

5 Robust estimation: RANSAC

Computer Vision Lab

Zoltan Kato

<http://www.cab.u-szeged.hu/~kato/>

Measuring Geometric Error

- E.g., **symmetric epipolar distance**: Overall point-to-line distances given by

$$\sum_i d(\mathbf{x}'_i, \mathbf{F}\mathbf{x}_i)^2 + d(\mathbf{x}_i, \mathbf{F}^T \mathbf{x}'_i)^2$$



Slide adopted from Christopher Rasmussen: CIS 489/689 -- Computer Vision

Nonlinear Iterative Refinement of the $\mathbf{F}$ Estimate

- Idea: Improve $\mathbf{F}$ estimate by minimizing geometric error (e.g., symmetric epipolar distance) with iterative method
- Enforcing rank-2 property
 - Shouldn't wait until minimization is over and project $\mathbf{F}$ onto nearest singular $\mathbf{F}'$ as with DLT
 - To ensure property holds at each step of minimization:
 - Penalize deviations from $\det \mathbf{F} = 0$ with some constraint function
 - Or, e.g., *over-parametrization*: Writing $\mathbf{F} = [\mathbf{t}]_x \mathbf{M}$ for arbitrary 3×3 matrix $\mathbf{M}$ guarantees that $\mathbf{F}$ is singular. So minimize over the 12 parameters of $\mathbf{t}, \mathbf{M}$ instead of the 7 of $\mathbf{F}$

Slide adopted from Christopher Rasmussen: CIS 489/689 -- Computer Vision

Matlab: Nonlinear Iterative Refinement of the $\mathbf{F}$ Estimate

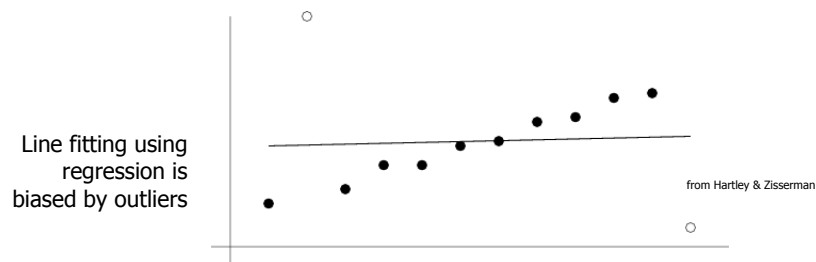
- Unconstrained nonlinear minimization: `fminsearch`
 - Specify starting point $\mathbf{x}_0$ (could be $\mathbf{F}$ directly or 12 parameters of $\mathbf{t}, \mathbf{M}$)
 - Specify error function f that takes parameters $\mathbf{x}, \mathbf{a}$, and $\mathbf{b}$ (such as sets of corresponding points) and returns scalar
 - Answer is $\mathbf{x} = \text{fminsearch}('f', \mathbf{x}_0, [], \mathbf{a}, \mathbf{b})$
- Constrained nonlinear minimization: `fmincon`
 - Can also specify nonlinear constraint function (e.g., use $\det(\mathbf{F})$ as penalty)

Slide adopted from Christopher Rasmussen: CIS 489/689 -- Computer Vision



The Problem with Outliers

- Least squares is a technique for fitting a model to data that exhibit a Gaussian error distribution
- When there are *outliers*—data points that are not drawn from the same distribution—the estimation result can be biased



Slide adopted from Christopher Rasmussen: CIS 489/689 -- Computer Vision

Robust Estimation

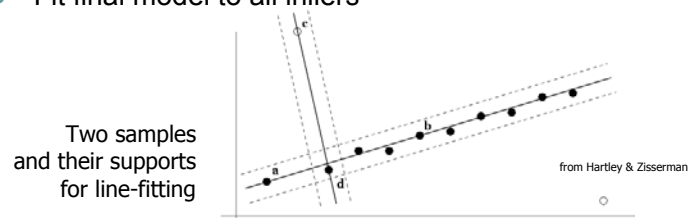
- View estimation as a two-stage process:
 - Classify data points as outliers or inliers
 - Fit model to inliers

Slide adopted from Christopher Rasmussen: CIS 489/689 -- Computer Vision



RANSAC (RANDOM SAMPLE CONSENSUS)

1. Randomly choose minimal subset of data points necessary to fit model (a *sample*)
2. Points within some distance threshold t of model are a *consensus set*. Size of consensus set is model's *support*
3. Repeat for N samples; model with biggest support is most robust fit
 - Points within distance t of best model are inliers
 - Fit final model to all inliers



Slide adopted from Christopher Rasmussen: CIS 489/689 -- Computer Vision



RANSAC: Picking the Distance Threshold t

- Usually chosen empirically

Slide adopted from Christopher Rasmussen: CIS 489/689 -- Computer Vision

RANSAC: How many samples?



- Using all possible samples is often infeasible
- Instead, pick N to assure probability p of at least one sample (containing S points) being all inliers

$$N = \frac{\log(1 - p)}{\log(1 - (1 - \epsilon)^S)}$$

where ϵ is probability that point is an outlier

- Typically $p = 0.99$

Slide adopted from Christopher Rasmussen: CIS 489/689 -- Computer Vision

RANSAC: Computed N ($p = 0.99$)



Sample size	Proportion of outliers ϵ						
S	5%	10%	20%	25%	30%	40%	50%
2	2	3	5	6	7	11	17
3	3	4	7	9	11	19	35
4	3	5	9	13	17	34	72
5	4	6	12	17	26	57	146
6	4	7	16	24	37	97	293
7	4	8	20	33	54	163	588
8	5	9	26	44	78	272	1177

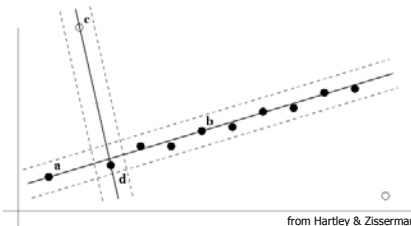
adapted from Hartley & Zisserman

Slide adopted from Christopher Rasmussen: CIS 489/689 -- Computer Vision

Example: N for the line-fitting problem



- $n = 12$ points
- Minimal sample size $S = 2$
- 2 outliers $\rightarrow \epsilon = 1/6 = 20\%$
- So $N = 5$ gives us a 99% chance of getting a pure-inlier sample
 - Compared to $N = 66$ by trying every pair of points



Slide adopted from Christopher Rasmussen: CIS 489/689 -- Computer Vision

RANSAC: Determining N adaptively



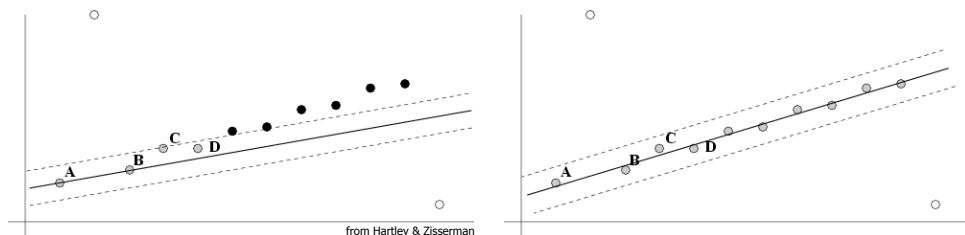
- If the outlier fraction ϵ is not known initially, it can be estimated iteratively:
 - Set $N=1$ and outlier fraction to worst case: $\epsilon=0.5$
 - For every sample, count number of inliers (support)
 - Update outlier fraction if lower than previous estimate:
 $\epsilon = 1 - (\text{number of inliers}) / (\text{total number of points})$
 - Set new value of N using formula
 - If number of samples checked so far exceeds current N , stop

Slide adopted from Christopher Rasmussen: CIS 489/689 -- Computer Vision



After RANSAC

- RANSAC divides data into inliers and outliers and yields estimate computed from minimal set of inliers with greatest support
- Improve this initial estimate with estimation over all inliers (i.e., standard minimization)
- But this may change inliers, so alternate fitting with re-classification as inlier/outlier



Slide adopted from Christopher Rasmussen: CIS 489/689 -- Computer Vision



Corner Detection

- Basic idea: Find points where two edges meet—i.e., high gradient in two directions
- “Cornerness” is undefined at a single pixel, because there’s only one gradient per point
 - Look at the gradient behavior over a small window
- Categories image windows based on gradient statistics
 - **Constant:** Little or no brightness change
 - **Edge:** Strong brightness change in single direction
 - **Flow:** Parallel stripes
 - **Corner/spot:** Strong brightness changes in orthogonal directions



Slide adopted from Christopher Rasmussen: CIS 489/689 -- Computer Vision



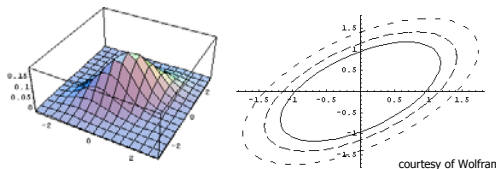
Corner Detection: Analyzing Gradient Covariance

- Intuitively, in corner windows both I_x and I_y should be high
 - Can’t just set a threshold on them directly, because we want rotational invariance



- Analyze distribution of gradient components over a window to differentiate between types from previous slide:

$$\mathbf{C} = \begin{pmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{pmatrix}$$



- The two eigenvectors and eigenvalues λ_1, λ_2 of $\mathbf{C}$ (Matlab: `eig(C)`) encode the predominant directions and magnitudes of the gradient, respectively, within the window
- Corners are thus where $\min(\lambda_1, \lambda_2)$ is over a threshold

Slide adopted from Christopher Rasmussen: CIS 489/689 -- Computer Vision

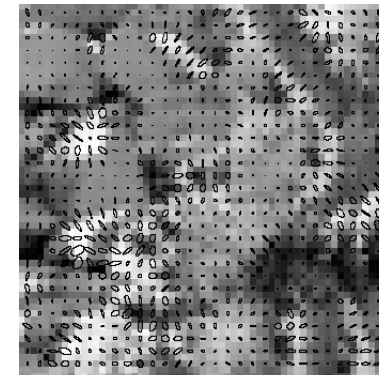


Example: Gradient Covariances

Corners are where both eigenvalues are big



Full image

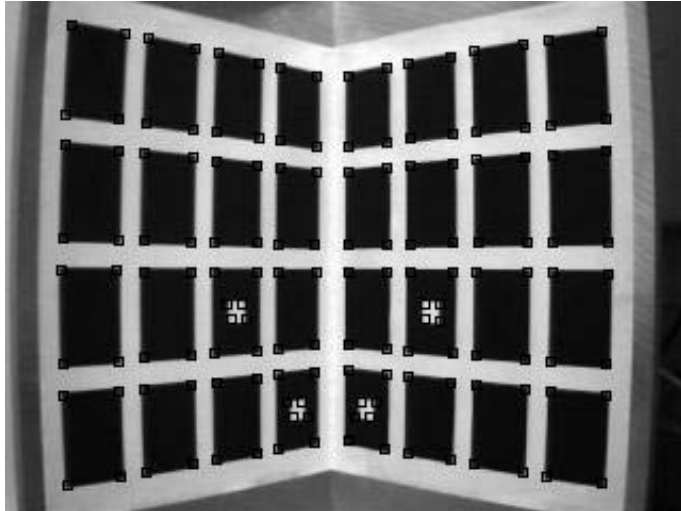


from Forsyth & Ponce

Detail of image with gradient covariance ellipses for 3 x 3 windows

Slide adopted from Christopher Rasmussen: CIS 489/689 -- Computer Vision

Example: Corner Detection (for camera calibration)



courtesy of B. Wilburn
Slide adopted from Christopher Rasmussen: CIS 489/689 -- Computer Vision

Example: Corner Detection



courtesy of S. Smith

SUSAN corners

Slide adopted from Christopher Rasmussen: CIS 489/689 -- Computer Vision

So, To Detect Corners



- Filter image with Gaussian to reduce noise
- Compute magnitude of the x and y gradients at each pixel
- Construct C in a window around each pixel (Harris uses a Gaussian window – just blur)
- Solve for product of λ s (determinant of C)
- If λ s are both big (product reaches local maximum and is above threshold), we have a corner (Harris also checks that ratio of λ s is not too high)

Automatic Homography H Estimation



- How to get correct correspondences without human intervention?



from Hartley & Zisserman

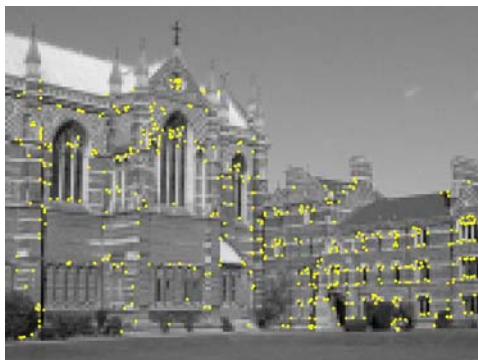


Slide adopted from Christopher Rasmussen: CIS 489/689 -- Computer Vision

Automatic H Estimation: Feature Extraction

- Find features in pair of images using corner detection—e.g., minimum eigenvalue over threshold of:

$$C = \begin{pmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{pmatrix}$$



from Hartley & Zisserman

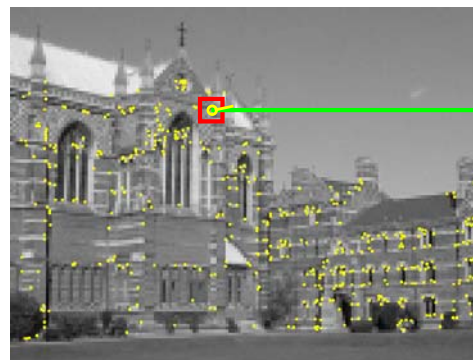


~500 features found

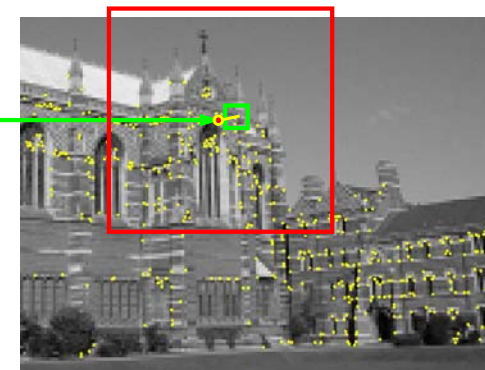
Slide adopted from Christopher Rasmussen: CIS 489/689 -- Computer Vision

Automatic H Estimation: Finding Feature Matches

- Best match over threshold within square search window using SSD or normalized cross-correlation



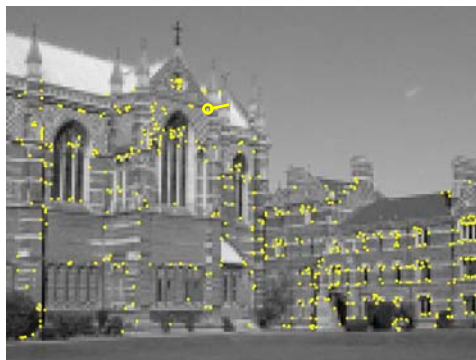
from Hartley & Zisserman



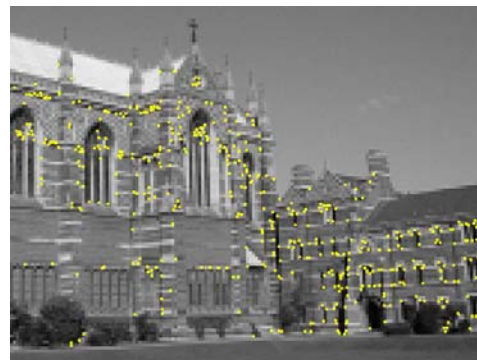
Slide adopted from Christopher Rasmussen: CIS 489/689 -- Computer Vision

Automatic H Estimation: Finding Feature Matches

- Best match over threshold within square search window using SSD or normalized cross-correlation



from Hartley & Zisserman



Slide adopted from Christopher Rasmussen: CIS 489/689 -- Computer Vision

Image Normalization

- Even when the cameras are identical models, there can be differences in gain and sensitivity.
- The cameras do not see exactly the same surfaces, so their overall light levels can differ.
- For these reasons and more, it is a good idea to normalize the pixels in each window:

$\bar{I} = \frac{1}{ W_m(x,y) } \sum_{(u,v) \in W_m(x,y)} I(u,v)$	Average pixel
$\ I\ _{W_m(x,y)} = \sqrt{\sum_{(u,v) \in W_m(x,y)} [I(u,v)]^2}$	Window magnitude
$\hat{I}(x,y) = \frac{I(x,y) - \bar{I}}{\ I - \bar{I}\ _{W_m(x,y)}}$	Normalized pixel

Slides adopted from Sebastian Thrun: CS223B Computer Vision (Stanford)

Images as Vectors

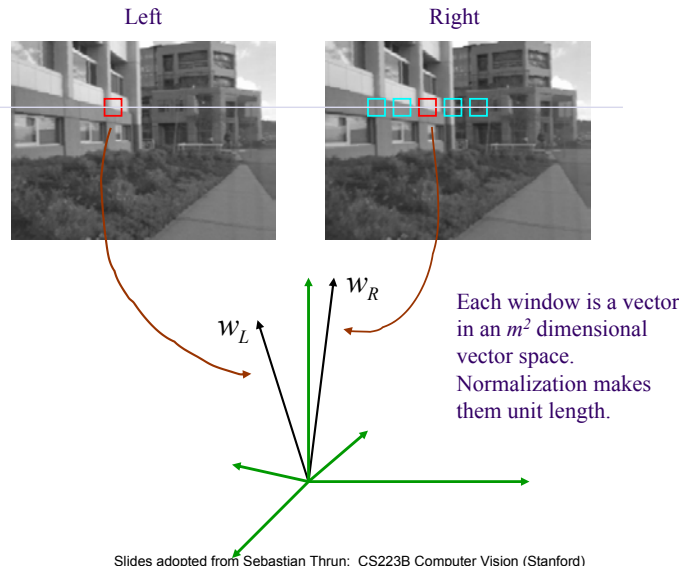
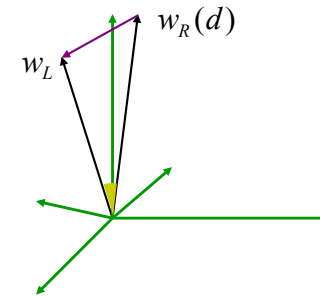


Image Metrics



(Normalized) Sum of Squared Differences

$$C_{SSD}(d) = \sum_{(u,v) \in W_m(x,y)} [\hat{I}_L(u,v) - \hat{I}_R(u-d,v)]^2$$

$$= \|w_L - w_R(d)\|^2$$

Normalized Correlation

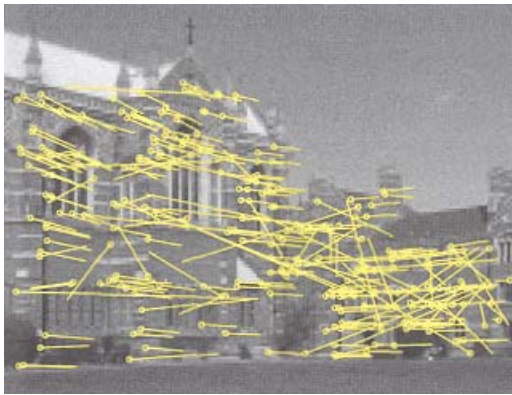
$$C_{NC}(d) = \sum_{(u,v) \in W_m(x,y)} \hat{I}_L(u,v) \hat{I}_R(u-d,v)$$

$$= w_L \cdot w_R(d) = \cos \theta$$

$$d^* = \arg \min_d \|w_L - w_R(d)\|^2 = \arg \max_d w_L \cdot w_R(d)$$

Slides adopted from Sebastian Thrun: CS223B Computer Vision (Stanford)

Automatic H Estimation: Initial Match Hypotheses



from Hartley & Zisserman

268 matched features (over SSD threshold) in left image pointing to locations of corresponding right image features

Slide adopted from Christopher Rasmussen: CIS 489/689 -- Computer Vision

Automatic H Estimation: Applying RANSAC

- Sampling
 - Size: Recall that 4 correspondences suffice to define homography, so sample size $S = 4$
 - Choice
 - Pick SSD threshold conservatively to minimize bad matches
 - Disregard degenerate configurations
 - Ensure points have good spatial distribution over image
- Distance measure
 - Obvious choice is symmetric transfer error:

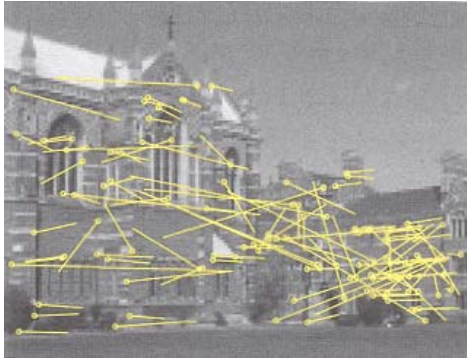
$$d_{transfer}^2 = d(\mathbf{x}, \mathbf{H}^{-1}\mathbf{x}')^2 + d(\mathbf{x}', \mathbf{H}\mathbf{x})^2$$

Slide adopted from Christopher Rasmussen: CIS 489/689 -- Computer Vision

Automatic H Estimation: Outliers & Inliers after RANSAC

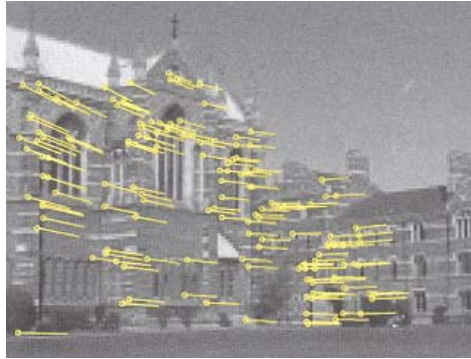


- 43 samples used with $t = 1.25$ pixels



from Hartley & Zisserman

117 outliers ($\epsilon = 0.44$)



151 inliers

Slide adopted from Christopher Rasmussen: CIS 489/689 -- Computer Vision

Homework (continue Lab03)



- Identify point correspondences by
 - Finding corners in overlapping areas
 - Apply RANSAC to select inliers (use SSD or NC) and compute planar homography based on the obtained point matches (inliers).
- Produce panorama (reuse code from Lab03)