

  - Likewise \(T(\vec{j}_0)\) must be a linear combination of \(\vec{i}_1\) and \(\vec{j}_1\),
    say \(T(\vec{j}_0) = t_{1,2}\vec{i}_1 + t_{2,2}\vec{j}_1\).

  - Finally \(T(O_0)\) must be an affine combination of \(\vec{i}_1\),
    \(\vec{j}_1\), and \(O_1\), say \(T(O_0) = t_{1,3}\vec{i}_1 + t_{2,3}\vec{j}_1 + O_1\).
– Then by substitution we get

\[ T(P) = x(t_{1,1} \vec{i}_1 + t_{2,1} \vec{j}_1) + y(t_{1,2} \vec{i}_1 + t_{2,2} \vec{j}_1) + t_{1,3} \vec{i}_1 + t_{2,3} \vec{j}_1 + O_1 \]

\[ = \begin{bmatrix} t_{1,1} \vec{i}_1 + t_{2,1} \vec{j}_1 & t_{1,2} \vec{i}_1 + t_{2,2} \vec{j}_1 & t_{1,3} \vec{i}_1 + t_{2,3} \vec{j}_1 + O_1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \]

\[ = \begin{bmatrix} \vec{i}_1 & \vec{j}_1 & O_1 \end{bmatrix} \begin{bmatrix} t_{1,1} & t_{1,2} & t_{1,3} \\ t_{2,1} & t_{2,2} & t_{2,3} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \]

Using \( M_T \) to denote the matrix, we see that \( F_0 = F_1 M_T \)

• Let \( T(P) = P' = x' \vec{i}_1 + y' \vec{j}_1 + O_1 \)
In vector form this is

\[ p' = \begin{bmatrix} \tilde{x}' \\ \tilde{y}' \\ 1 \end{bmatrix} = \begin{bmatrix} t_{1,1} & t_{1,2} & t_{1,3} \\ t_{2,1} & t_{2,2} & t_{2,3} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \]

So we see that

\[ \begin{bmatrix} \tilde{x}' \\ \tilde{y}' \\ 1 \end{bmatrix} = \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = M_{TP} \]

We can write this in shorthand \(-p' = M_{TP}\).
• **Translation**
  
  - Points are transformed as \([x' \ y' \ 1]^T = [x \ y \ 1]^T + [\Delta x \ \Delta y \ 0]^T\).
  
  - Vectors don’t change.
  
  - Thus translation is affine but not linear.
    
    If it were linear, we would have \(T(P + Q) = T(P) + T(Q)\), but point addition is undefined.
  
  - Translation can be applied to sums of vectors and vector-point sums.
  
  - Matrix formulation:
    
    \[
    \begin{bmatrix}
    x' \\
    y' \\
    1
    \end{bmatrix}
    =
    \begin{bmatrix}
    1 & 0 & \Delta x \\
    0 & 1 & \Delta y \\
    0 & 0 & 1
    \end{bmatrix}
    \begin{bmatrix}
    x \\
    y \\
    1
    \end{bmatrix}
    =
    \begin{bmatrix}
    x + \Delta x \\
    y + \Delta y \\
    1
    \end{bmatrix}
    \]
    
    \[
    \begin{bmatrix}
    x' \\
    y' \\
    0
    \end{bmatrix}
    =
    \begin{bmatrix}
    1 & 0 & \Delta x \\
    0 & 1 & \Delta y \\
    0 & 0 & 1
    \end{bmatrix}
    \begin{bmatrix}
    x \\
    y \\
    0
    \end{bmatrix}
    =
    \begin{bmatrix}
    x \\
    y \\
    0
    \end{bmatrix}
    \]
  
  - Shorthand for the above matrix: \(T(\Delta x, \Delta y)\)
• **Scale**
  
  – Linear transform — applies equally to points and vectors
  – Points transform as \( [x' \ y' \ 1]^T = [x \ S_x \ yS_y \ 1]^T \).
  – Vectors transform as \( [x' \ y' \ 0]^T = [x \ S_x \ yS_y \ 0]^T \).
  – Matrix formulation:

\[
\begin{bmatrix}
  x' \\
  y' \\
  1
\end{bmatrix}
= \begin{bmatrix}
  S_x & 0 & 0 \\
  0 & S_y & 0 \\
  0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
  x \\
  y \\
  1
\end{bmatrix}
= \begin{bmatrix}
  xS_x \\
  yS_y \\
  1
\end{bmatrix}
\]

\[
\begin{bmatrix}
  x' \\
  y' \\
  0
\end{bmatrix}
= \begin{bmatrix}
  S_x & 0 & 0 \\
  0 & S_y & 0 \\
  0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
  x \\
  y \\
  0
\end{bmatrix}
= \begin{bmatrix}
  xS_x \\
  yS_y \\
  0
\end{bmatrix}
\]

  – Shorthand for the above matrix: \( S(S_x, S_y) \)
  – Note that this is *origin sensitive*.
  – How do you do reflections?
**Rotate**

- Linear transform — applies equally to points and vectors

  \[
  \begin{bmatrix}
  x' \\
  y'
  \end{bmatrix}^T = \begin{bmatrix}
  x \\
  y
  \end{bmatrix}^T \begin{bmatrix}
  \cos(\theta) & -\sin(\theta) \\
  \sin(\theta) & \cos(\theta)
  \end{bmatrix} = [x' y']^T = [x' y']^T = \begin{bmatrix}
  x \\
  y
  \end{bmatrix}^T \begin{bmatrix}
  \cos(\theta) & \sin(\theta) \\
  -\sin(\theta) & \cos(\theta)
  \end{bmatrix} = \begin{bmatrix}
  x' \\
  y'
  \end{bmatrix}
  \]

- Vectors transform as:

  \[
  \begin{bmatrix}
  x \\
  y
  \end{bmatrix} = \begin{bmatrix}
  \cos(\theta) & \sin(\theta) \\
  -\sin(\theta) & \cos(\theta)
  \end{bmatrix} \begin{bmatrix}
  x \\
  y
  \end{bmatrix}
  \]

- Matrix formulation:

  \[
  \begin{bmatrix}
  x' \\
  y'
  \end{bmatrix} = \begin{bmatrix}
  \cos(\theta) & \sin(\theta) \\
  -\sin(\theta) & \cos(\theta)
  \end{bmatrix} \begin{bmatrix}
  x \\
  y
  \end{bmatrix}
  \]

- Shorthand for the above matrix: \( R(\theta) \)

- Note that this is origin sensitive.
• **Shear**
  
  - Linear transform — applies equally to points and vectors
  - Points transform as \([x' \ y' \ 1]^{T} = [x + \alpha y, \ y + \beta x, \ 1]^{T}\).
  - Vectors transform as \([x' \ y' \ 0]^{T} = [x + \alpha y, \ y + \beta x, \ 0]^{T}\).
  - Matrix formulation:

\[
\begin{bmatrix}
  x' \\
y' \\
1
\end{bmatrix} = \begin{bmatrix}
  1 & \alpha & 0 \\
  \beta & 1 & 0 \\
0 & 0 & 1
\end{bmatrix} \begin{bmatrix}
  x \\
y \\
1
\end{bmatrix} = \begin{bmatrix}
  x + \alpha y \\
y + \beta x \\
1
\end{bmatrix}
\]

\[
\begin{bmatrix}
  x' \\
y' \\
0
\end{bmatrix} = \begin{bmatrix}
  1 & \alpha & 0 \\
  \beta & 1 & 0 \\
0 & 0 & 1
\end{bmatrix} \begin{bmatrix}
  x \\
y \\
0
\end{bmatrix} = \begin{bmatrix}
  x + \alpha y \\
y + \beta x \\
0
\end{bmatrix}
\]

- Shorthand for the above matrix: \(Sh(\alpha, \beta)\)
• Composition of Transformations
  – Now we have some basic transformations, how do we create and represent arbitrary affine transformations?
  – We can derive an arbitrary affine transform as a sequence of basic transformations, then compose the transformations
  – Example — scaling about an arbitrary point $[x_c \ y_c \ 1]^T$
    1. Translate $[x_c \ y_c \ 1]^T$ to $[0 \ 0 \ 1] \ T(-x_c, -y_c)$
    2. Scale $[x' \ y' \ 1]^T = S(S_x, S_y) \ [x \ y \ 1]^T$
    3. Translate $[0 \ 0 \ 1]^T$ back to $[x_c \ y_c \ 1] \ T(x_c, y_c)$
  – The sequence of transformation steps is
    $T(-x_c, -y_c) \circ S(S_x, S_y) \circ T(x_c, y_c)$
– In matrix form this is

\[
\begin{bmatrix}
x' \\
y' \\
1
\end{bmatrix} = \begin{bmatrix}
1 & 0 & x_c \\
0 & 1 & y_c \\
0 & 0 & 1
\end{bmatrix} \begin{bmatrix}
S_x & 0 & 0 \\
0 & S_y & 0 \\
0 & 0 & 1
\end{bmatrix} \begin{bmatrix}
1 & 0 & -x_c \\
0 & 1 & -y_c \\
0 & 0 & 1
\end{bmatrix} \begin{bmatrix}
x \\
y \\
1
\end{bmatrix}
\]

\[
= \begin{bmatrix}
S_x & 0 & x_c(1 - S_x) \\
0 & S_y & y_c(1 - S_y) \\
0 & 0 & 1
\end{bmatrix} \begin{bmatrix}
x \\
y \\
1
\end{bmatrix}
\]

– Note that the matrices are arranged from right to left in the order of the steps.
– The order is important (why)?
• Three Dimensional Transformations
  – A point is $\mathbf{p} = [x \ y \ z \ 1]$, a vector $\mathbf{v} = [x \ y \ z \ 0]$
  – Translation:
    $T(\Delta x, \Delta y, \Delta z) = \begin{bmatrix}
      1 & 0 & 0 & \Delta x \\
      0 & 1 & 0 & \Delta y \\
      0 & 0 & 1 & \Delta z \\
      0 & 0 & 0 & 1
    \end{bmatrix}$
  – Scale:
    $S(S_x, S_y, S_z) = \begin{bmatrix}
      S_x & 0 & 0 & 0 \\
      0 & S_y & 0 & 0 \\
      0 & 0 & S_z & 0 \\
      0 & 0 & 0 & 1
    \end{bmatrix}$
  – Rotation:
    $R_z(\Theta) = \begin{bmatrix}
      \cos(\Theta) & -\sin(\Theta) & 0 & 0 \\
      \sin(\Theta) & \cos(\Theta) & 0 & 0 \\
      0 & 0 & 1 & 0 \\
      0 & 0 & 0 & 1
    \end{bmatrix}$
Projections and Projective Transformations

**Perspective Projection**

- Identify all points with a line through the eyepoint.
- Slide lines with viewing plane, take intersection point as projection.
- This is *not* an affine transformation, but a *projective transformation*.

**Projective Transformations:**

- Angles are not preserved.
- Distances are not preserved.
- Ratios of distances are not preserved.
- Affine combinations are not preserved.
- Straight lines are mapped to straight lines.
- Incidence relationships are preserved in a general way.
- *Cross ratios* are preserved.
Comparisons

<table>
<thead>
<tr>
<th><strong>Affine Transformations</strong></th>
<th><strong>Projective Transformations</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Image of 2 points on a line</td>
<td>Image of 3 points on a line</td>
</tr>
<tr>
<td>determine image of line</td>
<td>determine image of line</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Image of 3 points on a plane</td>
<td>Image of 4 points on a plane</td>
</tr>
<tr>
<td>determine image of plane</td>
<td>determine image of plane</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>In dimension ( n ) space,</td>
<td>In dimension ( n ) space,</td>
</tr>
<tr>
<td>image of ( n + 1 ) points/vectors</td>
<td>image of ( n + 2 ) points/vectors</td>
</tr>
<tr>
<td>defines affine map.</td>
<td>defines projective map.</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Vectors map to vectors</td>
<td>Vectors can map to vectors or points</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Points map to points</td>
<td>Points can map to vectors or points</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Can represent with matrix multiply</td>
<td>Can represent with matrix multiply and normalization.</td>
</tr>
</tbody>
</table>
Perspective Map

- Given a point \( P \), we want to find its projection \( P' \).

\[
p'=(x',y',n)
\]

Projection plane, \( z = n \)

- Similar triangles: \( P' = (xn/z, n) \)
- In 3D, \( (x', y', z') \mapsto (xn/z, yn/z, n) \)
- Have identified all points on a line through the origin with a point in the projection plane.
- Thus, \( (x, y, z) \equiv (kx, ky, kz), k \neq 0 \).
- These are known as homogeneous coordinates.
- If we have solids, or colored lines, then we need to know “which one is in front.”
- This map loses all \( z \) information, so it is inadequate.
Why Map Z

• 3D $\mapsto$ 2D projections map all $z$ to same value.
• Need $z$ to determine occlusion, so a 3D to 2D projective transformation doesn’t work.
• Further, we want 3D lines to map to 3D lines (this is useful in hidden surface removal).
• The mapping $(x, y, z, 1) \mapsto (xn/z, yn/z, n, 1)$ maps lines to lines, but loses all depth information.
• We could use

$$ (x, y, z, 1) \mapsto (xn/z, yn/z, z, 1) $$

Thus, if we map the endpoints of a line segment, these end points will have the same relative depths after this mapping.
BUT: It fails to map lines to lines
• The map

$$ (x, y, z, 1) \mapsto \left( \frac{xn}{z}, \frac{yn}{z}, \frac{zf + zn - 2fn}{z(f - n)}, 1 \right) $$

does map lines to lines, and it preserves depth information.
Mapping Z

• It’s clear how $x$ and $y$ map. How about $z$?

$$z \mapsto \frac{zf + zn - 2fn}{z(f - n)} = P(z)$$

• We know $P(f) = 1$ and $P(n) = -1$. What maps to 0?

$$P(z) = 0$$

$$\Rightarrow \quad \frac{zf + zn - 2fn}{z(f - n)} = 0$$

$$\Rightarrow \quad z = \frac{2fn}{f + n}$$

Note that $f^2 + 2f > 2fn/(f + n) > fn + n^2$ so

$$f > \frac{2fn}{f + n} > n$$
• What happens as map \( z \) to 0 or to infinity?

\[
\begin{align*}
\lim_{z \to 0^+} P(z) &= \frac{-2fn}{z(f - n)} \\
&= -\infty \\
\lim_{z \to 0^-} P(z) &= \frac{-2fn}{z(f - n)} \\
&= +\infty \\
\lim_{z \to +\infty} P(z) &= \frac{z(f + n)}{z(f - n)} \\
&= \frac{f + n}{f - n} \\
\lim_{z \to -\infty} P(z) &= \frac{z(f + n)}{z(f - n)} \\
&= \frac{f + n}{f - n}
\end{align*}
\]
• What happens if we vary \( f \) and \( n \)?

\[
\lim_{f \to n} P(z) = \frac{z(f + n) - 2fn}{z(f - n)} = \frac{(2zn - 2n^2)}{z \cdot 0}
\]

which is not surprising, since we’re trying to map a single point to a line segment.

\[
\lim_{f \to \infty} P(z) = \frac{zf - 2fn}{zf} = \frac{z - 2n}{z}
\]

• But note that this means we are mapping an infinite region to \([0,1]\) and we will effectively get a far plane due to floating point precision,

\[
\lim_{n \to 0} P(z) = \frac{zf}{zf} = 1
\]
i.e., the entire map becomes constant (again, we are mapping a point to an interval).
- Consider what happens as $f$ and $n$ move away from each other.

- We are interested in the size of the regions $[n, 2f/(f+n)]$ and $[2fn/(f+n), f]$.

- When $f$ is large compared to $n$, we have

\[
\frac{2fn}{f+n} = 2n
\]

So

\[
\frac{2fn}{f+n} - n = n
\]

and

\[
f - \frac{2fn}{f+n} = f - 2n
\]

Thus, as we move the clipping planes away from one another, the far interval is compressed more than the near one. With floating point arithmetic, this means we'll lose precision.

- In the extreme case, think about what happens as we move $f$ to infinity: we compress an infinite region to an finite one.

- Therefore, we try to place our clipping planes as close to one another as we can.
Clipping in Homogeneous Space

*Projection: linear transformations then normalize*

- **Linear transformation**
  \[
  \begin{bmatrix}
    nr & 0 & 0 & 0 \\
    0 & ns & 0 & 0 \\
    0 & 0 & \frac{f+n}{f-n} & -\frac{2fn}{f-n} \\
    0 & 0 & 1 & 0
  \end{bmatrix}
  \begin{bmatrix}
    x \\
    y \\
    z \\
    1
  \end{bmatrix}
  =
  \begin{bmatrix}
    \tilde{x} \\
    \tilde{y} \\
    \tilde{z} \\
    \tilde{w}
  \end{bmatrix}
  
  \]

- **Normalization**
  \[
  \begin{bmatrix}
    \bar{x} \\
    \bar{y} \\
    \bar{z} \\
    \bar{w}
  \end{bmatrix}
  =
  \begin{bmatrix}
    \tilde{x}/\tilde{w} \\
    \tilde{y}/\tilde{w} \\
    \tilde{z}/\tilde{w} \\
    1
  \end{bmatrix}
  =
  \begin{bmatrix}
    X \\
    Y \\
    Z \\
    1
  \end{bmatrix}
  \]
Region Mapping

\[ 8 \]
\[ \begin{array}{c}
10 \\
11 \\
12 \\
\hline
7 \\
8 \\
9 \\
\hline
4 \\
5 \\
6 \\
\hline
2 \\
3 \\
0 \\
\hline
x \\
8
\end{array} \]

\[ \begin{array}{c}
3 \\
2 \\
1 \\
\hline
10 \\
11 \\
12 \\
\hline
7 \\
8 \\
9 \\
\hline
4 \\
5 \\
6 \\
\hline
4 \\
5 \\
6 \\
\hline
8
\end{array} \]
Clipping not good after normalization:

- Ambiguity after normalization
  \[-1 \leq \frac{\bar{x}, \bar{y}, \bar{z}}{\bar{w}} \leq +1\]
  - Numerator can be positive or negative
  - Denominator can be positive or negative
- Normalization expended on points that are subsequently clipped.

Clipping in homogeneous coordinates:

- Compare unnormalized coordinate against \( \bar{w} \)
  \[-|\bar{w}| \leq x, y, z \leq +|\bar{w}|\]
Cross Ratio

Definition:
\[ x = \frac{CA}{CB} = \frac{DA}{DB} \]

\[
\begin{align*}
\text{area } OCA &= \frac{1}{2} \cdot h \cdot CA = \frac{1}{2} \cdot OA \cdot OC \sin \angle COA \\
\text{area } OCB &= \frac{1}{2} \cdot h \cdot CB = \frac{1}{2} \cdot OB \cdot OC \sin \angle COB \\
\text{area } ODA &= \frac{1}{2} \cdot h \cdot DA = \frac{1}{2} \cdot OA \cdot OD \sin \angle DOA \\
\text{area } ODB &= \frac{1}{2} \cdot h \cdot DB = \frac{1}{2} \cdot OB \cdot OD \sin \angle DOB
\end{align*}
\]
Hence

\[
\frac{CA}{DA} = \frac{OA \cdot OC \sin \angle COA}{OB \cdot OD \sin \angle DOB} = \frac{\sin \angle COA}{\sin \angle COB} = \frac{\sin \angle DOB}{\sin \angle DOA}
\]

Invariance of cross-ratio under central projection
Invariance of cross-ratio under parallel projection
\[(ABCD) > 0\]

\[(ABCD) < 0\]

Sign of cross-ratio

\[(ABCD) = \frac{CA}{CB} \left/ \frac{DA}{DB} \right. = \frac{x_3 - x_1}{x_3 - x_2} \left/ \frac{x_4 - x_1}{x_4 - x_2} \right. = \frac{x_3 - x_1}{x_3 - x_2} \cdot \frac{x_4 - x_2}{x_4 - x_1}\]
Cross-ratio in terms of coordinates.