Feature-Based Image Alignment

CS 650: Computer Vision

Bryan S. Morse

Image Registration

One common task is to align two images

- Different positions
- Different times
- Different types of imaging

This is called *image registration*.

Image Stitching

One common application of image registration is to stitch together or *mosaic* two or more images together.

Here's a link to an example.

Image Stitching

The key to image stitching is to be able to warp one image to match another, then combine.

Cases:

- Can't do in general without depth information due to *parallax*
- Can do if all of the images have the same focal point (pure 3-D rotation, panoramas)
- Can do if scene is planar (homographies)

Planar Case

If the scene is planar or approximately so (a wall, objects far away, etc.), you can warp one image to another using a homography:

$$\tilde{\mathbf{x}}' = \tilde{\mathbf{H}}\bar{\mathbf{x}}$$

or

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} \sim \begin{bmatrix} \tilde{x}' \\ \tilde{y}' \\ \tilde{w}' \end{bmatrix} = \begin{bmatrix} h_{00} & h_{01} & h_{02} \\ h_{10} & h_{11} & h_{12} \\ h_{20} & h_{21} & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

If $det(\tilde{\mathbf{H}}) \neq 0$, the homography *uniquely* maps 2-D homogeneous points to other 2-D homogeneous points.

Finding the Homography Between Two Images?

Common approaches to registration:

- Area matching (“direct” or “area-based”)
- Sparse set of feature points (“feature-based”)
- Segment, represent, then match objects
  —can be very robust for multimodal registration but very domain-specific

Direct Registration

Basic approach:

- Define some error metric that measures how well the images match (correlation, L-norms, normalized cross-correlation, etc.)
- Select the parameters for the homography that optimizes the quality of the match

We'll come back later to ways to do this.

Feature-Based Registration

Basic approach:

- Find “interesting” points (Moravec, Harris, SIFT, etc.)
- Match them somehow (neighborhoods, SIFT descriptors, etc.)
- Solve for the homography

Can solve for a homography with a minumum of four points, with no three of them colinear (the *4-point algorithm*).

Warning: don’t even get close to colinear!!

What about noise? What about mismatches?

The Four-Point Algorithm

Mapping:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} \sim \begin{bmatrix} h_{00} & h_{01} & h_{02} \\ h_{10} & h_{11} & h_{12} \\ h_{20} & h_{21} & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

or

$$x' = \frac{h_{00}x + h_{01}y + h_{02}}{h_{20}x + h_{21}y + 1} \qquad y' = \frac{h_{10}x + h_{11}y + h_{12}}{h_{20}x + h_{21}y + 1}$$

The Four-Point Algorithm

Using

$$x' = \frac{h_{00}x + h_{01}y + h_{02}}{h_{20}x + h_{21}y + 1} \qquad y' = \frac{h_{10}x + h_{11}y + h_{12}}{h_{20}x + h_{21}y + 1}$$

Let's re-write this:

$$\begin{aligned} h_{00}x + h_{01}y + h_{02} - x'\left(h_{20}x + h_{21}y + 1\right) &= 0 \\ h_{10}x + h_{11}y + h_{12} - y'\left(h_{20}x + h_{21}y + 1\right) &= 0 \end{aligned}$$

Can we write this in matrix form?

The Four-Point Algorithm

$$\begin{bmatrix} x & y & 1 & 0 & 0 & 0 & -x'x & -x'y \\ 0 & 0 & 0 & x & y & 1 & -y'x & -y'y \end{bmatrix} \begin{bmatrix} h_{00} \\ h_{01} \\ h_{02} \\ h_{10} \\ h_{11} \\ h_{12} \\ h_{20} \\ h_{21} \end{bmatrix} = \begin{bmatrix} x' \\ y' \end{bmatrix}$$

The Four-Point Algorithm

$$\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x_1'x_1 & -x_1'y_1 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -y_1'x_1 & -y_1'y_1 \\ x_2 & y_2 & 1 & 0 & 0 & 0 & -x_2'x_2 & -x_2'y_2 \\ 0 & 0 & 0 & x_2 & y_2 & 1 & -y_2'x_2 & -y_2'y_2 \\ x_3 & y_3 & 1 & 0 & 0 & 0 & -x_3'x_3 & -x_3'y_3 \\ 0 & 0 & 0 & x_3 & y_3 & 1 & -y_3'x_3 & -y_3'y_3 \\ x_4 & y_4 & 1 & 0 & 0 & 0 & -x_4'x_4 & -x_4'y_4 \\ 0 & 0 & 0 & x_4 & y_4 & 1 & -y_4'x_4 & -y_4'y_4 \end{bmatrix} \begin{bmatrix} h_{00} \\ h_{01} \\ h_{02} \\ h_{10} \\ h_{11} \\ h_{12} \\ h_{20} \\ h_{21} \end{bmatrix} = \begin{bmatrix} x_1' \\ y_1' \\ x_2' \\ y_2' \\ x_3' \\ y_3' \\ x_4' \\ y_4' \end{bmatrix}$$

Camera Calibration (Revisited)

- This general approach found in the four-point algorithm can be used in lots of other problems.
- Keep in mind that *it's the elements of the matrix that are the unknowns* and re-write the equations to make these the "unknown vector" and the other information the "matrix"
- Example: Camera Calibration

$$\mathbf{p} \sim \mathbf{P}\,\mathbf{p}_w$$

Write in terms of 11 unknowns of $\mathbf{P}$ and then use known 3-D points $\mathbf{p}_w$ and projected points $\mathbf{p}$

Dealing with Noise

Deal with noise in estimating any one point by *using more points than needed*—this is an *overconstrained* system of equations

Let $\mathbf{x}' = (x', y')$ denote the remapped points as before:

$$x' = \frac{h_{00}x + h_{01}y + h_{02}}{h_{20}x + h_{21}y + 1} \qquad y' = \frac{h_{10}x + h_{11}y + h_{12}}{h_{20}x + h_{21}y + 1}$$

and let $\hat{x}$ denote the correct matching position in the other image, then minimize the sum of the squared error:

$$\sum_i \|x - \hat{x}\|^2$$

and solve using least-squares fitting.

Least Squares

Your book has a more detailed discussion in 6.1.1, 6.1.3, A.2.

General Form:

Let for each point pair i and a set of transformation parameters p:

$x'_i = f(x_i, p)$	predicted (mapped) location
$\hat{x}_i$	measured location of matching point
$r_i = \hat{x}_i - x'_i$	error or *residual*

Goal: minimize the squared residuals

$$\sum_i \|r_i\|^2 = \sum_i \|x' - \hat{x}\|^2$$

Key piece: if x_i stays fixed and you change a parameter in p, how does the predicted matching point x'_i change?

Linear Least Squares

The Jacobian is an extension of the gradient for multiple outputs:

$$J = \frac{\partial f}{\partial p}$$

If the relationship between the point motion and the parameters p is *linear* (e.g., up to affine),

$$\Delta x = x' - x = J(x)\ p$$

The minimum-error result can then be found by a *pseudoinverse* technique:

$$A\, p = b$$

where

$$\begin{aligned} A &= \sum_i J^T(x_i) J(x_i) \\ b &= \sum_i J^T(x_i) \Delta x_i \end{aligned}$$

Nonlinear Least Squares

If the relationship between the point motion and the parameters of the transformation is *not* linear (e.g., homography, lots of other things in vision), you have to use an iterative solver.

- Basic idea: iteratively adjust parameters until residual reaches minimum — essentially variants on gradient descent
- The Jacobian is essential to all of these since it tells you how changing each parameter in p will change the result
- One popular technique for minimizing squared-error measures is *Levenberg-Marquardt*.
- Iterative solvers require a good starting point, so how do you get one? Use a linear technique to seed it (e.g., four-point method).

Dealing with Bad Point Matches

- Some point matches might be wrong—how do you deal with it?
- From a least-squares fitting view, these are *outliers*, which throw off the answer.
- Common approach: Use RANSAC and the four-point algorithm to simultaneously separate outliers and solve for initial estimate for $\tilde{\mathbf{H}}$.
- Throw all the resulting inliers into a least-squares solver.