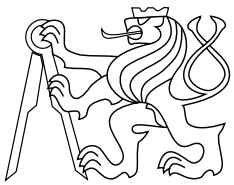
CENTER FOR
MACHINE PERCEPTION



CZECH TECHNICAL
UNIVERSITY IN PRAGUE

MASTER'S THESIS

Robust Two-View Geometry Estimation from Region Correspondences

Radek Matějka

radek.matejka@gmail.com

May 12, 2013

Thesis Advisor: prof. Jiří Matas

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 Two-View Geometry Estimation from Region Correspondences

Radek Matějka

May 12, 2013

There are people who guided me throughout this work and I want to thank them. Firstly, it is the supervisor Jiří Matas, who was kindly sharing his experience from the field and had a patience. I would like to thank Ondřej Chum who always took a time and explained me. I also admire Karel Lebeda, who however abroad and busy answered my e-mails.

Czech Technical University in Prague
Faculty of Electrical Engineering

Department of Cybernetics

DIPLOMA THESIS ASSIGNMENT

Student: Bc. Radek M a t ě j k a

Study programme: Open Informatics

Specialisation: Computer Vision and Image Processing

Title of Diploma Thesis: Robust Two-View Geometry Estimation from Region Correspondences

Guidelines:

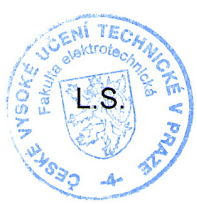
1. Familiarize yourself with the commonly used variants of the RANSAC method.
2. Study methods of two-view geometry estimation from region correspondences.
3. Suggest application of region correspondences in LO-RANSAC estimation.
4. Implement and validate the improved methods.

Bibliography/Sources:

[1] R. Hartley, A. Zisserman: Multiple View Geometry in Computer Vision. Cambridge University Press, 2004.
[2] O. Chum: Two-View Geometry Estimation by Random Sample and Consensus. PhD thesis, Czech Technical University in Prague, 2005.
[3] P.H.S. Torr: Outlier Detection and Motion Segmentation. PhD thesis, Dept. of Engineering Science, University of Oxford, 1995.
[4] K. Lebeda: Robust Sampling Consensus. MSc thesis, Czech Technical University in Prague, 2013.

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

Valid until: the end of the summer semester of academic year 2013/2014



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

prof. Ing. Pavel Ripka, CSc.
Dean

Prague, January 10, 2013

České vysoké učení technické v Praze
Fakulta elektrotechnická

Katedra kybernetiky

ZADÁNÍ DIPLOMOVÉ PRÁCE

Student: Bc. Radek M a t ě j k a

Studijní program: Otevřená informatika (magisterský)

Obor: Počítačové vidění a digitální obraz

Název tématu: Robustní odhad geometrie dvou obrazů z korespondencí oblastí

Pokyny pro vypracování:

1. Seznamte se s metodou RANSAC a jejími variantami.
2. Nastudujte metody odhadu geometrie z korespondencí oblastí.
3. Navrhněte použití korespondencí oblastí v metodě LO-RANSAC.
4. Vylepšení implementujte a ověřte.

Seznam odborné literatury:

[1] R. Hartley, A. Zisserman: Multiple View Geometry in Computer Vision. Cambridge University Press, 2004.
[2] O. Chum: Two-View Geometry Estimation by Random Sample and Consensus. PhD thesis, Czech Technical University in Prague, 2005.
[3] P.H.S. Torr: Outlier Detection and Motion Segmentation. PhD thesis, Dept. of Engineering Science, University of Oxford, 1995.
[4] K. Lebeda: Robust Sampling Consensus. MSc thesis, Czech Technical University in Prague, 2013.

Vedoucí diplomové práce: prof. Ing. Jiří Matas, Ph.D.

Platnost zadání: do konce letního semestru 2013/2014

L.S.

prof. Ing. Vladimír Mařík, DrSc.
vedoucí katedry

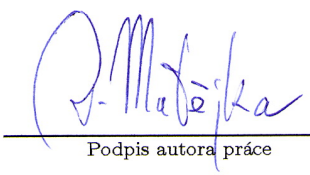
prof. Ing. Pavel Ripka, CSc.
děkan

V Praze dne 10. 1. 2013

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 8.5.2013



Podpis autora práce

Abstract

In this thesis, the algorithm LR73L for fundamental matrix estimation from 3 correspondences of local affine frames, i.e. the 3LAF2F problem, is proposed. The theoretical advantages of the approach are discussed first and the approach is justified. Among other alternatives, LO-RANSAC is chosen as the basis for the proposed algorithm. The LO-RANSAC consists of parts, which were considered with respect to the 3LAF2F problem. The considered parts of LO-RANSAC are: sampling, hypothesis generation and local optimization. The proposed algorithm differs when compared with the preceding algorithm for the 3LAF2F problem from [4]. The most important difference is the use of the 7-point algorithm with the priority for longer arms of LAFs for the F calculation. This method provides higher quality fundamental matrices and lower computing time when compared with the 9-point method used in the preceding algorithm. The suggested algorithm LR73L was experimentally compared with its predecessor and the traditional method for the fundamental matrix estimation from [13]. A speed-up of an order of magnitude was achieved when compared with the predecessor algorithm. The LR73L algorithm reaches quality and computing time of the traditional approach, however only for image pairs with more than 70% inlier ratio. Finally, it is stated that despite the improvements proposed in this work, the traditional approach remains the better option for general use.

Resumé

Předmětem této práce je odhadování fundamentální matice ze 3 korespondencí lokálních afiních rámců, t.j. problém 3LAF2F. Výsledkem je navržený a implementovaný algorithmus LR73L, který daný problém řeší. V textu jsou nejprve diskutovány výhody tohoto přístupu oproti tradičnímu odhadu F z korespondencí bodů. Dále jsou zváženy možnosti návrhu algoritmu, kde je vybrán LO-RANSAC jako základ řešení. LO-RANSAC je rozložen na části, které jsou jednotlivě navrhovány s ohledem na řešený problém. Části LO-RANSACu, které jsou předmětem zkoumání jsou: výběr vzorku, generátor hypotéz a lokální optimalizace. Navržený algoritmus se liší od svého předchůdce, algoritmu řešícího rovněž 3LAF2F problém, ze článku [4]. Významným rozdílem je použití sedmibodového algoritmu k výpočtu fundamentální matice. Ten navíc k výpočtu upřednostňuje od středu vzdálenější body LAFu. Díky tomu je dosaženo kvalitnějších výsledků a v kratším čase než u původního algoritmu, který používal devítibodovou metodu výpočtu fundamentální matice. Navržený algoritmus LR73L byl experimentálně porovnán s jeho předchůdcem a tradiční metodou pro odhad fundamentální matice zastoupenou programem z [13]. Z výsledků vyplývá, že nový algoritmus je o řád rychlejší než jeho předchůdce. Algoritmus LR73L dosahuje kvality a výpočetního času tradičního přístupu, avšak pouze pro obrázky s minimálně 70% podílem inlierů. V závěru je shrnuto, že přes všechna vylepšení navržená v této práci, zůstáva tradiční přístup lepší volbou pro všeobecné použití.

Contents

Abbreviations

3LAF2F	estimating F from 3LAFs
3LAFs	3 correspondences of LAFs
DR	distinguished region
F	fundamental matrix
LAF	local affine frame
LO	local optimization
LO-RANSAC	locally optimized RANSAC
LR73L	LO-RANSAC 7-point from 3LAFs
LSQ	least squares
MSER	maximally stable extremal regions
RANSAC	random sample consensus
SIFT	scale-invariant feature transform
SVD	singular value decomposition
TC	tentative correspondence

1. Introduction and goals

This thesis contributes to the problem of fundamental matrix estimation. The traditional approach is to use point correspondences between a pair of images. In this wok, fundamental martix is estimated from 3 correspondences of local affine frames (LAFs) [16].

The traditional approach is coupled with detectors, which describe image feature as a single point. However, present detectors (e.g. MSER [15]) are able to describe feature as a set of points, i.e. distinguished region. The complete object recognition framework involving distinguished regions was introduced in [18]. In this framework, local affine frames (LAFs) [16] are constructed from distinguished regions to normalize them. Despite this primary purpose of LAFs, it is also possible to use them for fundamental matrix estimation, which is a subject to this thesis.

Such an approach was pioneered by Chum et al. [4][5] in 2003. It was shown that the fundamental matrix can be estimated from 3 correspondences of local affine frames (3LAF2F) and that quality of outcome is comparable with fundamental matrices estimated from point correspondences. This thesis draws on Chum's et al. work and tries to improve F estimation from 3LAFs, especially its computational time.

Since the problem has been opened, a progress is expected to happen in the field. At the time of publication of the Chum's et. al original article, the MSER and LAFs were released new developments few years earlier. Now, more than ten years later, its application in computer vision has spread and become a standard. While MSER detector and LAFs were proving their utility, they were subject to ongoing research in order to be improved. The LAFs have been made more precise[17] and stable[19][18]. This was a motivation to reopen the 3LAF2F problem with expectation of improvement.

The 3LAF2F problem was shown to be a representative member of LO-RANSAC domain [2]. Recently, a paper improving local optimization was published by Lebeda et al. in 2012 [14]. It clarifies properties of using a LO step and it improves the state-of-the-art LO-RANSAC. This gives another promise of success in solving the problem.

1.1. Motivation

Solutions to many problems solved by computer vision require knowledge of correspondences between images. The correspondence computation is omnipresent and can be found at the very beginning of many computer vision algorithms. Thus, it is a very crucial problem.

In general, the correspondences are estimated by using local appearance (e.g. using SIFT) between images. Image features are extracted from images and they are tried to be matched between images. This method is valid for arbitrary images of rigid scenes and any camera setup. The images can capture an object under different conditions (e.g. various backgrounds) or under some transformation. Even then it is possible to make a match. This is a challenging procedure because of the numerous possibilities just indicated. However, when a specific condition applies to a captured scene, the number of possible correspondences decreases. This is where epipolar constraint comes into play.

When capturing a static scene, i.e. different viewpoints are allowed but the scene remains the same, the images are constrained with epipolar constraint. The object captured in the first image is constrained to a specific area in the second image. Thus, the matching problem is made easier in such cases. The tentative correspondences estimated using local appearance, which do not obey epipolar constraint are invalid and filtered out. This advantage has made the use of epipolar constraint integral part of every correspondence solver for static scenes. The application is wide and may involve:

1. stereo matching (figure 1.1)
2. outlier detection
3. rectification
4. image stitching
5. 3D reconstruction
6. stereo object tracking

The epipolar constraint is expressed by fundamental matrix F. The knowledge of F is not self-evident and has to be estimated. The F depends on relative pose of cameras. Therefore, when cameras relative pose changes the need for F re-estimation rises. Otherwise, without the knowledge of F, the epipolar constraint cannot be exploited. Thus, the F estimation is needed and is a frequent task in computer vision. The high quality F can be additionally required in a short time, which will be discussed next.

For a stereo rig the speed is in not the issue. The epipolar constraint is expressed only once, e.g. during calibration. The F is then re-used during the process. In this case the high quality of F is important, moving the computational time to the background. On the other hand, there are applications which emphasize speed of the algorithm. As an example, you can consider n-view reconstruction. The scene is captured from different viewpoints and there is no reference frame. That is, the fundamental matrix has to be re-estimated for more image pairs. In this case, computational time matters.

The arguments just presented imply the need for a precise and fast fundamental matrix estimator. In this thesis, the 3LAF2F approach which may lead to such an F estimator is studied and the LR73L algorithm is proposed.

1.2. Goal and objectives

An additional goal and objectives were specified apart from goals given by the thesis assignment. Those additional objectives and the need of their precise definition rose in the beginning of the work.

The goal of this work is to propose a precise and fast fundamental matrix estimator. The fundamental matrix have to be estimated using 3 correspondences of local affine frames. The objectives are defined as follows.

a) Fast - the proposed algorithm, or more precisely its implementation, should run with minimal time requirements. Wall-clock time is the subject to minimization.

b) Precise - the precision is measured in epipolar error. In the text, precision is also referred to as quality. The quality of estimated F has to be comparable with the state-of-the-art which is [13].

1.3. Thesis structure

The thesis begins with the theoretical background and review of related work. In the theory section, it is described what a local affine frame is and a little about epipolar

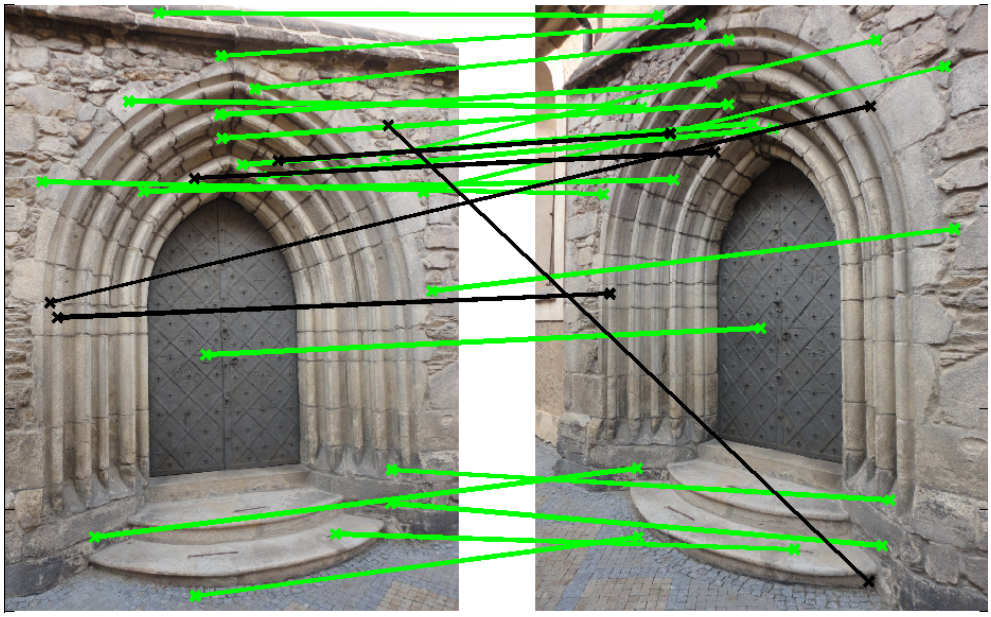


Figure 1.1. Correspondences constrained by fundamental matrix F. Green ones comply with F, black don't.

constraint and fundamental matrix is written. However, this work is not concerned with LAFs. They are mentioned because they are input to the 3LAF2F algorithm. The following chapter is devoted to LO-RANSAC. LO-RANSAC makes up the backbone of the algorithm proposed in this work. Therefore, to make a split between contributions and common knowledge, the LO-RANSAC basis is discussed in the separate chapter. LO-RANSAC is decomposed and each part is discussed. In the subsequent, chapter algorithm components solving the 3LAF2F problem are suggested and experimentally evaluated. In the 5^{th} chapter, the complete algorithm is suggested and benchmarked. Finally, there is a conclusion in the end followed by an appendix (e.g. dataset, complete experimental results).

1.4. Contributions

This thesis contributes to the field with the following work.

1. The new LR73L algorithm to the 3LAF2F problem was proposed.
2. An empirical study of the 3LAF2F problem was done. This brought answers to questions related to the problem. (chapter 4)
3. It was shown that using larger LAFs has benefical impact on quality of resulting Fs.
4. The problem of when to execute LO step was addressed. An empirical curve capturing execution strategy, quality and computational time was constructed.
5. A novel approach to the support measure when estimating F using SVD was suggested, evaluated and prototyped.

2. Overview, related work and theoretical background

This chapter introduces the topic of this thesis in the context of a related work. The related work does not denote just works with the similar goal, but it also includes works are concerned about the input for the problem being solved - local affine frames (LAFs).

The figure 2.1 gives an overview of the problem being solved and puts this work in the context. The figure displays four-stage pipeline used for estimating fundamental matrix F. The pipeline is composed of: detector, descriptor, matcher and F estimator.

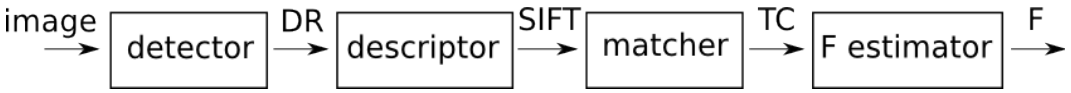


Figure 2.1. F estimating workflow.

Detector - a detector is responsible for extracting feature regions from images. The work presented in this thesis is based on MSER detector, which annotates distinguished regions (DR) in the image. Additionally, LAFs are constructed from DRs. LAFs are of great importance for the 3LAF2F problem, because they are the input. An attention is paid to them in the section 2.5.

Descriptor and matcher - descriptor followed by a matcher returns tentative correspondences (TCs). In our case, the TCs are composed of corresponding LAFs. In this work, there is no special requirement on those components, i.e. descriptor and matcher. Thus no extra attention is paid to them. For completeness, it should be stated that SIFT and the nearest neighbour matching strategy were used.

F estimator - tentative correspondences are used by an estimator to estimate a fundamental matrix. This part of the pipeline is important for this thesis and is the subject of research.

This chapter continues with state of the art F estimators. The attention is payed not only to traditional (i.e. per-point) approach, but also to more unusual ones. Consequently, more theoretical topics are considered. The 3LAF2F problem is theoretically justified in section 2.2. The fundamental matrix is discussed in the section 2.3.

2.1. State of the art approaches

Fundamental matrix estimation is crucial for some computer vision problems. This importance motivate to research and find a new ways how to estimate F. The circumstances of the application might require or allow to use a not generic method. This leads to such an approaches like estimating when low number of inliers, low number of correspondences or assuming affine cameras.

There were more approaches to fundamental matrix estimation. Some of them were proved in practice and became a standard. Some of them have tried to use a slightly different manner.

2.1.1. Estimating F from point correspondences.

It is assumed, that this is the most widely used method for F estimation. It's computation is fast and accessible in ordinary situation. There are two popular numerical methods:

1. 8-point algorithm [10][9],
2. 7-point algorithm [10].

To yield a complete F estimator, those numerical methods are usually embedded to some robust algorithm, e.g. RANSAC. [10]

2.1.2. Estimating affine F from two ellipses.

In the case of affine camera pair, i.e. cameras projections are affine, an epipolar constraint can be expressed with affine fundamental matrix F_A. This matrix has 4 degrees of freedom, which is 3 less than generic fundamental matrix. As a consequence, it can be calculated from less correspondences.

Zisserman and Arandjelovic´ presented a method for estimating affine fundamental matrix from two corresponding ellipses. [22]

2.1.3. Estimating F from 3 correspondences of LAFs

When LAFs became available, the new possibilities have appeared in the field. The LAFs provide additional correspondences between images. Every correspondence of center point is accompanied with two additional point correspondences. Because of this correspondence enrichment, the LAFs were exploited for F estimation.

The problem was opened by Chum et al.[4][5] and is the topic of this thesis. The theoretical advantages of this approach are considered in the section 2.2.

2.2. Theoretical properties of 3LAF2F problem

The traditional approach is to estimate F from point correspondences. The question arises, why to expect the estimation from 3 LAFs would perform better? This section presents arguments for 3LAFs usage. It will be shown that estimation from 3LAFs has some theoretical advantages over per-point estimation.

2.2.1. Probability of all-inlier match

The probability of uncontaminated sample (i.e. all inliers) s_{inl} is

$$P(s_{inl}) = e^n, \qquad e = \frac{inliersCnt}{corrsCnt}, \tag{2.1}$$

where e is the inlier ratio and n is a sample size. Inlier ration is defined as a fraction of inliers count to the number of tentative correspondences. Assume now, that e is an constant no matter if we use point correspondences or LAFs to estimate F. It will be shown that the probability of having a minimal all-inlier sample is higher for LAFs.

The fundamental matrix has 7 constraints and the minimal sample size for point correspondences is 7. This is the minimal number of correspondences we are able to estimate fundamental matrix from. For LAFs only 3 correspondences are required for the

minimal sample yielding 9 constraints (3 const. per LAF). Formally, the probabilities are:

$$\begin{aligned} P(s_{laf}) &= e^3, \\ P(s_{pt}) &= e^7, \\ P(s_{laf}) &> P(s_{pt}), \end{aligned} \tag{2.2}$$

where s_{laf} and s_{pt} denote all-inlier minimal samples for LAFs and point correspondences respectively.

The previous statements give theoretical advantage to 3LAF2F to be faster than per-point method. This is due to the stopping criterion. The RANSAC algorithm is stopped, when the probability of missing an all-inlier sample falls below given threshold. Now, compare again the 3LAF2F and per-point with respect to stopping criterion. The probability of missing an all-inlier sample is lower with LAFs. The probability of early success is high. Thus, the 3LAF2F requires lower number of iterations to satisfy the desired confidence. As a consequence, it can be faster.

2.2.2. Execution for lower number of correspondences

Situations with lack of correspondences may occur. Even though the fundamental matrix is required.

The LAF offers 3 point correspondences. Minimally, 3 corresponding regions have to be extracted to allow the problem to be solved. The per-point approach requires more correspondences. The minimal number for per-points is 7 correspondences.

Thus, the 3LAF2F algorithm can be executed on images with less than 7 correspondences. Of course, the per-points algorithm can be supplied with all 3 points of LAFs (not only centers), but 3LAF2F algorithm can handle such a situation naturally, i.e. without any modification.

The described low correspondence case is not expected to happen frequently in practice, nevertheless it is a positive property of LAF method.

2.3. The epipolar constraint

In the introduction, the need for fundamental matrices was justified. There were given few applications which give motivation for its estimation. In this section, a more detailed view on the epipolar constraint and F is given.

The geometry of the epipolar constraint is captured in the figure 2.2. Two cameras (C_1 and C_2) capture a static scene from two different viewpoints. As can be seen, viewing a point X in the camera C_1 constraints the possible occurrences of the image in the camera C_2. Or more precisely, knowing the image x in C_1 and the relative pose between cameras, we can identify the line l' in C_2, which expresses the possible correspondences of x in C_2. The line constraining occurrences in the image of C_2 is an epipolar line (eq. 2.4). The triple C_1, C_2 and X defines a plane, which is known as the epipolar plane.

As it has been mentioned, a relative pose between cameras is needed to exploit the epipolar constraint. The relative pose and thus the epipolar constraint is expressed using fundamental matrix. Fundamental matrix is $R^{3\times3}$ matrix of rank 2 usually denoted F. The well-know definition of F is

$$x'^T F x = 0, \quad rank(F) = 2, \tag{2.3}$$

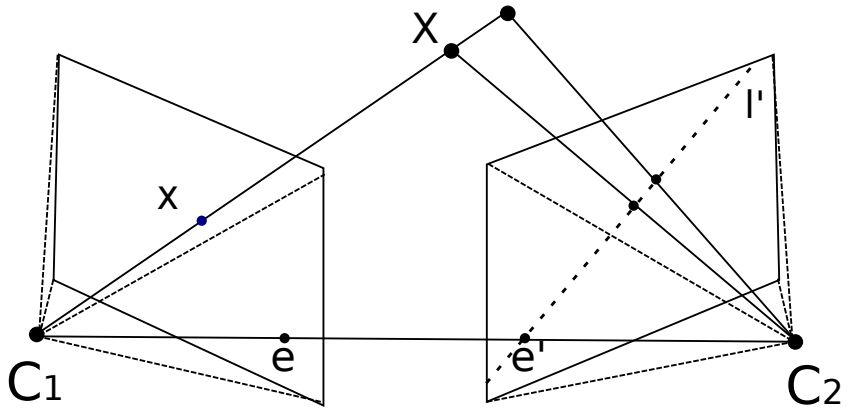


Figure 2.2. Epipolar constraint. The corresponding image of x in C_2 is constrained to l'.

where x and x' are homogeneous image coordinates in the images of the cameras respectively. Knowing a fundamental matrix F, x' and x are correspondences, only if this formula holds. The stereo setup has 7 degrees of freedom. Single correspondence (x, x') provides one constraint. Thus, to calculate F, at least 7 correspondences between image pair are needed.

The epipolar constraint also features epipoles. The epipoles are marked as e_1 and e_2 in the figure 2.2. An epipole is a projection of camera center of the second camera, e.g. C_2 is projected to e_1 in C_1. The relationship between epipoles and F is captured in the equation 2.4. The line connecting centers of a camera pair is known as the baseline.

$$\begin{aligned} \text{epipolar line:} \quad & l' = Fx \qquad l = F^T x' \\ \text{epipole:} \quad & Fe = 0 \qquad F^T e' = 0 \end{aligned} \tag{2.4}$$

2.4. Distinguished regions

Local affine frames are constructed from distinguished regions (DR). The distinguished regions (DRs) are therefore presented as a precursor to LAFs.

The distinguished region is a feature point generalization. While a feature point is defined as a point, the DR is defined as an image subset (i.e. pixel set). This is probably the most principal difference. Of course, as a generalization, those concepts have some properties in common.

The DR and a feature point as well are expected to be repeatable. To explain repeatability, consider we have some images of the same object from different viewpoints. Then we detect DRs in those images. The repeatability means, that the detected DRs in different images will denote the same thing in the scene, i.e. it is the property of being repeatedly detectable under changes of viewpoint, illumination changes and etc.. This property is very important for computer vision, because it enables matching. Matching, as it has been stated, is a basis for many computer vision algorithms.

The DRs, as they have been described, may look like a perfect theoretical concept without a method how to obtain them. This would however be a wrong conclusion. Actually, more detection methods were presented [15][18]. Among others, a detector of maximal stable extremal regions (MSER) is very popular. This one was used in this work.

2.5. Local affine frames

Distinguished regions together with local affine frames (LAFs) form an object recognition framework.

The LAF is comprised of 3 points (example in figure 2.4), i.e. the origin (center) and two axis points (side points). Those points form a local coordinate system for a distinguished region. The DRs are detected in various images and their LAFs are repeatedly detectable as well. The beneficial property of LAFs is that their coordinates relatively to distinguished regions remains the same in different views, i.e. LAFs express local coordinate systems for DRs.

Distinguished regions are subject to normalization when their comparison is needed. The DRs in different images representing the same object in the scene can be variously transformed depending on the viewpoint. To enable direct comparison of DRs a normalization is required. To facilitate such a normalization the LAFs were introduced. As they define a local coordinate system, they can be used to transform the DRs to canonical frames. That is to normalize image patches. The normalized DRs allow their comparison. This is useful for matching problem, where distinguished regions in different images can be compared with each other and correspondences can be suggested.

The LAFs are constructed from DRs. The DR is described with center of gravity and the matrix of second moments at first. The distinguished region is then normalized with the matrix. This enables to detect other geometric constructs, which are not affine-invariant. The region is fixed using the center of gravity and the matrix of moments up to an unknown rotation. This rotation is then additionally fixed using the mentioned geometric constructs. There are more geometric constructs, which can be used to construct LAFs. The LAFs can be then classified according to the geometric measure, which was used to their construction. There are examples of LAF constructions in the figure 2.3 (image from [18]). Those were constructed using various geometric measures.

The LAF cannot be constructed from every DR and is not constructed from every. The LAF cannot be constructed upon an elliptic region. Normalized ellipsis is a circle, which cannot be used to extract a rotation. LAFs are not extracted from every DR, but only the most likely repeatable and stable ones are used [19].

2.5.1. Used LAFs

However LAFs were presented as an comprehensive approach to matching or object recognition, they are used in another way in this thesis. They are used as correspondences for fundamental matrix estimation. Each LAF offers 3 point correspondences. An example of corresponding local affine frames is shown in the figure 2.4.

The feature extractor was set up to produce the following LAFs. These were than used as an input to 3LAF2F algorithm.

1. CM of DR + CoG of DR + point of maximal convex curvature
2. 2 tangent points on a concavity + point of the concavity most distant from the bitangent line
3. CM of DR + CoG of DR + bitangent direction
4. CM of concavity + CoG of concavity + bitangent direction
5. Junctions (converted to frames)
6. CM and affine point orientation

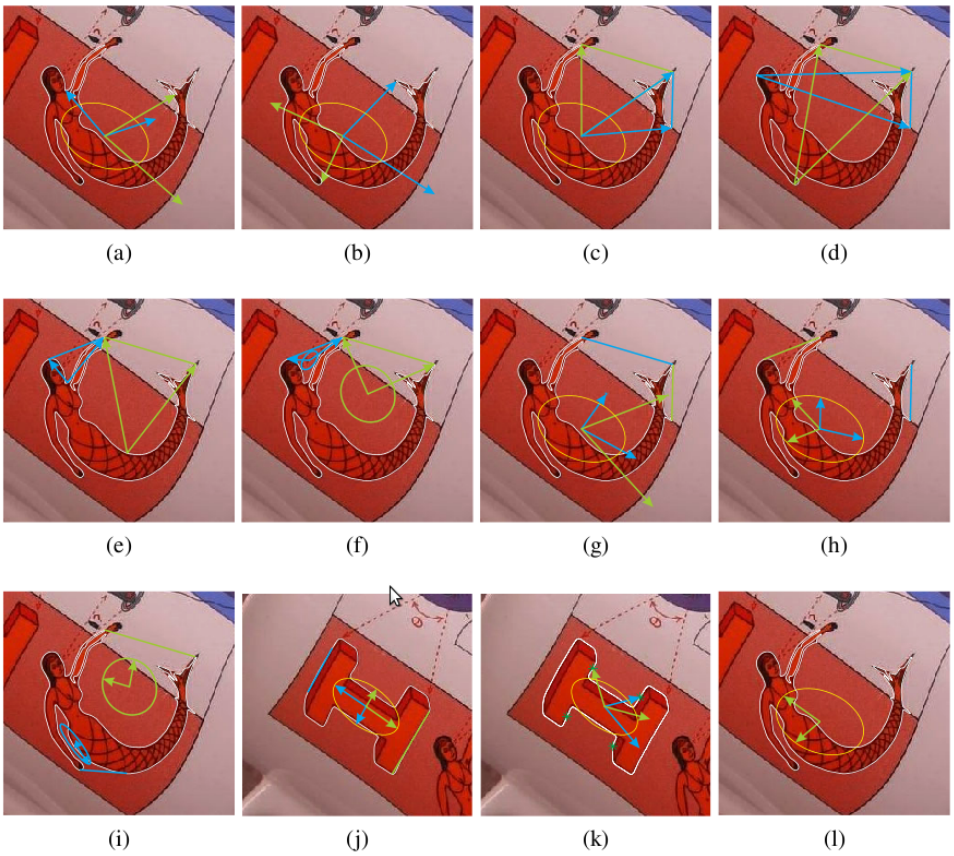


Figure 2.3. Various LAF constructs. (adopted image [18])

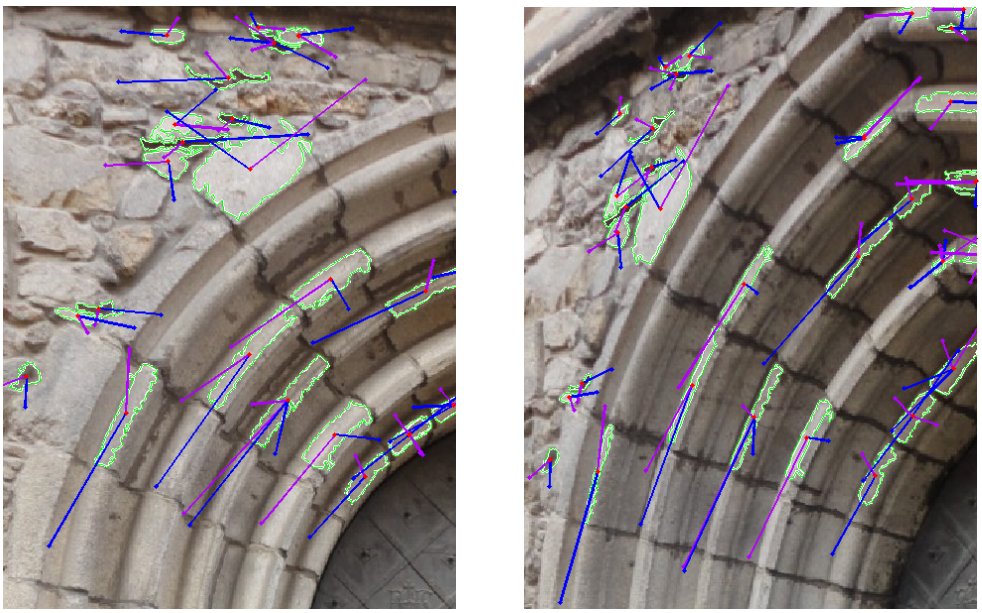


Figure 2.4. Corresponding LAFs example. (red points - LAF centers; blue and violet line segments denote the side points)

2.6. Solving for F with SVD

A method to save computational costs is suggested in 4.8. The suggestion is applicable for F calculation using SVD. Therefore, the topic is reviewed here and then in 4.8 the specific modifications are presented.

The fundamental matrix can be formulated as the least squares problem and subsequently solved using SVD. The algorithm is presented in three steps.

1. **Formulate homogeneous system** - in the first step, the $\tilde{F}$ matrix is described by system of linear equations. The $\tilde{F}$ matrix loosely represents the epipolar constraint, but can be a regular matrix, i.e have rank of 3. Therefore, it is not a fundamental matrix by definition and the distinguishing is necessary. The $\tilde{F}$ is calculated from correspondences. At least 8 of them are required to express the $\tilde{F}$ with a linear system. The linear system is then composed of 8 equations for 9 unknowns. Each correspondence brings 1 constraint for $\tilde{F}$. A correspondence is expressed with equation as stated next.

$$(xx' \quad yx' \quad x' \quad xy' \quad yy' \quad y' \quad x \quad y \quad 1) \cdot \vec{\tilde{f}} = 0 \tag{2.5}$$

   The (x, y) and (x', y') denotes correspondence between images I and I'. The $\vec{\tilde{f}}$ represents $\tilde{F}$ matrix, but is written as a single column vector. As follows, the system is represented with $R^{m\times 9}$ matrix A, where m is at least 8. The matrix A is constructed by filling its rows with equation 2.5, each of which representing different correspondence (eq. 2.6).

$$A = \begin{pmatrix} x_1x'_1 & y_1x'_1 & x'_1 & \cdots & 1 \\ x_2x'_2 & y_2x'_2 & x'_2 & \cdots & 1 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_mx'_m & y_mx'_m & x'_m & \cdots & 1 \end{pmatrix} \tag{2.6}$$

$$A\vec{\tilde{f}} = \vec{0} \tag{2.7}$$

2. **Solve using SVD** - the equation 2.7 is solved using SVD. Having $[U, S, V] = svd(A)$, where U and V are orthonormal matrices and S is a diagonal matrix with singular values. The solution corresponds to the smallest singular value (which is a zero in the case of 8 correspondences) and can be found in the last column of V. The last column needs to be reshaped to obtain $\tilde{F}$ matrix. This method produces a least squares solution in the case of overdetermined system.
3. **Singularize** $\tilde{F}$ - although comply with correspondences, the $\tilde{F}$ is not a fundamental matrix. The inaccuracy of correspondences causes $\tilde{F}$ "epilines" to not meet in the "epipole". The situation is illustrated in the figure 4.4 (could be more than dozen pages ahead). The left sub-figure shows the "epipole" of $\tilde{F}$, which is not a point. The desired state is in the right sub-figure, where all epipolar lines meet in a single point, i.e. epipole. This unwanted property of $\tilde{F}$ is indicated by its regularity.
   The fundamental matrix is defined to a be singular matrix of rank 2. To obtain such F, the $\tilde{F}$ have to be singularized. The singularization is a process of enforcing rank 2 to otherwise regular matrix $\tilde{F}$.
   The singularization can be done using SVD once again. The matrix $\tilde{F}$ is decomposed to $[U, D, V]$ and the diagonal matrix D is inspected. The matrix is singular only if

it has at least one zero entry in the diagonal of its D. Thus, the D is verified if it contains a zero. If don't, the singularization is done by zeroing the smallest singular number. The fundamental matrix F is then obtained by multiplying U, modified D and V' back together. Using this method, the $\tilde{F}$ will become fundamental matrix F.

3. LO-RANSAC

The original RANSAC algorithm was published by Fishler and Bolles in 1981 [7]. Since then, it has gained popularity in computer vision. It has became a standard for solving many computer vision problems. Its applications include: short baseline stereo stereo matching and wide baseline stereo matching, motion segmentation, mosaicing, detection of geometric primitives, robust eigenimage matching, structure and motion estimation. [2]

The wide success of RANSAC is assumed to be caused by its robustness. The robustness is wanted property for many computer vision algorithms. Lets consider the current problem of fundamental matrix estimation. The tentative correspondences, which are an input, are not only burdened with noise, but also contain invalid correspondences. Those are caused by matcher. Even though the TCs contain invalid correspondences, the algorithm should be able to estimate fundamental matrix. The algorithm is required to be immune to outliers, i.e. a robust algorithm.

However RANSAC is a favorite option in computer vision, there are more possibilities. The following two algorithm families also possess the robustness property. They are Hough transform and genetic algorithms. Both are used in computer vision. Hough transform is known for its use for shape extraction from images. A difficulty coming with Hough transform is the need to parametrize the space of solutions. This could be considered an disadvantage compared with genetic algorithms, i.e. another family of robust algorithms, because they don't need parametrization. Genetic algorithms are modest and flexible and were proved useful in solving computer vision problems. [1]

An unconventional F estimator was proposed, which is worth of notice.[6] The interesting point is the architecture of the algorithm, because it is a combination of RANSAC and Hough transform. The outer part is RANSAC, which samples less than minimal samples. The hypotheses are generated and used to to vote in hough transform. The final fundamental matrix is a result of the voting.

Returning back to RANSAC and reminding its utility for computer vision, its weaknesses have to be also mentioned. The RANSAC is a statistical algorithm and high quality results are not guaranteed. The algorithm can be analyzed statistically and probabilistic conclusions can be made. Thus, there are statements about RANSAC computational time or quality of its results. The drawback is that stable and reliable output is a common requirement which people intuitively requests and prefer. For example, consider a camera which is taking photos with 80% success rate.

The RANSAC itself is displayed in the algorithm 3. The main loop is shown enclosing its components. The algorithm repeatedly samples from given datapoints, i.e. the input. A hypothesis is generated from the gathered sample and it is subsequently verified. The verification evaluates the quality of hypothesis. The algorithm stores only the best-so-far hypothesis. Therefore, after the hypothesis is evaluated, it is considered if it is the best. If so, it is stored. Once enough iterations have been done, RANSAC stops and outputs the stored (best) hypothesis.

Although other options and weaknesses were mentioned, the choice was for RANSAC. It was successfully used in solving similar problems, thus it is expected to be also convenient for solving the 3LAF2F problem. The algorithm can be naturally separated into components, which was exploited in the following text. In the rest of this chapter,

the components are described and analyzed respectively.

Algorithm 1 The RANSAC algorithm

```
proc ransac(data)
  bestF = 0
  while 1
      s := minSample(data)
      F := makeHypotesis(s)
      sup := verifyHypothesis(F)
      if isBestSoFar(sup)
        bestF := F
      fi
      if shouldStop(conf)
        exit
      fi
  end
end
```

3.1. Sampling

The first step of RANSAC algorithm is the sampling. Its purpose is to select required number of datapoints. Those datapoints will be used to generate a hypothesis, which is a fundamental matrix in our case. The importance of the step is evident. It is at the beginning of the RANSAC and all other components rely on it. The sampling is what makes RANSAC a random algorithm, because it is the only step of the algorithm which is randomized.

The input datapoints are tentative correspondences of LAFs in our case. As it is given by the problem definition, 3 LAF correspondences have to be selected. Each LAF consists of 3 point correspondences and therefore 9 will be yielded in total. The minimal number of point correspondences to calculate F is 7, thus the input is sufficient. Moreover, the number of yielded point correspondences (i.e. 9) make it possible to use 7-point as well as 8-point method for F calculation.

The sampling is a random process. We have no information about datapoints being inliers or outliers, therefore we make a random guess. The random guess is required to select unique datapoints, otherwise degenerated hypotheses will occur. The uniqueness property calls for some mechanism to distinguish between points that were already used and those which are free to use. This problem may be solved using a random permutation, or more precisely, its subset. The indices to all datapoints will be generated, e.g. $1, 2, 3, \cdots, n$. Then, a k-subset of random permutation of indices will be calculated. The k-subset will contain indices to sampled points, i.e. sampling will be done. As it is suggested, the problem can be transformed to the generation of a random permutation. A convenient way to produce a random permutation is a Knuth shuffle [8][11]. Beneficially, this algorithm is able to output only the required k-subset of a permutation, thus it saves time. Additionally, it produces permutations with uniform distribution, i.e. all permutations are equally probable.

The Knuth shuffle algorithm is simple. Assume that we want to generate a random permutation of n items. We prepare the initialization vector, which contains all n

items. Then we iterate through the vector by single step and make random swaps. Random swap is swapping the actual value (i.e. pointed by iteration) with some value on random position. The random position can be generated as a random integer from $[1, n]$ interval using for example a standard library. As the algorithm proceeds, the vector is pair-wisely swapped and the random permutation is obtained. To generate only a k-subset, the Knuth shuffle can be set to do first k iterations and then quit instead of doing complete pass of n iterations.

Another approach to generate a random permutation is to use random numbers and sorting. Random numbers are generated and sorted using any sorting algorithm (e.g. quick-sort). The order of elements is used to generate the random permutation. The advantage of this approach is a simple control over distribution of random permutations, because it depends on the distribution of the initial random numbers. Thus, it can be used to generate permutations with other than uniform distribution. This approach, however interesting, exceeds our requirements and it costs more computational resources. The sorting is $O(n \cdot log(n))$ for quick-sort, while the Knuth shuffle can perform in $O(n)$ meeting all requirements. Therefore, the Knuth shuffle is used.

The discussed algorithm samples unique datapoints. This avoids to use the same datapoints in a single sample. Although this is avoided, a similar flaw is still possible. The same datapoints (i.e. whole sample) can be sampled in different iteration. To solve this issue a hashing was introduced to the problem[13]. The used samples are hashed and the hash is stored. Then, when a new sampling takes a place, only samples with a yet unseen hash are used. This solves the problem of repeating, but also increases the complexity. This improvement is not used by the algorithm proposed in this work.

3.2. Hypothesis generation

Sampled datapoints are passed to the hypothesis generator. The name is self-explanatory, a hypothesis is generated from a sample. A hypothesis is a particular parameter to a model. To be more specific, let us consider the case of our problem. The model is an epipolar constraint and the parameter is a fundamental matrix. A particular fundamental matrix is hypothesized in this step. An important observation is that hypothesis has the same form as the result, i.e. both are fundamental matrices.

As it has been introduced, the F needs to be calculated. Two possibilities are considered:

1. 7-point algorithm,
2. least squares.

The calculations are not analyzed here, because ready-to-use implementations were used. Thus, the implementation is not the topic. However the inner workings are omitted here, some relevant properties of those numerical methods should be mentioned.

The 7-point method has a variable number of solutions. Three solutions can be found at most. The question is which of those matrices to use. This decision making costs an additional computational resources. This is not the case of the least squares approach which produces a single result. It is assumed in this paragraph, that the correspondences supplied to the calculation are not degenerated, i.e. they provide enough constraints to determine epipolar geometry. If this assumption was false, the stated number of solutions would be invalid. There will be infinite number of solutions when underdetermined.

In this thesis, the 8-point algorithm is considered as a special case of the least squares approach, i.e. non-overdetermined system is provided and the sum of squared errors

is zero. The 8-point was criticized for being sensitive to noise in matched points and normalization was proposed [9] to solve this issue. The normalization of correspondences was adopted in this work.

The question of which of those numerical methods (or its implementations) is more suitable for 3LAF2F problem is further considered and experimentally decided in the chapter 4.

3.3. Hypothesis verification

Once the hypothesis (i.e. fundamental matrix) is generated the need for its evaluation rises. It must be decided whether the found hypothesis is the best-so-far, because we want only the best hypothesis and only the best is stored.

The RANSAC algorithm is or can be viewed as an optimization technique. A usual approach in optimization to express things "we want" is to use an objective function. In this case, the objective function was defined:

$$hypoError = \sum_{i=0}^{n} error(e_i^2, \theta^2) \tag{3.1}$$

and is a subject to minimization. The function expresses an error of fundamental matrix. The function $error(..)$ expresses an error of particular tentative correspondence. It is defined as a function of epipolar error and threshold. The error function and its inner design is the subject of upcoming text.

3.3.1. Error function

Even true tentative correspondences are burdened with error. This error is expected to be smaller than with false correspondences, nevertheless it is present and makes the problem more complicated. The complication is caused by the inability to simply identify inliers by checking if its epipolar error is zero. In other words, the epipolar error of an inlier can be non-zero.

To address this problem an error function was introduced to the system. Its purpose is to try to distinguish an inlier from an outlier. The function indicates outlier with higher output value.

Various definitions of error functions were presented [20]. The considered approaches are:

1. square of epipolar error,
2. thresholding epipolar error,
3. MSAC [20].

The disadvantage of the first approach is its not-being-robust property. A distant outlier (i.e with high epi. error) can significantly contribute to overall error of correct fundamental matrix. Therefore, the optimization process using square of epipolar error can tend to move its candidate solutions towards outliers.

The described issue is prevented by outputting smaller values for distant points. The thresholding (2) and MSAC (3) methods use this strategy. The threshold method introduces a user-defined parameter to the system called a threshold θ. Using the threshold, the piece-wise error function is constructed (3.2). If the absolute value of epipolar error falls bellow the threshold, then zero is returned denoting an inlier. Once the epipolar error exceeds the threshold, the function will return a constant value (e.g 1) denoting an outlier. This approach is simple and avoids the outlier influence issue.

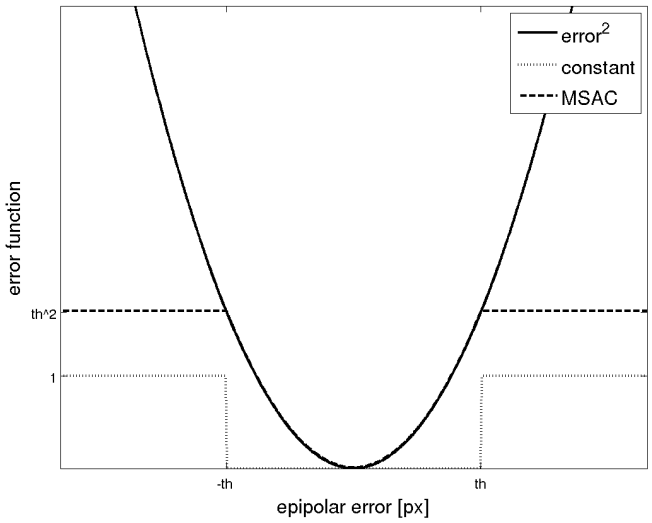


Figure 3.1. Examples of error functions.

$$error(x) = \begin{cases} const & \text{if } e^2 \geq \theta^2 \\ 0 & \text{if } e^2 < \theta^2 \end{cases} \tag{3.2}$$

The last method stated is MSAC(3). It is an improved thresholding method with additional treatment for inliers. It can also overcome the outlier influence issue and additionally it distinguishes between inliers. The function does not return a constant value for inliers, but it returns square of its epipolar error instead. The function is plotted in the figure 3.1 and defined in the equation 3.3.

$$error_{MSAC}(x) = \begin{cases} const & \text{if } e^2 \geq \theta^2 \\ e^2 & \text{if } e^2 < \theta^2 \end{cases} \tag{3.3}$$

3.3.2. Threshold

Although robust methods offer convenient properties, they have a drawback. They require a threshold to be defined. This is a disadvantage compared with the square of epipolar error based error function, which is working "out of the box", i.e. no additional parameter is needed. The threshold defines the defines the difference between an inlier and an outlier. Defining such a value is not that easy and was the subject of a previous work [13]. The definition of the threshold is discussed in this section.

The threshold is shown to be a part of the problem definition. Consider that we are looking for fundamental matrix F which has the lowest $hypoError$ given a threshold θ_1. Now, if we change the threshold to θ_2, a different fundamental matrix may be found. The possibly higher threshold θ_2 could mark more datapoints as inliers and consequently the outputted fundamental matrix may be different. Based on this reasoning, it can be stated that the threshold is a part of the problem definition, i.e. it specifies fundamental matrix we are searching for. Therefore, the appropriate value of threshold is crucial for the algorithm to success.

The problem of threshold selection can be studied analytically. In the following text, two phenomenons which influence the threshold value are considered.

Image quality - the threshold can vary depending on instance. Using high resolution images as the input may require a different threshold than low resolution images. This is because the precision of LAFs depend on an image resolution. Imprecise LAFs lead to imprecise fundamental matrices. Such Fs have higher variance of epipolar error for inliers. Therefore, F calculated from high-resolution images needs a smaller threshold.

Detector precision - this issue is coupled with the previous one. Various feature detectors are in use today. They have different precision capability. Consider detector with fine sub-pixel precision and some other else with worse precision (e.g. because of time trade-off). Each of both detectors may output inliers with different variance. The 3LAF2F algorithm is required to be detector independent and should not be coupled with any particular detector. Thus, this again leads to reconsider the threshold value.

Selecting the threshold too low will cause the error function to confuse an inlier for an outlier. This should lead to suboptimal solutions and instability. On the other hand, making the threshold too high will make various solutions having the same *hypoQlt* more possible. This will discredit the function *hypoQlt* at all, because it will lost its comparison functionality. This will also lead to instability.

In 2012, Lebeda et al. [14] proposed a method how to model the threshold. However the article is devoted to F estimation from point correspondences, it is presumed that it will apply well for the 3LAF2F problem. The method statistically models inliers (eq. 3.4). It is assumed that inliers are burdened by isotropic Gaussian noise. Therefore, the distribution of error on inliers is χ^2 with 1 DoF. This assumed error value is further scaled with σ^2 and σ_s^2. The σ^2 represents detector precision and is user-controllable. It is set to $\sigma = 0.3$ in our particular case. The σ_s^2 is an image-size factor. It addresses the input heterogeneity issue (i.e. various size of input images) and tries to make the threshold independent of input size. The image-size factor σ_s is defined as maximal dimension (e.g. 1024 in the case of 1024x768) divided by a constant factor 768.

$$\begin{aligned} \theta^2 &= \chi_1^2 \cdot \sigma^2 \cdot \sigma_s^2 \\ \sigma_s &= \frac{max(im_w, im_h)}{768} \end{aligned} \tag{3.4}$$

3.4. Stopping criterion

The RANSAC algorithm is a loop. The number of iterations can be controlled manually or statistically. The manual approach gives a user an opportunity to express how much time is he willing to spend running the algorithm. This is suitable, when time for execution is predefined. Running the algorithm for as long as possible maximizes the probability of better solution.

The statistical method uses a probability of sampling an all-inlier sample to control the number of iterations. The number of iterations required k can be calculated using 3.5. [10]

$$k = \frac{log(1-p)}{log(1-u^m)} \tag{3.5}$$

The u is a probability of selecting an inlier, m is the sample size (e.g. 3) and p is a probability that at least one of the samples will be an all-inlier. The probability u of selecting an inlier is not known in advance and is recalculated every iteration. The probability p is given by user (e.g. 0.95). The algorithm recalculates k during its run-time and quits when enough iterations has been done.

3.5. Local optimization

In 2003, Chum, Matas and Kittler[3] introduced a new step to the RANSAC algorithm. The step is known as the local optimization (LO) and the RANSAC alternation is called LO-RANSAC.

The assumption is made in a plain RANSAC about hypothesis generation. The hypothesis generated from an all-inlier sample is assumed to be correct. This assumption was shown not always true. The incorrectness can be observed in a number of iterations. In some cases, the number of iterations does not correspond to the theoretical value. The LO-RANSAC was suggested to address this problem.

It contains the local optimization step, which uses current inliers (i.e. at the time of LO execution) to make hypothesis more precise. Generally, it uses non-minimal samples from inliers to do that. As a consequence of higher quality hypotheses, the number of iterations needed is reduced and thus computational time is saved. The positive effect on quality of the result [14] is very important for this thesis and is exploited in 5.2.

The original paper doesn't define optimizations to use strictly. However, there are suggestions and their impact on performance. Some of them with brief description are listed next.

1. **Simple** - solves linear system to re-estimate a hypothesis from inliers. The inliers are enumerated from all datapoints using the best-so-far hypothesis and some threshold θ. Non-minimal samples leading to overdetermined systems can be used. This is then solved using least squares. Such a sample is expected to yield more precise hypotheses.
2. **Iterative** - this approach repeatedly run the previous algorithm with different θ. The strategy is to use a higher threshold in the beginning (e.g some multiply of θ). As the LO algorithm proceeds the threshold is gradually reduced to θ.
3. **Inner RANSAC** - the previous method is predisposed to stuck in local optimum. To avoid this, inner RANSAC was introduced. The LO step contains RANSAC, which starts a new sampling procedure. The inliers from outer RANSAC are used as a source for sampling. However, the verification is done against the whole data set. The inner RANSAC hypothesis generator uses non-minimal samples.
4. **Inner RANSAC with iteration** - this is the most heavier LO step presented in the original article. It uses inner RANSAC(3) with the iterative(2) method. The iterative method is executed for each sample of inner RANSAC.

The most powerful local optimization presented by Chum et al.[3] is inner RANSAC with iterative LSQ. According to presented results, it yields the highest quality and stable results. However, the quality is redeemed with time. Lebeda et al. [13] focused on this LO method and suggested an improvement. His variant uses weighted LSQ to avoid influence of imprecise points and therefore to obtain even better results. Additionally, he has fine-tuned the algorithm involving a lot of benchmarking. The result is the state-of-the-art LO-RANSAC for F estimation, which the LR73L algorithm proposed in this work tries to compete with.

3.5.1. Execution and computational time of the LO step

Computational time is very important in this thesis. It is one of the two objectives. Therefore, the LO step influence on time is considered.

The conclusion of LO-RANSAC article [3] states that, using the LO step increases quality of hypotheses as well as reduces computational time. This behavior is explained with respect to the stopping criterion. As the optimization proceeds, it increases the number of inliers detected. This increase could happen in early iterations of LO-RANSAC. Lets assume this happened for now. As a consequence of a high number of inliers, the number of iterations needed estimated by the stopping criterion (3.4), is decreased and the algorithm stops early saving time.

Later, it was shown that this is not always the case. Lebeda at al. [14] showed a case where LO-RANSAC ran longer than plain RANSAC. The results showed that more than a half of the run-time were spend in LO. Therefore, the LO step should be used with care.

To save time performing the LO step, it is executed only occasionally. The original design executes LO only for the best-so-far hypotheses. Additionally, recommendation [14] is to use some iteration offset for the LO step execution. The LO is then executed only when the best-so-far is found and enough iterations have happened. Another trick is to use a limit for correspondences entering the least squares calculation during the local optimization. The LSQ calculation is known to be time demanding and this is the method how to feather it. The problem of LO execution for 3LAF2F algorithm is studied in 5.2.1.

4. Designing algorithm components

The LO-RANSAC was selected as a suitable frame for the algorithm. The LO-RANSAC consists of several steps. Specific implementation of those steps is the subject of this chapter. In this chapter, a number of experiments are designed and performed to bring the answers to design issues.

4.1. Precision of LAFs

The very first question which rose when the 3LAF2F problem was reopened was about precision of LAFs. Are correspondences of LAFs as precise as point correspondences?

The question of LAFs stability and repeatability was studied before, but the research does not contain information about precision in the sense of epipolar error. Therefore, the experiment was done.

The experiment was set up to measure epipolar error of center and side points of LAFs respectively. The whole dataset, as it is defined in the appendix A, was used. The fundamental matrix was estimated by the state-of-the-art F estimator from [13] for every image pair from the dataset. Those fundamental matrices are considered correct and used to calculate the epipolar error.

The figure 4.1 shows the result using cumulative distribution function. The center points have obviously lower error. Both side points have similar error distribution. Therefore, the center points have more precise correspondences.

This discovery had a significant impact on the algorithm design. It follows, that using center points should be preferred to side points whenever it is possible. This rule was applied few times in the work.

4.2. The sample size problem

Sampling process have to select points, which will be used for F estimation. There are 3 LAFs randomly selected from the input data. Those LAFs gives 9 point correspondences. The problem is, which of those correspondences select for estimation.

The minimum points required is 7 for fundamental matrix estimation. This is clearly satisfied. However, the number of points does not have to be minimal. Three options are considered:

1. use 7 correspondences and the 7-point algorithm as generator,
2. use 8 correspondences and the 8-point algorithm,
3. use 9 correspondences and least squares.

As follows from the listing, each possibility is coupled with a numerical method, i.e. the method used for estimation. This observation yields the fact, that the sampling cannot be decided apart from the hypothesis generator. This dependency complicates the problem.

To find the answer to the sample size problem an experiment was conducted. The experiment was set up to examine the influence of sample size on quality of the fundamental matrix. Quality was observed, because it is the objective in this work as well

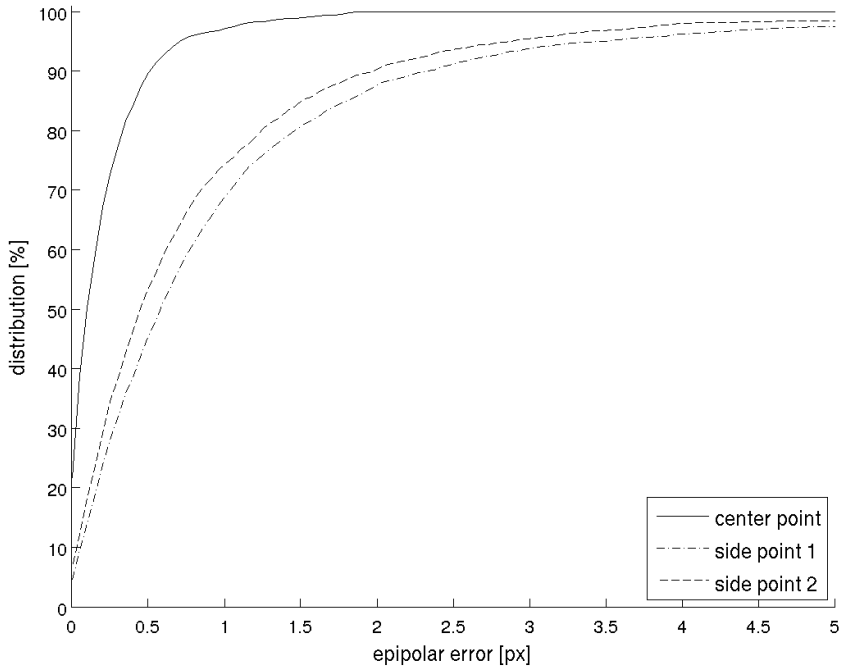


Figure 4.1. The precision of LAFs correspondences. The LAFs are decomposed to 3 point correspondences and are studied individually. The figure shows the distribution function of average epipolar error. Average is per-F across inliers.

as time. However, time is not considered here. This shortcoming was corrected in the following experiment.

The experiment was executed over all dataset and 22000 hypotheses were generated for each sample size. The important fact is that only inliers were used to generate hypotheses, i.e. fundamental matrices. Thus, only influence of verified correspondences is studied in the experiment. The wrong correspondences are of no interest, because they are assumed to lead to wrong F.

The results are presented in the figure 4.2. The figure displays cumulative distribution function of average epipolar error. As it is shown the 8-point sample with LSQ is inferior to the others, while the 7-point and the 9-point perform similarly with the 7-point doing a slightly better.

As a results of this experiment the 8-point sample was dropped from further reasoning. The decision between the 7-point and the 9-point is discussed next.

4.3. Computational time of hypothesis generator

The previous experiments showed that the sample size is tough to decide separately from the hypothesis generator. The 8-point sample was abandoned, because it produces low quality fundamental matrices. The remaining possibilities were performing similar.

High quality is not the only requirement for the solution, i.e. the algorithm being designed. Computational time is important as well. Thus, a stress test was performed. Two possibilities were left:

1. the 7-point sample and algorithm,
2. the 9-point with least squares.

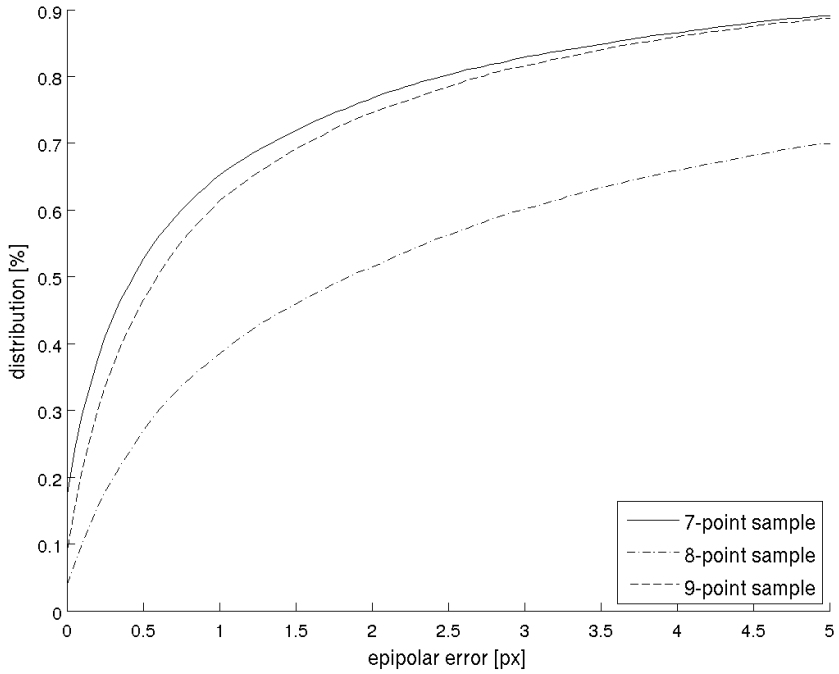


Figure 4.2. CDFs of average epipolar error reached using various sample sizes.

method	**time** [s] (10^6 runs)
7-point	57.90
9-point (LSQ)	205.16

Table 4.1. F calculation time for the two methods. Time of the decision between calculated models in the 7-point method is included.

It is not correct to say that algorithms are compared in this experiment. It is rather the comparison of their implementations. This work had implementations of both algorithms at disposal. The 9-point LSQ solver was based on SVD from GRASS library, while the 7-point implementation was written stand-alone. The 7-point was solving the nullspace using the Gaussian elimination and then solved the 3rd degree polynomial.

There is another issue which needs to be clarified. While the 9-point is working straightforward, i.e. you give 9 points and you always receive some F, the 7-point is not this kind. The 7-point estimates 2.6 models on average, but only one have to be output. It needs to select one. Thus, it calculates support over all tentative correspondences and returns the best F. The support calculation costs some computational time. It should be stated that this is included in the run-time of the 7-point algorithm.

The stress test was performed in the Matlab. The solvers are written in C. Therefore, the mex interface was used to execute the solvers. The solvers were executed 10^6 times and supplied with random 3 LAFs. For the 7-point sample, two random non-center LAF points were dropped.

The results are recorded in the table 4.1. As can be seen, the 7-point is doing almost 4 times better. As a consequence of this and the previous experiment, a hypothesis from 7 points proved to be the most accurate and the fastest. Thus, it is a well-founded choice for the 3LAF2F algorithm.

4.4. On quality and LAF magnitude

The sample consists of 7 point correspondences. To sample 7 points, 3 LAFs are selected, which brings 9 point correspondences. It was proved that LAF centers are the most precise. Thus, they are certain candidates to be included to the sample. The centers make 3 of 7 sample points, the remaining 4 points have to be selected from side points of selected LAFs. The question arises, which of the side points to use?

LAFs are of different size, i.e. the side points can be variously distant from the center points. It is argued that using larger LAFs would yield fundamental matrices with smaller epipolar error. The intention is to collect correspondences from all over the image. Such a sample would provide convenient situation for F estimation. It is assumed that having a correspondences only in a small part of an image will cause imprecision of the resulting F. Contrary, having correspondences across an entire image is more suitable for the estimation. It is assumed that using larger LAFs would lead to more suitable correspondences, than a random selection.

To achieve desired correspondence coverage, the LAFs side points are sorted by distance from the centers. The distant ones are preferred. It is not important to use both side points from a single LAF. Returning to the context of the first paragraph, there are 4 correspondences to be selected from the side points to complete the sample. There are 6 side points available. Those are sorted by their distance from center and 4 most distant are included to the sample.

To prove this approach an experiment was performed. Hypotheses, i.e. Fs, were generated for every image of the dataset. Two approaches were used.

1. The most distant side points from the centers (i.e. the discussed),
2. random side points (i.e. random side points are included to sample).

It was generated 22 000 fundamental matrices for each approach. Only the inliers were used in the generation process. The hypotheses from outliers are out of interest, because we assume only all-inlier sample will yield a high quality solution.

The figure 4.4 shows the distribution of error for both approaches. The assumption about influence of LAF size on F quality was proved true. A little higher probability of lower error can be seen. Because of the little difference between curves in the figure, the experiment was repeated several times. Every additional execution confirmed the result.

Therefore, it is beneficial to select 7 points with strategy presented in this section. In addition, it was discovered that measuring lengths of LAFs in a single view is sufficient. The original approach was using sum of lengths from both views. The single view method saves computational resources and works just as well.

4.5. Verification points

A generated hypothesis needs to be verified. Therefore, the support is calculated for a hypothesis. The support is defined as a number of inliers. However, it is a question how to define an inlier. The original 3LAF2F article [4] uses all 3 points of LAF. Epipolar error of all 3 points is considered for a datapoint to become an inlier. According to section 4.1, the center points have the most precise correspondences. Based on this observation, a different inlier definition was used. The epipolar error of a center point have to be below the threshold for LAF to become an inlier. The side points are not used in the verification.

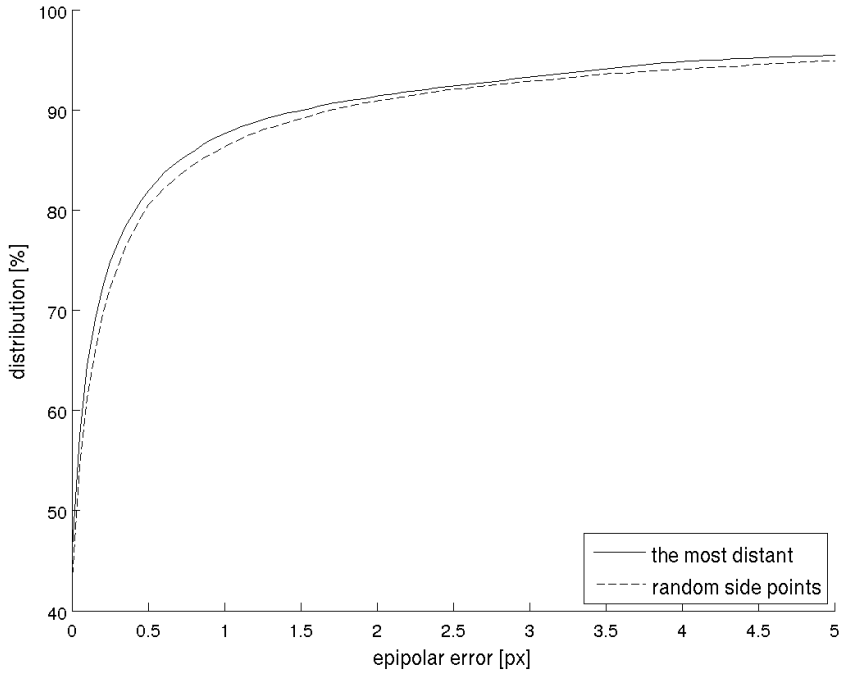


Figure 4.3. The influence of using the largest LAFs on hypothesis quality. The CDFs were constructed from 22 000 averages of epipolar error.

4.6. The LO step using LAFs

The LO-RANSAC algorithm requires lo step to be prepared for 3LAF2F problem. An intuitive approach would be to use LAFs for optimization. However as is has been shown is section 4.1 the LAFs correspondences are less precise then point correspondences. This measurement casts doubts. Thus, the problem of local optimization subject rises.

There are two possibilities considered. The first is to use all 3 points of LAFs in local LO step. Assume that the optimization consists of inner RANSAC and weighted least squares. The weighted least squares will therefore minimize error for the 3 points of LAF. The second possibility is to use center points only as it is done in traditional F estimation from points correspondences. To find out, an experiment was conducted.

The two possibilities were prototyped in Matlab. A complete LO-RANSAC F estimator with inner RANSAC and iterative weighted least squares was implemented. The only difference between those two implementations was the object used to calculate a new fundamental matrix in local optimization. The rest of the implementation was left the same to preserve otherwise similar conditions. Those programs were run 100 times on the dataset and the results were analyzed.

The results showed that these two approaches reaches almost the average epipolar error on the whole dataset, i.e. 10.06px for LAFs, 10.49px for points. However, after more detailed analysis it was discovered that these values are achieved in a different way. The study of CDF of average epipolar error (i.e. per-image average across executions) revealed the fact, that LAFs approach has more stable, but worse results. While using the centers for optimization the distribution function of average error was shifted towards lower epipolar error, but with notable extremal values. Those extremes are assumed to be responsible for influencing the average of otherwise better values.

The conclusion was made after inspecting results for easy images. The epipolar error for those (the first two in the figure 4.2) was notable worse. Consider Brussels which is no challenging image. The epipolar error for this image is required to be under 1px. Using the LO with LAFs error reached 2.35px on average. The inferior results for easy images discredited the local optimization using LAFs. The approach using the centers only was proved to be better choice.

			LO.pt	**LO.laf**
Brussels	inlier	[%]	76.08 ± 5.13	62.21 ± 10.74
	error	[px]	0.49 ± 0.70	2.35 ± 2.28
	time	[ms]	207.83	231.10
	sample		80.21 ± 32.21	82.50 ± 27.90
	LO		74.14 ± 13.75	72.43 ± 10.63
Dresden	inlier	[%]	92.28 ± 1.53	88.39 ± 7.23
	error	[px]	0.26 ± 0.35	2.06 ± 4.32
	time	[ms]	199.23	221.03
	sample		32.10 ± 12.47	28.01 ± 13.38
	LO		314.49 ± 50.47	328.07 ± 69.93
Leuven2	inlier	[%]	48.72 ± 8.30	41.52 ± 9.69
	error	[px]	29.24 ± 37.05	70.52 ± 65.60
	time	[ms]	275.52	245.91
	sample		165.32 ± 41.76	176.97 ± 48.12
	LO		12.72 ± 1.14	12.48 ± 1.28
booksh	inlier	[%]	57.98 ± 7.14	50.33 ± 5.03
	error	[px]	3.53 ± 3.74	7.64 ± 4.04
	time	[ms]	86.27	86.10
	sample		37.17 ± 10.10	38.61 ± 11.84
	LO		19.44 ± 2.15	19.40 ± 2.30
zoom	inlier	[%]	50.74 ± 12.51	55.19 ± 11.48
	error	[px]	15.92 ± 28.12	7.33 ± 20.53
	time	[ms]	163.90	167.53
	sample		88.79 ± 28.17	95.45 ± 30.49
	LO		22.48 ± 3.49	21.96 ± 2.74

Table 4.2. Results from experiment comparing LO steps. LO.laf uses the whole LAFs in optimization, while LO.pt uses only the center points. Note the zoom image, which is a scarce example where LO.laf performs notably better.

4.7. Stochastic hill climbing

LO-RANSAC constitutes of LO step. The state of the art LO step uses inner RANSAC and iterative LSQ. In this section, it is assumed, that the success of this approach originates in its two phase reset-and-converge architecture.

The first reset phase is randomized (i.e. RANSAC) and avoids to stuck in a local optimum. It seeds the search for local optimum by randomly selecting a subset of inliers. Its working does not depend on a result of LSQ. Thus, if LSQ stuck in local optimum, the RANSAC put the algorithm to a different position in the search space. The second phase (i.e. iterative LSQ) converges rapidly to a local optimum.

The assumed success of reset-and-converge strategy motivated to design a new local optimization. As it follows, it uses the same design pattern.

The new LO step uses inner RANSAC as well, but it uses stochastic hill climbing in its converge phase. The hill climbing randomly samples from current inliers. When a better solution is found, inliers are updated and the threshold is decreased. The hill climbing is executed for a given number of times. After its end, it returns to RANSAC, which initiate a hill climbing with different input data.

The LO step just described was prototyped in the Matlab language. The goal was to evaluate its performance, especially quality of proposed solutions. The prototype was benchmarked. The results revealed that this approach is inferior to the state-of-the-art. Thus, the local optimization research was quit and the focus was set on proved approaches and their application for 3LAF2F problem.

4.8. Fundamental matrix condition

Estimating F using the given implementation of LSQ was measured to be slow compared with the 7-point algorithm implementation. While the LSQ is not needed in the main loop, it is still a part of the local optimization. Thus, it slows down the LO step. The LO step has been observed to dominate run-time in some cases. This observation motivates to search for an enhancement.

A novel approach to speed up the LO step was introduced. The approach exploits the process of LSQ F calculation to express support of F. That this, using this method support calculations are saved. Therefore, run-time is saved as well.

F estimation using LSQ involves singularization step. The singularization step is undertaken to make epilines meet in an epipole (figure 4.4). That is, it makes regular $\tilde{F}$ to become singular F. It is argued, that the inaccuracy of $\tilde{F}$ implicates a low quality F. $\tilde{F}$ with "epilines" far from each other will not lead to high quality F. This statement will be proved true, at least for selected pictures.

The measure of $\tilde{F}$ inaccuracy was needed. There were two requirements for the measure.

1. Easy to compute - the measure should be computed more faster than support. Therefore, no loop over correspondences in $O(n)$ is possible, but preferably $O(1)$.
2. Express the accuracy - the value should express how far the 'epilines' from meeting in a single point are.

The $\tilde{F}$ accuracy can be expressed using matrix singularity. The more the $\tilde{F}$ has "epilines" meeting in a single point the more it is a singular matrix. Thinking about $\tilde{F}$ as the system of homogeneous linear equations, the system has a very poor condition. The system is near having infinite solutions. A small change to the system of equations will potentially cause a big difference to the solution. This problem is known as a matrix condition in linear algebra. The problem of system conditioning is faced with the condition number. [21]

The condition number is high when $\tilde{F}$ is near singular. Therefore, the higher the condition number of $\tilde{F}$ is, the higher quality F is expected. An advantage is the availability of the condition number. There are minimal computational costs to obtain it, because it can be gained during the singularization step.

The condition number was redefined for the purpose of evaluating $\tilde{F}$ as follows.

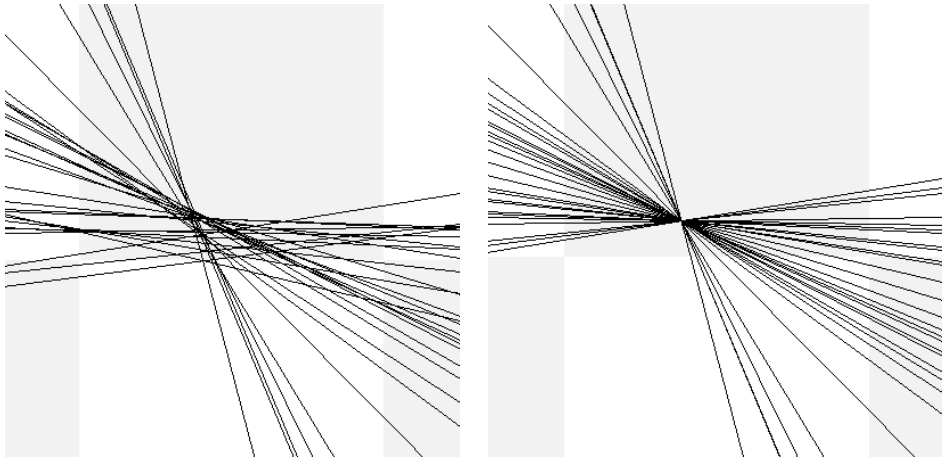
$$condition(\tilde{F}) = 1 - \frac{\sigma_{min}}{\sigma_{max}} \tag{4.1}$$

This limits the value to $< 0, 1 >$ interval, while preserving the higher the better behavior. The definition of condition reveals why it is easy to compute. The singular numbers are calculated during the singularization step. Therefore, the condition can be calculated with a little effort.

4.8.1. Experimental evaluation

To verify the condition number concept an experiment was conducted. The experiment was set up to study the dependency of F quality on $\tilde{F}$ condition. Random hypotheses from 9 correspondences (3LAF) were generated over the entire dataset. This yielded $22 \cdot 10^4$ hypotheses. Those hypotheses were classified to groups with respect to their support. Then, a cumulative distribution function of those groups was calculated with respect to condition.

The results are in the figure 4.8. As can be seen, there is a dependency between F quality and $\tilde{F}$ condition. However, the dependency is weakened by 10% of F matrices.



a) $\tilde{F}$ "*epipole*" **b)** F epipole

Figure 4.4. Geometric interpretation of fundamental matrix singularity. Pixels are represented with gray and white squares.

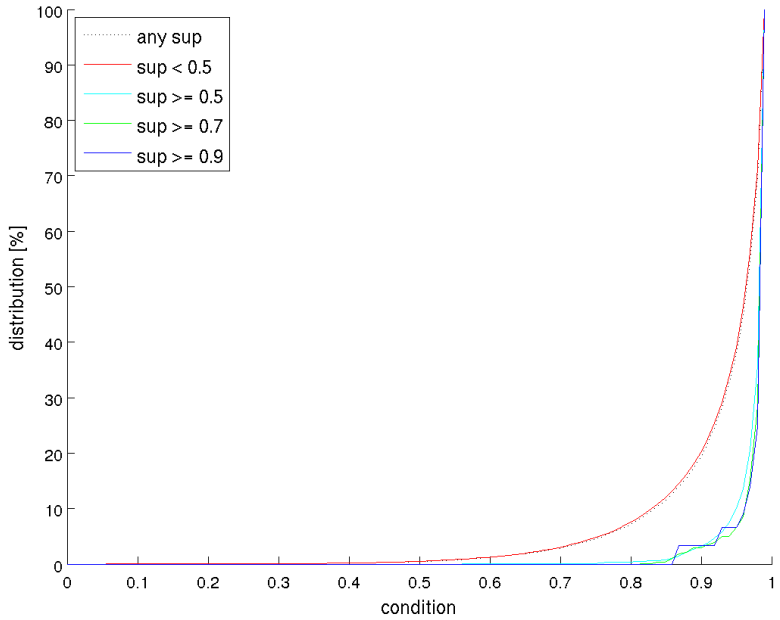


Figure 4.5. The distribution of F quality with respect to condition. The graph presents data gathered using the whole dataset. To enable inter-image comparison, the support is measured relatively to the best support reached in the individual image. Additional information in the table 4.8.1.

support	**number of Fs**
all	220000
sup < 0.5	211580
sup ≥ 0.5	8420
sup ≥ 0.7	1313
sup ≥ 0.9	108

Table 4.3. Complementary table to figure 4.8. The table shows how many Fs with given support are there.

Those matrices has high support, but lower condition. Additionally, it can be stated that high quality matrices have high condition with high probability. However, it can not be stated that low quality matrices have low condition with high probability. This makes impossible to evaluate fundamental matrices only on condition basis.

The experimental results proved the condition concept valid. However, there were some fundamental matrices that didn't obey the condition concept. There are 2 cases for those exceptional matrices. The first is, they are in fact a high quality estimation of fundamental matrix and the condition concept doesn't apply on them or secondly, they are a low quality estimation of F in fact, but they reach a high support. The second case can be caused by the assumption, that maximizing support maximizes F quality. If this assumption did not hold, some in fact high quality F would reach low support.

4.8.2. RANSAC condition prototype

To research more about condition and to determine if it is useful in practice, the RANSAC prototype with the condition concept was designed and implemented.

The condition concept was incorporated to plain RANSAC. The LSQ F estimator was modified to output the condition value. Then, RANSAC makes a use of this value to skip the support calculation. If the value of condition falls below certain threshold θ_{cnd}, the F is ignored, i.e. support calculation and the if-best-so-far is omitted. The pseudocode of the modification with respect to standard RANSAC is in the algorithm 4.8.2.

Algorithm 2 RANSAC with condition modification

```
proc ransac(data)
  while 1
        ...
        [F, cnd] := makeHypotesis(s)
        if cnd > θ_cnd
          sup := verifyHypothesis(F)
          if isBestSoFar(sup)
            bestF := F
          fi
        fi
        ...
  end
end
```

The prototype was benchmarked on the dataset. The results are presented in the

table 4.4. The table displays 4 selected images in its rows and two solving algorithms in its columns. The two solvers are: plain RANSAC and the modified with condition as presented recently. Although the benchmark was done on the whole dataset, only 4 selected measurements are shown.

The first two images Brussels and Dressed (image names are written sideways) are considered easy, because they have high inlier ratio. For such an easy images, the condition concept works perfectly. It saves time, while preserving quality. The situation is different with images low on inliers. The worst outcome was obtained for Leuven2. In 3% of cases in Leuven2 the algorithm omitted every hypothesis and returned no fundamental matrix, that is why the epipolar error is infinity. From further experiments, it was discovered that images with low a number of inliers require different θ_{cnd}.

In difficult images, the algorithm omitted a lot, thus computational time was lower. This speed-up was undeserved, because the quality was lower as well. Including those to time analysis will cause a bias. Therefore, those images were excluded. Even then time was reduced by 20% on average. This corresponds to 68% skipped fundamental matrices. It is assumed that the speedup will vary depending on a programming language used. In C, the speedup is expected to be lower.

The condition concept was proved useful for some images. Its reliability depends on selected threshold θ_{cnd}. This introduces a new parameter, which is controlled by a user.

		condNo	**cond997**
Brussels	inlier [%]	64.76 ± 6.11	64.57 ± 6.59
	error [px]	1.31 ± 0.98	1.18 ± 0.94
	time [ms]	46.17	35.10
	sample	100.00 ± 0.00	100.00 ± 0.00
	LO	0.00 ± 0.00	0.00 ± 0.00
Dresden	inlier [%]	91.74 ± 0.83	91.89 ± 0.50
	error [px]	0.33 ± 0.18	0.31 ± 0.12
	time [ms]	58.48	51.51
	sample	100.00 ± 0.00	100.00 ± 0.00
	LO	0.00 ± 0.00	0.00 ± 0.00
Leuven2	inlier [%]	43.98 ± 9.68	31.85 ± 12.28
	error [px]	39.62 ± 51.69	Inf ± NaN
	time [ms]	29.65	17.96
	sample	100.00 ± 0.00	100.00 ± 0.00
	LO	0.00 ± 0.00	0.00 ± 0.00
booksh	inlier [%]	62.24 ± 8.41	57.69 ± 7.87
	error [px]	4.17 ± 4.24	6.03 ± 4.95
	time [ms]	32.11	21.34
	sample	100.00 ± 0.00	100.00 ± 0.00
	LO	0.00 ± 0.00	0.00 ± 0.00

Table 4.4. Results of the RANSAC modification which uses the condition number. The *condNo* column shows the results of standard RANSAC. The *cond997* column shows the results of modified RANSAC, which skips every hypothesis with condition less than 0.997.

5. Experimental evaluation of the complete algorithm

5.1. Plain RANSAC version

The implementation of the 3LAF2F algorithm began with plain RANSAC version. This version contains no optimization step, although it was designed with LO keeping in mind, i.e. to add and enable LO in the future. The algorithm design is based on the findings presented in the previous chapter. It contains components, which should lead to the best performance. The goal was to achieve high quality fundamental matrices at low computational time as specified in the objectives section.

The design includes the following features.

The 7-point sample - only 7 point correspondences of 9 selected (3LAFS) are used.

All centers in the sample - the 7-point sample contains all 3 center correspondences. This is discussed in the section 4.1.

Largest LAFs in the sample - the sample is completed with the most distant side points of LAFs. (see 4.4)

The 7-point hypothesis generator - this generator was proved to be the fastest and the most accurate option. (see 4.3)

Center points support - the support is calculated using the center points only. (see 4.5)

The outlined design was implemented in the C programming language. The choice was obvious, because speed matters. Moving to such a low-level programming language will provide more control over a computer compared with the Matlab. Finally, the implementation was tested and benchmarked.

The benchmark was done running on the whole dataset for 1000 times. The complete results can be found in the appendix B. For illustration, results for selected image pairs are in the table 5.1. The table displays solvers used in its columns and images in the rows. There are 3 solver algorithms considered.

1. ransacF - represents standard, per-point F estimation approach. It is the state-of-the-art.
2. LR73L - this is the algorithm proposed in this thesis.
3. rEG3p - the original algorithm for the 3LAF2F problem. [4]

			ransacF	**LR73L**	**rEG3p**
Brussels	inlier	[%]	66.87 ± 5.95	34.27 ± 5.88	36.76 ± 6.10
	error	[px]	0.96 ± 0.72	6.55 ± 3.16	5.62 ± 2.65
	time	[ms]	1.42	1.30	88.83
	sample		80.42 ± 43.61	97.97 ± 36.34	379.28 ± 131.12
	LO		0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
Dresden	inlier	[%]	85.69 ± 4.59	52.73 ± 8.98	45.94 ± 8.50
	error	[px]	1.88 ± 3.24	14.32 ± 7.85	12.25 ± 8.13
	time	[ms]	1.14	1.47	18.40
	sample		10.32 ± 4.51	27.07 ± 11.48	61.22 ± 24.42
	LO		0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
Leuven2	inlier	[%]	55.00 ± 3.93	34.44 ± 6.78	38.26 ± 7.89
	error	[px]	6.55 ± 9.54	71.63 ± 66.18	65.32 ± 71.23
	time	[ms]	4.51	1.29	211.25
	sample		891.03 ± 339.00	211.07 ± 67.90	1084.21 ± 907.97
	LO		0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00

		ransacF	LR73L	rEG3p
booksh	inlier [%]	67.74 ± 5.34	49.28 ± 7.22	42.12 ± 7.09
	error [px]	1.69 ± 2.40	6.55 ± 4.22	7.27 ± 4.41
	time [ms]	0.83	0.33	28.14
	sample	107.92 ± 52.32	43.50 ± 15.46	130.09 ± 29.28
	LO	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00

Table 5.1. Benchmark results of the suggested algorithm LR73L with no local optimization. For the measured quantities please refer to the appendix B.

As can be read from the table, the 3LAF2F approach can not compete with per-point approach. The average epipolar error is so high, that it is considered unusable. The theoretical advantage of 3LAF2F should be lower number of samples. This is shown to be true only in particular cases. Comparing the both 3LAF2F estimators to each other, it can be stated that the LR73L algorithm runs faster and used lower number of samples. However, quality is slightly worse.

The performance lead of traditional approach is alarming and should indicate some error. Thus, an analysis was done to explain this behaviour. It was discovered that hypotheses from traditional generator are higher quality. As can be seen in the figure 5.1, the hypotheses from correspondences of 7 points (i.e. traditional) have higher probability of having lower epipolar error. Mentioning particular values from the figure, the probability of having hypothesis with an error less than 0.5px is approximately 80% for per-point and 40% for LAFs. This finding can be futher considered together with the LAF precision discussed earlier (see 4.1). It can be concluded that LAFs do not provide a good starting position for the F estimation and that the benchmark result is reasonable.

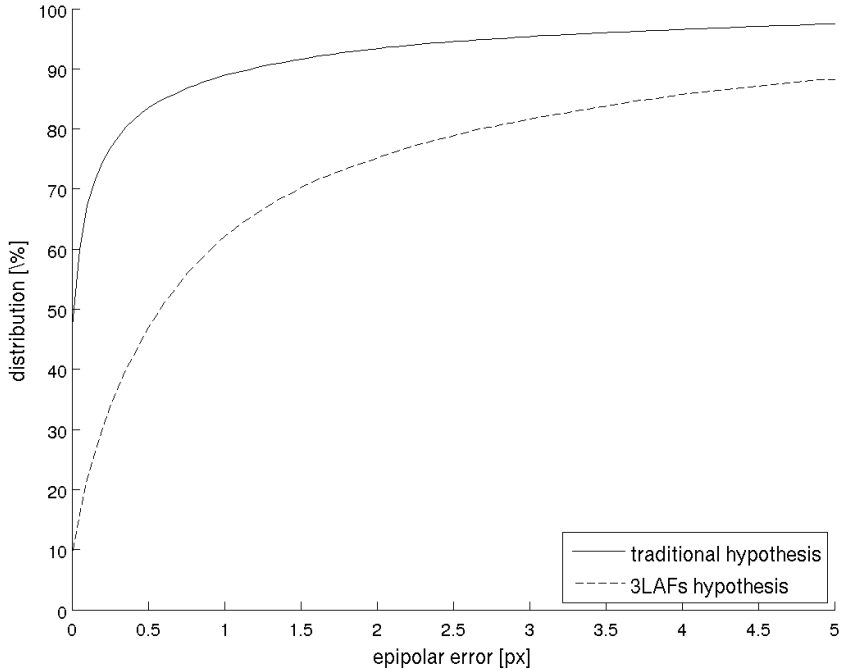


Figure 5.1. The comparison of Fs quality generated using traditional and 3LAF2F approach. The traditional approach is the estimation from 7 center correspondences. The 3LAF2F approach is represented with 9-point LSQ here. The empirical CDFs of average epipolar error show data gathered from 22000 hypotheses. The hypotheses were generated using inliers only.

In summary, the 3LAF2F approach was shown to be inferior to the traditional approach without optimization step. This fact has persisted since the original article on the problem, where similar conclusion was made. Thus, the use of local optimization is essential for the LR73L algorithm.

5.2. Involving the local optimization

The results of plain RANSAC motivated to use a local optimization. Thus, the LO step was incorporated to the algorithm.

There are more LO step possibilities as discussed in the theory section. The very time-proven possibility is to use inner RANSAC with iterative least squares [13]. This method is used by the state-of-the-art LO-RANSAC F estimator and provides excellent performance parameters. It is able to achieve high quality F, while using a low number of LSQ solver executions (e.g. 60). Consequently, it can run fast. Therefore, an implementation of such a local optimization was done. The implementation was done using best practices with respect to the 3LAF2F problem. For instance, consider the least number of inliers required to run. In the case of per-point method, at least 16 correspondences were required to run. Using this setting for the 3LAF2F problem will cause LO rarely to run for demanding images. This is caused by a deficiency of inliers (see 5.1). Thus, this value was adjusted to 9. Finally, the LO step was crafted and inserted to plain RANSAC algorithm.

5.2.1. Execution strategy for the LO step

When incorporating LO step to RANSAC, it is a question how to do it. The proved approach is to execute LO only on the best-so-far fundamental matrices. Aside from this, there are more options when to execute a local optimization.

The LO step does not have to be enabled from the beginning. That means, there is an iteration offset, where the LO step is disabled. For example, consider the first 20 iterations where even though the best-so-far F is found, the optimization is not executed. The intention of this method is to save time not optimizing too inferior hypotheses. Letting plain RANSAC to search for a better hypothesis during the first n iterations could yield higher quality optimization-worthy F. This F is then optimized and time is saved.

To see this theory in a practice an experiment was performed. Various offset settings ranging from 0 (i.e. optimize every best) to optimize only after plain RANSAC were evaluated. The 3LAF2F algorithm was run with each of this setting 1000 times on the whole dataset. Average computational and epipolar error was recorded.

The results are displayed in the figure 5.2. The curve was created interpolating measured datapoints. Its color denotes the offset. The experiment and curve generation was repeated several times. The epipolar error was proved similar for every repetition, however the time differs slightly. Anyway, the curve features remained the same, i.e. its decreasing tendency.

As the figure indicates, there is a straight way how to make the algorithm fastest or outputting the highest quality Fs. Using no offset and executing LO for each best-so-far causes a low number of iterations and the function to return very soon. The LO quits very quickly if called on a low number of inliers. No calculation can be done when no data available. This is the fastest option, but it offers the poorest quality.

On the other hand, if quality is the case, calling optimization only after a complete execution of plain RANSAC yields the highest quality option at the cost of time. The

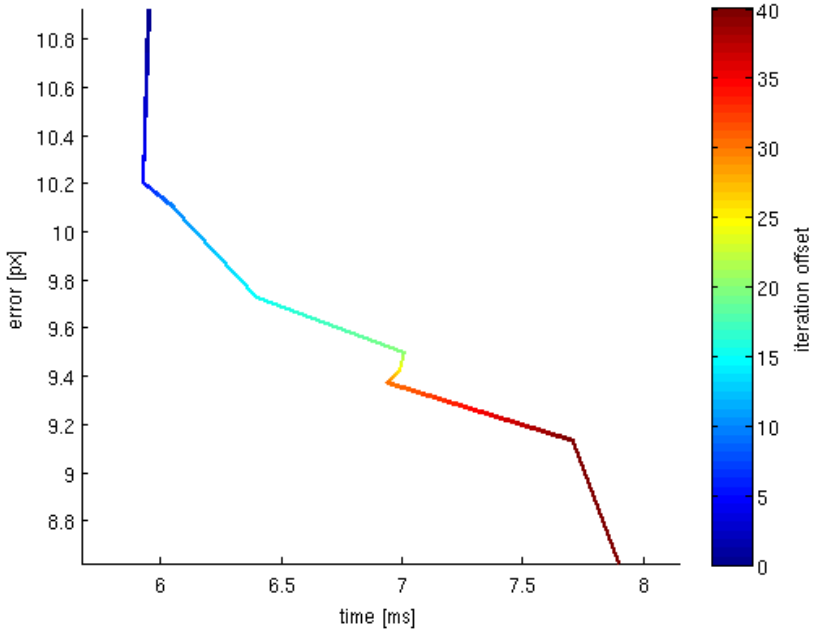


Figure 5.2. LO execution strategy. The graph displays the impact of when LO is executed on computational time and epipolar error. The LO step was permitted only from specified iteration and only for the best-so-far F matrices. The specified iteration is indicated with a color.

optimization step is provided with a more precise fundamental matrix compared with the previous case, i.e. the fastest option. This is how the success can be explained.

The options between the two extremes just discussed offer a trade-off between time and quality. Notice, how is epipolar error dependent on the iteration offset. The more offset the less error. Finally, the value of iteration offset was set to 20 as it is the midpoint. Additionally, the LO step is executed at least at the end if no optimization happened during a regular run.

5.2.2. Benchmarking

The benchmark from 5.1 was supplemented with LO-RANSAC F estimators. The conditions remained the same. That is, the estimators were run 1000 times on the whole dataset.

The selected results are presented in the table 5.2. The complete results are in the appendix B. It can be seen that optimization has a great impact on quality. In the plain benchmark, *LR7 3L* showed to be unusable due to its high epipolar error. The situation is different with LO enabled. The results for easy images are nearly the state-of-the-art in the meaning of quality and computational time. This can be observed in the first two rows (Brussels and Dresden) of the table. The situation is not that positive for more demanding image pairs with a lower number of inliers . The performance remains higher compared with LO disabled, but it is not sufficient to provide high quality F estimates. Consider results for booksh image where *ransacF.LO* (i.e. state-of-the-art LO-RANSAC F estimator) estimates Fs with average epipolar error below 2 pixels. The proposed *LR7 3L.LO* estimates Fs with error above 3 pixels, which is considered to be low quality F. The original *rEG3p.LO* is doing slightly better. This is caused by

its heavy local optimization, which is however infeasible due to its time consumption.

		ransacF	ransacF.LO	LR73L	LR73L.LO	rEG3p.LO
Brussels	inlier [%]	66.87 ± 5.95	77.56 ± 4.42	34.27 ± 5.88	76.16 ± 6.36	79.96 ± 0.59
	error [px]	0.96 ± 0.72	0.36 ± 0.20	6.55 ± 3.16	0.58 ± 1.05	0.22 ± 0.02
	time [ms]	1.42	8.49	1.30	10.00	604.80
	sample	80.42 ± 43.61	57.90 ± 27.05	97.97 ± 36.34	50.57 ± 32.26	18.95 ± 0.30
	LO	0.00 ± 0.00	1.00 ± 0.06	0.00 ± 0.00	1.00 ± 0.05	2.33 ± 0.95
Dresden	inlier [%]	85.69 ± 4.59	92.31 ± 2.23	52.73 ± 8.98	92.57 ± 0.49	92.22 ± 0.17
	error [px]	1.88 ± 3.24	0.51 ± 2.16	14.32 ± 7.85	0.21 ± 0.14	0.16 ± 0.04
	time [ms]	1.14	16.78	1.47	16.41	1588.65
	sample	10.32 ± 4.51	10.38 ± 4.64	27.07 ± 11.48	24.46 ± 9.51	4.00 ± 0.08
	LO	0.00 ± 0.00	1.00 ± 0.00	0.00 ± 0.00	1.00 ± 0.00	1.78 ± 0.70
Leuven2	inlier [%]	55.00 ± 3.93	57.90 ± 2.51	34.44 ± 6.78	49.07 ± 8.38	51.15 ± 8.29
	error [px]	6.55 ± 9.54	2.84 ± 5.82	71.63 ± 66.18	27.22 ± 34.46	22.70 ± 30.58
	time [ms]	4.51	10.50	1.29	4.67	322.65
	sample	891.03 ± 339.00	364.74 ± 188.80	211.07 ± 67.90	97.28 ± 71.41	1015.10 ± 979.55
	LO	0.00 ± 0.00	2.42 ± 1.17	0.00 ± 0.00	1.35 ± 0.66	1.00 ± 0.06
booksh	inlier [%]	67.74 ± 5.34	68.85 ± 5.36	49.28 ± 7.22	59.68 ± 8.38	66.93 ± 13.21
	error [px]	1.69 ± 2.40	1.56 ± 2.25	6.55 ± 4.22	3.49 ± 3.71	2.37 ± 4.11
	time [ms]	0.83	5.52	0.33	4.65	296.99
	sample	107.92 ± 52.32	90.82 ± 43.60	43.50 ± 15.46	36.11 ± 13.19	96.45 ± 23.31
	LO	0.00 ± 0.00	1.41 ± 0.67	0.00 ± 0.00	1.02 ± 0.14	2.17 ± 0.84

Table 5.2. Benchmark results. The solvers with ".LO" suffix has local optimization enabled.

It can be concluded that the proposed LR73L algorithm achieves nearly the state-of-the-art results for easy images. The performance is worse for images with inlier rate below 70%.

6. Conclusion

In this thesis, the problem of fundamental matrix estimation from 3 correspondences of local affine frames was addressed. The LR73L algorithm was proposed as the solution to the problem. The algorithm is based on LO-RANSAC, which was composed of experimentally evaluated components with respect to the problem being solved. Those components were: sampling, hypothesis generator and LO step.

The algorithm is targeted as the successor to the algorithm presented by Chum et al. in [4], i.e. the original algorithm. The LR73L algorithm uses the 7-point algorithm to generate hypotheses. This approach was shown to be faster and producing higher quality hypotheses than hypotheses from 9 points used by the original algorithm. Additionally, the proposed algorithm uses the largest LAFs for hypothesis generation. There are 3LAFs selected at random during the sampling. This yields 9 point correspondences and only 7 of them are needed. It was shown that the correspondences of LAFs center points are the most precise. Therefore, they are always included to the 7-point sample. The LAFs side points which are the most distant from their centers are used to complete the remaining 4 correspondences of the sample. This approach was shown superior to 7 random point correspondences. The hypothesis verification in LR73L is considering only center points compared with the original algorithm which was taking into account all 3 points of LAF. The center points are the most precise, their support is fast to compute and it is time-proven by traditional F estimation. The LO step used in the original algorithm was too time demanding to fit the requirements and therefore a different one was used. The used LO step is an implementation of local optimization proposed by Lebeda et al. in [14]. The algorithm was reimplemented with minor changes with respect to the 3LAF2F problem, i.e. different execution offset, different minimal number of required correspondences.

The LR73L algorithm was compared with the state-of-the-art LO-RANSAC F estimator by Lebeda et al. [14] and the original algorithm by Chum et al. [4]. It was shown that quality and time achieved by the LR73L algorithm compare well with the state-of-the-art for image pairs with high fraction of inliers (i.e. more than 70%). The computational time was proved to be a strong point of the algorithm surpassing the old algorithm by an order of magnitude. Quality of fundamental matrices estimated by LR73L is lower when compared with the original algorithm. This decrease is caused by the simpler and faster LO step used in LR73L. However quality was sacrificed, this step was unavoidable to reach computational time of the state-of-the-art.

The state-of-the-art was touched by the proposed LR73L algorithm. As it has been stated, the LR73L algorithm can provide equivalent performance in some cases. This is a newly achieved success in the domain of the 3LAF2F problem, because the original algorithm was not able to compete due to its time consumption. Despite this success, the performance is insufficient to set a new standard for fundamental matrix estimation. The traditional algorithm is still a better choice for general use.

Aside from the main problem, a method to save computational costs on support calculation was suggested. The method is applicable when solving for F using SVD. The method was experimentally evaluated followed by a prototype implementation and its benchmark. A noticeable time savings were observed. The method was measured to be image dependent. To solve this issue, a user controllable parameter was introduced.

Bibliography

[1] Tania M. Centeno, Heitor S. Lopes, Marcelo K. Felisberto, and Lúcia V. R. Arruda. Object detection for computer vision using a robust genetic algorithm. In Franz Rothlauf, Jürgen Branke, Stefano Cagnoni, DavidWolfe Corne, Rolf Drechsler, Yaochu Jin, Penousal Machado, Elena Marchiori, Juan Romero, GeorgeD. Smith, and Giovanni Squillero, editors, *Applications of Evolutionary Computing*, volume 3449 of *Lecture Notes in Computer Science*, pages 284–293. Springer Berlin Heidelberg, 2005. 16

[2] Ondřej Chum. *Two-view Geometry Estimation by Random Sample and Consensus*. Phd thesis, Center for Machine Perception, K13133 FEE Czech Technical University, Prague, Czech Republic, September 2005. 5, 16

[3] Ondřej Chum, Jiří Matas, and Josef Kittler. Locally optimized RANSAC. In J. van Leeuwen G. Goos, J. Hartmanis, editor, *DAGM 2003: Proceedings of the 25th DAGM Symposium*, number 2781 in LNCS, pages 236–243, Heidelberger Platz 3, 14197, Berlin, Germany, September 2003. Springer-Verlag. 22, 23

[4] Ondřej Chum, Jiří Matas, and Štěpán Obdržálek. Epipolar geometry from three correspondences. In Ondřej Drbohlav, editor, *Computer Vision — CVWW'03 : Proceedings of the 8th Computer Vision Winter Workshop*, pages 83–88, Prague, Czech Republic, February 2003. Czech Pattern Recognition Society. vii, 5, 9, 27, 34, 39

[5] Ondřej Chum, Jiří Matas, and Štěpán Obdržálek. Enhancing RANSAC by generalized model optimization. In Ki-Sang Hong and Zhengyou Zhang, editors, *Proc. of the Asian Conference on Computer Vision (ACCV)*, volume 2, pages 812–817, Seoul, Korea South, January 2004. Asian Federation of Computer Vision Societies. 5, 9

[6] Richard J. M. den Hollander and Alan Hanjalic. A combined ransac-hough transform algorithm for fundamental matrix estimation, 2007. 16

[7] Martin A. Fischler and Robert 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. 16

[8] Ronald A. Fisher and Frank Yates. *Statistical tables for biological, agricultural and medical research*. 1948. 17

[9] Richard I. Hartley. In defense of the eight-point algorithm. 1997. 9, 19

[10] Richard I. Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, second edition, 2004. 9, 21

[11] Donald Knuth. *The Art of Computer Programming vol. 2*. 1969. 17

[12] Karel Lebeda. Collection of datasets for computer vision. `http://cmp.felk.cvut.cz/data/geometry2view/index.xhtml`. 42

[13] Karel Lebeda. Robust sample consensus. MSc Thesis CTU–CMP–2013–01, Center for Machine Perception, K13133 FEE Czech Technical University, Prague, Czech Republic, January 2013. vii, 6, 18, 20, 22, 24, 36

[14] Karel Lebeda, Jiří Matas, and Ondřej Chum. Fixing the locally optimized ransac. In Richard Bowden, John Collomosse, and Krystian Mikolajczyk, editors, *Proceedings of the British Machine Vision Conference*, pages 1013–1023, London, UK, September 2012. BMVA. 5, 21, 22, 23, 39

[15] Jiří Matas, Ondřej Chum, Martin Urban, and Tomáš Pajdla. Robust wide baseline stereo from maximally stable extremal regions. In Paul L. Rosin and David Marshall, editors, *Proceedings of the British Machine Vision Conference*, volume 1, pages 384–393, London, UK, September 2002. BMVA. 5, 11

[16] Jiří Matas, Štěpán Obdržálek, and Ondřej Chum. Local affine frames for wide-baseline stereo. In R. Kasturi, D. Laurendeau, and Suen C., editors, *ICPR 02: Proceedings 16th International Conference on Pattern Recognition*, volume 4, pages 363–366, CA 90720-1314, Los Alamitos, US, August 2002. IEEE Computer Society. 5

[17] Andrej Mikulik, Jiří Matas, Michal Perdoch, and Ondřej Chum. Construction of precise local affine frames. In Roy Sterritt, editor, *ICPR'2010: Proceedings of the 20th International Conference on Pattern Recognition*, pages 3565–3569, 10662 Los Vaqueros Circle, Los Alamitos, USA, August 2010. IEEE Computer Society. 5

[18] Štěpán Obdržálek. *Object Recognition Using Local Affine Frames*. Phd thesis, Center for Machine Perception, K13133 FEE, Czech Technical University, Prague, Czech Republic, April 2007. 5, 11, 12, 13

[19] Štěpán Obdržálek and Jiří Matas. On the stability of local affine frames for the correspondence problem. In Allan Hanbury and Horst Bischof, editors, *Proceedings of the Computer Vision Winter Workshop 2005 (CVWW'05)*, pages 3–12, Wien, Austria, February 2005. PRIP TU Wien. 5, 12

[20] Philip H. S. Torr and Zisserman Andrew. MLESAC: A new robust estimator with application to estimating image geometry. *Computer Vision and Image Understanding*, 78:2000, 2000. 19

[21] Lloyd N. Trefethen and David Bau. *Numerical Linear Algebra*. Society for Industrial and Applied Mathematics, first edition, 1997. 30

[22] Andrew Zisserman and Relja Arandjelovic. Computing the affine fundamental matrixfrom two ellipses. 2010. 9

A. Dataset

Lebeda's dataset[12] for epipolar geometry was adopted including his ground truth correspondences. As stated by Lebeda, the images were gathered from various sources. The origin can be found at the webpage, which is listed in the bibliography. The used image pairs are listed next, including their names and resolutions.

image name width x height	Brussels 680x1024	Dresden 1024x683	Kyoto 2592x1944	Leuven1 1024x683
				
				

Leuven2 1024x683	booksh 768x576	box 1024x768	castle 768x576	corr 512x512
				
				

dino2 480x640	graff 800x640	head 1408x1056	kampa 800x543	leafs 1600x1200
				
				

plant
576x768

rotunda
1024x683

shout
768x576

temple2
480x640

valbonne
768x512

wall
2272x1704

wash
768x576

zoom
1024x768

B. Experimental results

Once the algorithm was implemented a need for benchmarking required. However there is existing benchmarking framework from a previous work, it does not meet the requirement to compare various F estimators, i.e. to compare 3LAF2F approach with point correspondence approach. Thus, a benchmarking framework was designed and implemented. The quantities studied are described next.

inliers The estimated F is used to decide which tentative correspondences comply with this F. The outcome is a fraction of inliers to a number of TCs (in %). An inliers output of a benchmarked function is ignored, only the outputted F is considered.

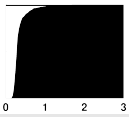
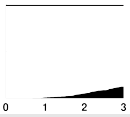
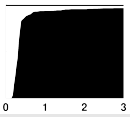
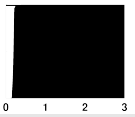
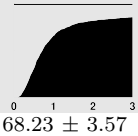
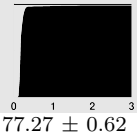
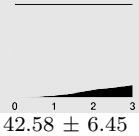
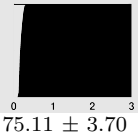
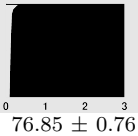
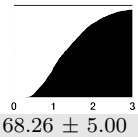
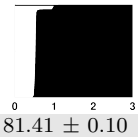
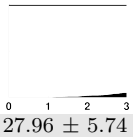
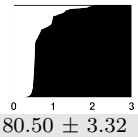
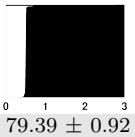
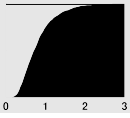
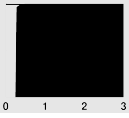
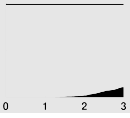
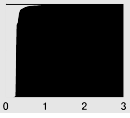
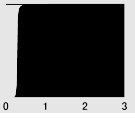
error This is average epipolar error in pixels calculated using Sampsons's error term.

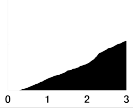
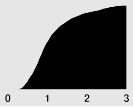
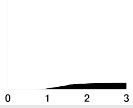
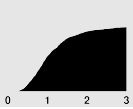
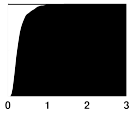
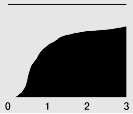
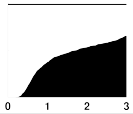
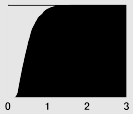
time The average run-time of a single run.

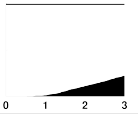
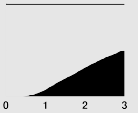
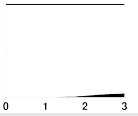
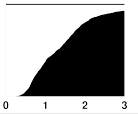
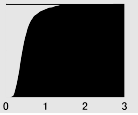
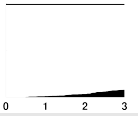
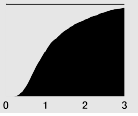
sample Number of samples, i.e. number of RANSAC iterations done.

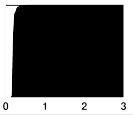
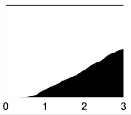
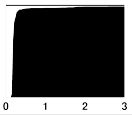
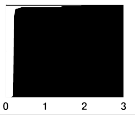
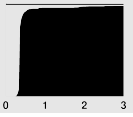
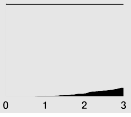
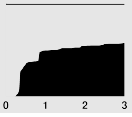
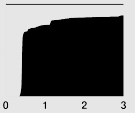
LO Number of LO steps executed. Note that, the relation between this and run-time may not be proportional. The LO step can quit early, while increasing the statistics.

error$_{cdf}$ This graph illustrated the cdf of average epipolar error. It gives a notion about algorithms stability. The horizontal axis is an average epipolar error (average for a single run) in pixels. It spans $[0, 3]$ interval, the rest is not shown. The vertical axis is a distribution, i.e. probability of having given or less error.

		ransacF	**ransacF.LO**	**LR73L**	**LR73L.LO**	**rEG3p.LO**
Brussels	inlier [%]	66.87 ± 5.95	77.56 ± 4.42	34.27 ± 5.88	76.16 ± 6.36	79.96 ± 0.59
	error [px]	0.96 ± 0.72	0.36 ± 0.20	6.55 ± 3.16	0.58 ± 1.05	0.22 ± 0.02
	time [ms]	1.42	8.49	1.30	10.00	604.80
	sample	80.42 ± 43.61	57.90 ± 27.05	97.97 ± 36.34	50.57 ± 32.26	18.95 ± 0.30
	LO	0.00 ± 0.00	1.00 ± 0.06	0.00 ± 0.00	1.00 ± 0.05	2.33 ± 0.95
	error$_{cdf}$					
Dresden	inlier [%]	85.69 ± 4.59	92.31 ± 2.23	52.73 ± 8.98	92.57 ± 0.49	92.22 ± 0.17
	error [px]	1.88 ± 3.24	0.51 ± 2.16	14.32 ± 7.85	0.21 ± 0.14	0.16 ± 0.04
	time [ms]	1.14	16.78	1.47	16.41	1588.65
	sample	10.32 ± 4.51	10.38 ± 4.64	27.07 ± 11.48	24.46 ± 9.51	4.00 ± 0.08
	LO	0.00 ± 0.00	1.00 ± 0.00	0.00 ± 0.00	1.00 ± 0.00	1.78 ± 0.70
	error$_{cdf}$					
Kyoto	inlier [%]	68.23 ± 3.57	77.27 ± 0.62	42.58 ± 6.45	75.11 ± 3.70	76.85 ± 0.76
	error [px]	1.51 ± 0.84	0.58 ± 0.10	11.68 ± 8.51	0.75 ± 0.47	0.54 ± 0.17
	time [ms]	2.09	11.66	1.40	11.02	1058.02
	sample	53.58 ± 20.66	45.12 ± 11.73	49.66 ± 20.20	36.72 ± 15.98	6.02 ± 0.15
	LO	0.00 ± 0.00	1.00 ± 0.03	0.00 ± 0.00	1.00 ± 0.00	1.98 ± 0.80
	error$_{cdf}$					
Leuven1	inlier [%]	68.26 ± 5.00	81.41 ± 0.10	27.96 ± 5.74	80.50 ± 3.32	79.39 ± 0.92
	error [px]	0.84 ± 0.43	0.29 ± 0.02	10.57 ± 9.28	0.36 ± 0.81	0.32 ± 0.06
	time [ms]	1.23	8.77	2.61	7.43	401.30
	sample	63.51 ± 31.04	47.49 ± 17.88	209.71 ± 88.83	66.53 ± 64.59	15.30 ± 2.86
	LO	0.00 ± 0.00	1.00 ± 0.00	0.00 ± 0.00	1.00 ± 0.00	1.52 ± 0.68
	error$_{cdf}$					

		ransacF	ransacF.LO	LR73L	LR73L.LO	rEG3p.LO
Leuven2	inlier [%]	47.43 ± 2.76	52.93 ± 1.71	27.76 ± 4.15	44.92 ± 7.78	45.68 ± 6.65
	error [px]	6.55 ± 9.54	2.84 ± 5.82	71.63 ± 66.18	27.22 ± 34.46	22.70 ± 30.58
	time [ms]	4.51	10.50	1.29	4.67	322.65
	sample	891.03 ± 339.00	364.74 ± 188.80	211.07 ± 67.90	97.28 ± 71.41	1015.10 ± 979.55
	LO	0.00 ± 0.00	2.42 ± 1.17	0.00 ± 0.00	1.35 ± 0.66	1.00 ± 0.06
	$error_{cdf}$					
booksh	inlier [%]	64.13 ± 5.25	65.08 ± 5.19	45.00 ± 6.09	55.47 ± 7.57	62.26 ± 12.67
	error [px]	1.69 ± 2.40	1.56 ± 2.25	6.55 ± 4.22	3.49 ± 3.71	2.37 ± 4.11
	time [ms]	0.83	5.52	0.33	4.65	296.99
	sample	107.92 ± 52.32	90.82 ± 43.60	43.50 ± 15.46	36.11 ± 13.19	96.45 ± 23.31
	LO	0.00 ± 0.00	1.41 ± 0.67	0.00 ± 0.00	1.02 ± 0.14	2.17 ± 0.84
	$error_{cdf}$					
box	inlier [%]	79.31 ± 2.57	83.25 ± 0.59	52.67 ± 9.73	83.18 ± 0.44	83.37 ± 0.24
	error [px]	36.74 ± 17.66	42.71 ± 13.99	48.06 ± 23.47	43.11 ± 13.15	46.81 ± 5.92
	time [ms]	0.58	8.28	0.60	8.36	726.27
	sample	15.41 ± 4.13	15.25 ± 4.04	31.14 ± 15.04	25.92 ± 11.07	6.14 ± 0.41
	LO	0.00 ± 0.00	1.00 ± 0.00	0.00 ± 0.00	1.00 ± 0.00	2.01 ± 0.82
	$error_{cdf}$					
castle	inlier [%]	63.83 ± 4.48	74.53 ± 1.05	39.58 ± 6.09	73.67 ± 2.25	74.32 ± 0.48
	error [px]	4.62 ± 7.60	0.42 ± 0.75	16.94 ± 11.64	0.82 ± 2.32	0.31 ± 0.25
	time [ms]	1.71	7.07	0.94	6.86	500.24
	sample	91.78 ± 36.97	54.69 ± 17.88	63.62 ± 22.92	42.38 ± 21.31	20.10 ± 0.59
	LO	0.00 ± 0.00	1.00 ± 0.00	0.00 ± 0.00	1.00 ± 0.00	1.98 ± 0.89
	$error_{cdf}$					
corr	inlier [%]	72.91 ± 5.12	72.31 ± 4.88	47.60 ± 5.93	49.62 ± 5.62	80.32 ± 2.82
	error [px]	0.32 ± 0.19	0.33 ± 0.21	1.66 ± 1.03	1.83 ± 1.15	0.25 ± 0.10
	time [ms]	0.69	1.88	0.44	4.82	227.49
	sample	36.59 ± 16.09	37.46 ± 16.09	32.86 ± 10.29	31.51 ± 9.67	31.02 ± 4.10
	LO	0.00 ± 0.00	1.09 ± 0.34	0.00 ± 0.00	1.10 ± 0.33	2.37 ± 0.98
	$error_{cdf}$					
dino2	inlier [%]	71.72 ± 3.05	75.77 ± 8.46	48.40 ± 5.56	60.62 ± 14.25	39.74 ± 1.81
	error [px]	1.77 ± 1.76	1.75 ± 2.94	4.21 ± 3.86	3.58 ± 4.17	9.18 ± 0.44
	time [ms]	0.37	3.35	0.30	2.84	178.21
	sample	33.53 ± 11.29	34.17 ± 10.78	34.38 ± 10.94	32.59 ± 10.77	815.41 ± 808.09
	LO	0.00 ± 0.00	1.01 ± 0.13	0.00 ± 0.00	1.07 ± 0.27	1.00 ± 0.00
	$error_{cdf}$					
graff	inlier [%]	64.52 ± 2.99	74.46 ± 1.27	43.65 ± 7.07	73.04 ± 1.69	73.36 ± 2.64
	error [px]	2.04 ± 1.35	2.03 ± 1.41	3.59 ± 1.67	2.57 ± 1.29	2.43 ± 1.26
	time [ms]	0.97	6.62	0.53	6.25	799.36
	sample	75.59 ± 24.71	60.54 ± 19.84	49.26 ± 20.59	35.88 ± 16.05	52.12 ± 10.08
	LO	0.00 ± 0.00	1.00 ± 0.00	0.00 ± 0.00	1.00 ± 0.00	3.26 ± 1.21
	$error_{cdf}$					
head	inlier [%]	76.37 ± 5.16	86.37 ± 0.05	40.38 ± 7.08	83.65 ± 9.54	82.84 ± 4.58
	error [px]	0.54 ± 0.24	0.28 ± 0.02	10.05 ± 16.45	1.85 ± 9.48	0.34 ± 0.36
	time [ms]	0.45	5.49	0.73	5.18	215.94
	sample	26.16 ± 12.69	25.98 ± 12.35	63.33 ± 25.07	41.92 ± 20.70	8.39 ± 5.75
	LO	0.00 ± 0.00	1.00 ± 0.00	0.00 ± 0.00	1.01 ± 0.08	1.37 ± 0.57
	$error_{cdf}$					

		ransacF	ransacF.LO	LR73L	LR73L.LO	rEG3p.LO
kampa	inlier [%]	54.68 ± 3.26	61.41 ± 4.11	37.02 ± 4.99	56.00 ± 6.08	57.79 ± 4.91
	error [px]	11.40 ± 8.98	8.36 ± 7.29	21.70 ± 12.47	11.18 ± 11.98	3.45 ± 5.68
	time [ms]	2.32	7.41	0.69	4.78	341.45
	sample	267.72 ± 90.04	131.94 ± 67.62	76.86 ± 25.81	47.25 ± 24.82	140.74 ± 105.88
	LO	0.00 ± 0.00	1.41 ± 0.65	0.00 ± 0.00	1.01 ± 0.11	2.21 ± 0.89
	$error_{cdf}$					
leafs	inlier [%]	59.04 ± 3.78	71.91 ± 0.00	35.11 ± 6.08	69.27 ± 7.95	68.44 ± 3.85
	error [px]	4.06 ± 3.13	1.34 ± 0.03	22.59 ± 18.31	2.97 ± 7.80	1.66 ± 1.32
	time [ms]	1.40	9.54	0.88	7.05	175.78
	sample	165.00 ± 69.11	74.24 ± 47.87	101.64 ± 42.51	54.62 ± 35.68	32.46 ± 29.62
	LO	0.00 ± 0.00	1.01 ± 0.13	0.00 ± 0.00	1.05 ± 0.23	1.84 ± 0.83
	$error_{cdf}$					
plant	inlier [%]	60.52 ± 2.59	62.95 ± 6.54	45.29 ± 4.66	53.16 ± 9.48	49.91 ± 8.62
	error [px]	19.76 ± 17.31	19.44 ± 19.30	23.44 ± 22.76	46.68 ± 45.38	6.07 ± 0.97
	time [ms]	0.95	3.77	0.33	3.30	431.94
	sample	132.12 ± 41.54	117.68 ± 45.51	41.04 ± 11.05	37.60 ± 11.38	1778.47 ± 1769.33
	LO	0.00 ± 0.00	1.34 ± 0.58	0.00 ± 0.00	1.04 ± 0.21	1.00 ± 0.03
	$error_{cdf}$					
rotunda	inlier [%]	79.89 ± 5.76	88.05 ± 0.83	39.51 ± 6.63	87.79 ± 1.79	86.77 ± 1.89
	error [px]	0.88 ± 0.52	0.35 ± 0.08	9.43 ± 7.31	0.37 ± 0.28	0.34 ± 0.07
	time [ms]	0.35	5.94	0.69	5.56	249.93
	sample	21.95 ± 12.59	20.86 ± 11.01	68.67 ± 27.97	42.63 ± 22.18	4.74 ± 2.13
	LO	0.00 ± 0.00	1.00 ± 0.00	0.00 ± 0.00	1.00 ± 0.00	1.38 ± 0.56
	$error_{cdf}$					
shout	inlier [%]	76.70 ± 4.29	79.91 ± 3.33	54.42 ± 8.51	77.71 ± 4.02	83.17 ± 1.61
	error [px]	1.57 ± 1.11	0.90 ± 0.71	4.33 ± 2.59	0.83 ± 0.47	0.40 ± 0.11
	time [ms]	0.31	5.61	0.26	5.39	331.75
	sample	23.55 ± 10.76	23.68 ± 9.92	25.58 ± 10.39	23.92 ± 9.21	9.28 ± 1.26
	LO	0.00 ± 0.00	1.00 ± 0.00	0.00 ± 0.00	1.00 ± 0.00	1.70 ± 0.72
	$error_{cdf}$					
temple2	inlier [%]	60.97 ± 3.53	71.43 ± 2.07	36.25 ± 4.68	60.91 ± 10.47	68.76 ± 3.77
	error [px]	0.52 ± 0.29	0.29 ± 0.15	3.89 ± 1.52	1.55 ± 1.82	0.34 ± 0.47
	time [ms]	2.11	7.60	0.85	7.66	479.82
	sample	123.24 ± 47.39	60.46 ± 26.13	77.45 ± 23.64	47.47 ± 25.72	96.82 ± 43.95
	LO	0.00 ± 0.00	1.03 ± 0.16	0.00 ± 0.00	1.00 ± 0.04	2.41 ± 0.87
	$error_{cdf}$					
valbonne	inlier [%]	71.90 ± 4.33	75.83 ± 6.84	44.49 ± 5.39	56.96 ± 12.40	44.68 ± 9.75
	error [px]	14.10 ± 8.43	14.15 ± 7.06	41.91 ± 21.04	35.68 ± 22.95	22.78 ± 12.81
	time [ms]	0.37	5.86	0.35	3.22	50.74
	sample	39.80 ± 18.64	39.91 ± 17.56	46.21 ± 15.25	41.26 ± 15.53	103.47 ± 43.74
	LO	0.00 ± 0.00	1.03 ± 0.18	0.00 ± 0.00	1.12 ± 0.37	1.33 ± 0.50
	$error_{cdf}$					
wall	inlier [%]	79.22 ± 5.49	90.68 ± 9.13	47.12 ± 7.71	87.56 ± 13.43	83.48 ± 10.60
	error [px]	1.31 ± 0.85	1.01 ± 8.77	31.27 ± 36.76	5.43 ± 20.92	1.26 ± 3.65
	time [ms]	0.32	5.59	0.42	5.82	211.96
	sample	20.92 ± 10.70	20.92 ± 10.68	38.94 ± 15.53	32.44 ± 13.01	8.95 ± 7.19
	LO	0.00 ± 0.00	1.00 ± 0.00	0.00 ± 0.00	1.04 ± 0.20	1.65 ± 0.73
	$error_{cdf}$					

		ransacF	ransacF.LO	LR73L	LR73L.LO	rEG3p.LO
wash	inlier [%]	79.01 ± 6.02	88.71 ± 0.00	44.05 ± 9.27	86.96 ± 8.09	86.95 ± 8.42
	error [px]	0.71 ± 0.38	0.22 ± 0.04	4.22 ± 4.64	0.45 ± 1.80	0.80 ± 7.46
	time [ms]	0.34	6.70	0.55	4.80	194.05
	sample	25.76 ± 15.64	25.21 ± 13.66	54.00 ± 22.61	39.35 ± 19.84	61.96 ± 68.86
	LO	0.00 ± 0.00	1.00 ± 0.03	0.00 ± 0.00	1.04 ± 0.20	1.36 ± 0.57
	error_{cdf}					
zoom	inlier [%]	57.06 ± 5.34	66.89 ± 3.10	33.37 ± 4.98	53.02 ± 12.02	60.92 ± 10.68
	error [px]	1.23 ± 1.28	0.65 ± 1.62	18.30 ± 20.43	15.56 ± 28.61	5.27 ± 17.23
	time [ms]	1.87	7.43	0.79	4.43	195.66
	sample	274.47 ± 149.67	122.90 ± 97.55	108.45 ± 36.39	62.00 ± 35.09	111.85 ± 110.51
	LO	0.00 ± 0.00	1.49 ± 0.79	0.00 ± 0.00	1.06 ± 0.25	1.84 ± 0.81
	error_{cdf}					

C. Content of the attached CD

This work is accompanied with the CD. The CD contains sources to all work done during this thesis. Selected features of its contents are listed next. The content features are marked bold, following with a path in a brackets and a brief description.

Source code (/wbs.tar/wbs/src/ransac/ranF3laf.c) - the source code of the implementation of the algorithm proposed in this work.

Experiments (/wbs.tar/wbs/exper/) - contains scripts to execute experiments presented in this thesis.

Benchmark results (/wbs.tar/wbs/bench/) - the results of benchmarks performed are stored here. The execution script is included.

Benchmarking framework (/wbs.tar/wbs/wbs-demo/ransac/bench/) - the benchmarking framework used is stored here. This includes the benchmark performing scripts as well as the result pdf outputting script.

Dataset (/wbs.tar/wbs/imageset/) - the image pairs used in the experimental evaluation.

Thesis (/matejra4_thesis.pdf) - this thesis in a pdf file.

Thesis sources (/matejra4_thesis.tar) - source codes of the written thesis, i.e. latex code and images.