



CENTER FOR
MACHINE PERCEPTION



CZECH TECHNICAL
UNIVERSITY IN PRAGUE

MASTER'S THESIS

ISSN 1213-2365

Robust Sampling Consensus

Karel Lebeda

karel@lebeda.sk

CTU-CMP-2013-01

January 1, 2013

Available at
http://lebeda.sk/masters_thesis.pdf

Thesis Advisor: Jiří Matas

The author was supported by the projects
SGS11/125/OHK3/2T/13 and GAČR P103/12/2310.

Research Reports of CMP, Czech Technical University in Prague, No. 1, 2013

Published by

Center for Machine Perception, Department of Cybernetics
Faculty of Electrical Engineering, Czech Technical University
Technická 2, 166 27 Prague 6, Czech Republic
fax +420 2 2435 7385, phone +420 2 2435 7637, www: <http://cmp.felk.cvut.cz>

Robust Sampling Consensus

Karel Lebeda

January 1, 2013

Czech Technical University in Prague
Faculty of Electrical Engineering

Department of Cybernetics

DIPLOMA THESIS ASSIGNMENT

Student: **Bc. Karel Lebeda**

Study programme: Open Informatics
Specialisation: Computer Vision and Image Processing

Title of Diploma Thesis: **Robust sampling consensus**

Guidelines:


1. Familiarize yourself with the commonly used variants of the RANSAC method.
2. Compare the variants and find their weakness and problems with their use.
3. Suggest improvements.
4. Implement and validate the improved methods.

Bibliography/Sources:

1. R. Hartley - A.Zissermann: Multiview Geometry
2. O. Chum: PhD thesis
3. P.H.S. Torr: PhD thesis

Diploma Thesis Supervisor: Prof.Ing. Jiří Matas, Ph.D.

Valid until the end of the summer semester of academic year 2012/2013


prof. Ing. Vladimír Mařík, DrSc.
Head of Department




prof. Ing. Pavel Ripka, CSc.
Dean

Prague, November 3, 2011

Hereby I would like to thank my supervisor Jiří Matas for his leading and advices. Further, I would like to mention my colleagues from G-2, mainly Ondřej Chum and Michal Perďoch, without whom this thesis would not be what it is. My thanks belong also to my family and friends for their support, mostly to Simona “Siss” Vrabková, who moreover holds credit for the flyleaf illustration. Special thanks go to Jan Sláma for many typographic suggestions.

I gratefully acknowledge grants SGS11/125/OHK3/2T/13 and GAČR P103/12/2310, which have supported me during my work on the thesis and related research.

Prohlášení autora práce

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

V Praze dne*3.1.2013*.....

.....*Libeda*.....

Podpis autora práce

Abstract

In this thesis, the problem of robust estimation of a multiple view geometry in the computer vision is studied. The main focus is put on random sampling techniques for an estimation of two-view geometries, in particular homography and epipolar geometry, in a presence of outliers. After a thorough analysis of LO-RANSAC, several improvements are proposed to make it more robust to the selection of the inlier/outlier error threshold and to the number of points. A new estimator, faster, more accurate and more robust than the *state-of-the-art* is the result. The improvements were implemented in the framework of CMP WBS-Demo and extensively tuned and experimentally tested on diverse data, using a newly created testing framework. The LO-RANSAC implementation for homography and epipolar geometry estimation has been separated from the rest of WBS-Demo and is now publicly available. The datasets were made available as well, including new manually annotated ground truth point correspondences.

Abstrakt

V rámci této práce byl studován problém robustního odhadu vícehledových geometrií v počítačovém vidění. Hlavní pozornost byla věnována technikám náhodného vzorkování pro odhad dvoupohledových geometrií, konkrétně homografie a epipolární geometrie, za přítomnosti extrémně vychýlených hodnot. Po zevrubné analýze LO-RANSACu je navrženo několik vylepšení pro zvýšení robustnosti k volbě prahu chyby a k počtu bodů. Výsledkem je nový estimátor, rychlejší, přesnější a robustnější než současně používané. Tato vylepšení byla implementována v rámci WBS-Dema CMP a rozsáhle laděna a experimentálně testována na různorodých datech s použitím nově vytvořeného testovacího systému. Implementace LO-RANSACu pro odhad homografie a epipolární geometrie byla osamostatněna od zbytku WBS-Dema a je veřejně dostupná. Použité datové sady byly také zveřejněny, včetně vytvořených manuálně anotovaných bodových korespondencí.

Contents

1. Introduction	3
1.1. Motivation	3
1.2. Contributions	5
1.3. Thesis structure	5
2. Random Sample Consensus	7
2.1. Termination Criterion	8
2.2. Hypotheses Validation	9
3. State of the Art – Members of RANSAC Family	11
3.1. Cost Function	11
3.2. Sampling	11
3.3. Evaluation	12
3.4. Degeneracy Handling	12
4. Locally Optimized RANSAC	15
4.1. Raison d’être	15
4.2. The Algorithm	16
4.3. Implementation Issues	18
5. On the Speed of Local Optimization	21
5.1. LO-RANSAC time consumption	21
5.2. Limiting the Number of Equations	21
5.3. Do Not Redo Done	22
5.4. Simpler Variants of Local Optimization	24
6. Cost Function – Measurement of Model Quality	29
6.1. Error scale	29
6.2. Thresholding or Truncating	31
6.3. Towards Maximum Likelihood Estimation	32
6.4. Uncertainty of Model	32
7. Conclusion	35
7.1. Summary	35
7.2. Outputs of the Work	35
Bibliography	37
A. Datasets	41
A.1. Test Images for Epipolar Geometry Estimation	41
A.2. Test Images for Homography Estimation	42
A.3. Sources	44

B. RANSAC Testing Framework	45
B.1. What Is Measured	45
B.2. Test Course	46
B.3. Output examples	47
C. LO-RANSAC software library	51
C.1. Dependencies	51
C.2. Building the library	51
C.3. Application Programming Interface	51
D. Abbreviations, Symbols, Constants	55
E. Content of Enclosed CD	57

1. Introduction

“ In this thesis, the problem of robust estimation of a multiple view geometry is studied. We propose a new estimator, faster, more accurate and more robust than the *state-of-the-art*. ”

1.1. Motivation

In the field of computer vision, images from multiple cameras are often used to get some more advanced information about the scene than just a field of pixels. Alternatively, those can be multiple images obtained by the same camera. Typical representatives of tasks requiring this are *Stereo vision* and *Structure from motion* and the subsequent use of their results for a *3D reconstruction*, *Simultaneous localization and mapping* in robotics, or other techniques involving three-dimensional information. Another usages can be *Image stitching* (creating a panorama from overlapping images) or *Visual tracking*, where the objective is to acquire a pose of an object in following frames of a videosequence. All of these techniques basically need an estimate of *geometric relations* between the cameras. The part of computer vision which tries to solve these is called *Multiple view geometry* (in particular 2-view, 3-view, or N -view).

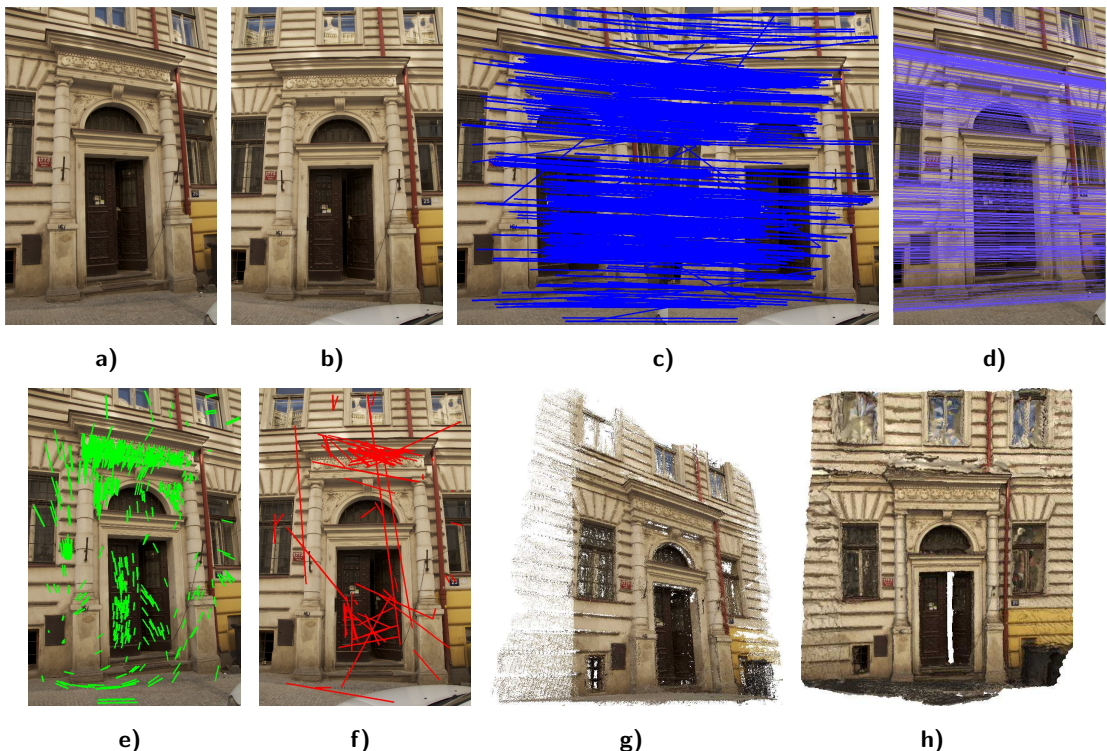


Figure 1.1. SFM for 3D reconstruction: From a set of images, **a)** and **b)** take a pair, and **c)** compute tentative correspondences. Then robustly estimate **d)** the geometry (epipolar lines shown) and divide correspondences into **e)** inliers and **f)** outliers. Figure **g)** then shows the resulting dense cloud of points and **h)** the reconstructed surface.

1. Introduction

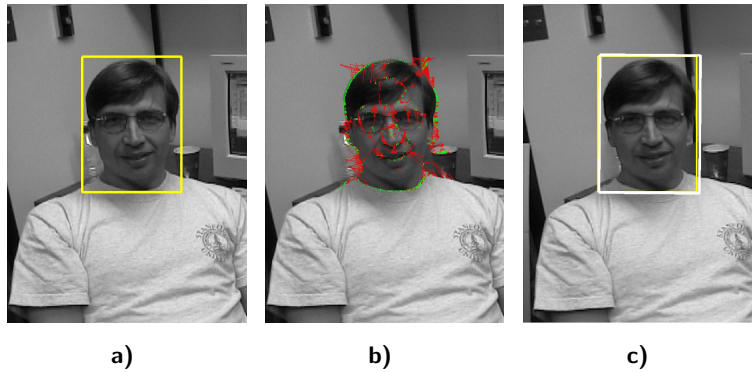


Figure 1.2. Estimating the geometry for tracking **a)** the object inside a bounding box. Edge-point correspondences **b)** contain some portion of outliers (red), therefore the new pose **c)** of the bounding box in a subsequent video frame should be estimated robustly. [17]

To estimate such a geometry, the first step is to extract geometric primitives, such as points, lines or ellipses. These are then locally matched to obtain tentative *correspondences* between a group – usually a pair or a triplet – of images (views). The geometry imposes global constraints on the local correspondences, which, on the other hand, yields equations constraining the geometry. The correspondences are always burdened by some error w.r.t. the geometry model. The error of a correctly matched correspondence is implied by inaccuracies in the primitives detection. Its impact in geometry estimation can be lowered by using as much of these as possible and solving an over-determined set of equations in a least squares sense. However, a presence of mismatches is the inherent property of local matching. For the case of mismatches, the error level can be higher than the noise by orders of magnitude, and even one correspondence with such a gross error, i.e. an outlier, can spoil the estimation. Therefore, a need arises for estimators robust to the presence of the outliers.

“ The robust estimation of the two-view geometry in the presence of the outliers constitutes an overall frame in which this thesis is situated. ”

In the field of robust statistics, a number of estimators has emerged in last decades. The most important for the computer vision are random sampling algorithms and estimators based on voting.

Random sampling techniques are based on the idea that outlier-free subset of the data yields a correct estimate of the geometry. Therefore small subsets are randomly sampled from the full set of data and the *best* geometry found is returned. There exist different approaches to what “best” means. In the family of RANSAC-like algorithms [5, 12], the inlier/outlier threshold is supplied by user and the quality of a model is measured as a number of inliers (or other gain function of this sort, see Chapter 6). In the opposite case of *Least Median of Squares* [33], no assumptions are made about the scale of the error. The percentage of inliers is nevertheless assumed to be higher than 50% and the winner is a geometry with the least median of squared error. For a different percentage this becomes *Least Quantile of Squares*.

Voting techniques make no such assumptions about the data. Every data point instead votes for all the models it is compatible with. The typical representative is *Hough Transformation* [16]. It works on a principle of dividing the model parameter space into small bins, where every single bin accumulates votes of the compatible data. This



Figure 1.3. Image stitching. Correspondences in one of image pairs are shown in **a)** and **b)**, using which the homographies were estimated. The resulting panorama in **c)** is then made by stitching the images together. Image courtesy of Jan Sláma (Alushta, Crimea, 2010).

algorithm can be used under the condition of low-dimensional parameter space. Furthermore, it must be possible to enumerate all the sets of model parameters compatible with the given datum. A combination of these techniques exists: *Randomized Hough Transformation* [41], which combines RANSAC-like sampling with voting for a model parameters.

1.2. Contributions

In a scope of this thesis, the following achievements were reached:

- different RANSAC-like algorithms were thoroughly tested using
- collected datasets with newly created ground truth points and
- a new RANSAC testing framework;
- numerous implementation issues were addressed;
- the claim that LO is not always an improvement at no cost was stated and
- several speed-ups were proposed, implemented and tested;
- the effect of error scale setting with different cost functions was studied and
- the superiority of a truncated quadratic one was claimed in the term of robustness to the error threshold selection;

the contributions were published at the British Machine Vision Conference [18] in a paper *Fixing the Locally Optimized RANSAC*.

The implementation and testing was focused on the point-based algorithms for estimation of homography and epipolar geometry. However, the LO procedure is not restricted to these and it can be used for any general geometry estimation task. E.g., in our another paper *Tracking the Untrackable: How to Track When Your Object Is Featureless* (ACCV DTCE 2012, [17]), LO-RANSAC based on the same principle is used in a tracker to estimate frame-to-frame transformation from correspondences of lines.

1.3. Thesis structure

The rest of this thesis is structured as follows: in Chapter 2 the principle of RANSAC is explained. The *state-of-the-art* improvements to different aspects of RANSAC are discussed in 3 with a special focus on the *Local Optimization* in 4. Then the speed issues of LO are addressed in Chapter 5. Chapter 6 is aimed at measuring the quality

1. Introduction

of a model hypothesis, including analysis of error scale settings, different cost functions and computing and exploiting model uncertainty. Finally, Chapter 7 draws conclusions.

The appendices bring information about the data and test procedures used for experimental evaluation. The details about the published LO-RANSAC software library are also given there. Furthermore, used symbols and constants can be found there.

2. Random Sample Consensus

The task of geometry estimation can be seen as a regression or fitting of a model to the data, burdened by inevitable errors. “Classical techniques for parameter estimation, such as least squares, optimize the fit of a model to all of the presented data. These techniques have no internal mechanisms for detecting and rejecting gross errors. They are averaging techniques that rely on the assumption that the maximum expected deviation of any datum from the assumed model is a direct function of the size of the data set, and thus regardless of the size of the data set, there will always be enough good values to smooth out any gross deviations” [12]. The assumption however hold only for an error attributable to a noise (measurement inaccuracy). As in the process of matching local features usually mismatches appear, there are points with gross error, which cannot be explained by the noise distribution – the *outliers* (the data consistent with the model we call *inliers*). An error of outlier can be arbitrarily high and with no relation to the size of the data set.

“ We call a datum an *outlier*, if its error with respect to the correct model is larger than a maximum allowed for a noise. ”

The least squares approach is to use as many data as possible to reduce an uncertainty of model parameters estimation. In the presence of outliers, the heuristic which may be used is to use all the data to estimate model parameters, remove the datum with the largest error w.r.t. estimated model and iterate until all the errors are lower then some predefined threshold. However, it can be shown (see Figure 2.1), that even one inlier can spoil the estimation process totally. Therefore the breakdown point (the least percentage of outliers to produce arbitrarily erroneous estimation [14]) of these techniques is 0 %.

“ Even one outlier can spoil the geometry estimation via least squares to produce arbitrarily wrong results. ”

On the contrary, the RANSAC (RANdom SAmple Consensus) algorithm proceeds in the opposite way. “Rather than using as much of the data as possible to obtain an initial solution and then attempting to eliminate the invalid data points, RANSAC uses as small an initial data set as feasible and enlarges this set with consistent data when possible” [12]. Basically, it draws randomly *minimal samples* out of the data (two points for line fitting, four point correspondences or two elliptical correspondences for a homography, etc.) to construct candidate hypotheses. The best of these hypotheses is returned. The smoothing then runs only on the consistent data.

Two questions remain unanswered to successfully use RANSAC. The first one is: How many samples we need to draw, how many hypotheses we need to check? The second is: How do we select the best hypothesis seen, how can we compare them? The answers lie in the following sections.

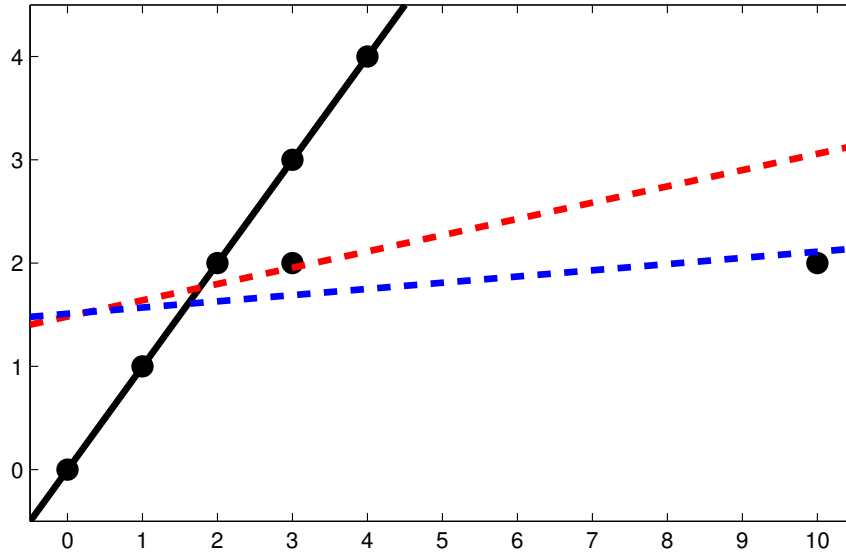


Figure 2.1. Given the set of seven points, we want to find a best fit line, assuming that no valid datum deviates from this line by more than 0.8 units. The ideal line is the solid black one, which would be selected by RANSAC. However, the least squares estimate is the dashed red line. If we started discarding points with highest residues in the iterative way, the final solution would be the dashed blue line, far away from the correct one. Adopted from [12].

2.1. Termination Criterion

To derive the necessary number of samples to be drawn in a probabilistic manner, we need to use the following assumption:

“ **A1:** An all-inlier sample generates model consistent with all inliers.[6] ”

Therefore the probability of getting a good model from a random sample is equal to the probability $P(I)$ of drawing an all-inlier (i.e. outlier-free) sample of size m out of N data points:

$$P(I) = \frac{\binom{I}{m}}{\binom{N}{m}} = \prod_{j=0}^{m-1} \frac{I-j}{N-j} \leq \varepsilon^m, \quad (2.1)$$

where I is (a priori unknown) number of inliers and $\varepsilon = I/N$ is the inlier ratio. Note that for $I \gg m$ holds

$$P(I) \approx \varepsilon^m. \quad (2.2)$$

The probability that at least one outlier-free sample was drawn in k samples is then

$$\eta = 1 - (1 - P(I))^k \geq \eta_0. \quad (2.3)$$

We usually want η to be higher than some user-supplied minimal probability η_0 of returning the optimal solution, which can be seen as a confidence of success. To reach this, RANSAC needs to draw K samples before termination:

$$K \geq \frac{\log(1 - \eta_0)}{\log(1 - P(I))}. \quad (2.4)$$

As the number of inliers I is one of RANSAC outputs, it is unknown during its run. Nevertheless, it can be conservatively approximated by $I^* = |\mathcal{I}^*|$, size of the largest support found thus far. From the other point of view, we may want to assure the probability we haven't missed any model, better the our current so-far-the-best one. Effectively this means to miss a sample from $I^* + 1$ inliers. Therefore, we can use 2.4 with probability $P(I^* + 1)$.

However, experimental results show a disagreement with this theory – first good sample is drawn after significantly longer time than expected. This issue was studied in [10, 34] and is also addressed in 4.

2.2. Hypotheses Validation

To distinguish between correct an incorrect solution (and in general to decide which one of two solutions is better) we need to accept the following assumption:

“ **A2:** A model consistent with a sample contaminated by at least one outlier has small support. ” [6]

Using this assumption, we can measure the quality of a model by size of its support, i.e. number of inliers. In later modifications of RANSAC, different cost functions have been used. Whilst taking the inlier count (thresholding) can be seen as a top-hat cost function, it is possible to use truncated (squared) error, log-likelihood cost function, etc. The only condition is that the selected cost function has to be robust, insensitive to gross error of the outliers. We will address the cost functions later in section 6.2.

We have defined an outlier as a datum, whose error is smaller than some threshold θ . This value (as well as the scale of other possible cost functions) is defined by the error scale of the inliers. For now, we will take it as a constant, supplied by the user and we will return to this topic in 6.1.

With a large number of data points, it can be time-consuming to validate all the hypotheses on all the data. For this reason, several improvements to RANSAC have been proposed, to reject bad hypotheses early, without testing all the data. These will be noted in section 3.3.

3. State of the Art – Members of RANSAC Family

The concept of RANSAC is widely used in the field of computer vision. However, during the decades of its existence, the algorithm has undergone several significant improvements in different aspects and the vanilla RANSAC is rarely used today. This chapter focus on some of these improvements as well as on other algorithms similar to RANSAC.

3.1. Cost Function

As mentioned, the thresholding in RANSAC can be seen as a top-hat cost function, giving a constant penalty to the data whose error exceeds a given level. However, this cost function suffers by scoring different model hypotheses as the same when the error threshold is overestimated. Torr and Zisserman proposed to truncate the error instead of thresholding in their MSAC [37] (M-estimator SAMPLE Consensus) algorithm to gain additional sensitivity at no cost. Later they introduced an algorithm dubbed MLESAC [38] (Maximum Likelihood Estimation SAMPLE Consensus), which models the error as a mixture of a Gaussian (for inliers) and a uniform distribution (outliers) and incorporates likelihood maximization into the sampling framework. Further description of these algorithms is given in section 6.2. Torr extended this work further by following fully Bayesian approach in MAPSAC [36] (Maximum A Posteriori SAMPLE Consensus). However, since it assumes uniform prior distribution, its cost function is similar to the previous ones.

3.2. Sampling

In standard RANSAC, the matching process is viewed as a black box that generates tentative correspondences with a priori unknown (i.e. uniformly distributed) probability of being inliers. However, there is usually available some real-valued measurement of correspondence quality from the matching stage. Guided Sampling and Consensus by Tordoff and Murray [34] exploits these measurements. They are used to estimate prior probability of each correspondence being valid. The uniform sampling is then replaced by a Monte-Carlo sampling with probabilities according to these priors.

Another approach, called PROSAC by Chum and Matas [8] (PROgressive SAMPLE Consensus) uses this knowledge as it draws samples semi-randomly from progressively larger sets of tentative correspondences. The basic assumption is that the ordering defined by the similarity used during matching is not worse than a random ordering. In practice there is often a strong correlation between the correspondence quality and the probability of being an inlier, which leads to a better efficiency. PROSAC can be viewed as an instance of a process that starts by deterministically testing the most promising hypotheses and then converges to the uniform sampling of RANSAC.

NAPSAC by Myatt et al. [26] (N Adjacent Points SAMPLE Consensus) changes the sampling strategy in a different way. It exploits the fact that the model to be estimated defines a lower-dimensional manifold in the space of the input data. The data points on this manifold (the inliers) tend to lie closer to each other than outliers, which are

uniformly distributed over the high-dimensional space. Thus it draws the first point randomly and the rest of $m - 1$ points inside a hypersphere centered at the selected point. This significantly reduces the consumed time, especially for high-dimensional problems.

3.3. Evaluation

As mentioned above, to validate all the candidate hypotheses with all the tentative correspondences may be unnecessarily time-consuming. The following approaches have emerged to address this issue.

The core idea of Randomized RANSAC by Chum and Matas [7, 9] is that a majority of hypotheses tested are influenced by the outliers. These can be tested only on a small number of data points and rejected early. The statistical test reduces the number of verified correspondences, but on the other hand it brings a possibility of unwanted rejection of a good sample. Lower number of the data points processed means higher computation savings, but also higher this probability, which (linearly) increases the necessary number of samples to be drawn. It is intended to find an optimum which minimizes the total running time. Two kinds of tests can be performed. The $T_{d,d}$ test is passed if all d data points out of d randomly selected are consistent with the hypothesized model. The optimal value for a range of problems was found as $d = 1$. The second is the Sequential Probability Ratio Test (SPRT), where the likelihood ratio is updated with every datum checked and if it exceeds the decision threshold, the model is early rejected. This threshold is set to minimize total running time. Models, which are not rejected early, are tested again all the data points in a RANSAC manner.

The bail-out test by Capel [3] works in a similar way. It is based on checking a subset of correspondences first. When the fraction of inliers in the subset is significantly lower than the overall inlier ratio of so-far-the-best sample, the model is rejected.

Nistér proposed an absolutely different approach – Preemptive RANSAC [27]. This is meant to be used in real-time applications, where it is necessary to find a solution in a fixed time. Instead of trying one hypothesis at a time with all the data, a group of hypotheses is generated in the first time. The data are randomly permuted and sequentially checked – one datum at a time with all the hypotheses. The worst hypotheses are being rejected during the test. The procedure terminates when there is only one hypothesis remaining or when all the data have been checked.

Raguram et al. proposed a technique called CovRANSAC (Covariance RANSAC), explicitly incorporating the model uncertainty to get the set of *possible inliers*. We will take a closer look at this topic in section 6.4.

3.4. Degeneracy Handling

The assumption **A2** states that an incorrect model has a small support. However, this assumption can be violated by a degenerate configuration as seen in Figure 3.1. In such a case a weaker assumption is needed:

“ **A2’**: A model consistent with a sample that is contaminated by at least one outlier and *does not* contain a degenerate configuration as a subset has small support. [6] ”

The first approach to detect and handle degeneracy in two-view geometry was PLUNDER by Torr et al. [39] (Pick Least UNDEgenerate Randomly). It tries to fit multiple

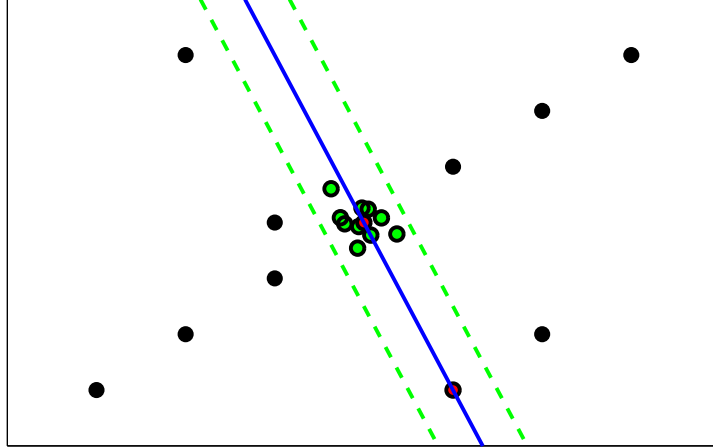


Figure 3.1. Degenerate situation, when an incorrect model has large support disregarding assumption **A2**. Adopted from [6].

models to the data (epipolar geometry, affine epipolar geometry, homography, etc.) and the most general non-degenerate model is returned. Since noise and outliers can make the degenerate data apparently non-degenerate, the decision whether to use a simpler model or not is based on a cardinalities of consensus sets of the models. This idea was later generalized in QDEGSAC by Frahm and Pollefeys [13] (Quasi-DEGenerate Sample Consensus) for estimation of any relation on any data. Furthermore, they proposed a robust matrix rank computation by thresholding the singular numbers.

Chum et al. proposed DegenSAC [11] (Degenerate Sample Consensus) algorithm to estimate the epipolar geometry in the presence of a dominant plane (where correspondences are degenerately related by a homography). When a sample (7-tuple) containing 5 or more points lying on the plane is drawn, the degeneracy is detected. Then a plane-and-parallax algorithm is incorporated, using outliers to the homography as the out-of-plane points.

4. Locally Optimized RANSAC

4.1. Raison d'être

As mentioned earlier, RANSAC relies on the assumption **A1** that an all-inlier sample gives a good model, consistent with all the inliers. However, in the presence of noise (see Figure 4.1), this is not always the case since a bad conditioning can make the sample generate a poor hypothesis even if it is not contaminated by the outliers. Results of such situations are an instability (different outputs in different runs) and an increased execution time. A procedure dubbed *Local Optimization* is used to improve promising model parameter hypotheses, based on a milder assumption:

“ **A1'**: All-inlier sample, even in the presence of noise, gives a model that is close to the optimal one. A local optimization of the model parameters reaches the optimal model that is supported by all inliers.[6] ”

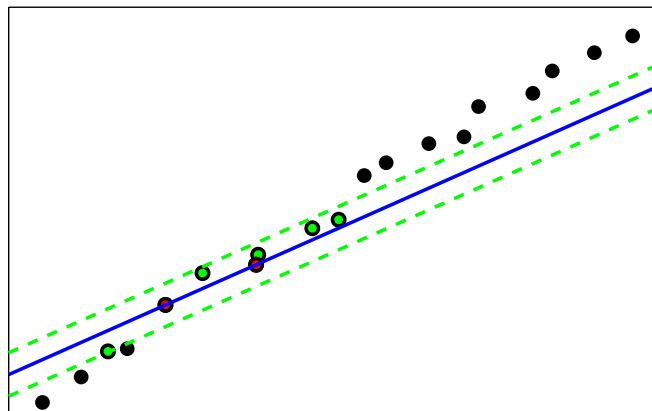


Figure 4.1. Noisy situation, when an all-inlier model is not consistent with all the inliers disregarding assumption **A1**. Adopted from [6].

The core idea of LO-RANSAC by Chum et al. [10] (Locally Optimized RANSAC) is to use promising samples as an initialization for an inlier count maximization. The observation is “that virtually all models estimated from an uncontaminated minimal sample contain large fraction of inliers within their support. An optimization process starting from the the-best-so-far hypothesized model is therefore inserted into RANSAC.”

One may argue that it does not worth the effort to perform such a procedure on every so-far-the-best sample, that it is enough to employ some post-processing step on the globally best sample. Furthermore, if there is some complex optimization procedure appended (such as Bundle Adjustment), it would not worth to do even this. Nevertheless, we can say the following in a defense of LO. Firstly, as the optimization (increasing inlier count) is done during the RANSAC run-time, it reduces the number of samples needed and it often lowers the total running time, especially for problems with low inlier ratios. There are even problems, which would be considered intractable by standard

4. Locally Optimized RANSAC

RANSAC without LO [6]. Secondly, since the Local Optimization is run on multiple promising samples (i.e. from multiple initializations), LO-RANSAC has a better chance of reaching the optimum.

It should be noted that LO does not compete with optimizations such as Bundle Adjustment. It rather tries to provide a better starting point in a shorter time, than standard RANSAC or *Gold Standard* (MSAC followed by linear least squares on inliers) advocated in Hartley and Zisserman’s book [15]. Since the Bundle Adjustment only works with inliers, increase in the number of inliers leads to more equations to be optimized and better results. The experiments can be found in the technical report [19].

4.2. The Algorithm

In the original paper, several methods of LO were proposed, with increasing power (and also time consumption), ranging from a simple linear-least-squares solution, to a complex iterative procedure:

- 1. Standard.** The standard implementation of RANSAC without any local optimization.
- 2. Simple.** Take all data points with error smaller than θ and use a linear algorithm to hypothesize new model parameters.
- 3. Iterative.** Take all data points with error smaller than $K \cdot \theta$ and use linear algorithm to compute new model parameters. Reduce the threshold and iterate until the threshold is θ .
- 4. Inner RANSAC.** A new sampling procedure is executed. Samples are selected only from I_k data points consistent with the hypothesized model of k -th step of RANSAC. New models are verified against whole set of data points. As the sampling is running on inlier data, there is no need for the size of sample to be minimal. On the contrary, the size of the sample is selected to minimize the error of the model parameter estimation. In our experiments the size of samples are set to $\min(I_k/2, 14)$ for epipolar geometry and to $\min(I_k/2, 12)$ for the case of homography estimation. The number of repetitions is set to ten in the experiments presented.
- 5. Inner RANSAC with iteration.** This method is similar to the previous one, the difference being that each sample of the inner RANSAC is processed by method 3. [10]

We have formalized the most powerful variant of LO-RANSAC (variant 5) in Algorithms 1–3. The detailed description of used symbols and constants can be found in Appendix D.

In the LO according to variant 5 (Algorithm 2), a model from the random sample is at first refined by computing least squares fit to inliers on a wider threshold ($m_\theta \cdot \theta$). Inliers to this refined model ($\mathcal{I}_{\text{base}}$) constitute a basic pool for inner sampling. From this pool, a fixed number (*reps*) of non-minimal samples \mathcal{S}_{is} is drawn and respective models (geometry estimates) are computed. These are then refined by iterative least squares with narrowing error threshold (see below). The best model seen during all the procedure is returned.

Algorithm 1 LO-RANSAC.

```

1: for  $k = 1 \rightarrow K(|\mathcal{I}^*|, \eta_0)$  do
2:    $\mathcal{S}_k \leftarrow$  randomly drawn minimal sample
3:    $M_k \leftarrow$  model estimated from sample  $\mathcal{S}_k$ 
4:    $\mathcal{I}_k \leftarrow \text{find\_inliers}(M_k, \theta)$ 
5:   if  $|\mathcal{I}_k| > |\mathcal{I}_s^*|$  then
6:      $M_s^* \leftarrow M_k; \mathcal{I}_s^* \leftarrow \mathcal{I}_k$ 
7:      $M_{\text{LO}}, \mathcal{I}_{\text{LO}} \leftarrow$  run Local Optimization (Alg. 2)
8:     if  $|\mathcal{I}_{\text{LO}}| > |\mathcal{I}^*|$  then
9:        $M^* \leftarrow M_{\text{LO}}; \mathcal{I}^* \leftarrow \mathcal{I}_{\text{LO}}$ 
10:      update  $K$ 
11:    end if
12:  end if
13: end for
14: return  $M^*$ 

```

Algorithm 2 Local Optimization step.

Require: $M_s^*, m_\theta, \text{reps}$

```

1:  $M_{m_\theta} \leftarrow$  model estimated by LSq on  $\text{find\_inliers}(M_s^*, m_\theta \cdot \theta)$ 
2:  $\mathcal{I}_{\text{base}} \leftarrow \text{find\_inliers}(M_{m_\theta}, \theta)$ 
3: for  $r = 1 \rightarrow \text{reps}$  do
4:    $\mathcal{S}_{\text{is}} \leftarrow$  sample of size  $s_{\text{is}}$  randomly drawn from  $\mathcal{I}_{\text{base}}$ 
5:    $M_{\text{is}} \leftarrow$  model estimated from  $\mathcal{S}_{\text{is}}$  by LSq
6:    $M_{\text{r}} \leftarrow$  Iterative Least Squares ( $M_{\text{is}}, m_\theta, \text{iters}$ ) (Alg. 3)
7: end for
8: return the best of  $M_s^*$ , all  $M_{\text{is}}$ , all  $M_{\text{r}}$ , with its inliers

```

Algorithm 3 Iterative Least Squares.

Require: $M_{\text{is}}, m_\theta, \text{iters}$

```

1:  $M' \leftarrow$  model estimated by LSq on  $\text{find\_inliers}(M_{\text{is}}, \theta)$ 
2:  $\theta' \leftarrow m_\theta \cdot \theta$ 
3: for  $i = 1 \rightarrow \text{iters}$  do
4:    $\mathcal{I}' \leftarrow \text{find\_inliers}(M', \theta')$ 
5:    $w' \leftarrow$  computed weights of  $\mathcal{I}'$  (depend on model)
6:    $M' \leftarrow$  model estimated by LSq on  $\mathcal{I}'$  weighted by  $w'$ 
7:    $\theta' \leftarrow \theta' - \Delta_\theta$ 
8: end for
9: return the best  $M'$ 

```

In the iterative least squares (Algorithm 3), the model from sample M_{is} is smoothed by linear least squares on its inliers in the first time. Then the iterative procedure begins: inliers \mathcal{I}' of the last model M' on a widened threshold are found, new model M' is estimated, and the error threshold is lowered by Δ_θ . This is iterated several times (*iters*) until θ' drops back to the normal level. The least squares computation may be weighted, this depends on a particular geometric model.

Every sample has to compete with all the $k - 1$ previous samples in being the best thus far seen. Since we assume the samples to be i.i.d., the probability, that k -th sample

4. Locally Optimized RANSAC

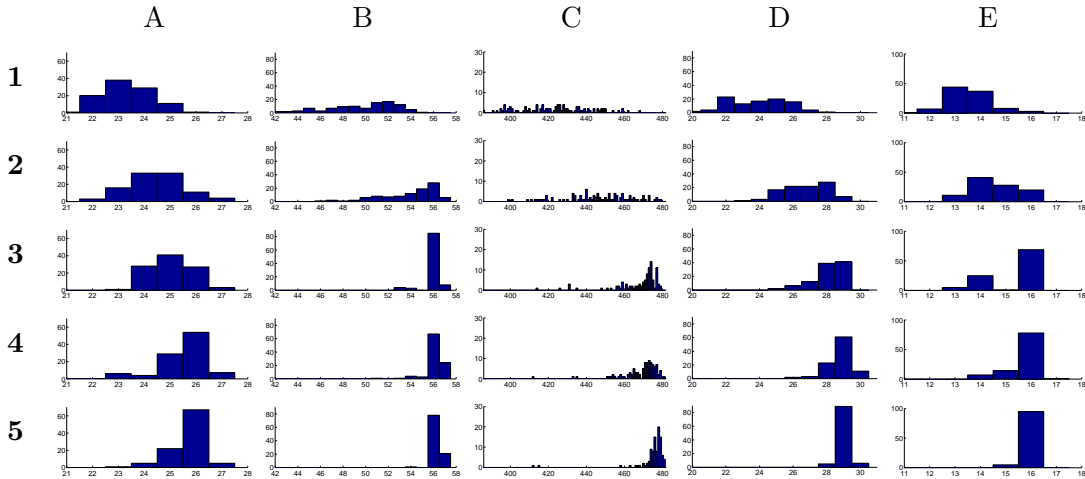


Figure 4.2. Histograms of the number of inliers. The methods 1 to 5 (original Chum’s, 1 stands for standard RANSAC) are stored in rows and different datasets are shown in columns. On each graph, there is a number of inliers on the x -axis and how many times this number was reached within one hundred repetitions on the y -axis. Adopted from [6].

will be so-far-the-best is $1/k$. Then the average number of times so-far-the-best sample is found within K samples can be bounded:

$$\sum_{k=1}^K \frac{1}{k} \leq \int_1^K \frac{1}{x} dx + 1 = \log(K) + 1 . \quad (4.1)$$

Thus LO is applied on average $\log(K)$ times, hence it is not expected to have a significant impact on the overall execution time of the algorithm, linear in K [10] (K is the number of samples drawn by RANSAC). Further insight into the issue of LO speed can be found in Chapter 5.

4.3. Implementation Issues

Since for some of the image pairs each call of the LO has an impact on the overall speed, it is efficient not to call the LO during first K_{start} iterations (set to 50 in our implementation; this number should correspond to the terminating K for an overoptimistic inlier ratio). In such a case it is necessary to ensure that LO is executed at least once, after K_{start} iterations, or at the end of RANSAC (as some scenes may require less than K_{start} samples). Then the LO is run for the best model found in the first K_{start} iterations.

We observed that the choice of numerical methods is critical for getting fast and precise model estimates. While Singular Value Decomposition (SVD) is a convenient and stable way to compute a least square solution of a system of linear equations, it is significantly faster to use eigen-decomposition of the covariance matrix, especially for large systems of equations. Similarly, for non-overdetermined sets of equations, the usage of LU (Lower-Upper) decomposition instead of SVD can bring the same results in a significantly lower execution times (nevertheless, we are losing the information about conditioning, contained in singular values).

In our experiments, we have encountered a significant drop in performance for epipolar geometry estimation caused by the instability of singular value decomposition when

using the CCMATH library [2]. This instability was observed during fundamental matrix singularization, especially for the image pairs with epipoles (one or both) in the image. The final implementation uses the LAPACK library [1] that does not suffer by such an instability.

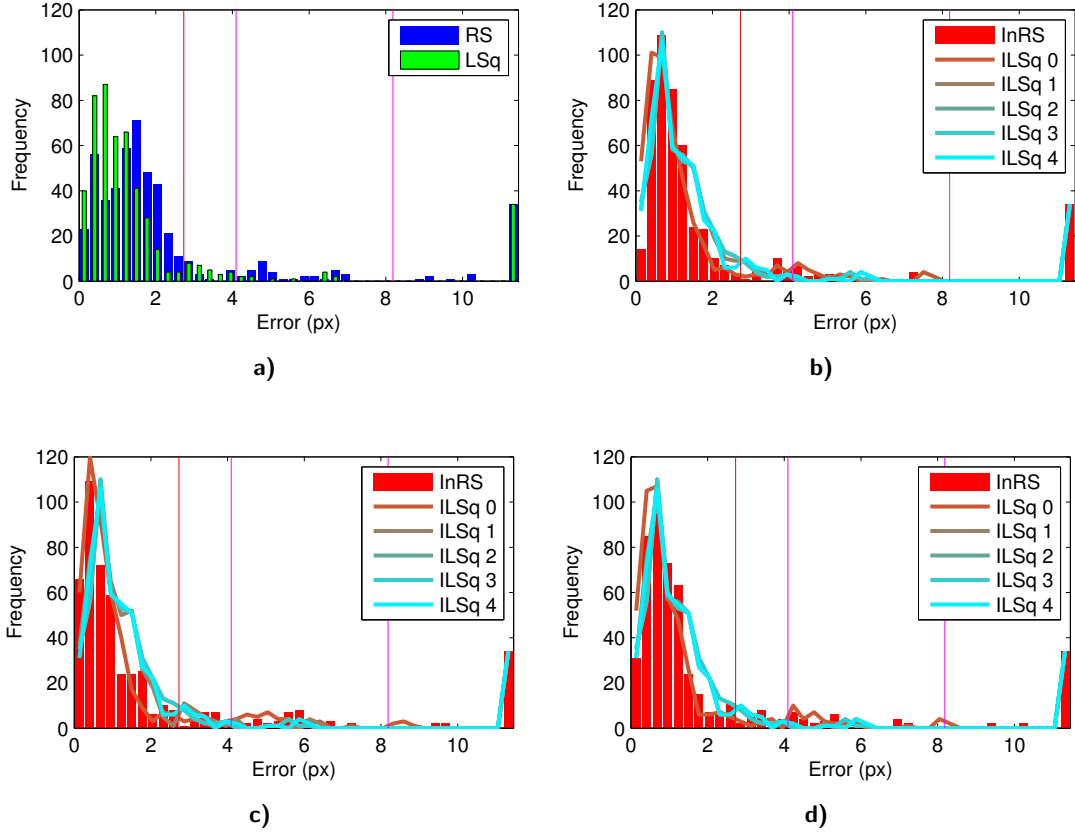


Figure 4.3. Histograms of errors in different stages of Local Optimization. The graphs demonstrate development of point errors (in absolute values) on the BRUSSELS image pair and the task of homography estimation. The blue bar graph in **a)** shows the errors of M_k . The errors of M_{m_θ} , plotted in green, are shifted to lower values significantly. Subfigures **b)** to **d)** reveal the development of the errors of M_{is} from three selected inner samples (red bars) and then the iterations of M' , from red to cyan. The red vertical lines denote the error threshold, while the magenta lines stands for a range of a widened quadratic cost function (see further) and its multiple by m_θ . In this case, the Local Optimization converged to the same solution in all ten cases, despite the different random inner samples drawn. This can be seen from the same shapes of the cyan curves.

4. Locally Optimized RANSAC

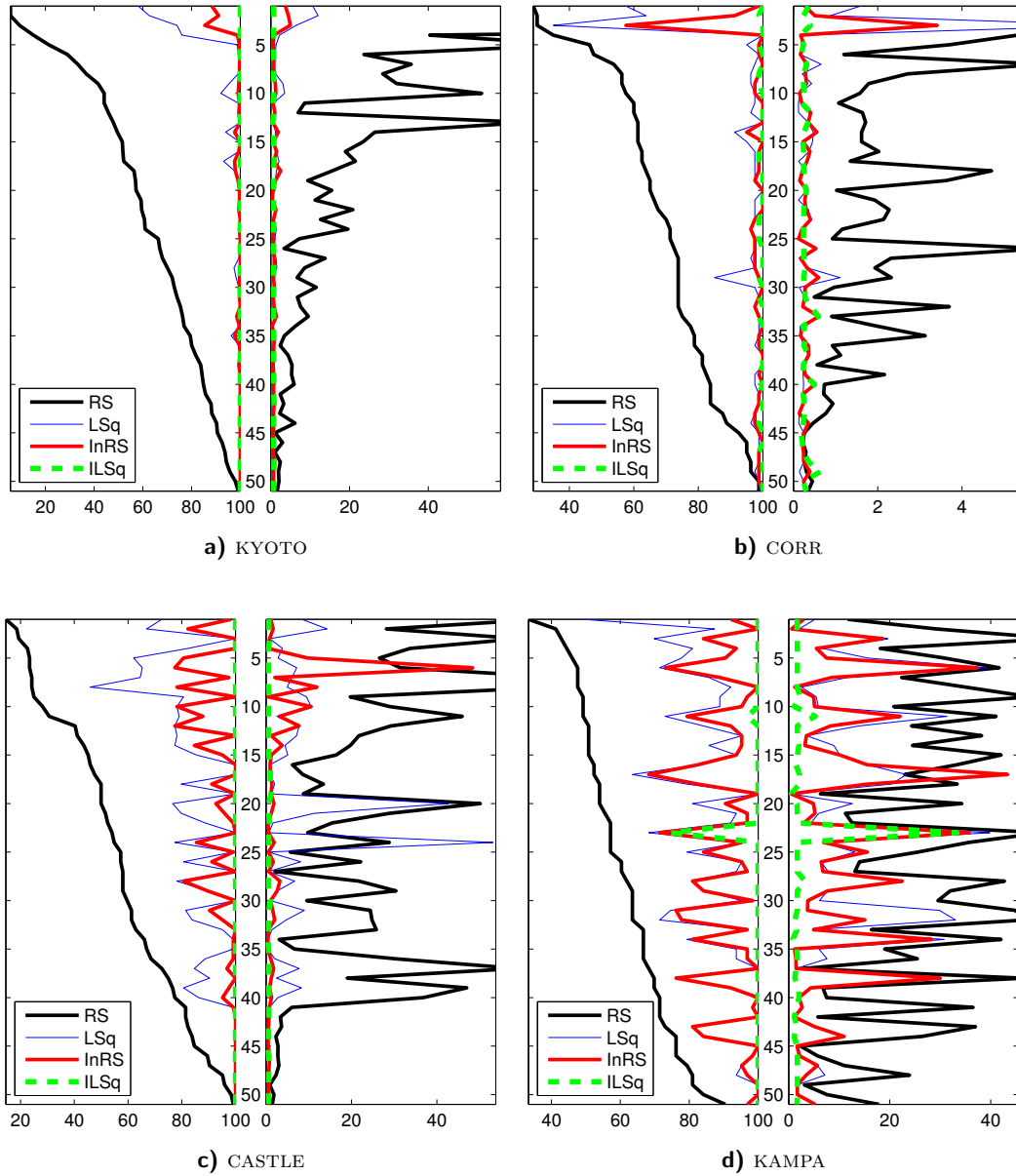


Figure 4.4. Numbers of inliers and geometric errors of geometries estimated in different steps of Local Optimizations run on all-inlier samples. Every point on a y -axis represents one run of LO (sorted according to the number of inliers from the sample). The left-hand part shows the fraction of inliers found and the right-hand part shows the RMS of Sampson's error on *ground truth* (GT) points. The assessed models are M_k , M_{m_θ} , the best M_{is} and the best M_r (see Algorithms 1 and 2). The subfigure **a)** shows the results on the easy image pair KYOTO, where even only simple linear least squares give nearly perfect solution. The CORR (**b**) pair is still rather easy, where simpler variant of LO would be sufficient. On the other hand, **c)** CASTLE pair is more difficult and **d)** KAMPA would have poor results without the full-power Local Optimization.

5. On the Speed of Local Optimization

The original paper [10] claimed that Local Optimization is an improvement at no cost. Since it is run $\log(K)$ -times only, its affect on the total execution time seems to be negligible. Furthermore, with an increased number of inliers during the run time, it is even supposed to make RANSAC faster. This definitely holds for problems with low inlier ratios and thus long runs. However, with a high number of inliers, standard RANSAC itself is fast enough and LO may dominate the computation time. Therefore we state the following.

“ For problems with a high fraction of inliers, there is a trade-off between speed and precision, when a stronger LO implies higher time demands. ”

“ In an experimental evaluation of RANSAC-like algorithms, we should never rely on the number of samples. The speed should be always measured by a wall clock time as well. ”

5.1. LO-RANSAC time consumption

Formally, the total execution time of LO-RANSAC is

$$t_{\text{tot}} = C_{\text{R}} \cdot K + C_{\text{LO}} \cdot \lceil \log(K) \rceil , \quad (5.1)$$

where C_{LO} is the average time of the Local Optimization procedure and C_{R} is the time of standard RANSAC single hypothesis generation and verification round (including multiple verifications for the cases of multiple models per one sample). For small K , the significant difference of C_{LO} and C_{R} means that $C_{\text{LO}} \cdot \lceil \log(K) \rceil \gg C_{\text{R}} \cdot K$. It happens often that even single run of LO takes more time than the rest of RANSAC. This may render LO-RANSAC considerably slower than standard RANSAC.

E.g. for a particular case of HEAD pair with 77 inliers (out of 86; 90 %) is $K \approx 13$. The standard RANSAC execution time $C_{\text{R}} \cdot K$ is only 0.3 ms while a single local optimization (C_{LO}) takes 5.8 ms on average. For the case of BOOKSH pair (30 inliers out of 41; 70 %) is $K \approx 41$, $C_{\text{R}} \cdot K = 0.6$ ms and C_{LO} takes 4.6 ms (see Figure 5.1).

The full local optimization step uses many repetitions of linear least squares to improve the quality of the estimated model. For a large number of inliers (both relative and absolute), this procedure can dominate the execution time. Basically, there are two different approaches to reduce the complexity. The first one is to speed up every repetition; the second approach is to reduce the number of repetitions. The following sections describe possible methods of doing so.

5.2. Limiting the Number of Equations

In the linear least squares computation, the time complexity is at best linear in the number of used data points (correspondences). This holds for computation by eigen-decomposition of a covariance matrix of the data. However, it can be even worse when

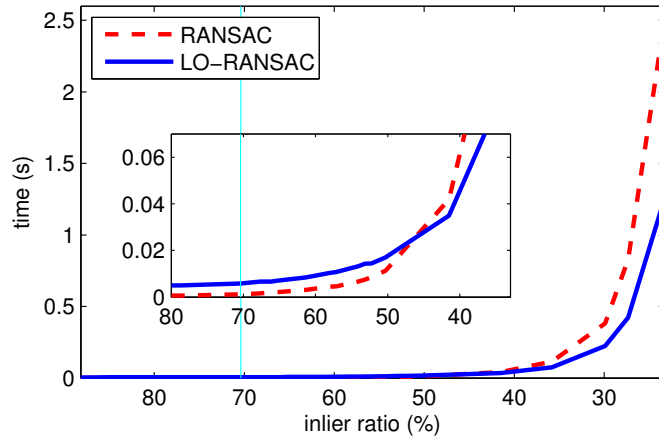


Figure 5.1. Dependence of the execution time on a fraction of inliers. The behavior of RANSAC with and without Local Optimization on the BOOKSH image pair (the percentage of inliers is controlled by the threshold of the second closest match during the matching process; the vertical cyan line marks the default value). In the cases of low inlier ratios, where the numbers of samples drawn by every RANSAC are high, LO-RANSAC is faster. However, LO introduces an additional overhead, so for high inlier ratios the curves cross.

using techniques such as Singular Value Decomposition (SVD). On the other hand, the uncertainty of model estimation is inversely proportional to a square root of the sample size only. This means that after reaching some number of points, the rise of precision is negligible and does not worth the effort of solving more and more equations.

Following this philosophy, we propose to introduce a limit on the number of inliers used for an estimation of M' (Algorithm 3). When this limit is reached, a limit-sized subset of inliers is randomly chosen and used for the estimation.

“ The approach of an inlier limit reduces the cost of the least squares estimation almost to a constant independent of the number of inliers. ”

In our implementation, the limit was empirically set to $7 \times m$ (minimal sample size), as this is the point, where the consumed time is already lower, but the precision is still at the unlimited level for most of the image pairs (see Figure 5.2). This is a rather conservative setting, for further speed-up it could be lowered to 6, or even $4 \times m$.

With these settings, the precision of the estimation is not negatively affected. In fact, it is sometimes even improved for a non-negligible number of epipolar geometry estimation tasks. We explain such a behavior as avoiding sticking in a local minimum by randomizing the set of points used in each iteration of the least squares.

5.3. Do Not Redo Done

During the least squares iterations, it can often happen that the same solution is found multiple times. The possible causes are similar initialization of Local Optimization, similar samples drawn in the inner sampling, or fast-enough convergence of the iteration. In all these cases, the same set of inliers may appear, in which case we may want to stop computation. To recognize such a situation, we store hashes of inlier sets and look for duplicities. When a duplicity is found, the iteration is stopped as everything what would happen in the future was probably already seen before.

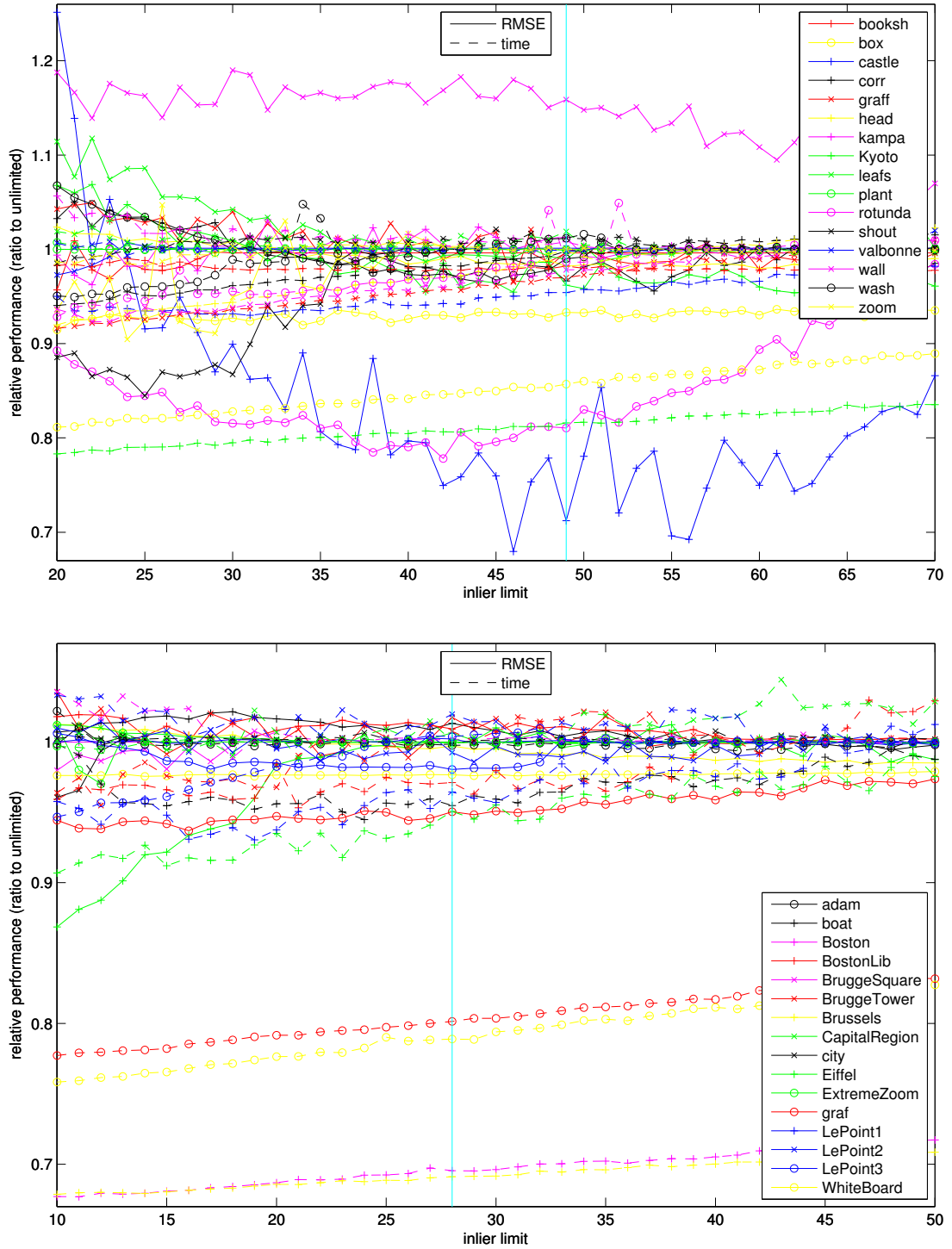


Figure 5.2. Dependence of LO-RANSAC performance on the inlier limit. The execution time and root-mean-square error on GT points were recorded (the ratios to results of unlimited LO are shown in the graphs). As conservative default values were selected limits denoted by vertical cyan lines – $7 \times m$, where speedup is already significant, but the geometric error is has not yet risen much above the unlimited case for a majority of image pairs. The upper part shows the effect on epipolar geometry estimation while the lower half illustrates the effect on homography estimation.

5. On the Speed of Local Optimization

Despite the intuitive insight that the hashing cuts off only exactly the same computational branches, this is not necessarily always the case. During the iteration with narrowing threshold, if the same set of inliers was seen at the different thresholds, the iterations can end up in different solutions. Furthermore, since for some geometric models (e.g. epipolar geometry) we use weighted least squares, the same set of inliers (even w.r.t. the same threshold) with different weights can give different solution. However, the practical effect of these differences is negligible.

5.4. Simpler Variants of Local Optimization

It was shown that every stronger variant of Local Optimization yield better and more stable results than simpler ones ([10] and Figure 4.2). However, for easier tasks, where no challenging camera motion or illumination changes are expected (e.g. tracking or Narrow Baseline Stereo, NBS) and where high inlier ratio can be expected, the results of weaker variants are close.

For such tasks, we propose to use a lightweight version of Local Optimization – LO', based on Chum's [10] variant 3 (Iterative). Instead of estimating models from non-minimal samples followed by iterative least squares, only a single iterative least squares are applied to each so-far-the-best model. The limit on the number of inliers used in each step of the least squares is applied. The experimental evaluation shows significant reduction in the execution time while sacrificing only little accuracy, especially for "easy" image pairs.

Table 5.1. Comparison of speed and precision of different LO speed-ups for the epipolar geometry estimation. Please note that the MSAC (truncated quadratic) cost function was used, see section 6.2 for details. The use of an inlier limit is denoted as LO⁺ according to [18]; # means that hashing was used.

Algorithm		MSAC	LO	LO'	LO ⁺	LO#	LO ⁺ #
booksh	Inliers	28.8 ± 1.7	29.8 ± 1.2	29.1 ± 1.5	29.8 ± 1.2	29.3 ± 1.5	29.3 ± 1.5
	Inliers (%)	70.2 ± 4.0	72.7 ± 2.8	71.0 ± 3.6	72.7 ± 2.8	71.4 ± 3.6	71.4 ± 3.6
	Error (px)	3.13 ± 4.41	1.84 ± 3.10	2.75 ± 4.19	1.84 ± 3.10	2.31 ± 3.52	2.32 ± 3.53
	Samples	43.6 ± 18.6	40.5 ± 14.4	41.3 ± 15.3	40.5 ± 14.4	41.4 ± 15.6	41.4 ± 15.6
	Time (ms)	0.6 ±	5.2 ±	1.2 ±	5.3 ±	3.9 ±	3.8 ±
	LO count	0.0 ± 0.0	1.0 ± 0.2	1.1 ± 0.3	1.0 ± 0.2	1.1 ± 0.3	1.1 ± 0.3
box	Inliers	195.8 ± 6.7	204.3 ± 12.1	207.9 ± 13.0	203.3 ± 12.2	203.8 ± 12.1	202.3 ± 12.3
	Inliers (%)	84.8 ± 2.9	88.5 ± 5.2	90.0 ± 5.6	88.0 ± 5.3	88.2 ± 5.3	87.6 ± 5.3
	Error (px)	38.67 ± 26.49	43.70 ± 32.69	35.03 ± 33.69	42.44 ± 29.64	45.20 ± 32.43	43.91 ± 28.45
	Samples	8.8 ± 1.7	8.8 ± 1.7	8.8 ± 1.7	8.8 ± 1.7	8.8 ± 1.7	8.8 ± 1.7
	Time (ms)	0.5 ±	8.5 ±	1.5 ±	6.9 ±	4.6 ±	4.0 ±
	LO count	0.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
castle	Inliers	110.0 ± 7.3	122.8 ± 2.0	119.6 ± 4.9	122.6 ± 1.9	122.3 ± 2.2	122.4 ± 1.9
	Inliers (%)	71.4 ± 4.8	79.7 ± 1.3	77.7 ± 3.2	79.6 ± 1.2	79.4 ± 1.4	79.5 ± 1.2
	Error (px)	4.41 ± 6.82	0.84 ± 1.68	2.10 ± 5.59	0.79 ± 1.22	0.84 ± 1.71	0.79 ± 1.19
	Samples	41.2 ± 17.6	37.6 ± 12.3	37.9 ± 12.7	37.6 ± 12.3	37.6 ± 12.3	37.6 ± 12.3
	Time (ms)	1.0 ±	7.5 ±	1.9 ±	6.8 ±	5.4 ±	6.1 ±
	LO count	0.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.1	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
corr	Inliers	73.3 ± 3.9	80.5 ± 0.7	77.8 ± 2.1	81.2 ± 0.8	80.5 ± 0.7	80.9 ± 0.9
	Inliers (%)	78.8 ± 4.2	86.5 ± 0.7	83.7 ± 2.3	87.3 ± 0.9	86.5 ± 0.8	87.0 ± 1.0
	Error (px)	0.55 ± 0.34	0.33 ± 0.06	0.28 ± 0.20	0.38 ± 0.07	0.34 ± 0.07	0.36 ± 0.08
	Samples	18.0 ± 7.2	18.0 ± 7.2	18.0 ± 7.2	18.0 ± 7.2	18.0 ± 7.2	18.0 ± 7.2
	Time (ms)	0.5 ±	6.2 ±	1.3 ±	6.0 ±	4.5 ±	5.0 ±
	LO count	0.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
graff	Inliers	94.6 ± 3.5	104.7 ± 1.1	102.8 ± 2.2	102.7 ± 1.9	104.6 ± 1.3	102.7 ± 1.9
	Inliers (%)	78.8 ± 2.9	87.2 ± 0.9	85.7 ± 1.9	85.6 ± 1.6	87.2 ± 1.0	85.6 ± 1.6
	Error (px)	2.34 ± 1.50	0.53 ± 0.45	0.68 ± 0.48	0.59 ± 0.44	0.55 ± 0.50	0.59 ± 0.44
	Samples	16.0 ± 4.5	16.0 ± 4.5	16.0 ± 4.5	16.0 ± 4.5	16.0 ± 4.5	16.0 ± 4.5
	Time (ms)	0.5 ±	6.8 ±	1.3 ±	6.2 ±	5.6 ±	6.1 ±
	LO count	0.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0

5.4. Simpler Variants of Local Optimization

head	Inliers	72.2 ± 4.1	77.0 ± 0.2	77.0 ± 0.2	77.0 ± 0.0	77.0 ± 0.1	77.0 ± 0.1
	Inliers (%)	84.0 ± 4.7	89.5 ± 0.2	89.5 ± 0.3	89.5 ± 0.0	89.5 ± 0.2	89.5 ± 0.1
	Error (px)	0.89 ± 0.62	0.32 ± 0.05	0.31 ± 0.04	0.30 ± 0.02	0.31 ± 0.04	0.31 ± 0.03
	Samples	13.1 ± 7.0	13.1 ± 7.0	13.1 ± 7.0	13.1 ± 7.0	13.1 ± 7.0	13.1 ± 7.0
	Time (ms) LO count	0.3 ± 0.0 ± 0.0	6.1 ± 1.0 ± 0.0	1.1 ± 1.0 ± 0.0	6.0 ± 1.0 ± 0.0	2.8 ± 1.0 ± 0.0	2.7 ± 1.0 ± 0.0
kampa	Inliers	54.0 ± 3.7	59.2 ± 2.1	57.9 ± 3.2	60.3 ± 2.7	59.0 ± 2.1	60.0 ± 2.6
	Inliers (%)	64.2 ± 4.4	70.5 ± 2.5	68.9 ± 3.8	71.7 ± 3.2	70.2 ± 2.5	71.5 ± 3.1
	Error (px)	12.11 ± 10.10	4.69 ± 5.53	7.89 ± 10.06	4.29 ± 5.38	4.93 ± 5.45	4.41 ± 5.27
	Samples	82.7 ± 31.4	53.6 ± 15.7	61.2 ± 21.7	53.3 ± 15.5	53.8 ± 15.7	53.3 ± 15.4
	Time (ms) LO count	1.4 ± 0.0 ± 0.0	6.5 ± 1.0 ± 0.2	1.9 ± 1.2 ± 0.5	6.5 ± 1.0 ± 0.2	5.9 ± 1.0 ± 0.2	6.0 ± 1.0 ± 0.2
Kyoto	Inliers	322.4 ± 15.0	346.2 ± 1.4	337.3 ± 10.1	346.7 ± 1.6	346.1 ± 1.5	346.8 ± 1.6
	Inliers (%)	72.5 ± 3.4	77.8 ± 0.3	75.8 ± 2.3	77.9 ± 0.4	77.8 ± 0.3	77.9 ± 0.4
	Error (px)	2.68 ± 1.54	0.76 ± 0.23	1.87 ± 1.21	0.80 ± 0.20	0.80 ± 0.25	0.80 ± 0.20
	Samples	32.7 ± 11.6	32.1 ± 10.1	32.1 ± 10.1	32.1 ± 10.1	32.1 ± 10.1	32.1 ± 10.1
	Time (ms) LO count	1.4 ± 0.0 ± 0.0	13.0 ± 1.0 ± 0.0	3.0 ± 1.0 ± 0.0	9.5 ± 1.0 ± 0.0	11.0 ± 1.0 ± 0.0	9.7 ± 1.0 ± 0.0
leafs	Inliers	52.8 ± 3.2	59.0 ± 0.3	58.8 ± 0.8	59.0 ± 0.3	58.9 ± 0.4	58.9 ± 0.4
	Inliers (%)	66.9 ± 4.1	74.6 ± 0.3	74.5 ± 0.9	74.6 ± 0.3	74.6 ± 0.4	74.5 ± 0.6
	Error (px)	7.71 ± 6.06	2.38 ± 0.19	2.53 ± 0.92	2.34 ± 0.26	2.39 ± 0.27	2.43 ± 0.43
	Samples	69.3 ± 34.0	47.7 ± 11.8	48.1 ± 12.4	47.7 ± 11.8	47.7 ± 11.8	47.7 ± 11.8
	Time (ms) LO count	0.9 ± 0.0 ± 0.0	6.0 ± 1.0 ± 0.1	1.4 ± 1.0 ± 0.2	6.1 ± 1.0 ± 0.1	3.7 ± 1.0 ± 0.1	3.9 ± 1.0 ± 0.1
plant	Inliers	19.5 ± 1.6	21.2 ± 2.0	20.6 ± 1.7	21.2 ± 2.0	21.1 ± 2.0	21.1 ± 2.0
	Inliers (%)	65.0 ± 5.2	70.7 ± 6.8	68.8 ± 5.6	70.7 ± 6.8	70.5 ± 6.7	70.5 ± 6.7
	Error (px)	16.96 ± 14.08	12.23 ± 14.38	15.77 ± 13.69	12.22 ± 14.38	12.37 ± 14.36	12.38 ± 14.36
	Samples	82.5 ± 35.4	69.6 ± 30.6	68.0 ± 28.9	69.6 ± 30.6	69.7 ± 30.6	69.7 ± 30.6
	Time (ms) LO count	1.0 ± 0.0 ± 0.0	4.0 ± 1.4 ± 0.7	1.6 ± 1.3 ± 0.6	3.9 ± 1.4 ± 0.7	3.9 ± 1.4 ± 0.7	3.9 ± 1.4 ± 0.7
rotunda	Inliers	72.0 ± 3.1	75.0 ± 0.2	74.2 ± 2.0	75.0 ± 0.0	74.9 ± 0.4	75.0 ± 0.3
	Inliers (%)	83.7 ± 3.6	87.2 ± 0.2	86.3 ± 2.4	87.2 ± 0.0	87.1 ± 0.5	87.2 ± 0.4
	Error (px)	1.28 ± 0.85	0.48 ± 0.14	0.58 ± 0.44	0.38 ± 0.09	0.47 ± 0.14	0.46 ± 0.15
	Samples	11.9 ± 5.7	11.9 ± 5.7	11.9 ± 5.7	11.9 ± 5.7	11.9 ± 5.7	11.9 ± 5.7
	Time (ms) LO count	0.3 ± 0.0 ± 0.0	6.0 ± 1.0 ± 0.0	1.1 ± 1.0 ± 0.0	5.9 ± 1.0 ± 0.0	3.1 ± 1.0 ± 0.0	3.0 ± 1.0 ± 0.0
shout	Inliers	41.0 ± 1.5	42.4 ± 0.8	41.5 ± 1.2	42.4 ± 0.8	42.1 ± 0.9	42.1 ± 0.9
	Inliers (%)	76.0 ± 2.7	78.6 ± 1.5	76.8 ± 2.3	78.6 ± 1.4	77.9 ± 1.7	77.9 ± 1.7
	Error (px)	1.66 ± 0.94	0.86 ± 0.51	1.48 ± 0.95	0.86 ± 0.51	0.85 ± 0.62	0.85 ± 0.62
	Samples	20.8 ± 6.4	20.8 ± 6.4	20.8 ± 6.4	20.8 ± 6.4	20.8 ± 6.4	20.8 ± 6.4
	Time (ms) LO count	0.4 ± 0.0 ± 0.0	5.3 ± 1.0 ± 0.0	1.0 ± 1.0 ± 0.0	5.4 ± 1.0 ± 0.0	3.9 ± 1.0 ± 0.0	4.0 ± 1.0 ± 0.0
valbonne	Inliers	24.8 ± 1.7	26.4 ± 0.9	25.3 ± 1.3	26.4 ± 0.9	25.9 ± 1.0	25.9 ± 1.0
	Inliers (%)	77.6 ± 5.2	82.6 ± 2.8	79.1 ± 4.1	82.6 ± 2.8	80.9 ± 3.2	80.9 ± 3.3
	Error (px)	30.25 ± 17.38	26.29 ± 8.97	28.40 ± 14.96	26.29 ± 8.99	27.71 ± 9.95	27.71 ± 9.96
	Samples	22.4 ± 11.2	22.3 ± 10.9	22.3 ± 10.9	22.3 ± 10.9	22.3 ± 10.9	22.3 ± 10.9
	Time (ms) LO count	0.4 ± 0.0 ± 0.0	4.0 ± 1.0 ± 0.0	1.0 ± 1.0 ± 0.0	4.0 ± 1.0 ± 0.0	3.1 ± 1.0 ± 0.0	2.9 ± 1.0 ± 0.0
wall	Inliers	83.6 ± 4.8	91.0 ± 0.0	91.0 ± 0.0	91.0 ± 0.0	91.0 ± 0.0	91.0 ± 0.2
	Inliers (%)	85.4 ± 4.9	92.9 ± 0.0	92.9 ± 0.0	92.9 ± 0.0	92.9 ± 0.0	92.8 ± 0.2
	Error (px)	2.42 ± 2.02	0.40 ± 0.03	0.39 ± 0.01	0.54 ± 0.06	0.39 ± 0.02	0.51 ± 0.12
	Samples	11.0 ± 5.5	11.0 ± 5.5	11.0 ± 5.5	11.0 ± 5.5	11.0 ± 5.5	11.0 ± 5.5
	Time (ms) LO count	0.3 ± 0.0 ± 0.0	6.1 ± 1.0 ± 0.0	1.0 ± 1.0 ± 0.0	5.8 ± 1.0 ± 0.0	3.2 ± 1.0 ± 0.0	3.4 ± 1.0 ± 0.0
wash	Inliers	49.3 ± 2.6	52.0 ± 0.0	52.0 ± 0.0	52.0 ± 0.0	52.0 ± 0.0	52.0 ± 0.0
	Inliers (%)	89.6 ± 4.8	94.5 ± 0.0	94.5 ± 0.0	94.5 ± 0.0	94.5 ± 0.0	94.5 ± 0.0
	Error (px)	1.15 ± 0.71	0.30 ± 0.05	0.29 ± 0.02	0.28 ± 0.02	0.30 ± 0.03	0.29 ± 0.03
	Samples	7.3 ± 3.9	7.3 ± 3.9	7.3 ± 3.9	7.3 ± 3.9	7.3 ± 3.9	7.3 ± 3.9
	Time (ms) LO count	0.2 ± 0.0 ± 0.0	5.5 ± 1.0 ± 0.0	0.9 ± 1.0 ± 0.0	5.7 ± 1.0 ± 0.0	2.3 ± 1.0 ± 0.0	2.6 ± 1.0 ± 0.0
zoom	Inliers	41.5 ± 2.7	44.9 ± 0.7	43.7 ± 2.0	44.9 ± 0.7	44.8 ± 0.8	44.8 ± 0.8
	Inliers (%)	59.2 ± 3.9	64.1 ± 1.0	62.4 ± 2.8	64.1 ± 1.0	64.0 ± 1.1	64.0 ± 1.1
	Error (px)	2.14 ± 1.98	0.71 ± 0.70	1.62 ± 1.87	0.71 ± 0.70	0.83 ± 0.72	0.83 ± 0.72
	Samples	168.7 ± 79.9	80.5 ± 34.9	110.6 ± 54.0	80.4 ± 34.9	81.5 ± 35.3	81.5 ± 35.3
	Time (ms) LO count	2.1 ± 0.0 ± 0.0	8.1 ± 1.4 ± 0.7	2.6 ± 1.8 ± 1.0	8.2 ± 1.4 ± 0.7	6.0 ± 1.4 ± 0.7	5.7 ± 1.4 ± 0.7

5. On the Speed of Local Optimization

Table 5.2. Comparison of speed and precision of different LO speed-ups for the homography estimation. Please note that the MSAC (truncated quadratic) cost function was used, see section 6.2 for details. The use of an inlier limit is denoted as LO⁺ according to [18]; # means that hashing was used. For further description how to read these tables and how they were generated, please refer to Appendix B.

Algorithm		MSAC	LO	LO'	LO ⁺	LO#	LO ⁺ #
adam	Inliers	11.9 ± 1.2	17.0 ± 0.1	16.4 ± 0.5	17.0 ± 0.1	16.5 ± 0.5	16.5 ± 0.5
	Inliers (%)	59.6 ± 5.8	84.9 ± 0.6	82.1 ± 2.5	84.9 ± 0.6	82.6 ± 2.5	82.6 ± 2.5
	Error (px)	2.19 ± 1.61	0.94 ± 0.03	0.92 ± 0.07	0.94 ± 0.03	0.93 ± 0.07	0.93 ± 0.07
	Samples	23.1 ± 8.3	23.1 ± 8.2	23.1 ± 8.2	23.1 ± 8.2	23.1 ± 8.2	23.1 ± 8.2
	Time (ms)	0.4 ±	3.9 ±	0.7 ±	3.8 ±	2.1 ±	2.1 ±
	LO count	0.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
boat	Inliers	66.6 ± 6.5	82.0 ± 0.0	82.0 ± 0.0	80.9 ± 1.8	82.0 ± 0.0	80.9 ± 1.8
	Inliers (%)	52.8 ± 5.1	65.1 ± 0.0	65.1 ± 0.0	64.2 ± 1.4	65.1 ± 0.0	64.2 ± 1.4
	Error (px)	2.29 ± 0.76	1.59 ± 0.00	1.59 ± 0.00	1.63 ± 0.11	1.59 ± 0.00	1.64 ± 0.11
	Samples	47.7 ± 20.5	41.1 ± 10.2	41.1 ± 10.2	41.1 ± 10.2	41.1 ± 10.2	41.1 ± 10.2
	Time (ms)	1.1 ±	7.2 ±	1.5 ±	6.4 ±	3.9 ±	5.9 ±
	LO count	0.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
Boston	Inliers	291.1 ± 18.1	305.0 ± 0.0	305.0 ± 0.0	305.0 ± 0.0	305.0 ± 0.0	305.0 ± 0.0
	Inliers (%)	76.2 ± 4.7	79.8 ± 0.0	79.8 ± 0.0	79.8 ± 0.0	79.8 ± 0.0	79.8 ± 0.0
	Error (px)	2.17 ± 1.43	0.66 ± 0.00	0.66 ± 0.00	0.67 ± 0.05	0.66 ± 0.00	0.69 ± 0.08
	Samples	9.7 ± 4.3	9.7 ± 4.3	9.7 ± 4.3	9.7 ± 4.3	9.7 ± 4.3	9.7 ± 4.3
	Time (ms)	0.7 ±	15.7 ±	2.1 ±	9.6 ±	6.1 ±	4.4 ±
	LO count	0.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
BostonLib	Inliers	47.3 ± 3.0	51.0 ± 0.0	51.0 ± 0.3	51.0 ± 0.1	51.0 ± 0.0	51.0 ± 0.1
	Inliers (%)	23.6 ± 1.5	25.5 ± 0.0	25.5 ± 0.1	25.5 ± 0.0	25.5 ± 0.0	25.5 ± 0.0
	Error (px)	2.31 ± 1.41	0.48 ± 0.01	0.49 ± 0.35	0.49 ± 0.04	0.48 ± 0.01	0.51 ± 0.05
	Samples	1142.1 ± 422.9	728.0 ± 78.4	744.3 ± 134.6	727.9 ± 77.9	728.0 ± 78.4	728.0 ± 78.2
	Time (ms)	8.5 ±	16.9 ±	7.1 ±	16.6 ±	10.6 ±	10.3 ±
	LO count	0.0 ± 0.0	2.9 ± 1.2	3.0 ± 1.2	2.9 ± 1.2	2.9 ± 1.2	2.9 ± 1.2
BruggeSquare	Inliers	18.3 ± 1.3	20.9 ± 0.7	20.8 ± 1.0	20.9 ± 0.7	20.9 ± 0.8	20.9 ± 0.8
	Inliers (%)	39.9 ± 2.9	45.5 ± 1.4	45.1 ± 2.1	45.4 ± 1.4	45.5 ± 1.6	45.5 ± 1.6
	Error (px)	6.38 ± 2.64	4.37 ± 2.43	4.90 ± 2.39	4.43 ± 2.48	4.59 ± 2.52	4.63 ± 2.54
	Samples	127.4 ± 34.9	68.5 ± 9.7	73.5 ± 16.4	68.7 ± 9.8	68.7 ± 11.0	68.8 ± 11.1
	Time (ms)	1.3 ±	5.5 ±	1.2 ±	5.6 ±	3.2 ±	3.1 ±
	LO count	0.0 ± 0.0	1.2 ± 0.5	1.3 ± 0.6	1.2 ± 0.5	1.2 ± 0.5	1.2 ± 0.5
BruggeTower	Inliers	38.0 ± 2.8	45.9 ± 1.2	44.4 ± 1.5	46.3 ± 1.4	45.7 ± 1.5	46.3 ± 1.5
	Inliers (%)	49.4 ± 3.6	59.6 ± 1.6	57.6 ± 1.9	60.2 ± 1.8	59.3 ± 1.9	60.2 ± 1.9
	Error (px)	5.33 ± 2.59	4.24 ± 0.71	3.68 ± 0.93	4.64 ± 0.20	4.13 ± 0.62	4.63 ± 0.27
	Samples	56.0 ± 18.8	47.2 ± 6.8	47.2 ± 6.8	47.2 ± 6.8	47.2 ± 6.8	47.2 ± 6.8
	Time (ms)	0.8 ±	5.6 ±	1.1 ±	5.2 ±	3.4 ±	4.2 ±
	LO count	0.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
Brussels	Inliers	378.9 ± 33.4	441.7 ± 2.5	444.3 ± 3.5	439.9 ± 5.6	444.0 ± 4.7	439.9 ± 5.7
	Inliers (%)	75.3 ± 6.6	87.8 ± 0.5	88.3 ± 0.7	87.5 ± 1.1	88.3 ± 0.9	87.5 ± 1.1
	Error (px)	3.70 ± 1.10	2.31 ± 0.02	2.26 ± 0.05	2.33 ± 0.08	2.26 ± 0.06	2.33 ± 0.08
	Samples	11.1 ± 5.2	11.1 ± 5.2	11.1 ± 5.2	11.1 ± 5.2	11.1 ± 5.2	11.1 ± 5.2
	Time (ms)	1.1 ±	20.9 ±	3.0 ±	11.8 ±	11.3 ±	11.4 ±
	LO count	0.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
CapitalRegion	Inliers	64.4 ± 6.0	75.7 ± 1.3	74.9 ± 1.1	75.9 ± 1.9	75.0 ± 1.2	75.9 ± 2.0
	Inliers (%)	49.6 ± 4.6	58.2 ± 1.0	57.6 ± 0.8	58.4 ± 1.5	57.7 ± 1.0	58.4 ± 1.5
	Error (px)	5.40 ± 0.60	4.63 ± 0.04	4.56 ± 0.03	4.66 ± 0.08	4.60 ± 0.07	4.66 ± 0.08
	Samples	59.4 ± 23.1	46.3 ± 7.8	46.3 ± 7.8	46.3 ± 7.8	46.3 ± 7.8	46.3 ± 7.8
	Time (ms)	1.2 ±	7.4 ±	1.5 ±	6.3 ±	4.3 ±	6.2 ±
	LO count	0.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
city	Inliers	11.6 ± 1.1	14.0 ± 0.1	14.0 ± 0.1	14.0 ± 0.1	14.0 ± 0.1	14.0 ± 0.1
	Inliers (%)	68.3 ± 6.6	82.3 ± 0.5	82.3 ± 0.4	82.3 ± 0.5	82.3 ± 0.5	82.3 ± 0.5
	Error (px)	1.73 ± 1.08	0.79 ± 0.63	0.79 ± 0.63	0.79 ± 0.63	0.79 ± 0.63	0.79 ± 0.63
	Samples	14.6 ± 7.1	14.6 ± 7.1	14.6 ± 7.1	14.6 ± 7.1	14.6 ± 7.1	14.6 ± 7.1
	Time (ms)	0.3 ±	3.7 ±	0.6 ±	3.7 ±	1.7 ±	1.7 ±
	LO count	0.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
Eiffel	Inliers	68.2 ± 4.8	76.0 ± 0.3	75.7 ± 1.8	76.0 ± 0.4	76.0 ± 0.3	75.8 ± 0.9
	Inliers (%)	34.1 ± 2.4	38.0 ± 0.2	37.9 ± 0.9	38.0 ± 0.2	38.0 ± 0.2	37.9 ± 0.5
	Error (px)	1.47 ± 0.67	1.05 ± 0.08	1.18 ± 0.39	1.07 ± 0.09	1.10 ± 0.08	1.10 ± 0.14
	Samples	265.7 ± 102.4	142.3 ± 5.2	149.2 ± 26.5	142.0 ± 4.2	142.3 ± 5.4	143.4 ± 7.7
	Time (ms)	3.1 ±	14.8 ±	3.0 ±	13.2 ±	7.8 ±	8.8 ±
	LO count	0.0 ± 0.0	1.9 ± 0.9	2.0 ± 1.0	1.9 ± 0.9	1.9 ± 0.9	1.9 ± 0.9

5.4. Simpler Variants of Local Optimization

ExtremeZoom	Inliers	12.5 ± 1.2	13.4 ± 0.9	13.2 ± 1.0	13.4 ± 0.9	13.4 ± 0.9	13.4 ± 0.9
	Inliers (%)	22.4 ± 2.1	24.0 ± 1.5	23.6 ± 1.8	24.0 ± 1.5	23.9 ± 1.6	23.9 ± 1.5
	Error (px)	13.16 ± 26.92	6.73 ± 20.48	9.68 ± 21.14	6.73 ± 20.48	6.78 ± 20.33	6.77 ± 20.33
	Samples	1789.9 ± 887.1	1264.4 ± 654.1	1367.8 ± 709.6	1264.4 ± 654.1	1268.8 ± 653.7	1268.8 ± 653.8
	Time (ms)	11.4 ±	13.7 ±	9.9 ±	13.7 ±	11.7 ±	11.7 ±
	LO count	0.0 ± 0.0	3.6 ± 1.5	3.7 ± 1.5	3.6 ± 1.5	3.6 ± 1.5	3.6 ± 1.5
graf	Inliers	177.8 ± 16.1	201.3 ± 0.7	201.6 ± 5.2	201.5 ± 1.5	202.4 ± 0.9	201.2 ± 1.6
	Inliers (%)	72.6 ± 6.6	82.1 ± 0.3	82.3 ± 2.1	82.2 ± 0.6	82.6 ± 0.4	82.1 ± 0.7
	Error (px)	1.69 ± 0.71	1.27 ± 0.01	1.30 ± 0.08	1.27 ± 0.10	1.28 ± 0.02	1.27 ± 0.13
	Samples	12.8 ± 5.6	12.8 ± 5.6	12.8 ± 5.6	12.8 ± 5.6	12.8 ± 5.6	12.8 ± 5.6
	Time (ms)	0.8 ±	11.0 ±	1.8 ±	7.8 ±	4.9 ±	6.1 ±
	LO count	0.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
LePoint1	Inliers	72.7 ± 6.7	85.0 ± 2.6	82.9 ± 3.5	85.2 ± 2.2	84.1 ± 3.2	85.1 ± 2.3
	Inliers (%)	49.1 ± 4.5	57.5 ± 1.8	56.0 ± 2.3	57.5 ± 1.5	56.9 ± 2.1	57.5 ± 1.5
	Error (px)	3.06 ± 0.49	2.51 ± 0.17	2.46 ± 0.32	2.55 ± 0.16	2.49 ± 0.25	2.55 ± 0.16
	Samples	60.6 ± 22.3	46.4 ± 7.0	46.5 ± 7.0	46.4 ± 7.0	46.4 ± 7.0	46.4 ± 7.0
	Time (ms)	1.6 ±	8.3 ±	1.9 ±	6.9 ±	6.1 ±	7.0 ±
	LO count	0.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
LePoint2	Inliers	46.5 ± 3.7	56.3 ± 1.1	56.7 ± 1.2	55.7 ± 1.7	56.6 ± 0.9	55.7 ± 1.8
	Inliers (%)	52.3 ± 4.1	63.2 ± 1.2	63.7 ± 1.3	62.6 ± 2.0	63.6 ± 1.0	62.6 ± 2.0
	Error (px)	2.31 ± 1.08	1.59 ± 0.07	1.64 ± 0.21	1.59 ± 0.10	1.64 ± 0.05	1.59 ± 0.10
	Samples	42.7 ± 12.7	40.4 ± 8.9	40.4 ± 8.9	40.4 ± 8.9	40.4 ± 8.9	40.4 ± 8.9
	Time (ms)	1.0 ±	6.4 ±	1.4 ±	5.7 ±	4.2 ±	5.8 ±
	LO count	0.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
LePoint3	Inliers	29.3 ± 3.1	35.9 ± 0.5	35.9 ± 0.4	36.5 ± 0.6	35.8 ± 0.4	36.0 ± 0.8
	Inliers (%)	60.9 ± 6.6	74.8 ± 1.1	74.8 ± 0.9	75.9 ± 1.3	74.6 ± 0.9	75.0 ± 1.7
	Error (px)	4.37 ± 1.71	2.44 ± 0.09	2.33 ± 0.13	2.45 ± 0.09	2.35 ± 0.12	2.45 ± 0.12
	Samples	26.2 ± 11.4	26.0 ± 10.8	26.0 ± 10.8	26.0 ± 10.8	26.0 ± 10.8	26.0 ± 10.8
	Time (ms)	0.5 ±	4.6 ±	0.8 ±	4.6 ±	2.3 ±	2.5 ±
	LO count	0.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
WhiteBoard	Inliers	170.8 ± 10.9	177.0 ± 0.0	177.0 ± 0.0	177.3 ± 0.4	177.0 ± 0.0	177.3 ± 0.5
	Inliers (%)	79.8 ± 5.1	82.7 ± 0.0	82.7 ± 0.0	82.8 ± 0.2	82.7 ± 0.0	82.8 ± 0.2
	Error (px)	1.69 ± 0.65	1.10 ± 0.00	1.10 ± 0.00	1.10 ± 0.04	1.10 ± 0.00	1.11 ± 0.07
	Samples	8.3 ± 4.2	8.3 ± 4.2	8.3 ± 4.2	8.3 ± 4.2	8.3 ± 4.2	8.3 ± 4.2
	Time (ms)	0.4 ±	9.6 ±	1.3 ±	7.0 ±	4.0 ±	3.2 ±
	LO count	0.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0

6. Cost Function – Measurement of Model Quality

“ In this chapter as well as in the rest of this text, the word “error” is used in the meaning *Sampson’s linear approximation of correspondence reprojection error*, unless explicitly noted. ”

6.1. Error scale

In this section, the attention will be paid to the scale of correspondence error. As the error of an outlying correspondence can be arbitrary, those will not concern us at all now. The error of inliers depends mostly on a quality of the used detector of feature points. It could be estimated using a variance of inliers and then an error threshold can be calculated. However, to decide, which points are inliers, we need to know the error threshold beforehand. This leads to a typical case of *chicken-and-egg problem*. In RANSAC, this is solved by assuming that the inlier/outlier threshold is supplied by the user and the inlier set is detected.

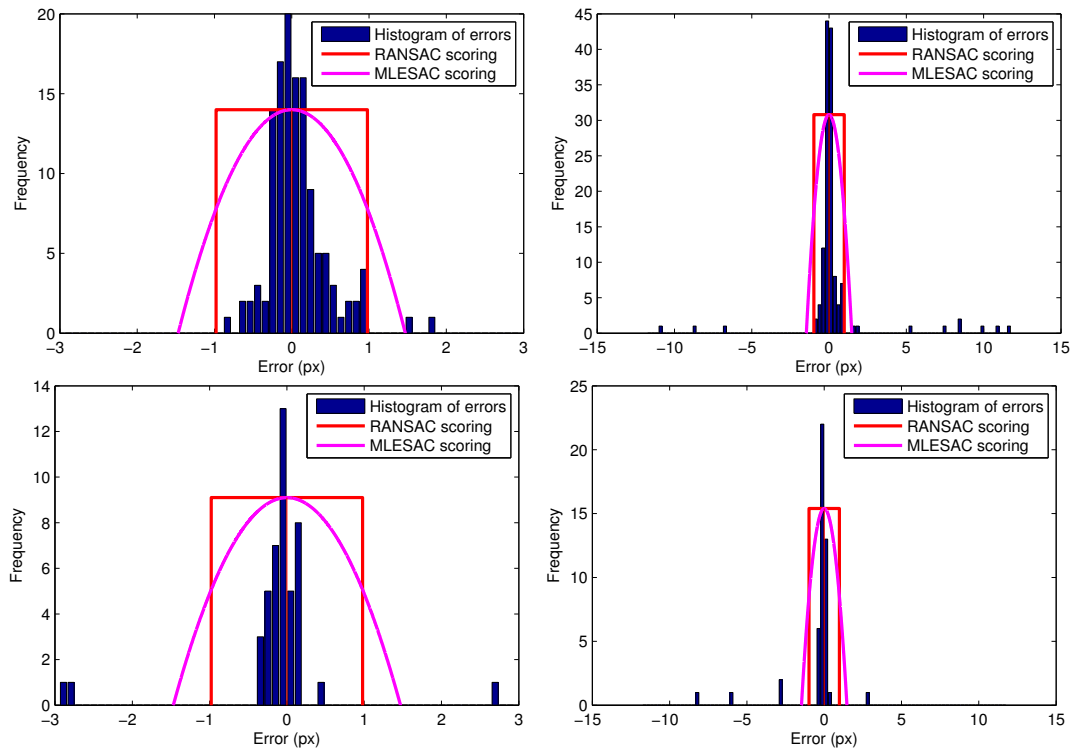


Figure 6.1. Error histograms. The upper row shows the histograms of errors of CASTLE image pair, where the error threshold is computed nearly optimal. The lower row shows distribution of error of SHOUT pair, where the measurement noise has been overestimated. Whilst the left close-ups show the distribution of inliers, the right images demonstrate the dichotomy in the data. There are inliers and nearly-inliers with low error (attributable to the noise), far outliers with error orders of magnitude higher and only little close outliers. Two scoring functions are superimposed – top-hat and truncated quadratic; these are discussed in section 6.2.

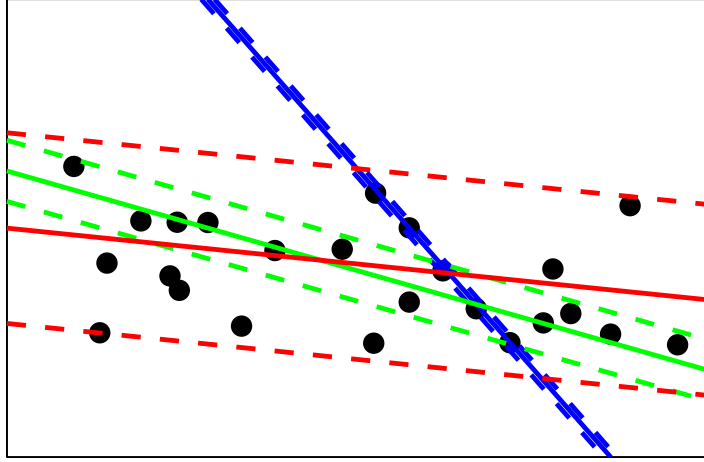


Figure 6.2. Different error threshold means a different problem to be solved. Data points and fitted lines with error bounds. In the case illustrated in this figure, three different models can be a global optima, each w.r.t. an another error scale, which is a part of a problem definition here.

This is sometimes referred to as a flaw, as the error threshold may vary widely in different tasks [24, 28]. However, the error threshold can be also seen as a part of problem definition, as illustrated in Figure 6.2. In this case, any automatic threshold selection would lead to the selection of *some* model, disregarding user preference. In the worst case, different models can be returned in different runs if the automatically selected threshold would be between the proper ones.

There are many ways how we can select the right threshold. E.g. we can just manually examine the distribution of errors of inliers and use some rule of thumb. As the other extreme, it is possible to use the Chebyshev inequality when we know only a variance of the error distribution (and not the distribution itself). Somewhere in between lies the approach of assuming for the error some known distribution with known variance. If the assumption is, for example, that image measurements are burdened by an isotropic Gaussian noise, then the distribution of error is χ^2 with two degrees of freedom (DoF) in the case of homography estimation and with one DoF for epipolar geometry (as a homography defines a 2D manifold in 4D joint-image-coordinates space while the manifold of epipolar geometry is three-dimensional). We can then set the error threshold to some (e.g. 95th) percentile of this distribution to capture the respective fraction of inliers:

$$\theta^2(\sigma) = \chi_{d,95\%}^2 \cdot \sigma^2, \quad (6.1)$$

where d is the number of DoF and σ^2 is expected variance of error of inliers.

In this approach, the error is treated solely as a noise on the correspondence coordinates while model is assumed to be exact. However, any model is constructed from the very data points, which means it has inherently some non-zero imprecision (even magnified in the ill-conditioned cases, e.g. from points lying close to each other). Thus even a good point may have large error, when it lies far from the model-generating sample where the constraints, imposed by the model, are rather extrapolated than interpolated. Raguram et al. have taken the uncertainty of the model directly into account in their CovRANSAC [32]; this will be seen in further detail in 6.4.

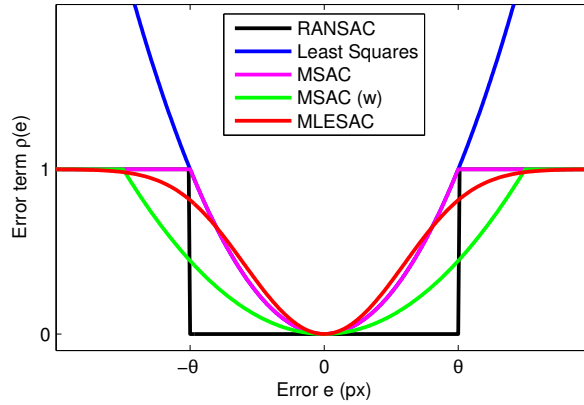


Figure 6.3. Overview of different cost functions. While Least Squares method (blue) highly penalizes gross errors, the others are more robust. Truncated quadratic (MSAC, magenta) one limits the error term at reaching the threshold θ . Top-hat cost function of RANSAC (black) only counts the number of inliers. The red curve of MLESAC is computed as a log-likelihood including outliers. The green cost function is the one used for experiments in this text, widened to have the same area as the top-hat one. (The functions have been shifted and scaled for easier comparison; this has no effect on the optimization results.) Inspired by [5].

6.2. Thresholding or Truncating

The original RANSAC of Fischler and Bolles maximizes the number of inliers.

“ The thresholding of correspondences to divide inliers and outliers can be seen as applying a *top-hat* cost function on the correspondence error. ”

The correspondences with an error under the threshold (the inliers) have zero cost, the outliers have constant cost; or alternatively, in the manner of inlier-count *gain* function to be maximized, inliers score one while outliers zero. The minimized cost function is defined as [38]:

$$C = \sum_{i=1}^N \rho(e_i^2), \quad (6.2)$$

$$\rho_R(e^2) = \begin{cases} 0 & e^2 \leq \theta^2 \\ \text{const.} & e^2 > \theta^2 \end{cases}. \quad (6.3)$$

The top-hat cost function works reasonably well, when the error threshold is correctly set. However, it is rather sensitive to the selection of the error scale. When the threshold is set too high, different models may have the same score, resulting in output instability. Torr and Zisserman proposed slightly different error term ρ_M for their MSAC [37], which simply truncates the squared error:

$$\rho_M(e^2) = \begin{cases} e^2 & e^2 \leq \theta^2 \\ \theta^2 & e^2 > \theta^2 \end{cases}. \quad (6.4)$$

This leads to a better precision at no additional cost. Furthermore, it was shown [18] that it is much more robust to the error threshold selection. In our experiments (unless noted otherwise) we use the truncated quadratic function widened by a factor 3/2, to have the same area as the top-hat cost function (so it would have the same score

on uniformly distributed data). In this case, it is also possible to follow the principle “the data optimized act in computation”, widening the error threshold by the same factor when marking the inliers for a least squares computation. The effect on results is positive, but not very significant.

6.3. Towards Maximum Likelihood Estimation

Torr and Zisserman later proposed an algorithm dubbed MLESAC [38]. They model the correspondence error as a mixture of Gaussian noise and uniformly distributed gross error of outliers:

$$P(e) = \varepsilon \cdot \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{e^2}{2\sigma^2}\right) + (1 - \varepsilon) \cdot \frac{1}{v}, \quad (6.5)$$

where ε is the mixing parameter (equal to the fraction of inliers) and v is a constant, given by geometric properties of the matching process (we expect error of outliers to fall within a range of $[-\frac{v}{2}, +\frac{v}{2}]$). The error minimized is the negative log likelihood.

The original paper suggested estimation of parameters, e.g. by the EM algorithm. However, Tordoff and Murray have shown [34] that monotonicity is preserved under changes of the mixing parameter – better hypotheses still score better, independent of the selection. Therefore the log likelihood may be fixed, becoming a function of the error only. Thus it can be seen as another cost function, as illustrated in Figure 6.3, while hypotheses are generated the same way as in standard RANSAC.

In the figure can be also observed the similarity between this cost function and truncated quadratic one of MSAC. In fact, MSAC can be perceived as a fast approximation to the maximum likelihood solution. Authors of algorithms indeed stated the performance of both as very close to each other, when MLESAC slightly more accurate and MSAC slightly faster.

6.4. Uncertainty of Model

The deviation of a datum from a model is a function of the error associated with the datum and the error associated with the model [12]. In standard RANSAC, the error is treated solely as a noise on the correspondence coordinates while model is assumed to be exact, as mentioned at the beginning of this chapter. This approximation has its role in the definition of inliers and in the derivation of RANSAC properties. However, since it does not hold exactly, the behavior of RANSAC differs from the theoretical predictions. E.g. [34] show that the first occurrence of all-inlier sample happens on average much later than predicted. This is caused by calculation of model from the noisy data, i.e. even the model is not exact. This issue was addressed also in [10], where the *effect* of imperfect model estimation is dealt with.

Raguram et al. in their CovRANSAC [32] have instead taken into account the uncertainty of the model explicitly. Using this, the covariance matrix of a transformed point (a point again in the case of homography or a line for epipolar geometry) can be established. Then a probability of a correspondence being correct may be observed and used to compute a set of “possible inliers”. In the CovRANSAC, when a high enough support is found, the loop is interrupted and a new RANSAC is run on this set, which has obviously a higher inlier ratio (hence it terminates after a small number of iterations). It is nevertheless possible to employ the detection of possible inliers in different ways, e.g. for the $\mathcal{I}_{\text{base}}$ computation.

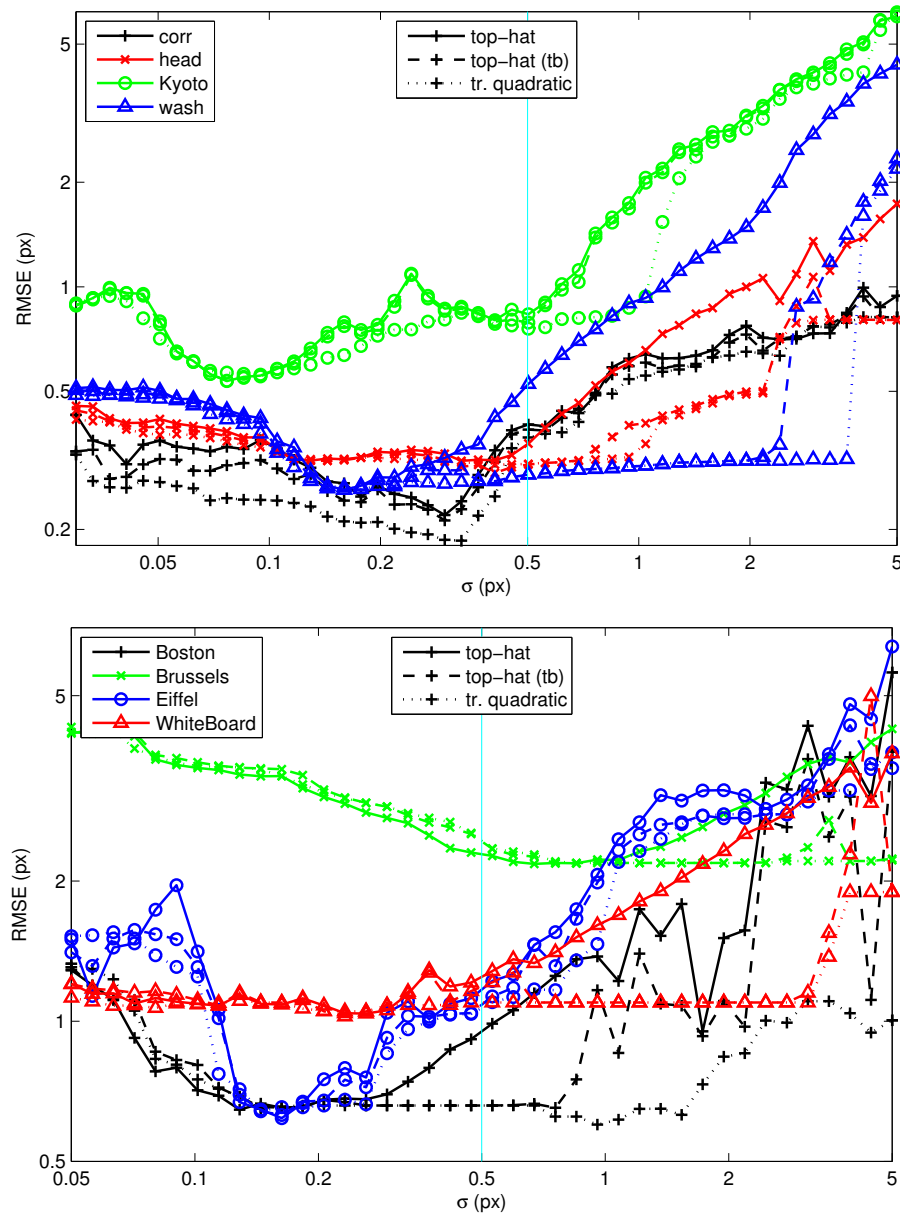


Figure 6.4. Effect of the used cost function on the robustness against changes of error scale.

The upper image shows results on selected image pairs for epipolar geometry estimation, the lower one shows results for homography estimation. On the x -axis is the expected standard deviation of inlier noise (giving the error threshold according to Equation 6.1) and the resulting RMS error on the ground truth points is on the y -axis. The vertical cyan line marks the default setting. The compared cost functions are top-hat, truncated quadratic (dotted) and top-hat with quadratic one used as a tie-breaker in a case of two models with equal numbers of inliers. As can be seen, in the vicinity of optimal error threshold (computed from error scale σ) both cost functions output similar results (MSAC is slightly more accurate). For larger thresholds this difference becomes more significant – the truncated quadratic cost function is more robust to the selection of the error scale. It provides a range of one order of magnitude of usable thresholds for most of the image pairs, making it easier to select a suitable one for a diverse set of data.

7. Conclusion

7.1. Summary

In this thesis, Locally Optimized RANSAC is revisited. It has been extensively tested, using the developed mass testing framework and datasets with newly created GT points. Several unexpected aspects have been discovered. Besides numerous implementation issues, the following fundamental two were addressed. Firstly, the behavior of RANSAC with different cost functions under the changing error threshold was examined. Secondly, it was observed that (contrarily to the theoretical predictions) under some circumstances the execution time of LO-RANSAC significantly exceeds the execution time of standard RANSAC.

To robustify the RANSAC to the error scale selection, we proposed to use the truncated quadratic cost function. This offers similar estimation performance near the optimal error threshold, but it keeps it for a substantially wider range, while the performance of the top-hat scoring drops quickly. To make the run-time performance of the Local Optimization less sensitive to the number of inliers (and subsequently the number of equations to be solved), we introduce a limit on this number. When the inlier count exceeds the limit, a limit-sized subset is randomly drawn and used. This procedure lowers the execution time of LO-RANSAC by up to 30% with negligible effect on the accuracy of the estimation. Furthermore we have suggested a different speed-up by remembering *already-seen* sets of inliers and then stopping the iteration on these. In our paper on the LO-RANSAC [18], we combined the improvements into a procedure dubbed LO⁺-RANSAC, using MSAC-like truncated quadratic cost function and the inlier limit. This gives a very stable robust estimation, despite its randomized nature. We have also shown, that the Locally Optimized RANSAC offers a significantly better starting point for a Bundle Adjustment than the Gold Standard.

For fast applications, where no challenging camera motion or illumination changes are expected, a lightweight version of Local Optimization (LO') is proposed. Instead of estimating models from non-minimal samples followed by iterative least squares, only a single iterative least squares are applied on each so-far-the-best model (the limit on the number of inliers used in each step of the least squares is applied). The experimental evaluation shows significant reduction in the execution time while sacrificing only little accuracy, especially for "easy" image pairs.

7.2. Outputs of the Work

The main contributions of this thesis were published at the British Machine Vision Conference 2012 in the paper *Fixing the Locally Optimized RANSAC* [18]. The speed-related issues are explained there as well as the question of the cost functions, and solutions (the LO⁺ and LO') are suggested. Additionally, a supplementary technical report with full experimental evaluation has been published as a CMP Research Report [19].

The implementation of LO-RANSAC in the CMP WBS-Demo was extensively tuned and all the improvements were incorporated. The LO-RANSAC for estimation of ho-

7. Conclusion

mography and epipolar geometry was separated and made publicly available under the GNU GPL license. The source codes can be found on the enclosed CD. Also, these are accessible from the home page at <http://cmp.felk.cvut.cz/software/LO-RANSAC/>.

The datasets, used for the experimental evaluation are made available as well. They are divided on account of the particular geometry by which the image pairs are related – there are datasets for homography estimation and datasets for epipolar geometry estimation. About ten ground truth point correspondences were manually annotated in each of the image pairs, which can be used as an external measurement of the estimation accuracy. The datasets (including GT) are available on the enclosed CD and at the web page <http://cmp.felk.cvut.cz/data/geometry2view/>.

Bibliography

- [1] LAPACK: Linear algebra package.
<http://www.netlib.org/lapack/>. 19, 51
- [2] D. A. Atkinson. CCMATH: Mathematics software library.
<http://freecode.com/projects/ccmath>. 19, 51
- [3] D. Capel. An effective bail-out test for RANSAC consensus scoring. In *Proceedings of the British Machine Vision Conference*, 2005. 12
- [4] J. Cech, J. Matas, and M. Perdoch. Efficient sequential correspondence selection by cosegmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1568–1581, 2009. 41, 44
- [5] S. Choi, T. Kim, and W. Yu. Performance evaluation of RANSAC family. In *Proceedings of the British Machine Vision Conference*, 2009. 4, 31
- [6] O. Chum. *Two-View Geometry Estimation by Random Sample and Consensus*. PhD thesis, Czech Technical University in Prague, 2005. 8, 9, 12, 13, 15, 16, 18
- [7] O. Chum and J. Matas. Randomized RANSAC with T(d,d) test. In *Proceedings of the British Machine Vision Conference*, pages 448–457, 2002. 12
- [8] O. Chum and J. Matas. Matching with PROSAC – progressive sample consensus. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 220–226, 2005. 11, 41, 44
- [9] O. Chum and J. Matas. Optimal randomized RANSAC. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(8):1472–1482, 2008. 12
- [10] O. Chum, J. Matas, and J. Kittler. Locally optimized RANSAC. In *Proceedings of DAGM-Symposium*, pages 236–243, 2003. 9, 15, 16, 18, 21, 24, 32
- [11] O. Chum, T. Werner, and J. Matas. Two-view geometry estimation unaffected by a dominant plane. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 772–779, 2005. 13, 41, 44
- [12] M. A. Fischler and R. C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981. 4, 7, 8, 32
- [13] J.-M. Frahm and M. Pollefeys. RANSAC for (Quasi-)degenerate data (QDEGSAC). In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 453–460, 2006. 13
- [14] F. R. Hampel. A general qualitative definition of robustness. *Annals of Mathematical Statistics*, 42(6):1887–1896, 1971. 7
- [15] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2004. 16

- [16] P. V. C. Hough. Method and means for recognizing complex patterns. U.S. Patent 3069654, 1962. 4
- [17] K. Lebeda, J. Matas, and R. Bowden. Tracking the untrackable: How to track when your object is featureless. In *Proceedings of the ACCV Workshop on Detection and Tracking in Challenging Environments 2012*, volume 7729 of *LNCS*, pages 343–355. Springer, 2013. 4, 5
- [18] K. Lebeda, J. Matas, and O. Chum. Fixing the locally optimized RANSAC. In *Proceedings of the British Machine Vision Conference*, pages 95.1–95.11, 2012. 5, 24, 26, 31, 35
- [19] K. Lebeda, J. Matas, and O. Chum. Fixing the locally optimized RANSAC – Full experimental evaluation. Research Report CTU–CMP–2012–17, Center for Machine Perception, Czech Technical University, Prague, Czech Republic, 2012. http://cmp.felk.cvut.cz/software/LO-RANSAC/Lebeda-2012-Fixing_LORANSAC-tr.pdf. 16, 35, 41
- [20] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004. 41
- [21] D. Martinec and T. Pajdla. 3D reconstruction by fitting low-rank matrices with missing data. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 198–205, 2005. 41, 44
- [22] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide baseline stereo from maximally stable extremal regions. In *Proceedings of the British Machine Vision Conference*, pages 384–396, 2002. 41, 44
- [23] K. Mikolajczyk and C. Schmid. Scale and affine invariant interest point detectors. *International Journal of Computer Vision*, 60(1):63–86, 2004. 41
- [24] S. Mittal, S. Anand, and P. Meer. Generalized projection based M-estimator: Theory and applications. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 2689–2696, 2011. 30
- [25] J.-M. Morel and G. Yu. ASIFT: A new framework for fully affine invariant image comparison. *SIAM Journal on Imaging Sciences*, 2(2):438–469, 2009. 41, 44
- [26] D. R. Myatt, P. H. S. Torr, S. J. Nasuto, J. M. Bishop, and R. Craddock. NAPSAC: High noise, high dimensional robust estimation – it’s in the bag. In *Proceedings of the British Machine Vision Conference*, 2002. 11
- [27] D. Nistér. Preemptive RANSAC for live structure and motion estimation. In *Proceedings of the International Conference on Computer Vision*, pages 199–206, 2003. 12
- [28] N. Noury, F. Sur, and M.-O. Berger. Fundamental matrix estimation without prior match. In *International Conference on Image Processing*, pages I513–I516, 2007. 30
- [29] M. Perdoch, J. Matas, and O. Chum. Epipolar geometry from two correspondences. In *Proceedings of the International Conference on Pattern Recognition*, pages 215–220, 2006. 41, 44









- [30] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, 2007. 45
- [31] M. Pollefeys, R. Koch, M. Vergauwen, and L. Van Gool. Automated reconstruction of 3D scenes from sequences of images. *ISPRS Journal Of Photogrammetry And Remote Sensing*, 55(4):251–267, 2000. 41, 44
- [32] R. Raguram, J.-M. Frahm, and M. Pollefeys. Exploiting uncertainty in random sample consensus. In *Proceedings of the International Conference on Computer Vision*, pages 2074–2081, 2009. 30, 32
- [33] P. J. Rousseeuw. Least median of squares regression. *Journal of the American Statistical Association*, 79:871–880, 1984. 4
- [34] B. Tordoff and D. W. Murray. Guided sampling and consensus for motion estimation. In *Proceedings of the European Conference on Computer Vision*, pages 82–98, 2002. 9, 11, 32
- [35] P. H. S. Torr. *Motion Segmentation and Outlier Detection*. PhD thesis, University of Oxford, 1995.
- [36] P. H. S. Torr. Bayesian model estimation and selection for epipolar geometry and generic manifold fitting. *International Journal of Computer Vision*, 50(1):35–61, 2002. 11
- [37] P. H. S. Torr and A. Zisserman. Robust computation and parametrization of multiple view relations. In *Proceedings of the International Conference on Computer Vision*, pages 727–732, 1998. 11, 31
- [38] P. H. S. Torr and A. Zisserman. MLESAC: A new robust estimator with application to estimating image geometry. *Computer Vision and Image Understanding*, 78:138–156, 2000. 11, 31, 32
- [39] P. H. S. Torr, A. Zisserman, and S. J. Maybank. Robust detection of degenerate configurations for the fundamental matrix. In *Proceedings of the International Conference on Computer Vision*, pages 1037–1042, 1995. 12
- [40] T. Tuytelaars and L. Van Gool. Wide baseline stereo matching based on local, affinely invariant regions. In *Proceedings of the British Machine Vision Conference*, pages 412–422, 2000. 41, 44
- [41] L. Xu, E. Oja, and P. Kultanen. A new curve detection method: Randomized Hough transform (RHT). *Pattern Recognition Letters*, 11(5):331–338, 1990. 5
- [42] G. Yang, C. V. Stewart, M. Sofka, and C.-L. Tsai. Registration of challenging image pairs: Initialization, estimation, and decision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(11):1973–1989, 2007. 41, 44

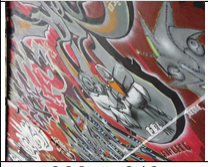







A. Datasets

The performance of different RANSAC variants was evaluated on a collection of 16 image pairs for epipolar geometry and 16 pairs for homography estimation. These image pairs were previously used for evaluation in a number of publications [4, 8, 11, 21, 22, 25, 29, 31, 40, 42]). The datasets are available (including manually annotated ground truth points) at <http://cmp.felk.cvut.cz/data/geometry2view/>.

















Tentative correspondences of image pairs were obtained by matching SIFT descriptors [20] of MSERs [22]. In the technical report [19], experiments using Hessian Affine detector [23] are presented as well. The results on Hessian Affine features are even more favorable for the LO methods because of lower inlier ratios.

A.1. Test Images for Epipolar Geometry Estimation









Filenames:	booksh[AB].png	box[AB].png	castle[AB].png	corr[AB].png
Image A:				
Dimensions:	768 × 576	1024 × 768	768 × 576	512 × 512
Image B:				
Dimensions:	768 × 576	1024 × 768	768 × 576	512 × 512
Err. scale θ :	0.6	0.8	0.6	0.4
TC:	41	231	154	93
Source:	(i)	(b)	(h)	(c)

Filenames:	graff[AB].png	head[AB].jpg	kampa[AB].png	Kyoto[AB].jpg
Image A:				
Dimensions:	800 × 640	1408 × 1056	800 × 543	2592 × 1944
Image B:				
Dimensions:	800 × 640	1408 × 1056	800 × 543	2592 × 1944
Err. scale θ :	0.6	1.1	0.6	2.0
TC:	120	86	84	445
Source:	(c)	(a)	(i)	(a)

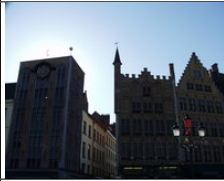







A. Datasets








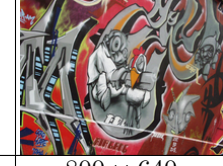
Filenames:	leafs[AB].jpg	plant[AB].png	rotunda[AB].png	shout[AB].png
Image A:				
Dimensions:	1600 × 1200	576 × 768	1024 × 683	768 × 576
Image B:				
Dimensions:	1600 × 1200	576 × 768	1024 × 683	768 × 576
Err. scale θ:	1.2	0.6	0.8	0.6
TC:	79	30	86	54
Source:	(l)	(g)	(j)	(g)
Filenames:	valbonne[AB].png	wall[AB].jpg	wash[AB].png	zoom[AB].png
Image A:				
Dimensions:	768 × 512	2272 × 1704	768 × 576	1024 × 768
Image B:				
Dimensions:	768 × 512	2272 × 1704	768 × 576	1024 × 768
Err. scale θ:	0.6	1.7	0.6	0.8
TC:	32	98	55	70
Source:	(c)	(l)	(g)	(k)




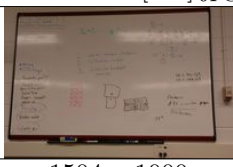



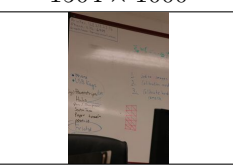
A.2. Test Images for Homography Estimation

Filenames:	adam[AB].png	boat[AB].png	Boston[AB].jpg	BostonLib[AB].png
Image A:				
Dimensions:	600 × 450	850 × 680	1712 × 1368	1504 × 1000
Image B:				
Dimensions:	600 × 450	850 × 680	1712 × 1368	1504 × 1000
Err. scale θ:	0.6	0.8	1.6	1.4
TC:	20	126	382	200
Source:	(f)	(c)	(d)	(d)

A.2. Test Images for Homography Estimation

Filenames:	BruggeSquare [AB].jpg	BruggeTower[AB].png	Brussels[AB].jpg	CapitalRegion [AB].jpg
Image A:				
Dimensions:	1712 × 1368	856 × 684	1712 × 1368	1368 × 1712
Image B:				
Dimensions:	1712 × 1368	856 × 684	1712 × 1368	1368 × 1712
Err. scale θ :	1.6	0.8	1.6	1.6
TC:	46	77	503	130
Source:	(d)	(d)	(d)	(d)

Filenames:	city[AB].png	Eiffel[AB].png	ExtremeZoom[AB].png	graf[AB].png
Image A:				
Dimensions:	329 × 278	1198 × 958	1519 × 1006	800 × 640
Image B:				
Dimensions:	329 × 278	1198 × 958	1519 × 1006	800 × 640
Err. scale θ :	0.3	1.1	1.5	0.8
TC:	17	200	56	245
Source:	(e)	(d)	(d)	(c)

Filenames:	LePoint1[AB].png	LePoint2[AB].png	LePoint3[AB].png	WhiteBoard[AB].jpg
Image A:				
Dimensions:	600 × 450	600 × 450	600 × 450	1504 × 1000
Image B:				
Dimensions:	600 × 450	600 × 450	600 × 450	1000 × 1504
Err. scale θ :	0.6	0.6	0.6	1.4
TC:	148	89	48	214
Source:	(f)	(f)	(f)	(d)

A.3. Sources

(a)	[4]	Cech et al.: Sequential Correspondence Verification, http://cmp.felk.cvut.cz/~cechj/SCV/
(b)	[11]	Chum et al.: DegenSAC (Two-view Geometry Estimation Unaffected by a Dominant Plane)
(c)	various	Visual Geometry Group (University of Oxford), http://www.robots.ox.ac.uk/~vgg/data.html
(d)	[42]	Stewart et al.: Testsuite of 22 challenging pairs of images, http://www.vision.cs.rpi.edu/gdbicp/dataset/
(e)	–	Centre for Remote Imaging, Sensing and Processing, http://www.crisp.nus.edu.sg/~research/tutorial/opt_int.htm .
(f)	[25]	Morel and Yu: ASIFT, http://www.cmap.polytechnique.fr/~yu/research/ASIFT/demo.html
(g)	[40]	Tuytelaars, http://homes.esat.kuleuven.be/~tuytelaa/
(h)	[31]	Pollefeys, Leuven castle image sequence, http://www.cs.unc.edu/~marc/
(i)	[22]	Matas: Robust Wide Baseline Stereo from Maximally Stable Extremal Regions
(j)	[21]	Martinec: St. George rotunda, http://cmp.felk.cvut.cz/projects/is3d/
(k)	[29]	Perdoch: Epipolar Geometry from Two Correspondences, http://cmp.felk.cvut.cz/~perdom1/
(l)	[8]	Chum and Matas: Matching with PROSAC – progressive sample consensus

B. RANSAC Testing Framework

LO-RANSAC has a number of variations and parameters. Thus it is not sufficient to say that it works correctly; furthermore we need to compare different variants against each other. As it is from definition a randomized algorithm, such a comparison should be carried out on extensive datasets with results averaged over a number of iterations.

This condition is necessary, but not sufficient to achieve the best repeatability possible. Unless we are willing to perform a huge numbers of runs, we should assure that all the samples drawn by different variants of RANSAC are the same. The intuitive solution to this requirement is to fix the random seed before start of every RANSAC run. Moreover, since there can be further usage of the pseudorandom generator during RANSAC (e.g. inner sampling in LO), its state should be stored after and restored before each draw of a minimal sample, to fix the samples disregarding any further processing.

B.1. What Is Measured

In standard RANSAC, the number of data points consistent with the estimated model (inliers) is optimized. Such a measure is a good indicator how well the image pair is matching [30]. Thus the number of inliers (both absolute and relative, for a fast insight) is the first information reported.

For some applications such as 3D reconstruction, the precision of the estimated model is of high importance. To compare the precision, we record the RMS error of the estimated model on manually annotated ground truth correspondences (available at the [page with datasets](#)). These correspondences are *not included* in the estimation process. Note that such a measure includes the error in the manual selection as well as error induced by deviation of the imaging process from a pin-hole camera model (such as radial distortion). Statistically, smaller error indicates better precision, however, small differences in small values are insignificant.

The time complexity is indicated by the number of samples drawn by RANSAC. However, Local Optimization and other techniques beyond sample-and-consensus can constitute a significant portion of the total running time. Thus it is appropriate to measure the wall clock time as well, according to Chapter 5. Naturally, all the experiments have to be carried out on the same computer, whose parameters should be reported along. The time measurement is the only one, which is not absolutely repeatable. Its variation decreases with the number of repetitions, but it never falls to zero. To make the variation as low as possible, the running time is measured only once over all the runs. Thus no statistics such a minimum, maximum or standard deviation are available. For a better insight into the running time, the number of performed Local Optimizations is tabulated as well.

As a fast visual indicator of algorithm stability, graphs are included in the tables (in the case of PDF reports, see below). These graphs show the observed probability of every point of being an inlier in repeated runs and a histogram of such probability. Figures B.1 and B.2 show the details. Since the graphs are supposed to provide a quick and intuitive overview of the stability of the results over repeated executions, there is

B. RANSAC Testing Framework

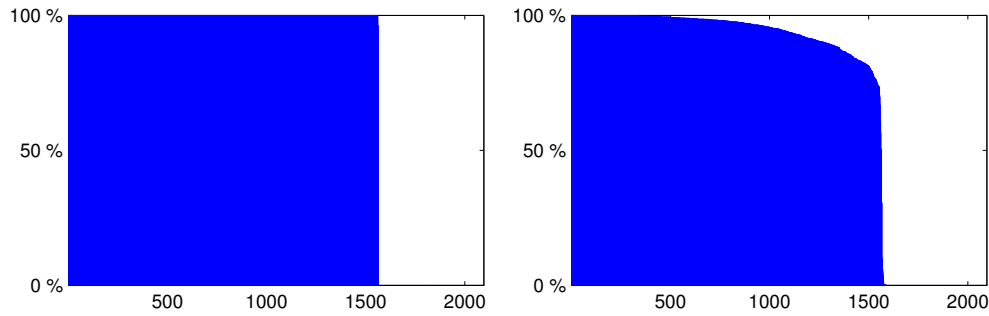


Figure B.1. The probability of a TC being an inlier, collected over 1000 executions. Correspondences on the horizontal axis are ordered so that the values on the vertical axis – the fraction of executions the correspondence has been labeled as an inlier – are non-increasing. The left plot shows an ideal case when the inlier and outlier dichotomy is identical for all executions. The plot on the right side depicts an example of less stable estimation, some tentative correspondences alternate between being an inlier and outlier in different executions.

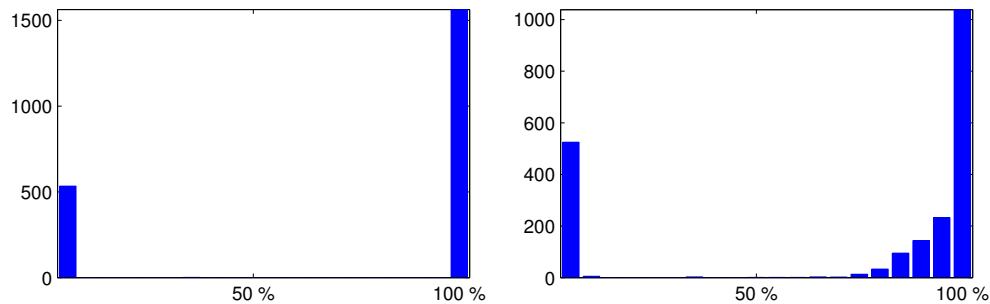


Figure B.2. The histogram of the probability of a TC being an inlier, collected over 1000 executions. Vertical axis: the number of tentative correspondences that have been output as inliers in a fraction of executions (horizontal axis). This plot summarizes the plot in Figure B.1. The left plot shows an ideal case: two bars, one stands for outliers labeled as outliers in all executions, the other stands for inliers consistently labeled in all executions. Plot on right-hand side demonstrates results of a less stable estimation process: for example about 40 tentative correspondences were output as inliers in 80% of executions.

no scale or axis labels in the tables with experimental results. Rather than the exact values in the plots, the shape is important.

B.2. Test Course

Each experiment consists of a number of RANSAC runs on multiple images with the same settings. The testing framework goes through all the image pairs in a dataset. For each pair, features are extracted and matched. Then RANSAC is run multiple times with the same settings. Since we want to start with the same state of pseudorandom generator over different experiments, but with the different states in runs of one experiment, we seed the generator by an index of the run (increasing series of natural numbers). With this manner of seeding, we obtain *exactly the same results* in two independent experiments, started with the same settings. Of course, except the wall clock time¹, as mentioned above. During the test course, necessary statistics are collected.

¹To keep this as stable and repeatable as possible, the experiments were carried out on a dual-core machine with no other resources-demanding process running. Thus RANSAC had always one core solely for its use.

There are different ways the results can be output (a user can select one or more). The basic one is to print the statistics into Command Window of Matlab directly at the end of every experiment (see Figure B.4 for an example). This is useful e.g. when it is desired to just quickly see the results on new data, or to check an effect of a minor change without storing the results permanently.

The next possibilities are to print the results into a file, containing a group of columns in either CSV format, or with marks for \LaTeX tabular environment. One or multiple of these files are then loaded by a Bash script and merged together. In the case of the CSV output, headers are added and the final CSV file is stored. The CSV file may become useful as it is essentially plain text and thus readable for humans, but it can be also passed to any other software as easily processed but still compact format. E.g. all the major table/spreadsheet processors (MS Excel, OpenOffice.org Calc, Gnumeric, KSpread, ...) can read it and allow customizable views, such as sorting, filtering, displaying/hiding/rearranging particular columns or rows, etc., any high-level processing or a creation of graphs. See Figure B.5 for an example of the raw CSV file and its view in Calc spreadsheet processor.

The files containing exported columns for \LaTeX are processed in a similar way. Moreover, the script adds row headers and footers, table header and footer and then uses the table to create a report, adding image thumbnails and additional information. This output is meant for a human inspection, when columns can be easily compared against each other and the small graphs give quick and intuitive overview of algorithm stability. The table with image information is similar to those in Appendix A, for an example of the tabulated results see Figure B.3.

B.3. Output examples

In this section, several examples of the experimental results are shown on a model situation of comparing MSAC and LO-MSAC.

Solver→		M		M.LO			
Detectors→		MSER+ MSER-		MSER+ MSER-			
Descriptors→		SIFT		SIFT			
Image	Qty↓	10000 runs, $\sigma = 0.5$, conf = 95%					
booksh	I	28.8 ± 1.7	(22-33)		29.8 ± 1.2	(23-33)	Inlss
	I (%)	70.2 ± 4.0	(54-80)		72.7 ± 2.8	(56-80)	
	Samp	43.6 ± 18.6	(11-139)		40.5 ± 14.4	(11-125)	
	Time _(ms)	0.6	(NA)		5.2	(NA)	H _{miss}
	Error	3.13 ± 4.41	(0.4-26.2)		1.84 ± 3.10	(0.4-25.8)	
LO count	0.0 ± 0.0	(0-0)		1.0 ± 0.2	(1-4)		
box	I	195.8 ± 6.7	(177-222)		204.3 ± 12.1	(194-223)	Inlss
	I (%)	84.8 ± 2.9	(77-96)		88.5 ± 5.2	(84-97)	
	Samp	8.8 ± 1.7	(2-18)		8.8 ± 1.7	(2-18)	
	Time _(ms)	0.5	(NA)		8.5	(NA)	H _{miss}
	Error	38.67 ± 26.49	(0.7-111.6)		43.70 ± 32.69	(1.2-72.2)	
LO count	0.0 ± 0.0	(0-0)		1.0 ± 0.0	(1-1)		
castle	I	110.0 ± 7.3	(90-124)		122.8 ± 2.0	(105-124)	Inlss
	I (%)	71.4 ± 4.8	(58-81)		79.7 ± 1.3	(68-81)	
	Samp	41.2 ± 17.6	(12-130)		37.6 ± 12.3	(12-130)	
	Time _(ms)	1.0	(NA)		7.5	(NA)	H _{miss}
	Error	4.41 ± 6.82	(0.3-59.9)		0.84 ± 1.68	(0.4-16.4)	
LO count	0.0 ± 0.0	(0-0)		1.0 ± 0.0	(1-1)		

Figure B.3. Example of \LaTeX output. Each column represents one experiment. Image thumbnails are included in the row headers. Column headers contain the used settings. The tabulated results are arranged as follows: **mean value** \pm *standard deviation (minimum-maximum)*. After these, the stability of each experiment is visualized in the graphs, see Figures B.1 and B.2 for details.

B. RANSAC Testing Framework

```

M has finished its 1000 runs on files "bookshX.png" with statistics (min/avg/max/+std):
Inliers found: 24/ 28.9/ 33/+ 1.6
Inliers found: 58.5%/ 70.4%/ 80.5%/+ 3.9%
Samples drawn: 13/ 42.7/ 131/+ 18.4
Time elapsed: 0.7
Ei: 0.4/ 3.07/ 24.8/+ 4.32
L0 count: 0/ 0.0/ 0/+ 0.0
M has finished its 1000 runs on files "boxX.png" with statistics (min/avg/max/+std):
Inliers found: 180/ 196.1/ 222/+ 6.9
Inliers found: 77.9%/ 84.9%/ 96.1%/+ 3.0%
Samples drawn: 3/ 8.8/ 16/+ 1.7
Time elapsed: 0.4
Ei: 0.9/ 38.37/ 107.6/+ 26.71
L0 count: 0/ 0.0/ 0/+ 0.0
M has finished its 1000 runs on files "castleX.png" with statistics (min/avg/max/+std):
Inliers found: 90/ 110.3/ 124/+ 7.4
Inliers found: 58.4%/ 71.6%/ 80.5%/+ 4.8%
Samples drawn: 13/ 41.2/ 130/+ 18.0
Time elapsed: 1.0
Ei: 0.4/ 4.49/ 59.4/+ 7.65
L0 count: 0/ 0.0/ 0/+ 0.0
.
.
.

M.L0 has finished its 1000 runs on files "bookshX.png" with statistics (min/avg/max/+std):
Inliers found: 26/ 30.0/ 33/+ 0.8
Inliers found: 63.4%/ 73.2%/ 80.5%/+ 2.0%
Samples drawn: 11/ 25.9/ 75/+ 7.6
Time elapsed: 12.8
Ei: 0.5/ 1.37/ 22.4/+ 1.67
L0 count: 1/ 3.3/ 9/+ 1.3
M.L0 has finished its 1000 runs on files "boxX.png" with statistics (min/avg/max/+std):
Inliers found: 195/ 213.0/ 223/+ 11.9
Inliers found: 84.4%/ 92.2%/ 96.5%/+ 5.2%
Samples drawn: 2/ 5.1/ 9/+ 2.8
Time elapsed: 16.9
Ei: 1.5/ 22.00/ 71.9/+ 30.27
L0 count: 1/ 2.0/ 5/+ 0.8
M.L0 has finished its 1000 runs on files "castleX.png" with statistics (min/avg/max/+std):
Inliers found: 120/ 123.0/ 124/+ 0.8
Inliers found: 77.9%/ 79.9%/ 80.5%/+ 0.5%
Samples drawn: 12/ 12.8/ 16/+ 0.8
Time elapsed: 19.7
Ei: 0.4/ 0.65/ 1.2/+ 0.07
L0 count: 1/ 3.0/ 7/+ 1.2
.
.
.

```

Figure B.4. Example of text output in the Matlab Command Window. As the experiments are not piled close to each other, this manner is not suitable for a comparison of methods.

```
"FileNames","Qty","M MEAN","M MIN","M MAX","M STD","M.LO MEAN","M.LO MIN","M.LO MAX","M.LO STD"
"bookshA.png","I",28.9,24,33,1.6,30.0,26,33,0.8
"bookshB.png","I (%)",70.4,58.5,80.5,3.9,73.2,63.4,80.5,2.0
,"Samp",42.7,13,131,18.4,25.9,11,75,7.6
,"Time",0.7,0,0,0,12.8,0,0,0
,"Error",3.07,0.4,24.8,4.32,1.37,0.5,22.4,1.67
,"LO Count",0.0,0,0,0.0,3.3,1,9,1.3
"boxA.png","I",196.1,180,222,6.9,213.0,195,223,11.9
"boxB.png","I (%)",84.9,77.9,96.1,3.0,92.2,84.4,96.5,5.2
,"Samp",8.8,3,16,1.7,5.1,2,9,2.8
,"Time",0.4,0,0,0,16.9,0,0,0
,"Error",38.37,0.9,107.6,26.71,22.00,1.5,71.9,30.27
,"LO Count",0.0,0,0,0.0,2.0,1,5,0.8
"castleA.png","I",110.3,90,124,7.4,123.0,120,124,0.8
"castleB.png","I (%)",71.6,58.4,80.5,4.8,79.9,77.9,80.5,0.5
,"Samp",41.2,13,130,18.0,12.8,12,16,0.8
,"Time",1.0,0,0,0,19.7,0,0,0
,"Error",4.49,0.4,59.4,7.65,0.65,0.4,1.2,0.07
,"LO Count",0.0,0,0,0.0,3.0,1,7,1.2
:
:
.
```

	A	B	C	D	E	F	G	H	I	J
1	FileNames	Qty	M MEAN	M MIN	M MAX	M STD	M.LO MEAN	M.LO MIN	M.LO MAX	M.LO STD
2	bookshA.png	I	28.9	24	33	1.6	30.0	26	33	0.8
3	bookshB.png	I (%)	70.4	58.5	80.5	3.9	73.2	63.4	80.5	2.0
4		Samp	42.7	13	131	18.4	25.9	11	75	7.6
5		Time	0.7	0	0	0	12.8	0	0	0
6		Error	3.07	0.4	24.8	4.32	1.37	0.5	22.4	1.67
7		LO Count	0.0	0	0	0.0	3.3	1	9	1.3
8	boxA.png	I	196.1	180	222	6.9	213.0	195	223	11.9
9	boxB.png	I (%)	84.9	77.9	96.1	3.0	92.2	84.4	96.5	5.2
10		Samp	8.8	3	16	1.7	5.1	2	9	2.8
11		Time	0.4	0	0	0	16.9	0	0	0
12		Error	38.37	0.9	107.6	26.71	22.00	1.5	71.9	30.27
13		LO Count	0.0	0	0	0.0	2.0	1	5	0.8
14	castleA.png	I	110.3	90	124	7.4	123.0	120	124	0.8
15	castleB.png	I (%)	71.6	58.4	80.5	4.8	79.9	77.9	80.5	0.5
16		Samp	41.2	13	130	18.0	12.8	12	16	0.8
17		Time	1.0	0	0	0	19.7	0	0	0
18		Error	4.49	0.4	59.4	7.65	0.65	0.4	1.2	0.07
19		LO Count	0.0	0	0	0.0	3.0	1	7	1.2
20	corrA.png	I	73.3	61	82	3.9	80.3	74	82	0.7
21	corrB.png	I (%)	78.8	65.6	88.2	4.2	86.3	79.6	88.2	0.7

Figure B.5. Example of CSV output. The upper part shows a raw CSV file with results of sample experiment “MSAC vs. LO-MSAC”. In the lower part, this file is open in the OpenOffice.org Calc spreadsheet processor, allowing further operations with the data.

C. LO-RANSAC software library

In the scope of this work, LO-RANSAC implementation in CMP WBS-Demo was improved and tuned. The LO-RANSAC source code has been separated and published as a C library. It is now publicly available under the GNU GPL license. The home page can be found at <http://cmp.felk.cvut.cz/software/LO-RANSAC/>.

C.1. Dependencies

The library has only one dependency – LAPACK: Linear Algebra Package [1]. In Linux distributions, this is often available via package management system (e.g. package `liblapack`). We were informed that on Windows it is possible to replace LAPACK by Intel MKL library. Then, besides changing the path in a Makefile (see below), it is necessary to replace LAPACK `#includes` by `<mk1.h>`. This was reported as working with MS Visual Studio 2008 on Windows 7 x64. However, we were not able to check the reliability of this information.

The LO-RANSAC library also uses several procedures from CCMATH library [2]. Nevertheless, since this is included in the distribution archive, it is not necessary to be concerned about those at all.

C.2. Building the library

The unpacked LO-RANSAC directory contains a license file, information about authors with contact information, README file with content similar to this chapter and `src` subdirectory, containing the source files. In the `src` directory, the Makefile is prepared for an easy build. The debug/release build can be switched there by simply (un)commenting lines (turning on/off optimization or debugging information).

As the LAPACK installation usually does not include the header files, it is necessary to obtain those elsewhere. E.g., Matlab carries a set of headers along. The path to the headers should be specified in the Makefile as well.

Running the Makefile compiles all the source files into C objects and then link these together with LAPACK library. A file `libransac.a` is the result. Then it is possible to link this file to other software using `-lransac -L.` E.g. for the use in Matlab, MEX-sources are prepared. These should be compiled as follows:

```
mex loransach.mex.c -o loransach.mexglx -lransac -L. -llapack.
```

This example is for 32-bit Linux. The proper MEX-file extension should be used, according to the particular platform (use the `mexext` command in Matlab).

C.3. Application Programming Interface

The main gateway to the library consists of two functions for geometry estimation: `ransachH` and `ransachF` for homography and epipolar geometry estimation (respectively). Their arguments are summarized in Table C.3. Since the library is often called from Matlab MEX-files, the matrices are stored column-wise (in a column-major order). In these arguments, all the essential RANSAC parameters are possible to set. Also, the

Table C.1. Application Programming Interface. A user can choose between a more detailed parameter settings and the simple variant where as many as possible parameters are set to their respective default values. See table C.3 for a description of the arguments.

Score	ransacH	(double *u, int len, double th, double conf, int max_sam, double *H, unsigned char * inl, int do_lo, int inlLimit);
Score	ransacF	(double *u, int len, double th, double conf, int max_sam, double *F, unsigned char * inl, int do_lo, int inlLimit);
void	ransacHsimple	(double *u, int len, double th, double *H);
void	ransacFsimple	(double *u, int len, double th, double *F);

Table C.2. Matlab MEX-files Application Programming Interface. Again, the only mandatory inputs are the tentatively corresponding pairs of points (here called TC_PAIRS instead of u) and the inlier/outlier error threshold. A matrix, representing the geometry, is always returned; optionally also with the inliers/outliers separation. One additional parameter is present here – the seed for pseudorandom generator (srand in C).

[H]	=	loransacH(TC_PAIRS, THRESHOLD)
[H, INL]	=	loransacH(TC_PAIRS, THRESHOLD [, LO_ON, CONF, INL_LIMIT, MAX_SAM, RAND_SEED])
[F]	=	loransacF(TC_PAIRS, THRESHOLD)
[F, INL]	=	loransacF(TC_PAIRS, THRESHOLD [, LO_ON, CONF, INL_LIMIT, MAX_SAM, RAND_SEED])

ransac* functions return inliers/outliers bipartitioning of the data points by filling the inl array by ones and zeroes. The score of the best geometry hypothesis found is returned in a special structure, containing the number of inliers I and the widened truncated quadratic score J (note that this is implemented as a *gain* function to be maximized, to be consistent with the inlier count). Its definition is:

```
typedef struct { unsigned I; double J; } Score;
```

If it is necessary to change some of the inner LO-RANSAC constants or parameters (e.g. the number of inner samples in LO, or cost function), this can be done in the file rtools.h. On the other hand, for the cases when an “as simple as possible” interface is needed, we have prepared wrapping functions ransacHsimple and ransacFsimple. In these, all the parameters are set to their default values wherever it makes any sense. Furthermore, nothing but the geometry matrix is returned. The default values can be found in the Table C.3 as well.

For an example of a typical LO-RANSAC usage see the files testH.c and testF.c. Furthermore, these indicate a proper function of the LO-RANSAC library. The Makefile compiles them with the library, thus they can be easily run from the system terminal (e.g. ./testH for homography).

The input matrix u is of size $6 \times N$. It consists of the homogeneous coordinates of both tentatively corresponding points – i^{th} column can be expressed as $[x_i, y_i, w_i, x'_i, y'_i, w'_i]^T$ or $[\mathbf{x}_i^T, \mathbf{x}'_i{}^T]^T$. On the output, a model of the desired geometry is returned, satisfying

$$\text{rank } \mathbf{H} = 3 \tag{C.1}$$

and

$$\mathbf{x}_i \times \mathbf{H}\mathbf{x}'_i = 0 \quad \forall i: \text{inl}(i) = 1 \tag{C.2}$$

Table C.3. The list of LO-RANSAC arguments. This table applies to C interface, all the properties nevertheless hold even for the MEX interface. For the interface of the MEX-files please refer to the Table C.2.

Name	Direction	Type	Size	Default value (F/H)
F/H	output	double	3×3	NA
	fundamental matrix / homography matrix (M^*)			
inl	output	binary	$1 \times \text{len}$	NULL (not returned)
	inliers/outliers separation (\mathcal{I}^*)			
u	input	double	$6 \times \text{len}$	NA
	input data – tentative correspondences			
len	input	int	scalar	NA
	length of input tentative correspondences (N)			
th	input	double	scalar	NA
	squared inlier-outlier error threshold (θ^2 , px ²)			
conf	input	double	scalar	0.95
	user-required probability of obtaining the best solution (η_0)			
max_sam	input	int	scalar	1,000,000
	maximal number of samples drawn			
do_lo	input	binary	scalar	1
	turning LO on/off			
inlLimit	input	int	scalar	49/28
	maximal number of inliers for iterative least squares, 0 = no limit			

for the homography estimation, and

$$\text{rank } \mathbf{F} = 2 \quad (\text{C.3})$$

and

$$\mathbf{x}_i^\top \mathbf{F} \mathbf{x}'_i = 0 \quad \forall i : \text{inl}(i) = 1 \quad (\text{C.4})$$

for the epipolar geometry estimation, with the tolerance θ of the correspondence Sampson's error ($e_i^2 \leq \theta^2$). Please note that the notation used in the CMP WBS-Demo is used here, instead of the more common $\mathbf{x}'_i \times \mathbf{H} \mathbf{x}_i = 0$ and $\mathbf{x}'_i^\top \mathbf{F} \mathbf{x}_i = 0$.

D. Abbreviations, Symbols, Constants

2D, 3D, 4D Two-, Three-, Four-Dimensional
BA Bundle Adjustment
Bash Bourne again shell
CMP Center for Machine Perception (Centrum Strojového vnímání ČVUT)
CovRANSAC Covariance RANdom SAMple Consensus (using uncertainty estimation)
CSV Comma Separated Values
DegenSAC Degenerate SAMple Consensus
DoF Degree of Freedom
EG Epipolar Geometry
GNU GNU's Not Unix
GPL General Public License
GT Ground Truth
HG HomoGraphy
i.i.d. Independent and Identically Distributed
ILSq Iterative Least Squares
InRS Inner Random Sample
LAF Local Affine Frame
LAPACK Linear Algebra PACKage
LMS Least Median of Squares
LO Local Optimization
LO-RANSAC Locally Optimized RANdom SAMple Consensus
LSq Least Squares solution, normalized and possibly weighted
MAPSAC Maximum A Posteriori Estimation SAMple Consensus
MEX Matlab EXecutable
MLESAC Maximum Likelihood Estimation SAMple Consensus
MSAC M-estimator SAMple Consensus
MSER Maximally Stable Extremal Region
NAPSAC N Adjacent Points SAMple Consensus
NBS Narrow Baseline Stereo
PLUNDER Pick Least UNDEgenerate Randomly
PROSAC PROgressive SAMple Consensus
QDEGSAC Quasi-DEGenerate SAMple Consensus
RANSAC RANdom SAMple Consensus
RS Random Sample
SFM Structure From Motion
SIFT Scale-Invariant Feature Transform
SPRT Sequential Probability Ratio Test
TC Tentative Correspondence(s)
WBS Wide Baseline Stereo

C cost function.

d number of DoF of model error.

e correspondence error.

D. Abbreviations, Symbols, Constants

find_inliers find points with error smaller than θ w.r.t. model M .

I number of inliers.

\mathcal{I} (\mathcal{I}^* , \mathcal{I}_s^*) inlier set (the largest found, the largest from minimal sample).

K number of samples drawn by RANSAC.

m size of minimal sample (e.g. 2 for fitting 2D line to points, 4 for homography from point correspondences, etc.)

M (M^* , M_s^*) model (the best found, the best from minimal sample).

N number of data points.

$P(I)$ probability of a random sample being outlier-free.

\mathcal{S} random sample.

ε inlier ratio, i.e. I/N .

η probability of drawing at least one outlier-free sample.

η_0 user-supplied probability η to terminate, confidence in returned solution.

θ inlier/outlier error threshold.

ρ error term of the minimized cost function (some robust function of the correspondence error).

σ standard deviation of error of inliers.

iters iterations of LSq (4).

iters' iterations of LSq – LO' (10 for EG, 4 for HG).

K_{start} number of samples, when LO is suppressed (50).

m_θ threshold multiplier (2).

m'_θ threshold multiplier – LO' ($4 \cdot m_\theta$ for EG, m_θ for HG).

reps inner sampling repetitions (10).

s_{is} size of inner sample ($\min(14, |\mathcal{I}_{\text{base}}|/2)$ for EG, $\min(12, |\mathcal{I}_{\text{base}}|/2)$ for HG).

Δ_θ threshold decrement ($\frac{m_\theta \cdot \theta - \theta}{\text{iters} - 1}$).

E. Content of Enclosed CD



CD root



README.txt – Content info, similar to this section.



index.html – Homepage of CD.



resources – Data of HTML pages on CD.



text – Text of the thesis (this file).



lebedkar.thesis.pdf – Resulting PDF document.



lebedkar.thesis.zip – \LaTeX source codes.



loransac_dist – Distribution package of LO-RANSAC library.



src – Source codes.



manual.pdf – Manual for building and usage.



LICENSE, AUTHORS – Authors and legal information.



README – Basic information about the package.



BMVC2012paper – The published paper and related materials.



Lebeda-2012-Fixing_LORANSAC-BMVC.pdf – The paper.



...LORANSAC-BMVC_abstract.pdf – Extended abstract.



...LORANSAC-BMVC_poster.pdf – Conference poster.



...LORANSAC-tr.pdf – Tech. report with full exp. evaluation.



data – Datasets used for experimental evaluation.



homogr.tar.gz – Homography dataset.



kusvod2.tar.gz – Epipolar geometry dataset.

The source of the icons used on this page is Gnome.org.