**View Morphing** (Seitz & Dyer, SIGGRAPH 96)

View interpolation (ala McMillan) but
- no depth
- no camera information

---

**But First: Multi-View Projective Geometry**

Last time (single view geometry)
- Vanishing Points
- Points at Infinity
- Vanishing Lines
- The Cross-Ratio

Today (multi-view geometry)
- Point-line duality
- Epipolar geometry
- The Fundamental Matrix

All quantities on these slides are in *homogeneous coordinates* except when specified otherwise.
The Projective Plane

Why do we need homogeneous coordinates?
- represent points at infinity, homographies, perspective projection, multi-view relationships

What is the geometric intuition?
- The projective plane

- Each point \((x,y)\) on the plane is represented by a ray \(s(x,y,1)\)
  - Cartesian coordinates \((x,y,z)\rightarrow(x/z, y/z)\)

Projective Lines

A point is a ray in projective space
- How would we represent a line?

- A line is a plane of rays
  - all rays \((x,y,z)\) satisfying: \(ax + by + cz = 0\)

in vector notation:

\[
\begin{bmatrix}
 x \\
 y \\
 z
\end{bmatrix} = \lambda \begin{bmatrix}
 a \\
 b \\
 c
\end{bmatrix}
\]

- A line is also represented as a homogeneous 3-vector \(l\)
Point and Line Duality

- A line \( l \) is a homogeneous 3-vector (a ray)
- It is \( \perp \) to every point (ray) \( p \) on the line: \( l^T p = 0 \)

- What is the intersection of two lines \( l_1 \) and \( l_2 \)?
  - \( p \) is \( \perp \) to \( l_1 \) and \( l_2 \) \( \Rightarrow \)
  - \( p = l_1 \times l_2 \)
- Points and lines are dual in projective space
  - every property of points also applies to lines (e.g., cross-ratio)

Homographies of Points and Lines

We’ve seen lots of names for these
- Planar perspective transformations
- Homographies
- Texture-mapping transformations
- Collineations

Computed by 3x3 matrix multiplication
- To transform a point: \( p' = H p \)
- To transform a line: \( l^T p = 0 \) \( \Rightarrow \) \( l'^T p' = 0 \)
  - \( 0 = l^T p = l^T H^T H p = l^T H^T p' \Rightarrow l'^T = l^T H^T \)
  - lines are transformed by \((H^T)^T\)
3D Projective Geometry
These concepts generalize naturally to 3D

- Homogeneous coordinates
  - Projective 3D points have four coords: \( \mathbf{X} = (X,Y,Z,W) \)
- Duality
  - A plane \( \Pi \) is also represented by a 4-vector
  - Points and planes are dual in 3D: \( \Pi^T \mathbf{p} = 0 \)
- Projective transformations
  - Represented by 4x4 matrices \( \mathbf{T} \): \( \mathbf{p}' = \mathbf{T} \mathbf{p} \), \( \Pi' = (\mathbf{T}^T \Pi \mathbf{T}) \)
- Cross-ratio of planes

However
- Can’t use cross-products in 4D. We need new tools
  - Grassman-Cayley Algebra
    » generalization of cross product, allows interactions between points, lines, and planes via “meet” and “join” operators
  - Won’t get into this stuff today

3D to 2D: Perspective Projection
Matrix Projection:
\[
\mathbf{p} = \begin{bmatrix} s_x & * & * & * \\ s_y & * & * & * \\ s_z & * & * & * \\ 1 & * & * & * \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \Pi \mathbf{p}
\]

It’s useful to decompose \( \Pi \) into \( \mathbf{T} \rightarrow \mathbf{R} \rightarrow \text{project} \rightarrow \mathbf{A} \)

\[
\Pi = \begin{bmatrix} s_x & 0 & -t_x & 0 \\ 0 & s_y & -t_y & 0 \\ 0 & 0 & 1/f & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} R \mathbf{t} & 0 & 0 & 0 \\ 0 & I \mathbf{t} & 0 & 0 \\ 0 & 0 & T \mathbf{t} & 0 \end{bmatrix}
\]

Then we can write the projection as:
\[
\mathbf{p} = \Pi \mathbf{p} = \mathbf{A} \mathbf{R} (\mathbf{P} + \mathbf{T})
\]
Multi-View Projective Geometry

How to relate point positions in different views?
- Central question in image-based rendering
- Projective geometry gives us some powerful tools
  - constraints between two or more images
  - equations to transfer points from one image to another

Epipolar Geometry

What does one view tell us about another?
- Point positions in 2\textsuperscript{nd} view must lie along a known line

Epipolar Constraint
- Extremely useful for stereo matching
  - Reduces problem to 1D search along conjugate epipolar lines
- Also useful for view interpolation...
Transfer from Epipolar Lines

What does one view tell us about another?
- Point positions in 2nd view must lie along a known line

Two views determines point position in a third image
- But doesn’t work if point is in the *trifocal plane* spanned by all three cameras
  - bad case: three cameras are colinear

Epipolar Algebra

How do we compute epipolar lines?
- Can trace out lines, reproject. But that is overkill

Note that \( p' \) is \( \perp \) to \( T \times p' \)
- So \( 0 = p^T T \times p = p'^T T \times (Rp + T) = p'^T T \times (Rp) \)
Simplifying: \( p^T T \times (Rp) = 0 \)

We can write a cross-product \( a \times b \) as a matrix equation

- \( a \times b = A_a b \) where
  \[
  \begin{pmatrix}
  x \\
  y \\
  z
  \end{pmatrix} = \begin{pmatrix}
  0 & -z & y \\
  z & 0 & -x \\
  -y & x & 0
  \end{pmatrix}
  \]

Therefore: \( 0 = p'^T E p \)

- Where \( E = T_x R \) is the 3x3 “essential matrix”
- Holds whenever \( p \) and \( p' \) correspond to the same scene point

Properties of \( E \)

- \( Ep \) is the epipolar line of \( p \); \( p'^T E \) is the epipolar line of \( p' \)
  - \( p'^T E p = 0 \) for every pair of corresponding points
  - \( 0 = E e = e'^T E \) where \( e \) and \( e' \) are the epipoles
    » \( E \) has rank < 3, has 5 independent parameters
- \( E \) tells us *everything* about the epipolar geometry

Linear Multiview Relations

The Essential Matrix: \( 0 = p'^T E p \)

- First derived by Longuet-Higgins, Nature 1981
  - also showed how to compute camera \( R \) and \( T \) matrices from \( E \)
  - \( E \) has only 5 free parameters (three rotation angles, two transl. directions)
- Only applies when cameras have same internal parameters
  - same focal length, aspect ratio, and image center

The Fundamental Matrix: \( 0 = p'^T F p \)

- \( F = (A^{*-1})^T E A^{-1} \), where \( A_{3x3} \) and \( A^*_{3x3} \) contain the internal parameters
- Gives epipoles, epipolar lines
- \( F \) (like \( E \)) is defined only up to a scale factor and has rank 2 (7 free params)
  - Generalization of the essential matrix
  - Can’t uniquely solve for \( R \) and \( T \) (or \( A \) and \( A^* \)) from \( F \)
  - Can be computed using linear methods
    » R. Hartley, *In Defence of the 8-point Algorithm*, ICCV 95
  - Or nonlinear methods
    » Xu & Zhang, *Epipolar Geometry in Stereo, Motion and Object Recognition*, 1996
The Trifocal Tensor

What if you have three views?

• Can compute 3 pairwise fundamental matrices
• However there are more constraints
  – it should be possible to resolve the trifocal problem
• Answer: the trifocal tensor
  – introduced by Shashua, Hartley in 1994/1995
  – a 3x3x3 matrix T (27 parameters)
    » gives all constraints between 3 views
    » can use to generate new views without trifocal probs. [Shai & Avidan]
    » linearly computable from point correspondences

How about four views? five views? N views?

• There is a quadrifocal tensor [Faugeras & Morrain, Triggs, 1995]
• But: all the constraints are expressed in the trifocal tensors, obtained by considering every subset of 3 cameras

View Morphing (Seitz & Dyer, SIGGRAPH 96)

View interpolation (ala McMillan) but

• no depth
• no camera information
Uniqueness Result

Given
- Any two images of a Lambertian scene
- No occlusions

Result: *all views along* $C_1C_2$ *are uniquely determined*

View Synthesis is solvable when
- Cameras are uncalibrated
- Dense pixel correspondence is not available
- Shape reconstruction is impossible

Uniqueness Result

Relies on *Monotonicity* Assumption
- Left-to-right ordering of points is the same in both images
  - used often to facilitate stereo matching
- Implies no occlusions on line between $C_1$ and $C_2$
Image Morphing

Photograph  Morphed Image  Photograph

Linear Interpolation of 2D shape and color

Image Morphing for View Synthesis?

We want to high quality view interpolations
  • Can image morphing do this?

Goal: extend to handle changes in viewpoint
  • Produce valid camera transitions
Morphing parallel views \(\rightarrow\) new parallel views

- Projection matrices have a special form
  - third rows of projection matrices are equal
- Linear image motion \(\leftrightarrow\) linear camera motion

Uncalibrated Prewarpping

Parallel cameras have a special epipolar geometry

- Epipolar lines are horizontal
- Corresponding points have the same \(y\) coordinate in both images

What fundamental matrix does this correspond to?

\[
\hat{F} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix}
\]

Prewarp procedure:

- Compute \(F\) matrix given 8 or more correspondences
- Compute homographies \(H\) and \(H'\) such that
  \[
  H'^T FH = \hat{F}
  \]
  - each homography comprises two rotations, a scale, and a translation
- Transform first image by \(H^{-1}\), second image by \(H'^{-1}\)
1. Prewarp
   ➔ align views

2. Morph
   ➔ move camera
1. Prewarp
   - align views

2. Morph
   - move camera

3. Postwarp
   - point camera
Face Recognition

Corrected Photographs

View Morphing Summary

Additional Features
- Automatic correspondence
- N-view morphing
- Real-time implementation

Pros
- Don’t need camera positions
- Don’t need shape/Z
- Don’t need dense correspondence

Cons
- Limited to interpolation (although not with three views)
- Problems with occlusions (leads to ghosting)
- Prewarp can be unstable (need good F-estimator)
Some Applications of Projective Geometry

Homogeneous coordinates in computer graphics

Metrology
  • Single view
  • Multi-view

Stereo correspondence

View interpolation and transformation

Camera calibration

Invariants
  • Object recognition
  • Pose estimation

Others...