Automatic Panoramic Image Stitching

Dr. Matthew Brown, University of Bath
AutoStitch iPhone

“Create gorgeous panoramic photos on your iPhone”
- Cult of Mac

“Raises the bar on iPhone panoramas”
- TUAW

“Magically combines the resulting shots”
- New York Times
Case study – Image mosaicing

Any two images of a general scene with the same camera centre are related by a planar projective transformation given by:

$$\tilde{w}' = KRK^{-1}\tilde{w}$$

where $K$ represents the camera calibration matrix and $R$ is the rotation between the views.

This projective transformation is also known as the homography induced by the plane at infinity. A minimum of four image correspondences can be used to estimate the homography and to warp the images onto a common image plane. This is known as mosaicing.
Meanwhile in 1999…

- David Lowe publishes “Scale Invariant Feature Transform”
- 11,572 citations on Google scholar
- A breakthrough solution to the correspondence problem
- SIFT is capable of operating over much wider baselines than previous methods

[ Lowe ICCV 1999 ]
Scale Invariant Feature Transform

- Start by detecting points of interest (blobs)

- Find maxima of image Laplacian over scale and space

$$L(I(x)) = \nabla \cdot \nabla I = \frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2}$$
Scale Invariant Feature Transform

- Describe local region by distribution (over angle) of gradients

- Each descriptor: 4 x 4 grid x 8 orientations = 128 dimensions
Scale Invariant Feature Transform

- Extract SIFT features from an image

- Each image might generate 100’s or 1000’s of SIFT descriptors
Feature Matching

• Goal: Find all correspondences between a pair of images

• Extract and match all SIFT descriptors from both images
Feature Matching

• Each SIFT feature is represented by 128 numbers
• Feature matching becomes task of finding a nearby 128-d vector
• All nearest neighbours:

\[ \forall j \quad \text{NN}(j) = \arg \min_i ||x_i - x_j||, \quad i \neq j \]

• Solving this exactly is $O(n^2)$, but good approximate algorithms exist
• e.g., [Beis, Lowe ’97] Best-bin first k-d tree
• Construct a binary tree in 128-d, splitting on the coordinate dimensions
• Find approximate nearest neighbours by successively exploring nearby branches of the tree
2-view Rotational Geometry

- Feature matching returns a set of noisy correspondences
- To get further, we will have to understand something about the geometry of the setup
2-view Rotational Geometry

• Recall the projection equation for a pinhole camera

\[ \tilde{u} = \begin{bmatrix} K & R & t \end{bmatrix} \tilde{X} \]

\( \tilde{u} \sim [u, \ v, \ 1]^T \) : Homogeneous image position

\( \tilde{X} \sim [X, \ Y, \ Z, \ 1]^T \) : Homogeneous world coordinates

\( K (3 \times 3) \) : Intrinsic (calibration) matrix

\( R (3 \times 3) \) : Rotation matrix

\( t (3 \times 1) \) : Translation vector
2-view Rotational Geometry

• Consider two cameras at the same position (translation)
• WLOG we can put the origin of coordinates there

\[\tilde{u}_1 = K_1 [ R_1 \mid t_1 ] \tilde{X}\]

• Set translation to 0

\[\tilde{u}_1 = K_1 [ R_1 \mid 0 ] \tilde{X}\]

• Remember \(\tilde{X} \sim [X, Y, Z, 1]^T\) so

\[\tilde{u}_1 = K_1 R_1 X\]

(where \(X = [X, Y, Z]^T\))
• Add a second camera (same translation but different rotation and intrinsic matrix)

\[
\tilde{u}_1 = K_1 R_1 X \\
\tilde{u}_2 = K_2 R_2 X
\]

• Now eliminate \( X \)

\[
X = R_1^T K_1^{-1} \tilde{u}_1
\]

• Substitute in equation 1

\[
\tilde{u}_2 = K_2 R_2 R_1^T K_1^{-1} \tilde{u}_1
\]

This is a 3x3 matrix -- a (special form) of homography
Finding Consistent Matches

- Raw SIFT correspondences (contains outliers)
RANSAC

• **RA**ndom **SA**mple Consensus [Fischler-Bolles ’81]
• Allows us to robustly estimate the best fitting homography despite noisy correspondences
• **Basic principle:** select the smallest random subset that can be used to compute $H$
• Calculate the support for this hypothesis, by counting the number of **inliers** to the transformation
• Repeat sampling, choosing $H$ that maximises # inliers
RANSAC

\[ H = \text{eye}(3,3); \ n\text{Best} = 0; \]

for (int \( i = 0; i < n\text{Iterations}; i++ \))
{
   P4 = \text{SelectRandomSubset}(P);
   Hi = \text{ComputeHomography}(P4);
   nInliers = \text{ComputeInliers}(Hi);
   if (nInliers > n\text{Best})
   {
      H = Hi;
      n\text{Best} = n\text{Inliers};
   }
}
Finding Consistent Matches

- Raw SIFT correspondences (contains outliers)
Finding Consistent Matches

• SIFT matches consistent with a rotational homography
Finding Consistent Matches

- Warp images to common coordinate frame