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 23, 2021

Open Informatics Master's Course

Deriving A Robust Matching Model by Approximate Marginalization

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

$$\mathbf{F}^* = \arg\max_{\mathbf{F}} p(E, D, \mathbf{F}) \tag{23}$$

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

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

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

- 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

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

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

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

Robust Matching Model (cont'd)

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

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

$$= (1-P_0)\, p_1(e_{ij}, d_{ij}, \mathbf{F}) + P_0\, p_0(e_{ij}, d_{ij}, \mathbf{F}) \qquad (24)$$

- 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) (→115 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.

$$p_1(e_{ij} \mid \mathbf{F}) = \frac{1}{T_e(\sigma_1)}\, e^{-\frac{e_{ij}^2(\mathbf{F})}{2\sigma_1^2}}, \quad p_1(d_{ij}) = \frac{1}{T_d(\sigma_d, \dim \mathbf{d})}\, e^{-\frac{\|\mathbf{d}(x_i)-\mathbf{d}(y_j)\|^2}{2\sigma_d^2}} \qquad (25)$$

- $\mathbf{F}$ 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 (24)

▶Simplified Robust Energy (Error) Function

- assuming the choice of p_1 as in (25), we are simplifying

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

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

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

- we define the 'potential function' as: $V(x) = -\log p(x)$, then

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

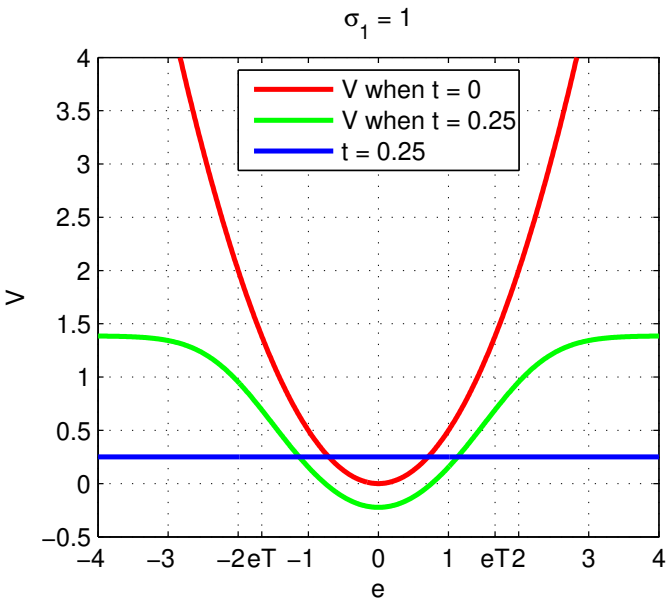
$t \geq 0$

$$= m\, n\, \Delta + \sum_{i=1}^{m} \sum_{j=1}^{n} \underbrace{-\log\Big(e^{-\frac{e_{ij}^2(\mathbf{F})}{2\sigma_1^2}} + t \Big)}_{\hat{V}(e_{ij})} \qquad (26)$$

- the terms in (26) are: (constant) + (total robust error for all pairs in M)
- 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

Example for $\hat{V}(e_{ij})$ from (26):



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 (26)

- 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 (26)
- 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 $t \approx 0$

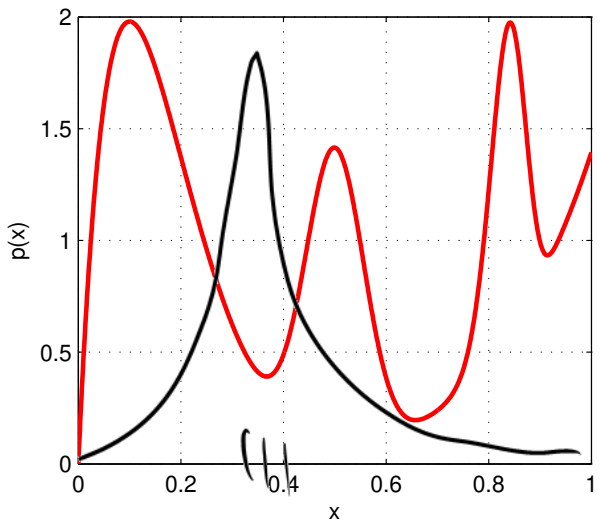
$$e_T = \sigma_1 \sqrt{-\log t^2}, \quad t = e^{-\frac{1}{2}\left(\frac{e_T}{\sigma_1}\right)^2} \quad \text{e.g. } e_T = 4\sigma_1 \quad (27)$$

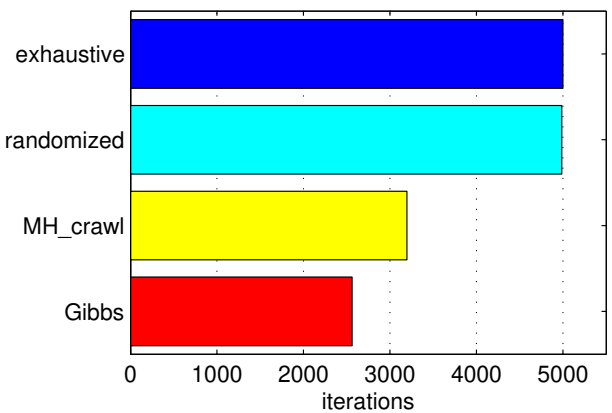
The full optimization problem (23) uses (26):

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

- 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 $\mathbf{F}$
- 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?





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

- given the function $p(x)$ at left

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

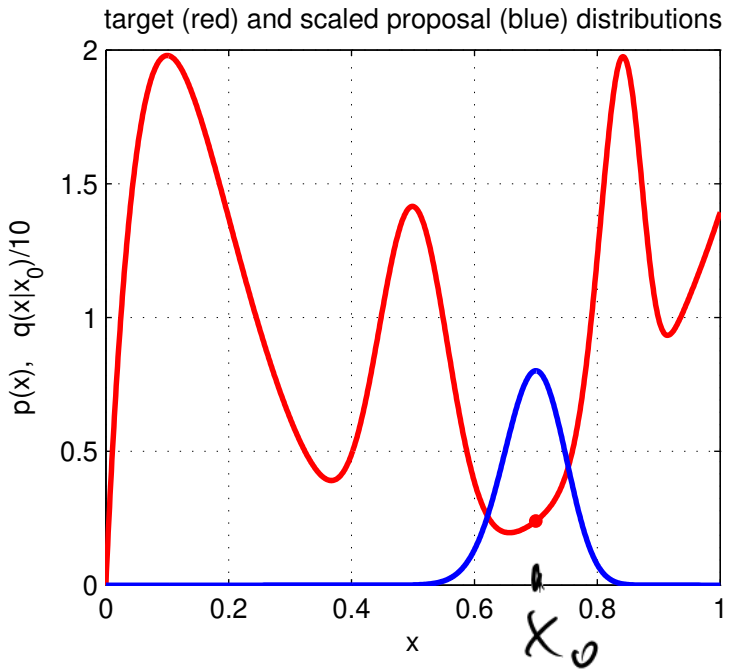
2. randomized search with uniform sampling

```
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

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 (of all variable values) e.g. $C = x$ and $\pi(C) = \pi(x)$ from →118

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

- setup a Markov chain with a suitable transition probability to generate the sequence

Sampling procedure

proposal

1. given current config. C_t, draw a random config. sample S from $q(S \mid C_t)$
   q may use some information from C_t (Hastings)
2. compute acceptance probability — 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

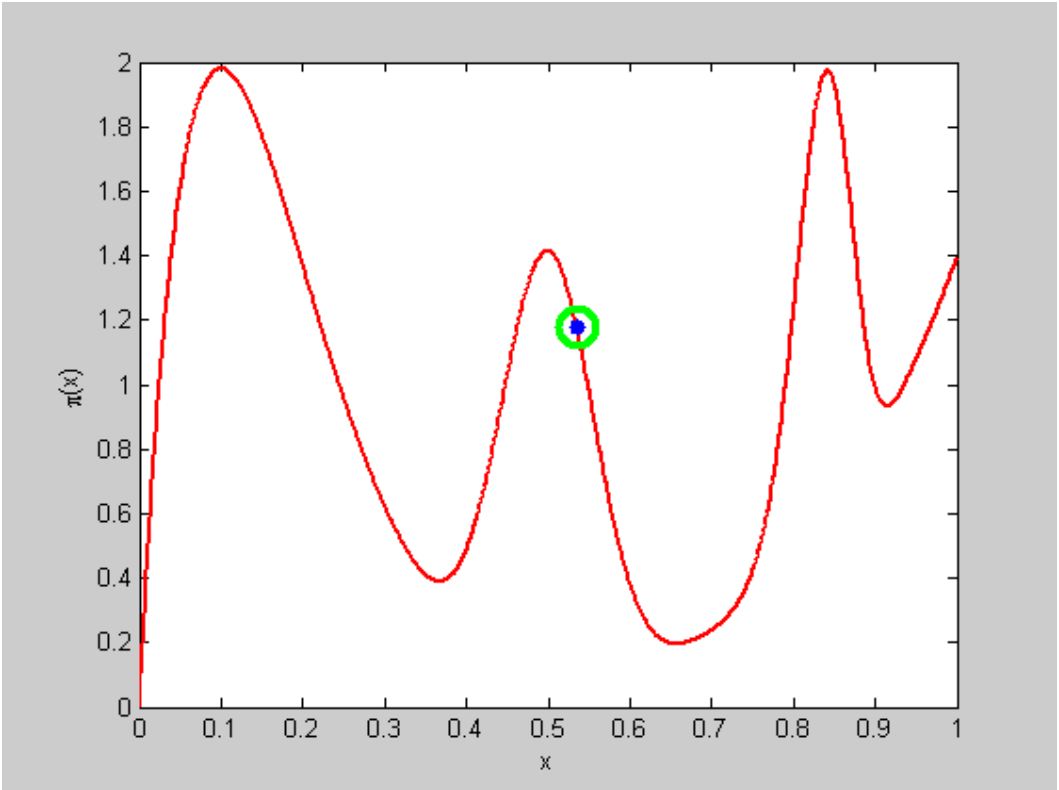
1. design a proposal distribution (mixture) q and a sampler from q
2. express functions $q(C_t \mid S)$ and $q(S \mid C_t)$ as proper distributions — not always simple

Finding the mode

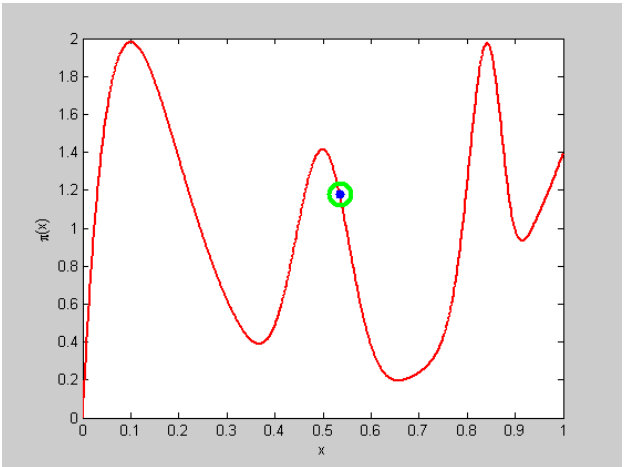
- remember the best sample — fast implementation but must wait long to hit the mode
- use simulated annealing — very slow
- start local optimization from the best sample — good trade-off between speed and accuracy

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

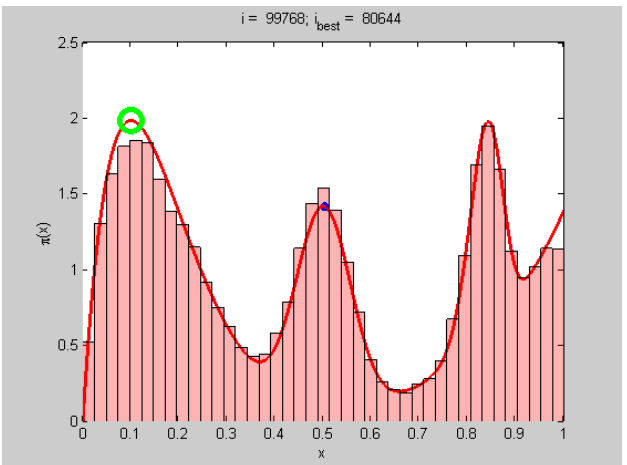
MH Sampling Demo



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



initial sample



final distribution
of visited states

- blue point: current sample
- green circle: best sample so far quality $= \pi(x)$
- histogram: current distribution of visited states
- the vicinity of modes are the most often visited states

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
```

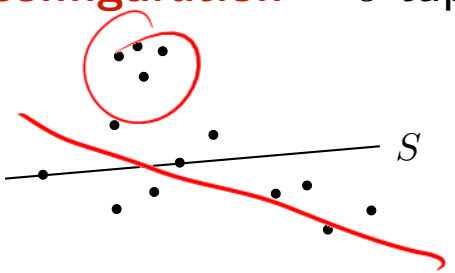
▶The Nine Elements of a Data-Driven MH Sampler

data-driven = proposals $q(S \mid C_t)$ are derived from data

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

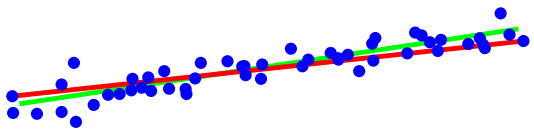


the minimization will then be over a discrete set:
   - of point pairs in line fitting (left)
   - of match 7-tuples in epipolar geometry estimation

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 $\mathbf{F}$ (26)
     - robustified point distance from the line $\boldsymbol{\theta} = \mathbf{n}$
     - robustified Sampson error for $\boldsymbol{\theta} = \mathbf{F}$, etc
   - posterior likelihood $p(E, D \mid \boldsymbol{\theta})p(\boldsymbol{\theta})$ can be used MAPSAC ($\pi(S)$ includes the prior)

5. parameter distribution follows the **empirical distribution** of s-tuples. Since the proposal is done via the minimal problem solver, it is 'data-driven',



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

6. **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'

7. (optional) hard **inlier/outlier discrimination** by the threshold (27)

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

8. **local optimization** from promising proposals

- can use the hard inliers or just the robust error (26) (more expensive but more stable)
- cannot be used to replace C_t (it would violate 'detailed balance' required for the MH scheme)

9. **stopping** based on the probability of proposing an all-inlier configuration →124

▶Data-Driven Sampler Stopping

- The number of proposals N needed to hit the "true parameters" = an all-inlier config?

this will tell us nothing about the accuracy of the result

P ... probability that the last proposal is all-inlier
ε ... the fraction of inliers among primitives, $\varepsilon \leq 1$
s ... minimal configuration size

$1 - P$ ... all previous N proposals were outliers

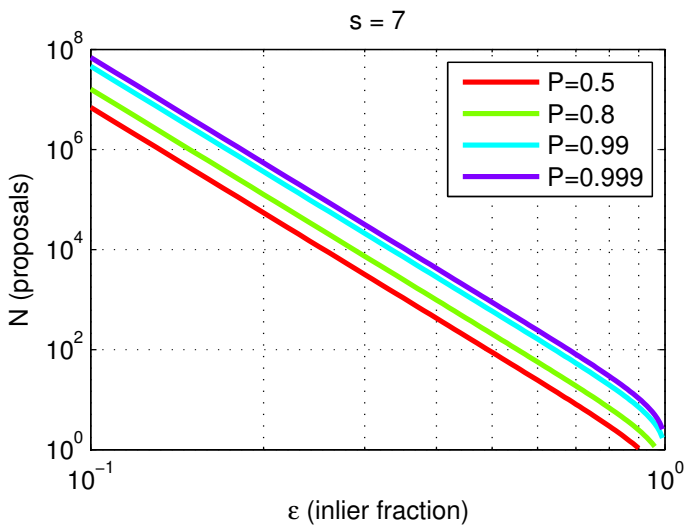
2 in line fitting, 7 in 7-point algorithm, 4 in homography fitting,...

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

- 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	$1.6 \cdot 10^7$	$4.6 \cdot 10^7$

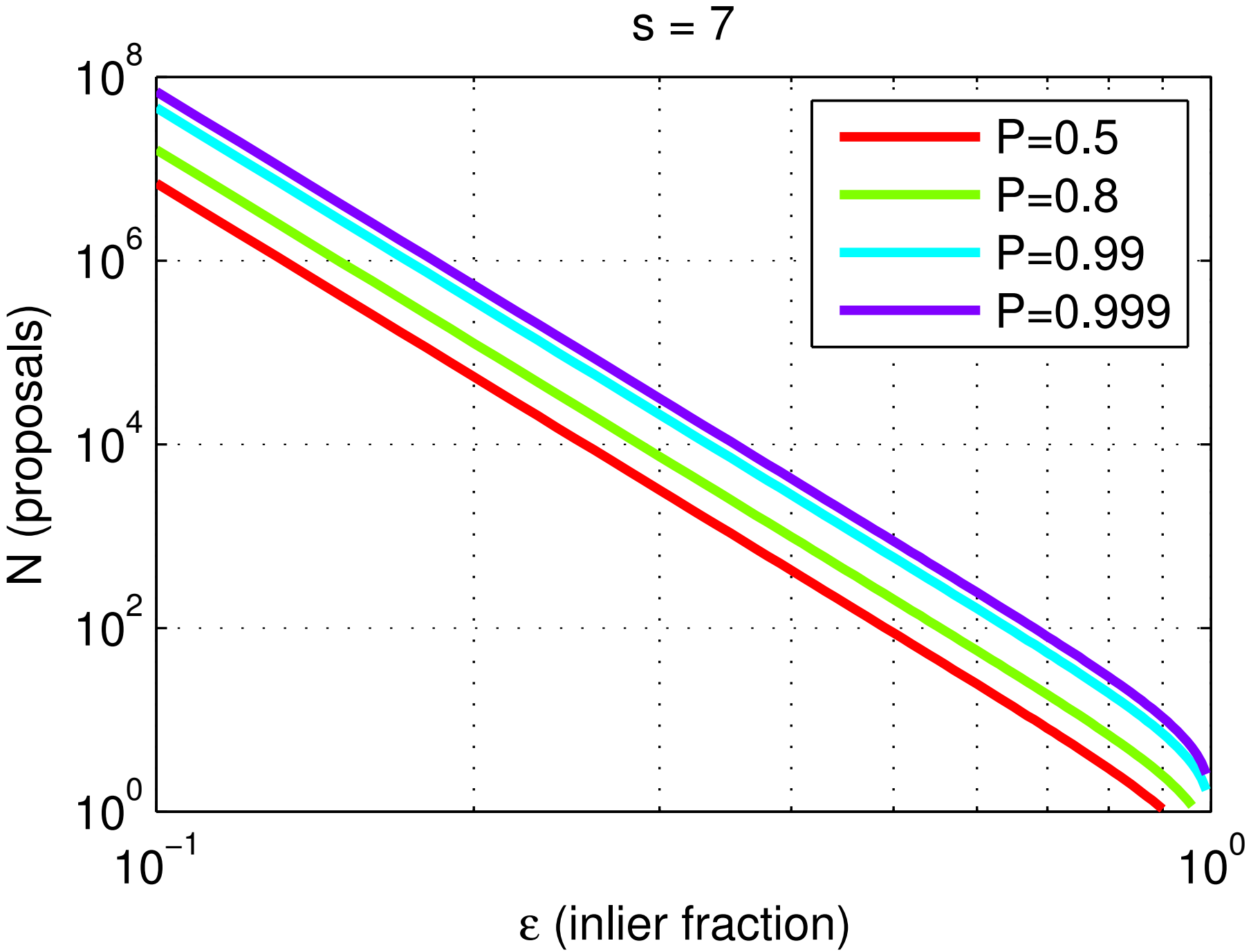


- N can be re-estimated using the current estimate for ε (if there is LO, then after LO)

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

Thank You


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