

Capturing, Modeling, Rendering 3D Structures

Image Warping

Image Warping

- We can divide image warping into
 - Simple spatial transformations (no per-pixel depth information)
 - Full 3D transformations (needs per-pixel depth information) (sometimes called "3D Image Warping")

Spatial Transformations

- A geometric relationship between input (u,v) and output pixels (x,y)

– Forward mapping:
 $(x,y) = (X(u,v), Y(u,v))$

– Inverse mapping:
 $(u,v) = (U(x,y), V(x,y))$



Spatial Transformations

- Matrix form is
 $[x, y, w] = [u, v, w] T$

$$\text{where } T = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$$

and operates in the "homogeneous coordinate system".

(examples from Wolberg)

Affine Transformations

- Matrix form is
 $[x, y, w] = [u, v, w] T$

$$\text{where } T = \begin{bmatrix} a_{11} & a_{12} & 0 \\ a_{21} & a_{22} & 0 \\ a_{31} & a_{32} & 1 \end{bmatrix}$$

and accommodates translations, rotations, scale, and shear.

(examples from Wolberg)

Affine Transformations

- The T matrix can be inferred from correspondences

Affine Transformations

- The T matrix can be inferred from correspondences

$$\begin{bmatrix} x_0 & y_0 & 1 \\ x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \end{bmatrix} = \begin{bmatrix} u_0 & v_0 & 1 \\ u_1 & v_1 & 1 \\ u_2 & v_2 & 1 \end{bmatrix} \begin{bmatrix} a_{11} & a_{12} & 0 \\ a_{21} & a_{22} & 0 \\ a_{31} & a_{32} & 1 \end{bmatrix}$$

- Given 3 correspondences solve for the T matrix...

Perspective Transformations

- Matrix form is

$$[x, y, w] = [u, v, w] T$$

$$\text{where } T = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$$

and in addition accommodates foreshortening of distant line and convergence of lines to a vanishing point (only parallel lines parallel to the projection plane remain parallel).

Perspective Transformations

- The T matrix can be inferred from correspondences...

Perspective Transformations

- The T matrix can be inferred from correspondences...

$$\begin{pmatrix} u_0 & v_0 & 1 & 0 & 0 & 0 & -u_0 v_0 & -v_0 x_0 \\ u_1 & v_1 & 1 & 0 & 0 & 0 & -u_1 x_1 & -v_1 x_1 \\ u_2 & v_2 & 1 & 0 & 0 & 0 & -u_2 x_2 & -v_2 x_2 \\ u_3 & v_3 & 1 & 0 & 0 & 0 & -u_3 x_3 & -v_3 x_3 \\ 0 & 0 & 0 & u_0 & v_0 & 1 & -u_0 y_0 & -v_0 y_0 \\ 0 & 0 & 0 & u_1 & v_1 & 1 & -u_1 y_1 & -v_1 y_1 \\ 0 & 0 & 0 & u_2 & v_2 & 1 & -u_2 y_2 & -v_2 y_2 \\ 0 & 0 & 0 & u_3 & v_3 & 1 & -u_3 y_3 & -v_3 y_3 \end{pmatrix} A = X$$

where A is the vector of unknown coefficients a_i

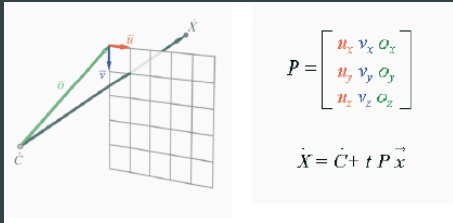
3D Image Warping

- Introduction
 - What's it good for?
- Equations
 - How the heck do you do it?
- Examples
 - What does it look like?
- Misc. Issues
 - Disocclusions
 - Rendering order
 - Splatting

3D Image Warping

- Goal: "warp" the pixels of the image so that they appear in the correct place for a new viewpoint
- Advantage:
 - Don't need a geometric model of the object/environment
 - Can be done in time proportional to screen size and (mostly) independent of object/environment complexity
- Disadvantage:
 - Limited resolution
 - Excessive warping reveals several visual artifacts (see examples)

3D Image Warping Equations

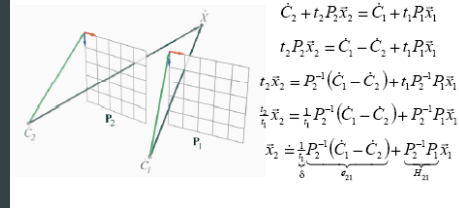


$$P = \begin{bmatrix} u_x & v_x & o_x \\ u_y & v_y & o_y \\ u_z & v_z & o_z \end{bmatrix}$$

$$\dot{X} = \dot{C} + t P \vec{x}$$

Some pictures courtesy of SIGGRAPH '99 course notes (Leonard McMillan)

3D Image Warping Equations



$$\dot{C}_2 + t_2 P_2 \vec{x}_2 = \dot{C}_1 + t_1 P_1 \vec{x}_1$$

$$t_2 P_2 \vec{x}_2 = \dot{C}_1 - \dot{C}_2 + t_1 P_1 \vec{x}_1$$

$$t_2 \vec{x}_2 = P_2^{-1} (\dot{C}_1 - \dot{C}_2) + t_1 P_2^{-1} P_1 \vec{x}_1$$

$$\frac{t_2}{t_1} \vec{x}_2 = \frac{1}{\delta} P_2^{-1} (\dot{C}_1 - \dot{C}_2) + P_2^{-1} P_1 \vec{x}_1$$

$$\vec{x}_2 = \frac{1}{\delta} P_2^{-1} (\dot{C}_1 - \dot{C}_2) + \frac{P_2^{-1} P_1 \vec{x}_1}{z_{21}}$$

3D Image Warping Equations

McMillan & Bishop Warping Equation:

$$\dot{x}_2 = \delta(x_1) P_2^{-1} (c_1 - c_2) + P_2^{-1} P_1 x_1$$

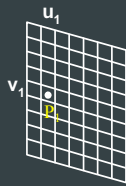
Move pixels based on distance to eye

~ Texture mapping

- Per-pixel distance values are used to warp pixels to their correct location for the current eye position

3D Image Warping Equations

- Images enhanced with per-pixel depth [McMillan95]



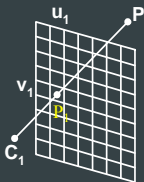
3D Image Warping Equations

$$\dot{P} = \dot{C}_1 + (\bar{c}_1 + u_1 \bar{a}_1 + v_1 \bar{b}_1) w_1$$

$$w_1 = \frac{C_1 P}{C_1 P_1}$$

- 1/w₁ also called generalized disparity

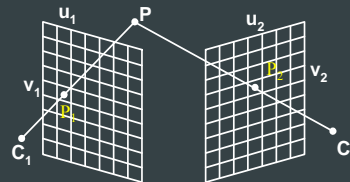
- another notation $\delta(u_1, v_1)$



3D Image Warping Equations

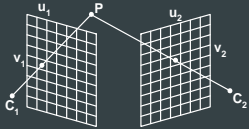
$$\dot{P} = \dot{C}_1 + (\bar{c}_1 + u_1 \bar{a}_1 + v_1 \bar{b}_1) w_1$$

$$\dot{P} = \dot{C}_2 + (\bar{c}_2 + u_2 \bar{a}_2 + v_2 \bar{b}_2) w_2$$

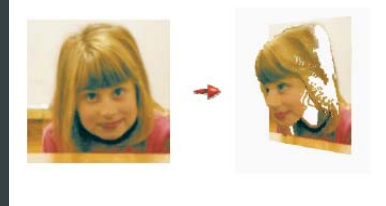


3D Image Warping Equations

$$u_2 = \frac{w_{11} + w_{12} \cdot u_1 + w_{13} \cdot v_1 + w_{14} \cdot \delta(u_1, v_1)}{w_{31} + w_{32} \cdot u_1 + w_{33} \cdot v_1 + w_{34} \cdot \delta(u_1, v_1)}$$
$$v_2 = \frac{w_{21} + w_{22} \cdot u_1 + w_{23} \cdot v_1 + w_{24} \cdot \delta(u_1, v_1)}{w_{31} + w_{32} \cdot u_1 + w_{33} \cdot v_1 + w_{34} \cdot \delta(u_1, v_1)}$$



3D Image Warping Example



3D Image Warping Example



- DeltaSphere
– Lars Nyland *et al.*

3D Image Warping Example



3D Image Warping Example



3D Image Warping Example



3D Image
Warping Example

