Aliasing and Anti-Aliasing
• Hearn & Baker, chapter 4-17

2D transforms
• Hearn & Baker, chapter 5

Problem: “jaggies”

Also known as “aliasing.” It results from sampling a pattern (signal) at limited resolution.
Aliasing and Anti-Aliasing - Examples

Aliasing

Anti-Aliasing

Aliasing in photos

- A piece of practical knowledge
  - Example web page
  - Often better to blur before resizing
Aliasing and Anti-Aliasing

To fully understand the causes of aliasing requires the use of Signal Processing and Fourier Transforms.

We can look at Aliasing from a more intuitive basis.
Aliasing and Anti-Aliasing

Effects of Sampling

Aliasing

Aliasing and Anti-Aliasing

Aliasing
Aliasing and Anti-Aliasing

Anti-Aliasing is the process that attempts to prevent or fix the problem.

More Resolution
Unweighted Area Sampling
Weighted Area Sampling
Post image creation filtering
Aliasing and Anti-Aliasing

Unweighted Area Sampling

Pixel

Sub Pixels

$255 \times \frac{4}{9} = 113$
Aliasing and Anti-Aliasing

Weighted Area Sampling

Pixel

Sub Pixels

Weights

1 2 1
2 4 2
1 2 1

\[ 255 \times \frac{(1+2+1+2)}{16} = 96 \]

Post image creation filtering

\[ = \frac{3}{9} \times 255 = 85 \]
Aliasing and Anti-Aliasing

Post-image processing:
- fewer samples (less work)
- more blur (worse quality)

Weighted area sampling:
- slightly less blurry than unweighted

Matrix Transformations

Basic Graphics Transforms
- Translation
- Scaling
- Rotation
- Reflection
- Shear

All can be expressed as linear functions of the original coordinates:

\[
x' = Ax + By + C
\]
\[
y' = Dx + Ey + F
\]

\[
\begin{bmatrix}
x' \\
y'
\end{bmatrix} = \begin{bmatrix}
A & B & C \\
D & E & F \\
0 & 0 & 1
\end{bmatrix} \begin{bmatrix}
x \\
y \\
1
\end{bmatrix}
\]
Affine Transformations

- Preserve Parallel Lines
- Finite Points Map to Finite Points
- Translation, Rotation, and Reflection Preserve:
  - Angles
  - Lengths

2-D Transformations - Translation

\[ x' = x + T_x \]
\[ y' = y + T_y \]

Vectors: \( \vec{p}' = \vec{p} + \vec{T} \)

Matrices: \( [P'] = [P] + [T] \)

\[
\begin{bmatrix}
  P'_x \\
  P'_y
\end{bmatrix}
= \begin{bmatrix}
  P_x \\
  P_y
\end{bmatrix}
+ \begin{bmatrix}
  T_x \\
  T_y
\end{bmatrix}
= \begin{bmatrix}
  P_x + T_x \\
  P_y + T_y
\end{bmatrix}
\]

Transform Polygons by Transforming Each Vertex
2-D Transformations - Scaling

\[ x' = S_x \cdot x \]
\[ y' = S_y \cdot y \]

\[
\begin{bmatrix}
  P'_{x} \\
  P'_{y}
\end{bmatrix} =
\begin{bmatrix}
  S_x & 0 \\
  0 & S_y
\end{bmatrix}
\begin{bmatrix}
  P_{x} \\
  P_{y}
\end{bmatrix} =
\begin{bmatrix}
  S_x \cdot P_{x} \\
  S_y \cdot P_{y}
\end{bmatrix}
\]

Uniform Scaling: \( S_x = S_y \)

Differential Scaling: \( S_x \neq S_y \)

2-D Transformations - Rotation

\[ x' = r \cos \Theta = x \]
\[ y' = r \sin \Theta = y \]

\[ x' = r \cos(\Theta + \phi) = r \cos \Theta \cos \phi - r \sin \Theta \sin \phi \]
\[ y' = r \sin(\Theta + \phi) = r \cos \Theta \sin \phi + r \sin \Theta \cos \phi \]

\[ \begin{bmatrix}
  P'_{x} \\
  P'_{y}
\end{bmatrix} =
\begin{bmatrix}
  \cos \phi & -\sin \phi \\
  \sin \phi & \cos \phi
\end{bmatrix}
\begin{bmatrix}
  P_{x} \\
  P_{y}
\end{bmatrix}
\]
Shear: An action or stress resulting from applied forces that causes two contiguous parts of a body to slide relatively to each other.

\[
\begin{bmatrix}
1 & a \\
0 & 1 \\
\end{bmatrix}
\]
Shear Translation in X

\[
\begin{bmatrix}
1 & 0 \\
0 & b \\
\end{bmatrix}
\]
Shear Translation in Y
Shear Transforms

Properties:
It is an Affine Transform
a and b are the proportionality constant

A shear transform application:
Fast and efficient rotation
It will be used in perspective transformations

Shear is easily invertable

Shear rotation

\[
\begin{bmatrix}
1 & \alpha \\
0 & 1
\end{bmatrix}
\begin{bmatrix}
1 & 0 \\
\beta & 1
\end{bmatrix}
\begin{bmatrix}
1 \\
0
\end{bmatrix}
=
\begin{bmatrix}
\cos \theta & -\sin \theta \\
\sin \theta & \cos \theta
\end{bmatrix}
\]

\[\alpha = -\tan \left(\frac{\theta}{2}\right)\]
\[\beta = \sin \theta\]
\[\lambda = -\tan \left(\frac{\theta}{2}\right)\]
Reflection Transforms

Reflection is a transform that allows vectors to be flipped about an axis as if reflected in a mirror.

\[
\begin{bmatrix}
-1 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 1
\end{bmatrix}
\quad \begin{bmatrix}
1 & 0 & 0 \\
0 & -1 & 0 \\
0 & 0 & 1
\end{bmatrix}
\quad \begin{bmatrix}
1 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & -1
\end{bmatrix}
\]

Reflection About YZ  Reflection About XZ  Reflection About XY
2-D Transformations

- Unify Transformation Operations
- All Transformations Represented as Matrix Products

\[
[T] = \begin{bmatrix} 1 & 0 & T_x \\ 0 & 1 & T_y \\ 0 & 0 & 1 \end{bmatrix}, \quad [S] = \begin{bmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad [R(\phi)] = \begin{bmatrix} \cos \phi & -\sin \phi & 0 \\ \sin \phi & \cos \phi & 0 \\ 0 & 0 & 1 \end{bmatrix}
\]

\[
[P'] = [T][P], \quad [P'] = [S][P], \quad [P'] = [R(\phi)][P]
\]

Homogeneous Coordinates
\[
[P] = \begin{bmatrix} P_x \\ P_y \\ 1 \end{bmatrix}
\]

NOT the z coordinate

Transform Combinations

- Scaling About a Point
- Translate to Origin
- Apply Scale Matrix
- Translate Back to Original Position

\[
[P'] = [T_2][S][T_1][P]
\]
Transform Combinations

- Rotation About a Point
  - Translate to Origin
  - Apply Rotation Matrix
  - Translate Back to Original Position

\[
[P'] = [T_2][R(\phi)][T_1][P]
\]

Homogeneous coordinates in 2D

- How to represent perspective projection
  - It is hard!

Given the coordinates of the orange point find the coordinates of the blue point

Points: \((x,y) \rightarrow (x,y,1)\)

Equivalency:
\((x,y,w) \equiv (x/a, y/a, w/a)\)
An arbitrary camera?

- Need some systematic way of doing this

Approach:

One thing at a time
- First transform everything so that the camera is in the canonic position
  - Matrix M
- Then perform projection
  - Matrix P
- Combine the result $P^*M$