3D Computer Vision

Radim Šára   Martin Matoušek

Center for Machine Perception
Department of Cybernetics
Faculty of Electrical Engineering
Czech Technical University in Prague

https://cw.fel.cvut.cz/wiki/courses/tdv/start

http://cmp.felk.cvut.cz
mailto:sara@cmp.felk.cvut.cz
phone ext. 7203

rev. November 21, 2023

Open Informatics Master's Course

▶Local Optimization for Fundamental Matrix Estimation

Summary so far

- Given a set $X = \{(x_i, y_i)\}_{i=1}^k$ of $k \gg 7$ inlier correspondences, compute a statistically efficient estimate for fundamental matrix $\mathbf{F}$.
  1. Find the conditioned (→93) 7-point $\mathbf{F}_0$ (→85) from a suitable 7-tuple
  2. Improve the $\mathbf{F}_0^*$ using the LM optimization (→110–111) and the Sampson error (→112) on all inliers, reinforce rank-2, unit-norm $\mathbf{F}_k^*$ after each LM iteration using SVD

Partial conceptualization

- inlier = a correspondence (a true match)
- outlier = a non-correspondence
- binary inlier/outlier labels are hidden
- we can get their likely estimate only, with respect to a model

We are not yet done

- if there are no wrong correspondences (mismatches, outliers), this gives a local optimum given the 7-point initial estimate
- the algorithm breaks under contamination of (inlier) correspondences by outliers
- the full problem involves finding the inliers!
- in addition, we need a mechanism for jumping out of local minima (and exploring the space of all fundamental matrices)

Example Matching Results for the 7-point Algorithm with Random-Sampling Optimization

input images | interest points (ca. 3600) | tentative corresp. (416) | matching (340)

- descriptors used to obtain tentative matches but no descriptors used in the final matching
- without local optimization the minimization is over a discrete set of epipolar geometries proposable from 7-tuples
- notice the mismatches (they have wrong depth, even negative) remember: hidden labels →113
- they are considered as random outliers to the epipolar model
- inlier matches will be treated as correspondences for the SfM problem

▶A Preview: Optimization by Random Sampling of Geometric Primitives

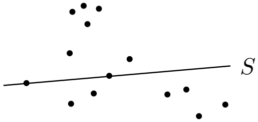
Given an optimization problem, define:

- parameters $\boldsymbol{\theta} \in \mathrm{domain}(\boldsymbol{\theta})$
- primitive geometric element $x_i \in \mathcal{P}$
- generator q of random minimal proposal s-tuples $S \in \mathcal{P}^s$ of primitive elements
- minimal-problem solver computing $\boldsymbol{\theta}$ from the s-tuples: $\mathrm{solver}: \mathcal{P}^s \to \mathrm{domain}(\boldsymbol{\theta})$
- objective function $\pi(\mathcal{P} \mid \boldsymbol{\theta})$

Examples:	$\boldsymbol{\theta}$	primitive	s	solver	$\pi(\cdot)$ terms
line fitting in 2D	$\underline{\mathbf{n}} \in \mathbb{R}^3$	point	2	$\underline{\mathbf{n}} \simeq \underline{\mathbf{x}}_1 \times \underline{\mathbf{x}}_2$	point-to-line distances
plane fitting in 3D	$\underline{\mathbf{p}} \in \mathbb{R}^4$	point	3	$\underline{\mathbf{p}} \simeq \mathrm{null}\big([\underline{\mathbf{x}}_1, \underline{\mathbf{x}}_2, \underline{\mathbf{x}}_3]^\top\big)$	point-to-plane distances
fundamental matrix fitting	$\mathbf{F}$	match 2D–2D	7	7-pt alg	Sampson errors
exterior orientation	$(\mathbf{R}, \mathbf{t})$	match 3D–2D	3	P3P alg	projection errors

Algorithm sketch:

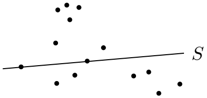
- propose a random s-tuple of primitives S using $q(\cdot)$
- run the solver on S to obtain parameters $\boldsymbol{\theta}$
- compute the value of $\pi(\mathcal{P} \mid \boldsymbol{\theta})$ on all primitives $\mathcal{P}$
- remember the sample which gave the best $\pi(\mathcal{P} \mid \boldsymbol{\theta})$



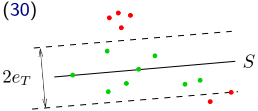
►A Preview: RANSAC with Local Optimization and Early Stopping

Given: minimal configuration C definition, proposal distribution $q(\cdot)$, minimal-problem solver, objective $\pi(\cdot)$:

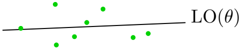
1. initialize the best parameters $\boldsymbol{\theta}_{\text{best}} := \emptyset$, $\pi_{\text{best}} := -\infty$, and proposal index $k := 0$
2. estimate the total number of needed proposals as $N := \binom{n}{s}$ — n – No. of primitives, s – minimal config size
3. while $k \le N$:



   a) **propose** a random s-tuple S from $q(\cdot)$
   b) solve the minimal problem on S to obtain $\boldsymbol{\theta}$
   c) if $\pi(\mathcal{P} \mid \boldsymbol{\theta}) > \pi_{\text{best}}$ then **accept**
      i) update the best $\boldsymbol{\theta}_{\text{best}} := \boldsymbol{\theta}$ — $\pi(S)$ marginalized as in (29); $\pi(S)$ includes a prior $\Rightarrow$ MAP
      ii) threshold-out outliers using e_T from (30)



      iii) locally **optimize** $\boldsymbol{\theta}$ from the inliers of $\boldsymbol{\theta}_{\text{best}}$ — LM optimization with robustified (→121) Sampson error possibly weighted by posterior $\pi(m_{ij})$ [Chum et al. 2003]



      iv) update $\boldsymbol{\theta}_{\text{best}}$, update inliers using (30), re-estimate the **stopping criterion** N from inlier counts — →117 for derivation

$$N = \frac{\log(1-P)}{\log(1-\varepsilon^s)}, \quad \varepsilon = \frac{|\operatorname{inliers}(\boldsymbol{\theta}_{\text{best}})|}{n},$$

   d) $k := k + 1$
4. output C_{best}

- see the MPV course for RANSAC details — see also [Fischler & Bolles 1981], [25 years of RANSAC]

▶Data-Driven Stopping Criterion

- The number of proposals N needed to hit the "true parameters" $=$ an all-inlier configuration:

  this will tell us nothing about the accuracy of the result

P ... probability that the last proposal is an all-inlier for the first time — $1-P$ ... all previous N proposals contained outlier(s)

ε ... the fraction of inliers among primitives, $\varepsilon \leq 1$

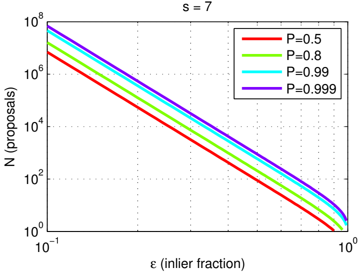
s ... No. of primitives in a minimal configuration — 2 in line fitting, 7 in 7-point algorithm, 4 in homography fitting,...

$$N \geq \frac{\log(1-P)}{\log(1-\varepsilon^s)}$$

- ε^s ... proposal is all-inlier
- $1-\varepsilon^s$ ... proposal contains at least one outlier
- $(1-\varepsilon^s)^N$ ... N previous proposals contained an outlier $= 1-P$

N for $s=7$

	P	
ε	0.8	0.99
0.5	205	590
0.2	$1.3{\cdot}10^5$	$3.5{\cdot}10^5$
0.1	$1.6{\cdot}10^7$	$4.6{\cdot}10^7$



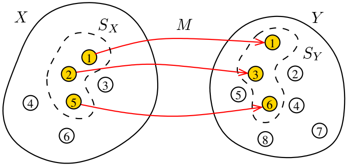
- N can be re-estimated using the current estimate for ε (if there is LO, then after LO)

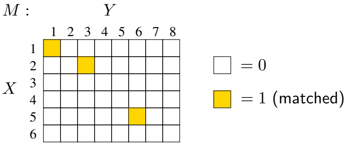
  the quasi-posterior estimate for ε is the average over all samples generated so far
- this shows we have a good reason to limit all possible matches to tentative matches only
- for $\varepsilon \to 0$ we gain nothing over the standard MH-sampler stopping rule — not covered in this course

►Towards $\pi(\cdot)$: The Full Problem of Matching and Fundamental Matrix Estimation

Problem: Given image keypoint sets $X = \{x_i\}_{i=1}^m$ and $Y = \{y_j\}_{j=1}^n$ and their descriptors D, find the most probable

1. inlier keypoints $S_X \subseteq X$, $S_Y \subseteq Y$
2. one-to-one perfect matching $M: S_X \to S_Y$ — perfect matching: 1-factor of the bipartite graph
3. fundamental matrix $\mathbf{F}$ such that $\operatorname{rank} \mathbf{F} = 2$
4. such that for each $x_i \in S_X$ and $y_j = M(x_i)$ it is probable that
   a) the image descriptor $D(x_i)$ is similar to $D(y_j)$, and
   b) the total reprojection error $E = \sum_{ij} e_{ij}^2(\mathbf{F})$ is small — note a slight change in notation: e_{ij}
5. inlier-outlier and outlier-outlier matches are improbable





$$(M^*, \mathbf{F}^*) = \arg\max_{M,\mathbf{F}} \pi(E, D, \mathbf{F}, M) \qquad (E, D) \sim \mathcal{P}, \ (\mathbf{F}, M) \sim \boldsymbol{\theta} \tag{24}$$

- probabilistic model: an efficient language for problem formulation — it also unifies 4.a and 4.b
- the (24) is a Bayesian probabilistic model — there is a constant number of random variables!
- binary matching table $M_{ij} \in \{0, 1\}$ of fixed size $m \times n$
  - each row/column contains at most one unity
  - zero rows/columns correspond to unmatched point x_i / y_j

Deriving A Robust Matching Model by Approximate Marginalization

For algorithmic efficiency, instead of $(M^*, \mathbf{F}^*) = \arg\max_{M,\mathbf{F}} p(E, D, \mathbf{F}, M)$ solve

$$\mathbf{F}^* = \arg\max_{\mathbf{F}} p(E, D, \mathbf{F}) \tag{25}$$

by marginalization of $p(E, D, \mathbf{F}, M)$ over the set of all matchings $\mathcal{M}$ s.t. $M \in \mathcal{M}$ — this changes the problem!

drop the assumption that M is a 1:1 matching, assume correspondence-wise independence:

$$p(E, D, \mathbf{F}, M) = p(E, D, \mathbf{F} \mid M)P(M) = \prod_{i=1}^{m} \prod_{j=1}^{n} p_e(e_{ij}, d_{ij}, \mathbf{F} \mid m_{ij}) P(m_{ij})$$

- e_{ij} represents (reprojection) error for match $x_i \leftrightarrow y_i$: e.g. $e_{ij}(x_i, y_i, \mathbf{F})$
- d_{ij} represents descriptor similarity for match $x_i \leftrightarrow y_i$: e.g. $d_{ij} = \|\mathbf{d}(x_i) - \mathbf{d}(y_j)\|$

Approximate marginalization: — take all the 2^{mn} terms in place of M

$$p(E, D, \mathbf{F}) \approx \sum_{m_{11} \in \{0,1\}} \sum_{m_{12}} \cdots \sum_{m_{mn}} p(E, D, \mathbf{F} \mid M) P(M) =$$

$$= \sum_{m_{11}} \cdots \sum_{m_{mn}} \prod_{i=1}^{m} \prod_{j=1}^{n} p_e(e_{ij}, d_{ij}, \mathbf{F} \mid m_{ij}) P(m_{ij}) \overset{\circledast 1}{=} \cdots =$$

$$= \prod_{i=1}^{m} \prod_{j=1}^{n} \underbrace{\sum_{m_{ij} \in \{0,1\}} p_e(e_{ij}, d_{ij}, \mathbf{F} \mid m_{ij}) P(m_{ij})}_{\text{we will continue with this term}} \tag{26}$$

$$\sum_{m_{ij}\in\{0,1\}} p_e(e_{ij}, d_{ij}, \mathbf{F} \mid m_{ij}) P(m_{ij}) =$$

$$= \underbrace{p_e(e_{ij}, d_{ij}, \mathbf{F} \mid m_{ij}=1)}_{p_1(e_{ij},d_{ij},\mathbf{F})} \underbrace{P(m_{ij}=1)}_{1-P_0} + \underbrace{p_e(e_{ij}, d_{ij}, \mathbf{F} \mid m_{ij}=0)}_{p_0(e_{ij},d_{ij},\mathbf{F})} \underbrace{P(m_{ij}=0)}_{P_0} =$$

$$= (1-P_0)\, p_1(e_{ij}, d_{ij}, \mathbf{F}) + P_0\, p_0(e_{ij}, d_{ij}, \mathbf{F}) \qquad (27)$$

- the $p_0(e_{ij}, d_{ij}, \mathbf{F})$ is a penalty for 'missing a correspondence' but it should be a p.d.f. (cannot be a constant)
→121 for a simplification

$$\text{choose} \quad P_0 \to 1, \quad p_0(\cdot) \to 0 \quad \text{so that} \quad \frac{P_0}{1-P_0}\, p_0(\cdot) \approx \text{const}$$

- the $p_1(e_{ij}, d_{ij}, \mathbf{F})$ is typically an easy-to-design term: assuming independence of reprojection error and descriptor similarity:

$$p_1(e_{ij}, d_{ij}, \mathbf{F}) = p_1(e_{ij} \mid \mathbf{F})\, p_F(\mathbf{F})\, p_1(d_{ij})$$

- we choose, e.g.

$$p_1(e_{ij} \mid \mathbf{F}) = \frac{1}{T_e(\sigma_1)}\, e^{-\frac{e_{ij}^2(\mathbf{F})}{2\sigma_1^2}}, \quad p_1(d_{ij}) = \frac{1}{T_d(\sigma_d, \dim \mathbf{d})}\, e^{-\frac{\|\mathbf{d}(x_i)-\mathbf{d}(y_j)\|^2}{2\sigma_d^2}} \qquad (28)$$

- $\mathbf{F}$ is a random variable and σ_1, σ_d, P_0 are parameters
- the form of $T_e(\sigma_1)$ depends on the error definition, it may depend on x_i, y_j but not on $\mathbf{F}$
- we will continue with the result from (27)

Simplified Robust Energy (Error) Function

- assuming the choice of p_1 as in (28), we are simplifying (26) to

$$p(E,D,\mathbf{F}) = p(E,D \mid \mathbf{F})\, p_F(\mathbf{F}) = p_F(\mathbf{F}) \prod_{i=1}^{m} \prod_{j=1}^{n} \Big[(1-P_0)\, p_1(e_{ij}, d_{ij} \mid \mathbf{F}) + P_0\, p_0(e_{ij}, d_{ij} \mid \mathbf{F})\Big]$$

- we choose $\sigma_0 \gg \sigma_1$ and omit d_{ij} for simplicity; then the square-bracket term is

$$\frac{1-P_0}{T_e(\sigma_1)}\, e^{-\frac{e_{ij}^2(\mathbf{F})}{2\sigma_1^2}} + \frac{P_0}{T_e(\sigma_0)}\, e^{-\frac{e_{ij}^2(\mathbf{F})}{2\sigma_0^2}} = \frac{1-P_0}{T_e(\sigma_1)} \left(e^{-\frac{e_{ij}^2(\mathbf{F})}{2\sigma_1^2}} + \frac{T_e(\sigma_1)}{1-P_0}\,\frac{P_0}{T_e(\sigma_0)}\, e^{-\frac{e_{ij}^2(\mathbf{F})}{2\sigma_0^2}} \right)$$

- we define the 'error function' as: $V(x) = -\log p(x)$ — smaller V is better

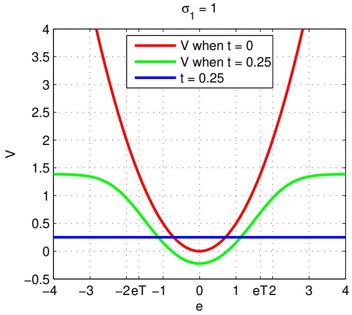
$$V(E,D \mid \mathbf{F}) = \sum_{i=1}^{m}\sum_{j=1}^{n} \left[\underbrace{-\log \frac{1-P_0}{T_e(\sigma_1)}}_{\Delta\,=\,\text{const}} - \log\left(e^{-\frac{e_{ij}^2(\mathbf{F})}{2\sigma_1^2}} + \underbrace{\frac{P_0}{1-P_0}\,\frac{T_e(\sigma_1)}{T_e(\sigma_0)}\, e^{-\frac{e_{ij}^2(\mathbf{F})}{2\sigma_0^2}}}_{t\,\approx\,\text{const}} \right) \right] =$$

$$= m\,n\,\Delta + \sum_{i=1}^{m}\sum_{j=1}^{n} \underbrace{-\log\left(e^{-\frac{e_{ij}^2(\mathbf{F})}{2\sigma_1^2}} + t\right)}_{\hat{V}(e_{ij})} \qquad (29)$$

- the terms in (29) are: (constant) + (total robust error for all pairs in M) — expensive but explicit matching is avoided
- note we are summing over all $m\,n$ matches (m, n are constant!)
- when $t = 0$ we have quadratic inlier error function $\hat{V}(e_{ij}) = e_{ij}^2(\mathbf{F})/(2\sigma_1^2)$

▶The Action of the Robust Matching Model on Data

Ex: Error function $\hat{V}(e_{ij})$ (29):



red – the (non-robust) quadratic error $\hat{V}(e_{ij})$ when $t = 0$

blue – the rejected match penalty t

green – robust $\hat{V}(e_{ij})$ from (29)

- if the error of a correspondence exceeds a limit, it is ignored
- then $\hat{V}(e_{ij}) = \text{const}$ and we just count outliers in (29)
- t controls the 'turn-off' point
- the inlier/outlier threshold is e_T – the error for which $(1 - P_0)\, p_1(e_T) = P_0\, p_0(e_T)$: note that $t \approx 0$

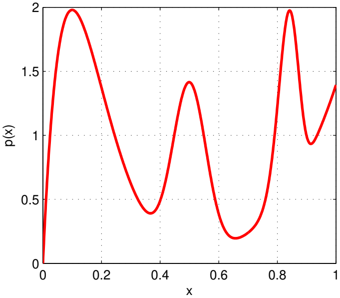
$$e_T = \sigma_1 \sqrt{-\log t^2},\ t = e^{-\frac{1}{2}\left(\frac{e_T}{\sigma_1}\right)^2} \quad \text{e.g. } e_T = 4\sigma_1 \ \to\ t \approx 3.4 \cdot 10^{-4} \tag{30}$$

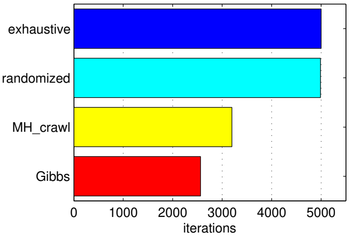
The full optimization problem (25) uses (29):

$$\mathbf{F}^* = \arg\max_{\mathbf{F}} \frac{\overbrace{p(E, D \mid \mathbf{F})}^{\text{data model}} \cdot \overbrace{p(\mathbf{F})}^{\text{prior}}}{\underbrace{p(E, D)}_{\text{evidence}}} \approx \arg\min_{\mathbf{F}} \left[V(\mathbf{F}) + \sum_{i=1}^{m} \sum_{j=1}^{n} \log\left(e^{-\frac{e_{ij}^2(\mathbf{F})}{2\sigma_1^2}} + t \right) \right]$$

- typically we take $V(\mathbf{F}) = -\log p(\mathbf{F}) = 0$ unless we need to stabilize a computation, e.g. when video camera moves smoothly (on a high-mass vehicle) and we have a prediction for $\mathbf{F}$
- the evidence is not needed unless we want to compare different models (e.g. homography vs. epipolar geometry)

How To Find the Global Maxima (Modes) of a PDF?





- number of proposals before $|x - x_{\text{true}}| \leq \text{step}$
- averaged over 10^4 trials

- given a toy probability distribution $p(x)$ at left $\theta = x$, p.d.f. on $[0, 1]$, mode at 0.1

consider several methods:

1. exhaustive search

```
step = 1/(iterations-1);
for x = 0:step:1
 if p(x) > bestp
  bestx = x; bestp = p(x);
 end
end
```

   - slow algorithm (definite quantization)
   - fast to implement

2. randomized search with uniform sampling

```
while t < iterations
 x = rand(1);
 if p(x) > bestp
  bestx = x; bestp = p(x);
 end
 t = t+1;  % time
end
```

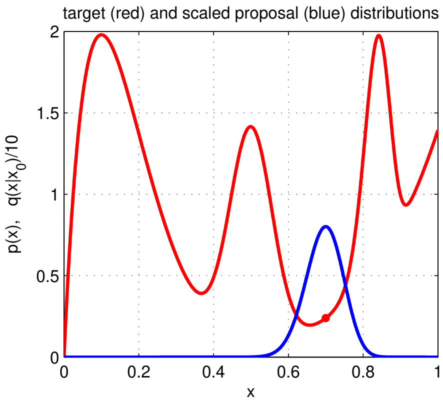
   - equally slow algorithm
   - fast to implement

3. random sampling from $p(x)$ (Gibbs sampler)
   - faster algorithm • fast to implement but often infeasible (e.g. when $p(x)$ is data dependent (our case in correspondence prob.))

4. Metropolis-Hastings sampling
   - almost as fast (with care) • not so fast to implement
   - rarely infeasible • RANSAC belongs here

- simpler (unimodal) distributions result in faster convergence

How To Generate Random Samples from a Complex Distribution?



- red: probability density function $\pi(x)$ of the toy distribution on the unit interval — target distribution

$$\pi(x) = \sum_{i=1}^{4} \gamma_i \, \mathrm{Be}(x; \alpha_i, \beta_i), \quad \sum_{i=1}^{4} \gamma_i = 1, \;\; \gamma_i \geq 0$$

$$\mathrm{Be}(x; \alpha, \beta) = \frac{1}{\mathrm{B}(\alpha, \beta)} \cdot x^{\alpha-1} (1-x)^{\beta-1}, \quad \alpha, \beta \geq 0$$

- alg. for generating samples from $\mathrm{Be}(x; \alpha, \beta)$ is known
- $\Rightarrow$ we can generate samples from $\pi(x)$ — how?

- suppose we cannot sample from $\pi(x)$ but we can sample from some 'simple' proposal distribution $q(x \mid x_0)$, given the previous sample x_0 (blue)

$$q(x \mid x_0) = \begin{cases} \mathrm{U}_{0,1}(x) & \text{(independent) uniform sampling } = \mathrm{Be}(x, 1, 1) \\ \mathrm{Be}(x; \frac{x_0}{T} + 1, \frac{1-x_0}{T} + 1) & \text{'beta' diffusion (crawler)} \quad T \text{ – temperature} \\ \pi(x) & \text{(independent) Gibbs sampler} \end{cases}$$

- note we have unified all the random sampling methods from the previous slide
- how to redistribute proposal samples $q(x \mid x_0)$ to target distribution $\pi(x)$ samples?

▶Metropolis-Hastings (MH) Sampling

C, S – configurations: carry information about $\boldsymbol{\theta}$ e.g. $C = \theta = x$ in →124, C - s-tuple on →115

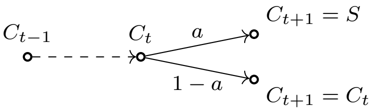
Goal: Generate a sequence of random samples $\{C_t\}$ from target distribution $\pi(C)$

Idea: Setup a Markov chain with a suitable transition probability to generate the sequence

Sampling procedure

1. given current configuration C_t, propose (draw a random) configuration sample S from $q(S \mid C_t)$
   q may use some information from C_t (Hastings)
2. compute acceptance probability the redistribution filter; note the evidence term drops out

$$a = \min\left\{1,\ \frac{\pi(S)}{\pi(C_t)} \cdot \frac{q(C_t \mid S)}{q(S \mid C_t)}\right\}$$



3. accept S with probability a
   a) draw a random number u from unit-interval uniform distribution $\mathrm{U}_{0,1}$
   b) if $u \leq a$ then $C_{t+1} := S$ else $C_{t+1} := C_t$

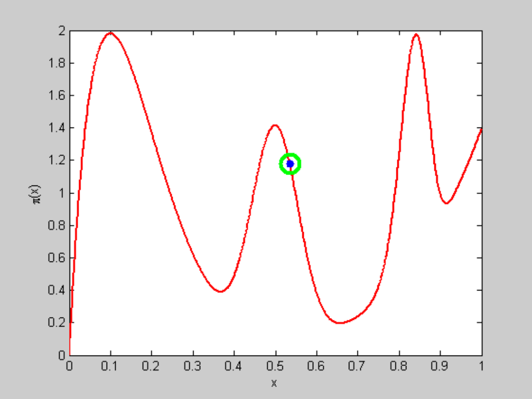
'Programming' an MH sampler

1. design a proposal distribution (mixture) q and a sampler from q
2. express functions $q(C_t \mid S)$ and $q(S \mid C_t)$ as proper distributions not always simple

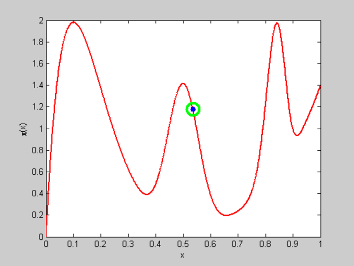
Finding the mode

- remember the best sample fast implementation but must wait long to hit the mode
- use simulated annealing very slow
- use the sampler as an explorer and do local optimization from the accepted sample a trade-off between speed and accuracy

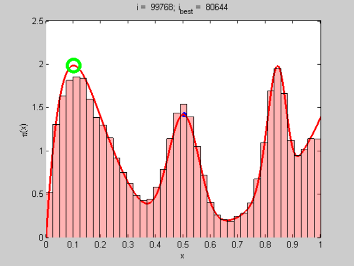
an optimal algorithm does not use just the best sample: a Stochastic EM Algorithm (e.g. SAEM)



sampling process (100k samples; video, 7:33) click for video



initial sample



final distribution of visited states

- blue point: current sample
- green circle: best sample so far
- histogram: current distribution of visited states
- the vicinity of modes are the most often visited states

quality $= \pi(x)$

Demo Source Code (Matlab)

```
function x = proposal_gen(x0)
% proposal generator q(x | x0)

 T = 0.01; % temperature
 x = betarnd(x0/T+1,(1-x0)/T+1);
end


function p = proposal_q(x, x0)
% proposal distribution q(x | x0)

 T = 0.01;
 p = betapdf(x, x0/T+1, (1-x0)/T+1);
end


function p = target_p(x)
% target distribution p(x)

 % shape parameters:
 a = [2   40   100 6];
 b = [10  40   20   1];

 % mixing coefficients:
 w = [1 0.4 0.253 0.50]; w = w/sum(w);
 p = 0;
 for i = 1:length(a)
  p = p + w(i)*betapdf(x,a(i),b(i));
 end
end
```

```
%% DEMO script

k = 10000;    % number of samples
X = NaN(1,k); % list of samples

x0 = proposal_gen(0.5);
for i = 1:k
 x1 = proposal_gen(x0);
 a = target_p(x1)/target_p(x0) * ...
     proposal_q(x0,x1)/proposal_q(x1,x0);
 if rand(1) < a
  X(i) = x1;  x0 = x1;
 else
  X(i) = x0;
 end
end

figure(1)
x = 0:0.001:1;
plot(x, target_p(x), 'r', 'linewidth',2);
hold on
binw = 0.025; % histogram bin width
n = histc(X, 0:binw:1);
h = bar(0:binw:1, n/sum(n)/binw, 'histc');
set(h, 'facecolor', 'r', 'facealpha', 0.3)
xlim([0 1]); ylim([0 2.5])
xlabel 'x'
ylabel 'p(x)'
title 'MH demo'
hold off
```

▶Stripping MH Down To Get RANSAC [Fischler & Bolles 1981]

- when we are interested in the best config only. . . and we need fast data exploration. . .
- . . . then Steps 2–4 below make no difference when waiting for the best sample configuration:

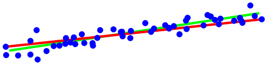
From sampling to RANSACing

1. ~~given C_t,~~ draw a random sample S from ~~$q(S \mid C_t)$~~ $q(S)$ — independent sampling; no use of information from C_t
2. ~~compute acceptance probability~~
$$a = \min\left\{1,\ \frac{\pi(S)}{\pi(C_t)} \cdot \frac{q(C_t \mid S)}{q(S \mid C_t)}\right\}$$
3. ~~draw a random number u from unit-interval uniform distribution $U_{0,1}$~~
4. ~~if $u \leq a$ then $C_{t+1} := S$ else $C_{t+1} := C_t$~~
5. if $\pi(S) > \pi(C_{\text{best}})$ then remember $C_{\text{best}} := S$

- this is (almost) the 'stupid' Method 2 from →123 but(!) the data-driven sampling via higher-order primitives
- it has a good overall exploration but slow convergence in the vicinity of a mode where C_t could serve as an attractor
- getting a good accuracy configuration might take very long this way
- (possibly robust) 'local optimization' necessary for reasonable performance
- unlike the full sampler, it cannot use the past generated configurations to estimate any parameters

The Elements of a Data-Driven MH Sampler

data-driven = proposals $q(S \mid C_t)$ are derived from data ⇒ parameter distribution follows the **empirical distribution** of the s-tuples of primitives. The parameter proposal is done via the minimal problem solver.

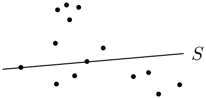


- pairs of points define line distribution $p(\mathbf{n} \mid X)$ (left)
- random correspondence 7-tuples define epipolar geometry distribution $p(\mathbf{F} \mid M)$

Then

1. **primitives** = elementary measurements
   - points in line fitting
   - matches in epipolar geometry or homography estimation

2. **configuration** = s-tuple of primitives — minimal subsets necessary for parameter estimate



the minimization will then be over a discrete set:

- of point pairs in line fitting (left)
- of match 7-tuples in epipolar geometry estimation

3. a map from configuration C to parameters $\boldsymbol{\theta} = \boldsymbol{\theta}(C)$ by solving the **minimal problem**

- line parameters $\mathbf{n}$ from two points — $(\mathbf{x}^1, \mathbf{x}^2) \mapsto \mathbf{n}$
- fundamental matrix $\mathbf{F}$ from seven matches — $\{(\mathbf{x}_i^1, \mathbf{x}_i^2)\}_{i=1:7} \mapsto \mathbf{F}$
- homography $\mathbf{H}$ from four matches, etc — $\{(\mathbf{x}_i^1, \mathbf{x}_i^2)\}_{i=1:4} \mapsto \mathbf{H}$

4. **target likelihood** $p(E, D \mid \boldsymbol{\theta}(C))$ is represented by $\pi(C)$

- can use log-likelihood: then it is the sum of robust errors $\hat{V}(e_{ij})$ given $\mathbf{F}$ (29)
  - robustified point distance from the line $\boldsymbol{\theta} = \mathbf{n}$
  - robustified Sampson error for $\boldsymbol{\theta} = \mathbf{F}$, etc
- posterior likelihood $p(E, D \mid \boldsymbol{\theta}(C))p(\boldsymbol{\theta}(C))$ can be used — MAPSAC ($\pi(S)$ includes the prior)

5. **proposal distribution** $q(\cdot)$ is just a constant(!) distribution of the s-tuples:
   a) q uniform, independent $q(S \mid C_t) = q(S) = \binom{mn}{s}^{-1}$, then $a = \min\left\{1, \frac{p(S)}{p(C_t)}\right\}$
   b) q dependent on descriptor similarity — PROSAC (similar pairs are proposed more often)
   c) q dependent on the current configuration C_t — e.g. 'not far from C_t'

6. (optional) hard **inlier/outlier discrimination** by the threshold (30)

$$\hat{V}(e_{ij}) < e_T, \qquad e_T = \sigma_1 \sqrt{-\log t^2}$$

7. **local optimization** from promising proposals

- can use the hard inliers or just the robust error (29) — more expensive but more stable
- cannot be used to replace C_t — it would violate 'detailed balance' required for the MH scheme

8. **stopping** based on the probability of proposing an all-inlier configuration →117

Harnessing The Full Power of MH Sampler

By marginalization in (25) we have lost constraints on M (e.g. uniqueness). One can choose a better model when not marginalizing:

$$\pi(M,\mathbf{F},E,D) = \underbrace{p(E \mid M,\mathbf{F})}_{\text{reprojection error}} \cdot \underbrace{p(D \mid M)}_{\text{similarity}} \cdot \underbrace{p(\mathbf{F})}_{\text{prior}} \cdot \underbrace{P(M)}_{\text{constraints}}$$

this is a global model: decisions on m_{ij} are no longer independent!

In the MH scheme

- one can work with full $p(M,\mathbf{F} \mid E,D)$, then configuration $C = M$ — $\mathbf{F}$ computable from M
  - explicit labeling m_{ij} can be done by, e.g. sampling from

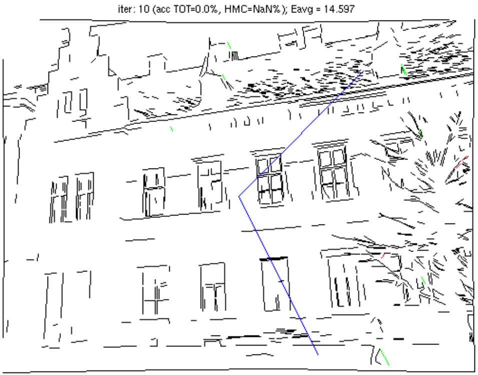
    $$q(m_{ij} \mid \mathbf{F}) \sim \big((1-P_0)\,p_1(e_{ij} \mid \mathbf{F}),\; P_0\,p_0(e_{ij} \mid \mathbf{F})\big)$$

    when $P(M)$ uniform then always accepted, $a = 1$ — ⊛ derive
  - we can compute the posterior probability of each match $p(m_{ij})$ by histogramming m_{ij} from the sequence $\{C_i\}$
  - local optimization can then use explicit inliers and $p(m_{ij})$
- error can be estimated for the elements of $\mathbf{F}$ from the sequence $\{C_i\}$ — does not work in RANSAC
- large error indicates problem degeneracy — this is not directly available in RANSAC
- good conditioning is not a requirement — we work with the entire distribution $p(\mathbf{F})$
- one can find the most probable number of models (epipolar geometries, homographies, ...) — by reversible jump MCMC

  if there are multiple models explaning data, RANSAC will return one of them randomly

Example: MH Sampling for a More Complex Problem

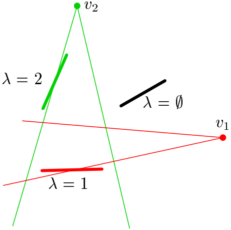
Task: Find two vanishing points from line segments detected in input image. Principal point is known, square pixel.



click for video

- primitives = line segments
- latent variables
  1. each line has a vanishing point label $\lambda_i \in \{\emptyset, 1, 2\}$, $\emptyset$ = outlier
  2. 'mother line' parameters θ_L (they pass through their vanishing points)
- explicit variables
  1. two unknown vanishing points v_1, v_2
- marginal proposals (v_i fixed, v_j proposed)
- minimal configuration $s = 2$
- Gibbs sampling for λ_i

$$\arg \min_{v_1, v_2, \Lambda, \theta_L} V(v_1, v_2, \Lambda, \theta_L)$$



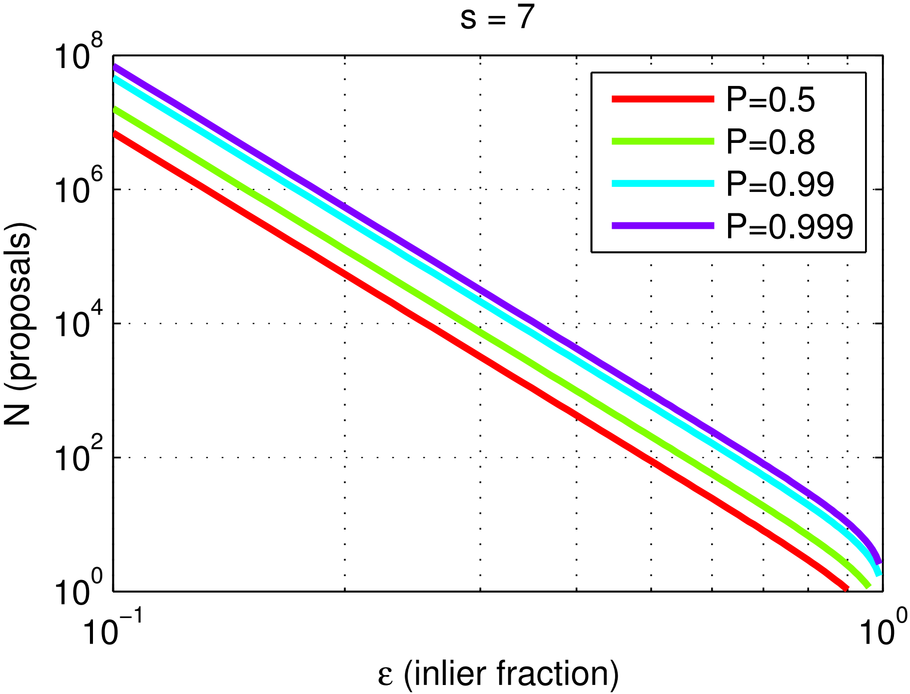
simplifications

- vanishing points restricted to the set of all pairwise segment intersections
- mother lines fixed by segment centroid, then θ_L uniquely given by λ_i, and the configuration is

$$C = \{v_1, v_2, \Lambda\}$$

- blue lines point away from the vanishing points
- proposal acceptance: 20%
- ca. 150 iterations to a good solution

Thank You


s = 7
10^8
10^6
10^4
10^2
10^0
N (proposals)
10^{-1}
10^0
ε (inlier fraction)
P=0.5
P=0.8
P=0.99
P=0.999