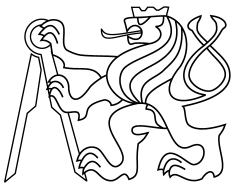
CENTER FOR
MACHINE PERCEPTION



CZECH TECHNICAL
UNIVERSITY IN PRAGUE

PhD THESIS

ISSN 1213-2365

Stable wave detector and tracker

Doctoral thesis

Jan Dupač

dupac@rsdynamics.com

CTU–CMP–2011–02

February 24, 2011

Available at
ftp://cmp.felk.cvut.cz/pub/cmp/articles/dupac/StableWavePhDthesisDupac2011.pdf

Thesis Advisor: Prof. Ing. Václav Hlaváč, CSc.

The presented research was supported by the Czech Ministry of Education under Project 1M0567, by EC projects FP7-ICT-247870 NIFTi, FP7-ICT-247525 HUMAVIPS and by the PhD candidate's employer, the company RS-Dynamics.

Research Reports of CMP, Czech Technical University in Prague, No. 2, 2011

Published by

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

Stable wave detector and tracker

Doctoral Thesis

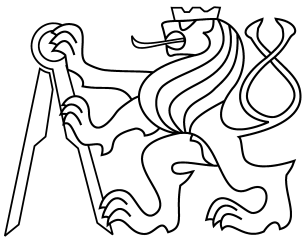
presented to the Faculty of Electrical Engineering of the Czech Technical University in Prague in partial fulfilment of the requirements for the Ph.D. degree in Study Programme No. P 2612 – Electrical Engineering and Information Technology, branch No. 3902V035 – Artificial Intelligence and Biocybernetics, by

Ing. Jan Dupač

February 2010

Thesis Advisor

Prof. Ing. Václav Hlaváč, CSc.



Center for Machine Perception
Department of Cybernetics
Faculty of Electrical Engineering
Czech Technical University in Prague
Karlovo náměstí 13, 121 35 Prague 2
Czech Republic
phone: +420 224 357 465
fax: +420 224 357 385
http://cmp.felk.cvut.cz

Research Reports of CMP, Czech Technical University in Prague, **No. 02, 2011**
ISSN 1213-2365
Published by
Center for Machine Perception, Department of Cybernetics
Faculty of Electrical Engineering, Czech Technical University in Prague
Technická 2, 166 27 Prague 6, Czech Republic
fax: +420 224 357 385, phone: +420 224 357 637
`http://cmp.felk.cvut.cz`

Abstract

A new salient semantics-less blob detector in the intensity named the Stable Wave Detector (SWD) is suggested. The detector proved to be particularly very useful in tracking a target in a videosequence. The thesis explains and experimentally verifies both the SWD detector and its use in tracking.

SWD is a multiscale blob detector in the intensity image. The targets are blobs which correspond to local maxima/minima of the image intensities, i.e. to positive and negative peaks. SWD belongs to a group of interest point/regions-like operators aiming at detecting repeatedly distinguished entities regardless of their meaning. Precision, robustness and speed of the SWD detector are main issues. The detector is based on the phase of the first harmonic wave in the window sliding across the image. Its localization is a result of an integral transformation rather than a usual derivative. Thus, the blob detector is inherently robust to noise and blur in the image.

The SWD provides a subpixel localization of blobs together with the estimate of its precision, the measure of the strength/significance and the estimate of the size/scale for each blob. Detected blobs are further hierarchically decomposed according to scale. Blobs at each hierarchy level have some minimal distance given by the period of the detector. Both grouping of detected blobs and their minimal mutual distance can facilitate matching of corresponding scene points. The properties of the SWD detector itself did not beat the state of the art considerably.

The situation has improved dramatically when the SWD detector was employed in tracking a target, i.e. in establishing a point to point (blob to blob) correspondences across the videosequence. The outcome is an ultrafast tracker able perform better than the state-of-the-art. The tracker takes two inputs: (a) the location of the blob in the previous frame which serves as the estimate for the location in the current frame and (2) the current frame in which the blob should be found. The problem of tracking is considered in several levels of abstraction. At a low level, correspondences between individual points are established at different scales independently. The high level of tracking system incorporates a model of the movement and a model of the target. The trade off between the precision and the range of trackable displacement at different scales must be solved. It is shown how the movement and target model can be used in a coarse to fine approach solving the trade off between the precision and the range.

The SWD detector and the tracker utilizing it were extensively tested experimentally. Test results are presented in the thesis. The implementation of the involved algorithms in C language is provided to the research community.

Acknowledgments

I would like to express my thanks to my supervisor Professor Václav Hlaváč for introducing me to the academic research during my master studies and for advising me since. I appreciate interactions and discussion with many colleagues from the Center for Machine Perception of the Department of Cybernetics at the Czech Technical University. I learned from them a lot.

The idea of applying the key idea of this thesis, the Stable Wave, to the tracking in a videosequence originates in discussions with Professor Jiří Matas. I like to thank also to Karel Zimmermann and Michal Perdoch with whom we considered different aspects of tracking.

The diploma thesis by Martin Pejčoch supervised by me and defended in January 2009 was devoted to extensive experiments with the stable wave. I thank to Martin Pejčoch for these efforts.

Professor Mirko Navara helped me with expressing the Stable Wave idea more clearly mathematically. Tomáš Pajdla assisted me by putting some of my dreams to a doable and realistic ground. Vojtěch Franc, my fellow student and friend from our master studies, has been inspiring and helping me since.

There is a long way for an idea to be clearly explained. For me, it is much more easy to implement the idea than to clearly express it to be understandable by other. Therefore, I would like to express my thanks to Professor Václav Hlaváč, Professor Mirko Navara and Vojtěch Franc who patiently read my thesis and helped me to clarify my ideas.

I like to express my thanks to Jiří Bláha, the director and founder of the technological company RS Dynamics, in which I learned R&D craft already as a student. He has been supporting my part-time PhD research since I have been working full time in the company.

My warm thanks go to my wife Naoko who has exhibited tremendous patience and understanding while I focussed my attention to the research published in this thesis. And I would like to apologize to my daughter Jana who will be one year old in this April that I could not find free time to spend with here. I would like to thank my parents who helped us a lot in this bussy time.

The presented research was supported by the Czech Ministry of Education under Project 1M0567, by EC projects FP7-ICT-247870 NIFTi, FP7-ICT-247525 HUMAVIPS and by the PhD candidate's employer, the company RS-Dynamics.

Typeset in LaTeX, a macro system for the TeX document preparation system.

Contents

Notation

Symbols

$\mathbb{R}$	set of real numbers
$\mathbb{R}^n$	n-dimensional Euclidean space
$\mathbb{Z} = \{\ldots, -1, 0, 1, 2, \ldots\}$	set of all integers
$\mathbb{Z}^+ = \{0, 1, 2, 3, \ldots\}$	set of all non-negative integers
$\mathbb{N} = \{1, 2, 3, \ldots\}$	set of all natural numbers
$\langle a, b\rangle = \{x \mid a \leq x \leq b\}$	closed interval
$(a, b) = \{x \mid a < x < b\}$	open interval

Abbreviations

1D	one dimension, one dimensional
2D	two dimensions, two dimensional
CNW	Consistent Neighbouring Window
FT	Fourier transform
FFT	Fast Fourier transform
KLT	Kanade-Lucas-Tomasi tracker
SLAM	Simultaneous Localization and Mapping
SWD	Stable Wave detector
ZSP	Zero Shift Point

1 Motivation and problem formulation

1.1 Motivation

The author of this thesis has been a member of the development team in a small Czech R&D company RS Dynamics, which designs and produces special and highly precise measuring instruments for the environment protection, security, etc. One of main products is the ECOPROBE series, which is dedicated to detection of traces of hydrocarbon contamination in the soil environment. The author has been in charge of the firmware and the electronics development for several different measuring instruments.

Let start with the original motivation of the research presented here. The ECOPROBE is used in outdoor measurements in places where the soil contamination occurs. It is necessary to log geographical coordinates of the location where the sample of earth gas was taken and analyzed. The differential GPS secures this task well in the cases when the satellites are visible. If the direction between the location of interest and satellites is occluded by the terrain or vegetation then GPS-based localization fails. This happens rather often in real-life measurements.

The initial idea was to use a stereo camera rig for the task known from mobile robotics as the simultaneous localization and mapping (SLAM). The thought is to capture sequence of images, establish correspondences among key points in them, estimate the movement of the observer and the 3D coordinates of the key points in the scene. The required precision was in the order of 0.1 meter in the area spanning several hundreds meters.

The usual approach is that several hundreds of key points are detected in the scene. Many methods for doing so were developed and have been used for this purpose. If the single typical representative has to be chosen then Harris corner detector [14] would pop out. It detects quite reliably points in the intensity image, in which strong partial derivatives in two perpendicular directions exist. These key points do not have any semantic meaning in the scene. Detected key points typically serve in a seek for a parametric transformation with relatively few parameters (say, up to 10). The problem is changed into a robust solution of an overdetermined system of equations usually solving a least square minimization task. As many more key points are available than necessary, the correct parameters are found quickly and reliably in most cases. Similar ideas are used in the case when key points are used in tracking in a videosequence.

When thinking about the described assignment in the context of a typical scene, in which ECOPROBE instrument is used, the idea of a Stable Wave emerged. The inspiration came from the nature. It is known that dogs are short-sighted and cannot see sharply. They still live well in the 3D world and are able to hunt. The thesis author is short-sighted too. He can also perceive approximately a 3D structure of the world without glasses. Another observation concerns fast movements. If an object moves very fast then a human is unable to recognize details of it, however he still perceives a position and speed of the moving object. These observations

motivated to seek some integral entities as blobs rather than differential structures as corners or edges.

This thesis explains the Stable Wave principle, suggests efficient algorithms to detect and refine it in images. The second key idea was to use the Stable Wave in tracking. The result is the detection/tracking system which is ultra fast and competitive with the state-of-the-art methods.

The thesis explicates the Stable Wave idea theoretically, suggests related algorithms, describes their implementation, and reports extensive experimental validation of the idea. In addition, the related C++ and MATLAB software is made publicly available to the research community. On the other hand, the author did not get the Stable Wave to the original application with ECOPROBE because he had to dedicate his efforts to other R&D activities in the company he works for.

1.2 Problem formulation

The research presented in the thesis aims at

- detecting *salient semantic-less regions of interest (stable waves)* in the image automatically. It is assumed that the input data come from a single camera videosequence which implies a short baseline;
- building *point correspondences* among detected stable waves in consequent frames of the videosequence;
- utilizing detected stable waves and established correspondences among them in a fast, *low-level tracking*.

The thesis builds on the key idea – the *stable wave*. The *wave is locally convex* and it is represented by a *unique point – the maximum*. The neighbourhood of this point constitutes the attraction basin leading to the mentioned point. This concepts provides a competitive advantage as compared to the state-of-the *interest points* approaches as *Harris points* [14], its simplified versions as FAST [35, 36] or its enhanced modifications as affine detector [28] or SHIFT [23]. The advantage comes from the *attraction basin* existence which does not require testing all points in the neighbourhood. The attraction basins can be observed with the evergreen *Kanade-Lucas-Tomasi tracker* [25] (abbreviated KLT in the literature) and in recent linear predictors trackers [15, 45].

We explicate the stable wave idea in one dimension (1D) first. The task of a *peak detection in 1D signals* was solved long ago and has many applications. The thesis offers a new alternative solution to the task which is fast, accurate, insensitive to noise, and computationally simple. In the thesis, the 1D case serves as a point of departure to a two dimensional (2D) case needed in image/video analysis. Actually, the 1D stable wave idea is utilized in the RS Dynamics device for detecting explosives. Unfortunately, the real experimental data are company confidential and cannot be presented in the thesis.

The *tasks solved in the thesis* can be formulated in the three simple items:

- Explain the stable wave idea in 1D.
- Design the 2D version of the stable wave and study its properties.
- Use the 2D stable wave in tracking and evaluate its properties.

1.3 Organization of the thesis

The thesis is structured as follows. First, the state of the art related to the work is given in Chapter 2. Next, the origin of all the methods presented, Stable Wave Detector of peaks in 1D discrete signal is explicated in Chapter 3. Chapter 4 extends the Stable Wave detector to 2D. The low-level tracker based on Stable Wave idea is then described and evaluated in Chapter 5. The adjective 'low-level' means that the tracker establishes point-to-point correspondences between consecutive frames and each point is treated independently. Chapter 6 shows how to integrate the low-level tracker into a higher level system able to track a real object in a scene or to estimate the camera pose and/or maps the environment, in which the camera moves. The main contributions are summarized and the directions for future research are given in Chapter 7.

2 The state-of-the-art

The thesis deals, broadly speaking, with two areas:

1. *Detecting* of semantic-less salient entities – stable waves.

2. *Tracking* above mentioned stable waves in videosequences.

Both these areas have been extensively studied by others. Our intention is to keep the state-of-the-art description rather compact. We select mainly methods related to the key concept of the thesis, to the stable wave. The two next sections deal with detection (Section 2.1) and tracking (Section 2.2).

2.1 Interest point/region detection

The rationale of representing the image or part of it by a relatively *small number of semantic-less interest points/regions* is to provide reliable affirmations about image geometry and appearance. Sometimes these points/regions serve as basis for image description which is sparser and more informative than the image intensities (colours) itself. Points/regions are mainly used for matching slightly different images, stereo, 3D reconstruction, image retrieval, object categorization, object recognition, object tracking, etc.

Points/regions of interest have to be detected and many detectors were proposed for this purpose. The detected good points/regions should fulfil the following *four requirements* [9]:

1. *Robustness* against noise, image intensity variations due to lighting, geometric distortions, etc.

2. *Sparseness* to reduce the amount of data, keep relevant information and increase speed.

3. *High speed* because the detector/descriptors constitute low-level parts of more complicated algorithms.

4. *Completeness* because the detected points/regions should comprise much information about the image for subsequent processing.

The extensive survey of related detectors and methods is given in [43]. Detected entities are often divided into *corner detectors* (also interest point, key point), *blob detectors* and *region detectors*. Corners represented as points are essentially zero dimensional. Blobs and regions represent two dimensional entities. *Blobs* usually denote entities attached to a particular image location representing light or dark areas around it. *Regions* are explicitly determined by their boundaries.

To save space, detectors of one dimensional entities as edge detectors and line detectors will not be discussed here even they might be used for corner/region detection.

A popular group of corner detectors originated in *Moravec detector* [30], improved later by Zuniga [46]. The important breakthrough was the *Harris corner detector* [14] which is probably the most often used one also due to several easy to get public domain implementations.

A fast implementation of the Harris detector is described in [31]. Comparison of interest point detectors is given in [38]. An overview of existing interest point detectors can be found in [28].

Corners are not inherently scale invariant, i.e., a *multi-scale Harris detector* does not localize the same local structure at the same point in different scales. Mikolajczyk [28] choose the 'correct' scale as maxima of Laplacian-of-Gaussian in the scale space. Deriche [8] approached this problem by fitting the line through the locations at different scales and searched where the series of points converge. *Lowe's detector* searches maxima of Difference-of-Gaussian [23] in the scale space. The recent thorough analysis of Harris corner detector is provided in [22].

It was observed by many that richer structures in images compared to local, 'derivative-based' corners can bring additional benefit. The extensions of Harris-like interest point detectors based on an affine normalization around Harris points were proposed [28], [37].

The other big group of methods relates to *region/blob detectors*. The seminal work from the same group as this work (the Centre for Machine Perception of the Czech Technical University in Prague) suggested *Maximally Stable Extremal Regions* (MSERs) [26]. It is based on an idea that thresholding the intensity image by all possible thresholds and observing the change of the area of detected regions allows to detect highly stable regions, those which correspond to the smallest speed of the area change. The approach was later extended by others, e.g. to color images [10].

The other regions detectors are, e.g., the edge-based region detector [41]; the detector based on intensity extrema [42], and the detector of 'salient regions' [16]. The performance of above region detectors is compared in [13], [29], where the performance to change in viewpoint, scale, illumination, defocus and image compression are considered.

The *blob detector* proposed in this thesis was motivated by practical observations and a some practical experience with applying signal processing theory. Robustness and speed stems from basic properties of Fourier transformation, dot product and *phase of the first harmonics*. A global difference of phase of the first harmonics between two omnidirectional images was used to find their relative orientation [33]. However, the blob detection method suggested in the thesis uses a moving window focusing to individual landmarks in the image than to the whole image.

The blobs detected by method proposed here seek for local extrema similarly as [42, 26]. The difference of our method is in the requirement that the blob has to exhibit some degree of symmetry around its centre. The recent work of Maver [27] suggested a new interest point detector which explores symmetries. The approach uses much more complicated computational approach and is slower.

2.2 Tracking

The *aim of tracking* is to locate continuously the *target* in a videosequence. It is usually assumed that the target is given at the beginning of tracking. The *target trajectory* is built incrementally by repeatedly estimating relative displacement of the target between successive frames.

Many different tracking methods have been proposed, from global template-based trackers [1], shape-based methods, probabilistic models using mean-shift [6], particle filtering [20] to local key-point based trackers [32] or flow-based trackers [39].

The *original concern in this work* was in *tracking salient blobs* serving as natural landmarks in a videosequence captured in bush or forest-like scenes. The intended purpose was to self-localize the observer in the 3D environment. Similar task is approached in Simultaneous Localization and Mapping (SLAM) in mobile robotics [31, 3].

The term tracking covers a wide range of techniques which can be categorized according to a number of criteria: the character of the tracked object, e.g. a square image patch, a free-form region, an articulated structure or a dynamic texture, by the ability to adapt, learn and recover from failures, by the speed of tracking, by the choice of a predictor, and other properties.

This thesis focuses on *tracking image patches* and thus belongs to the category of *low-level tracking algorithms*. Two subtasks are integrated into the tracker: (1) the choice of objects to track and (2) the method of establishing correspondence between the instance of the objects in consecutive frames.

The *Kanade-Lucas-Tomasi (KLT) tracker* constitutes a milestone and the widely used tool in low-level tracking. The tracker is based on the early work of Lucas and Kanade on image registration [25]. The idea was applied to tracking by Tomasi and Kanade [40]. Later the approach was explained clearly in the paper by Shi and Tomasi [39]. Additional information useful to understand the implementation were provided in [2].

KLT tracker establishes correspondence by (iterative) *gradient-based minimisation*. Typically, the tracked objects are image patches with two high eigenvalues of the structure tensor [39], which are essentially the same as regions centred around Harris interest points [14]. These so called 'good features to track' or Harris regions possess the property that they are different from all regions in their neighbourhood, a necessary condition for establishing reliable point-to-point correspondence. However, these points do not have, at least not by design, any property that would facilitate the minimisation step.

Other popular low-level tracking methods, e.g. [3, 19], rely on very fast detectors and establish correspondence by matching of detected points, which is feasible if only a small number of alternatives must be verified, i.e. when the error in prediction and hence the search radius is small with respect to the density of points.

Inspired by the active appearance models of Cootes et al. [7], Jurie and Dhome [15] realised that in some image regions an approximately linear relationship exists between observations and displacements, allowing ultra-fast tracking by performing a few dot-products in a high-dimensional spaces. This family of trackers is usually

named linear predictor trackers.

The *linear tracker* has been recently generalised by K. Zimmermann et al. [45] in our Centre for Machine Perception at the Czech Technical University in Prague. They made the approach more precise by applying a sequence of lower dimensional linear operations which make progressively more accurate predictions. However, the significant weakness of linear prediction methods is the need, before tracking starts, to perform both a time-consuming search for suitable regions and learning of the linear mapping.

This work keeps the advantage of the linear predictor method – an extremely fast tracking, but removes the search for suitable regions to be tracked.

3 Stable Wave detector in 1D

This chapter describes the origin of all the methods described in the theses – the *Stable Wave Detector* (SWD) in 1D for the peak detection in discrete signal. We consider the discrete data to be the sequence of measurements of physical quantity changing continuously in time and therefore the location of the real peak is a real and not an integer number. The time instances of measurements are called samples. Let us start our explanation with an experiment in 1D which motivated our approach and gave it the name, Section 3.1. Single scale Stable Wave Detector is described in Section 3.2. Multiscale issues are disscused and multiscale SWD algorithm is decsribed in Section 3.3 The results of experiments are shown in Section 3.4.

Please, notice that all equations work with discrete time and the coefficients are always computed inside the window with period T and never from whole data.

3.1 What is the Stable Wave?

The word *'wave'* comes from Fourier transform which is a basic tool in signal/image processing. The term 'stable' says that the wave in the signal (or image) should be stable / well localized in some sense.

The *idea of the stable wave* originates from a simple experiment in 1D. The aim is to find a peak in 1D data reflecting measurements of a continuously varying physical quantity in equidistant discrete instants called samples. We start form a data vector $\mathbf{I}$ of length N. Suppose that the peak is approximately $T/2$ wide, where $T < N/2$, and it is located somewhere in the closed interval $\langle x_0, x_0 + T\rangle$, where $1 < x_0 < N - T$ (symbols $x_i \in \mathbb{R}$ measure the distance with subsample resolution from the begining of data). This inequality allows the algorithm to move the window during its iterations. The reader may ask what the peak width is. The exact definition is not provided because it is difficult to establish it. For our purposes, the peak width is the half of period of the cosine which best fits the peak. The algorithm may *search the nearest peak of any polarity* or select the maximum or the minimum.

For the estimation of the phase, we compute the first harmonic in the form

$$A \cos\left(\frac{2\,\pi\,t}{T} - \varphi\right) = a \sin\frac{2\,\pi\,t}{T} + b \cos\frac{2\,\pi\,t}{T}\,, \tag{3.1}$$

where $a, b \in \mathbb{R}$ are sine and cosine coefficients of the first harmonic (computed by formulas for Fourier series), variable $t \in \mathbb{R}$ is relative to the position of the window (in contrast to x_i related to the beginning of data).

$$A = \sqrt{a^2 + b^2} \tag{3.2}$$

is the amplitude and φ is the unique angle in $(-\pi, \pi\rangle$ satisfying Equation 3.1, see Figure 3.1. The case $A = 0$ is rare and not assumed here.

Let $t_{\min}, t_{\max} \in \langle 0, T)$. The first harmonic has the minimum

$$t_{\min} = \left(\frac{1}{2} + \frac{\varphi}{2\,\pi}\right) T$$

and the maximum

$$t_{\max} = \begin{cases} \left(1 + \frac{\varphi}{2\,\pi}\right) T & \text{if} \quad \varphi < 0\,, \\ \frac{\varphi}{2\,\pi}\, T & \text{if} \quad \varphi \geq 0\,. \end{cases}$$

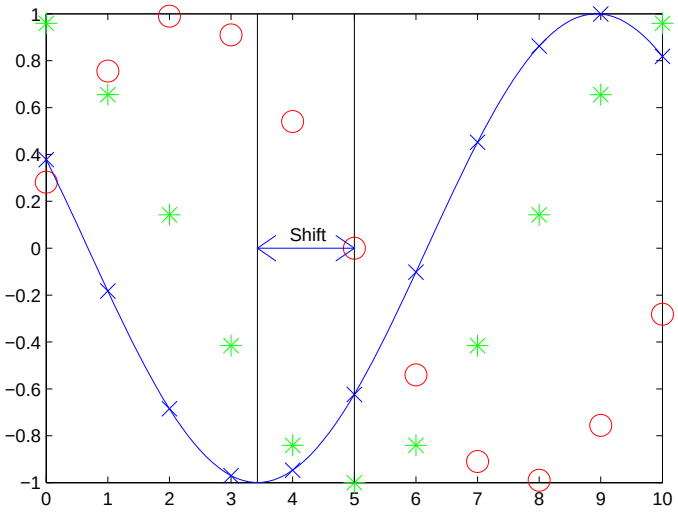


Figure 3.1: The sine (red) and the cosine (green) base functions. The arbitrarily shifted cosine wave (blue).

$$\delta = \frac{T \arctan\left(\frac{a}{b}\right)}{2\pi} + d\,, \tag{3.3}$$

where the offset $d \in \{0, -T/2, +T/2\}$ depends on the signs of a, b and the desired task (the nearest peak/maximum/minimum) according to Table 3.1. Division by zero has to be treated in the implementation (e.g., if $b = 0$ one can set b to some small number like 0.0000001). The coefficients a, b are computed from the window of length T, the same as the period of the base functions according to Equation 3.5.

Let $\mathbf{F}$ be a vector of intensity values in the window of length T. In the discrete case, sine and cosine are represented by vectors $\mathbf{S}$ and $\mathbf{C}$, henceforth called the base vectors, which are defined as

$$\begin{aligned} \mathbf{S}(j) &= k_T \sin(2\pi((j+\tau)/T))\,, \\ \mathbf{C}(j) &= k_T \cos(2\pi((j+\tau)/T))\,, \\ k_T &= \sqrt{T/2}\,, \\ j &= 0, \ldots, T-1\,, \end{aligned} \tag{3.4}$$

where $\tau \in \langle -1, 1 \rangle$ tunes the position of extremes relative to the grid and k_T is a normalization constant. For the localization purpose, k_T can be replaced by 1 or any other nonzero number (e.g. by 2^{15} for 32-bit integer arithmetic). However, if the amplitude is used to compare waves with different periods then a proper scaling is necessary.

The *Fourier coefficients* a and b are computed as dot products

$$a = \sum_{j=1}^{T} \mathbf{F}(j)\,\mathbf{S}(j) = \mathbf{F} \cdot \mathbf{S}\,, \qquad b = \sum_{j=1}^{T} \mathbf{F}(j)\,\mathbf{C}(j) = \mathbf{F} \cdot \mathbf{C}\,. \tag{3.5}$$

The peak can be localized iteratively by Algorithm 1.

1. Define the window $\mathbf{W}_1 = \langle \hat{x}_0, \hat{x}_0 + T - 1 \rangle$, $\hat{x}_0 = \mathrm{round}(x_0)$, $i = 1$.
2. Compute Fourier coefficients a_i, b_i from $\mathbf{F}_i = \mathbf{I}(\mathbf{W_i})$ of the first harmonic wave (i.e., with a period T).
3. Estimate the peak shift δ_i in window $\mathbf{W}_i$ according to Equation 3.3.
4. $x_i = x_{i-1} + \delta_i$, $\hat{x}_i = \mathrm{round}(x_i)$.
if $\hat{x}_i = \hat{x}_{i-1}$ **then**
| Finish. The peak location is $x_i + T/2$.
end
if $|x_i - x_0| > T/2$ **then**
| Fail – divergence.
end
5. $i = i + 1$.
if $i < T$ **then**
| Define a new window $\mathbf{W}_i = \langle \hat{x}_{i-1}, \hat{x}_{i-1} + T - 1 \rangle$.
| Go to Step 2.
else
| Fail – oscillation.
end

Algorithm 1: Iterative SWD algorithm in 1D.

Algorithm 1 typically converges in the first or the second iteration for the ideal peak without noise (Section 3.1.1) if the period T fits well the peak width as shown in Figure 3.2. The algorithm works because the phase of the first harmonic wave

Signs of the coeff	$b > 0$ & $a \geq 0$	$b > 0$ & $a < 0$	$b < 0$ & $a < 0$	$b < 0$ & $a \geq 0$
Extreme/edge	min/falling	min/rising	max/rising	max/falling
Go to nearest	0	0	0	0
Go to minimum	0	0	$-T/2$	$+T/2$
Go to maximum	$-T/2$	$+T/2$	0	0

Table 3.1: The additional offset d in Equation 3.3. The second row interprets what can be seen in the window.

determines the location of the peak. The *higher harmonics* make the peak shape more and more precise. Intuitively, the *symmetric peak* occurs in the location where the waves of different near frequencies meet with the same phase. If their phases differ then the peak becomes asymmetric, e.g. the rising slope is more steep than the falling slope.

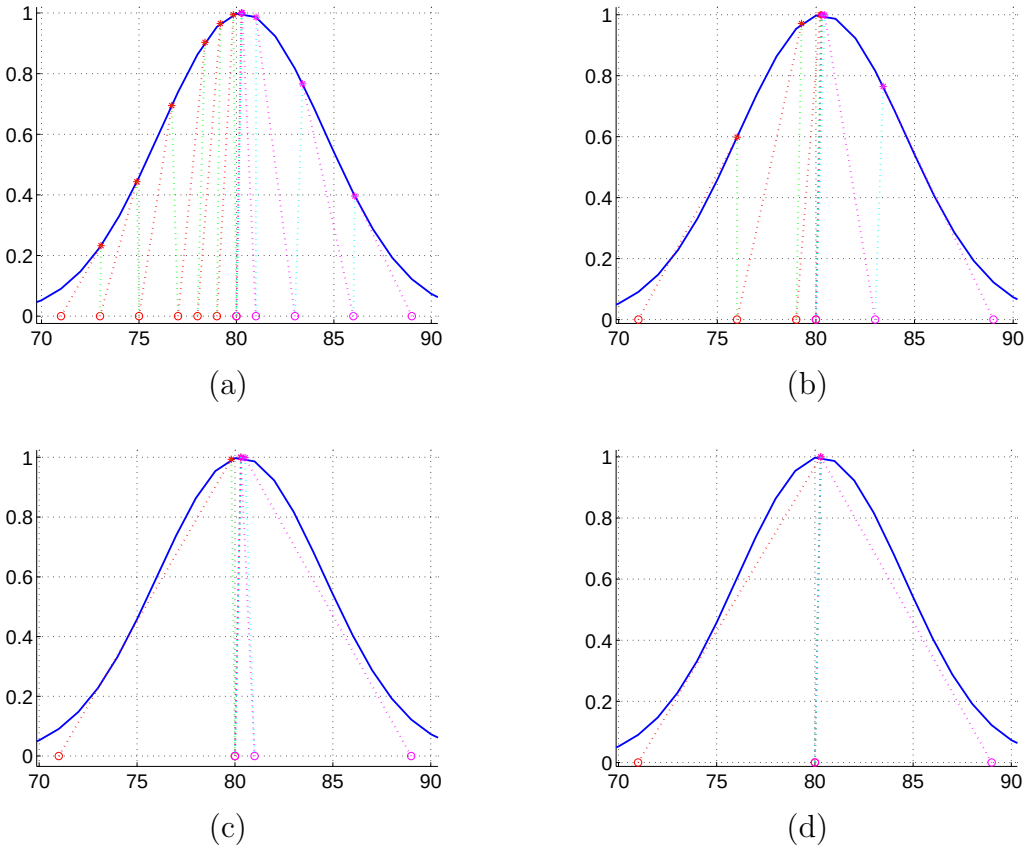


Figure 3.2: Examples of the convergence of Algorithm 1. The algorithm runs twice – first, from the leftmost circle; second, from the rightmost circle with periods a) 8, b) 16, c) 32, d) 64. Each red/magenta line connecting the stars with the circles shows a single iteration of Algorithm 1. The stars show estimates of the extreme location and the circles mark the window centres. Green/cyan lines connect the estimates from ith iteration with the new position of the window for $(i+1)$th iteration.

An *interesting observation* is that the *precise knowledge of the peak width is not crucial for the localization* of an ideal peak (see Figure 3.2). The algorithm works well for the period T more than one octave below or over the optimal period for isolated peak in flat area. If the period is too long, Algorithm 1 may be blocked by the presence of the other peaks. The meaning of the octave is the same as in the music theory, i.e. the interval between one musical pitch and another with half or double frequency (half or double period).

The amplitude A (Equation 3.2) can measure the strength of the response and suitability of the period T for a given peak.

The algorithm can work also for edges. Edges are zero crossings instead of extremes. In such a case, the phase of the sine would be used.

3.1.1 Ideal peaks

Examples of ideal shapes from the SWD point of view are cosine, Gaussian, rectangle and other similar symmetric shapes. The detector localizes peaks of the *'ideal shape'* with the subsample precision similarly as humans would localize them intuitively when looking at them. For example, the detector finds the maximum of a Gaussian or the centre of a rectangle. The application of SWD is not limited to symmetric shapes. However, the interpretation may not be so straightforward for other shapes.

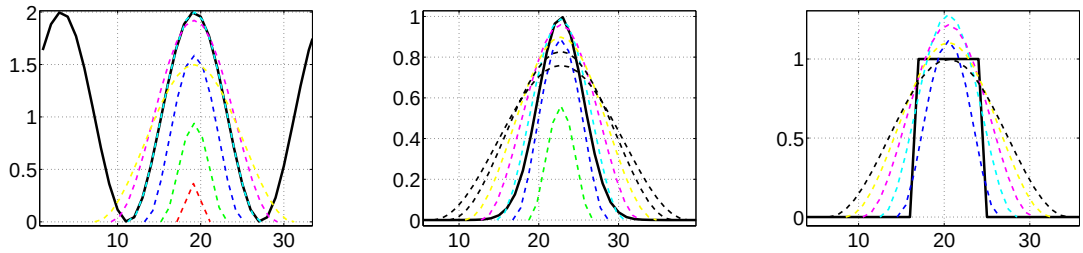


Figure 3.3: Solid curves show the examples of ideal peaks. Dotted curves visualize the results of SWD at different periods T: 4 (red), 8 (green), 12 (blue), 16 (cyan), 20 (magenta), 24 (yelow),28 (black), 32 (black). The dotted curves are cosines with the amplitude and phase found by SWD (the curves are vertically shifted to have positive values). a) Sine wave ($T = 16$, $\varphi = 0.69$). b) Gaussian $y = \exp(-(x - 22.76)^2/16))$. c) 8-sample-wide rectangle.

Figure 3.3 shows the examples of ideal peaks (solid curves) suitable for SWD with the windows of width 16. The dotted curves visualize the results of SWD algorithm at variable window lengths. The dotted curves are cosine waves with the amplitude found by SWD centred at the peak location found by SWD. SWD algorithm successfully found peaks even if the window length was far from the period of the signal or from the double of the peak width. For periodic signal, SWD period T should be shorter then double of signal period. For the peak with a flat tip, SWD period should be longer than the width of the flat tip of peak.

3.2 The single scale Stable Wave detector in 1D

This section describes method *searching peaks in discrete data*. The data are covered by overlapping windows of the length T which is the period of the base functions, Equation 3.4. The windows overlap by Ω samples.

Inputs are a vector $\mathbf{I}$ of the length N containing data and a scalar $T < 2N$ giving a SWD period which is assumed to be divisible by 4.

Outputs are peak locations and SWD amplitudes, separately for maxima and minima. The method is described by Algorithm 2.

1. Compute Fourier coefficients a_i, b_i of the first harmonic wave for individual windows $\mathbf{W}_i$.
2. Find the candidate windows using CNW criterion (Section 3.2.2).
3. For candidate windows, compute the shift δ_i (Equation 3.3) and amplitude A of the Stable Wave (Equation 3.2).
4. The extreme location relative to the candidate window $\mathbf{W}_i$ is $\delta_i + T/2$.
5. Replace double detections of a single peak (a peak detected in two overlapping windows) by their mean (both in the location and in the amplitude).

Algorithm 2: The single scale Stable Wave Detector in 1D

3.2.1 The efficient computation of Fourier coefficients

The coefficients are defined by Equation 3.5. Notice that *only the discrete case is considered in the whole work*. The number of multiplications $M(N, \Omega)$ necessary to compute all coefficients a_i, b_i for the whole data of the length N, the window length T, and the overlap Ω is equal to

$$M(N, \Omega) = 2N \frac{T}{T - \Omega} \,. \tag{3.6}$$

For example $M(N, 2/3) = 6N$, $M(N, 3/4) = 8N$, $M(N, 4/5) = 10N$.

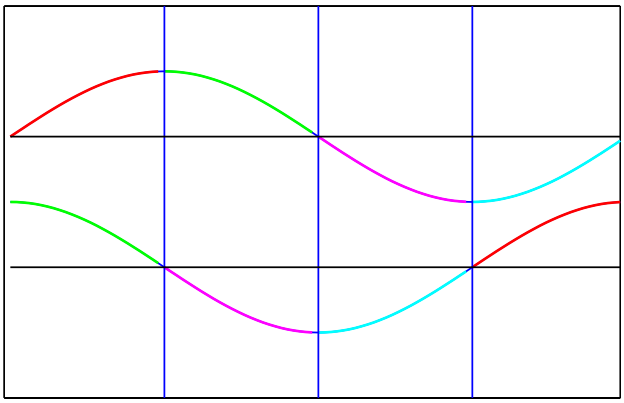


Figure 3.4: Similarity of the quarters of the sine and the cosine waves. The colour coding in the image represents the parts of the signal which are the same.

The choice of the *window overlap* is an important decision which influences both the speed and the precision. Looking at the properties of the sine and the cosine functions, 3/4 of the period is the best choice. The reason is the well known fact that the sine wave (the first base function) is the cosine wave (the second base function) shifted by a quarter of a period, i.e., $\cos\varphi = \sin(\varphi + \pi/2)$. Exploring this fact, a significant amount of computations can be saved for the window shift equal to $T/4$ which is equivalent to the phase shift of the base function of $\pi/2$. Notice that

equations require the period T divisible by 4. This divisibility is assumed from this point on in all formulae in Chapter 3 and in Chapter 4.

The dot product $a_i = F_i \cdot S$ can be written as

$$\begin{aligned} a_i &= \sum_{j=1}^{T} \mathbf{F}_i(j)\mathbf{S}(j) \\ &= D_i^s\left(1,\tfrac{T}{4}\right) + D_i^s\left(\tfrac{T}{4}+1,\tfrac{T}{2}\right) + D_i^s\left(\tfrac{T}{2}+1,3\tfrac{T}{4}\right) + D_i^s\left(3\tfrac{T}{4}+1,T\right) , \end{aligned} \tag{3.7}$$

$$\begin{aligned} b_i &= \textstyle\sum_{j=1}^{T} \mathbf{F}_i(j)\mathbf{C}(j) \\ &= D_i^c\left(1,\tfrac{T}{4}\right) + D_i^c\left(\tfrac{T}{4}+1,\tfrac{T}{2}\right) + D_i^c\left(\tfrac{T}{2}+1,3\tfrac{T}{4}\right) + D_i^c\left(3\tfrac{T}{4}+1,T\right) , \end{aligned} \tag{3.8}$$

where D_i^c are dot products of a part of the window with a part of the cosine wave and similarly D_i^s are dot products of a part of a window with a part of the sine wave. The two neighbouring windows have the 3/4 period overlap. Thus, six out of the eight partial dot products differ only in its signs (see Fig 3.4). It is not necessary to compute them twice. A similar idea is behind the Fast Fourier Transform algorithm.

$$\begin{aligned} D_i^s\left(\tfrac{1}{4}T+1,\tfrac{1}{2}T\right) &= D_{i+1}^c\left(1,\tfrac{1}{4}T\right) , & D_i^c\left(\tfrac{1}{4}T+1,\tfrac{1}{2}T\right) &= -D_{i+1}^s\left(1,\tfrac{1}{4}T\right) , \\ D_i^s\left(\tfrac{1}{2}T+1,\tfrac{3}{4}T\right) &= -D_{i+2}^s\left(1,\tfrac{1}{4}T\right) , & D_i^c\left(\tfrac{1}{2}T+1,\tfrac{3}{4}T\right) &= -D_{i+2}^c\left(1,\tfrac{1}{4}T\right) , \\ D_i^s\left(\tfrac{3}{4}T+1,T\right) &= -D_{i+3}^c\left(1,\tfrac{1}{4}T\right) , & D_i^c\left(\tfrac{3}{4}T+1,T\right) &= D_{i+3}^s\left(1,\tfrac{1}{4}T\right) . \end{aligned} \tag{3.9}$$

The data vector is divided into $N_f \leq 4N/T$ non-overlapping subwindows of the length $T/4$ and

$$D_i^{s1} = D_i^s\left(1,\tfrac{1}{4}T\right) = \sum_{j=1}^{T/4} \mathbf{F}_i(j)\,\mathbf{S}(j), \qquad D_i^{c1} = D_i^c\left(1,\tfrac{1}{4}T\right) = \sum_{j=1}^{T/4} \mathbf{F}_i(j)\,\mathbf{C}(j), \tag{3.10}$$

Finally, the coefficients are computed:

$$a_i = D_i^{s1} + D_{i+1}^{c1} - D_{i+2}^{s1} - D_{i+3}^{c1}, \qquad b_i = D_i^{c1} - D_{i+1}^{s1} - D_{i+2}^{c1} + D_{i+3}^{s1} \tag{3.11}$$

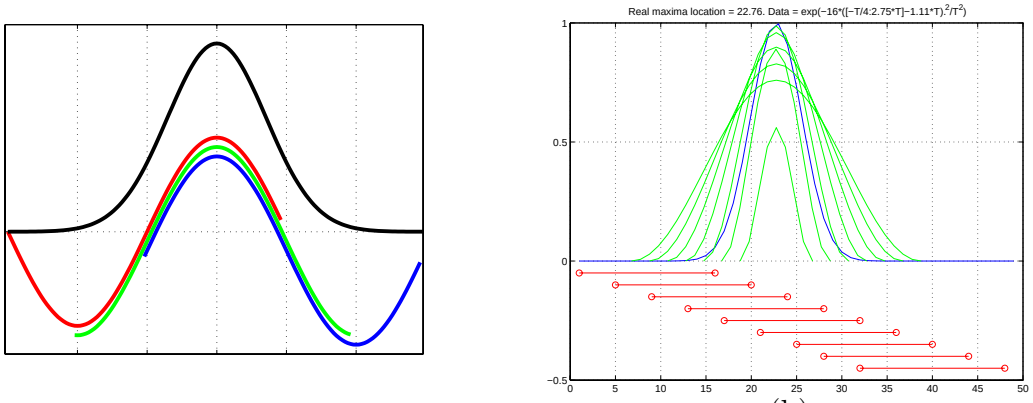
for $j = 1 \ldots N_f - 3$.

As a result, only $2N$ (instead of $8N$) multiplications are needed for data containing N samples.

The special case – the period $T = 8$

The period 8 allows further improvement of the efficiency. The base vectors become

$$\begin{aligned} S &= [\ 0,\ 1/\sqrt{2},\ 1,\ 1/\sqrt{2},\ 0,\ -1/\sqrt{2},\ -1,\ -1/\sqrt{2}\ ] \\ C &= [\ 1,\ 1/\sqrt{2},\ 0,\ -1/\sqrt{2},\ -1,\ -1/\sqrt{2},\ 0,\ 1/\sqrt{2}\ ]. \end{aligned} \tag{3.12}$$



(a) (b)

Figure 3.5: a)The principle of the CNW criterion. The minus sine wave (red, $a < 0$) corresponds to the rising slope. The minus cosine wave (green, $-b \gg |a|$) corresponds to the tip of the peak and sine wave (blue, $a > 0$) corresponds to the falling slope. b)The overapping windows marked by red horizontail lines are used by Algorithm 2.

Combining Equation 3.12 and Equation 3.10 brings

$$\begin{aligned} D_i^{s1} &= \mathbf{F}_i(2)/\sqrt{2} \\ D_i^{c1} &= \mathbf{F}_i(1) + \mathbf{F}_i(2)/\sqrt{2} = \mathbf{F}_i(1) + D_i^{s1}\,. \end{aligned} \tag{3.13}$$

Looking at Equation 3.13, the base can be changed to reduce the amount of computations:

$$\mathbf{S}' = \mathbf{S}\sqrt{2}, \qquad \mathbf{C}' = \mathbf{C}\sqrt{2}\,. \tag{3.14}$$

Consequently

$$\begin{aligned} D_i^{s1'} &= \mathbf{F}_i(2) \\ D_i^{c1'} &= \mathbf{F}_i(1)\sqrt{2} + \mathbf{F}_i(2). \end{aligned} \tag{3.15}$$

It can be observed that just a single multiplication and a single addition is necessary to compute both partial sums for each sample. SWD with the period $T = 8$ can be used for any peaks which are more than one sample wide. For peaks wider than 6 samples, the decimation is recommended. It can be achieved by filtration and downsampling, for instance by replacing k samples by their average, which would reduce the peak width to approximately four samples.

3.2.2 Consistent Neighboring Window (CNW) criterion

The Consistent Neighboring Window (CNW) criterion is used to *test the existence of an extreme inside a given window*. The principle of the criterion can be illustrated on the example of a Gaussian depicted in Figure 3.5a.

Table 3.2 shows the *first two Fourier coefficients, phases and amplitudes* for the windows of length 16. The windows and the peak are depicted in Figure 3.5. The window 4 containing the maximum of the peak is a negative cosine window ($-b > |a|$) surrounded by the two sine windows. Windows containing a rising edge have

window	1	2	3	4	5	6	7	8
a	-0.01	-0.12	-0.46	-0.31	0.38	0.42	0.09	0.00
b	0.01	0.11	0.12	-0.38	-0.31	0.25	0.18	0.02
Amplitude	0.01	0.16	0.47	0.49	0.49	0.49	0.21	0.02

Table 3.2: Gaussian and windows of SWD, a is the sine coefficient, b is the cosine coefficient.

negative sine coefficients and windows containing a falling edge have positive sine coefficients. This could be expected considering the similarity to the sine wave. Not each cosine window contains a peak. For example, a falling edge of a Gaussian produces cosine windows (windows 7, 8 in Table 3.2) which do not contain any peak. CNW criterion is based on this observation and can be summarized as:

- A window F_i contains a stable minimum if
  $b_i > k|a_i|$ & $b_{i+1} > k|a_{i+1}|$ & $a_i > 0$ & $a_{i+1} < 0$ or
  $b_i > k|a_i|$ & $b_{i+1} < |a_{i+1}|$ & $b_{i-1} < |a_{i-1}|$ & $a_{i-1} > 0$ & $a_{i+1} < 0$.
- A window F_i contains a stable maximum if
  $-b_i > k|a_i|$ & $-b_{i+1} > k|a_{i+1}|$ & $a_i < 0$ & $a_{i+1} > 0$ or
  $-b_i > k|a_i|$ & $-b_{i+1} < |a_{i+1}|$ & $-b_{i-1} < |a_{i-1}|$ & $a_{i-1} < 0$ & $a_{i+1} > 0$.

The number $k < 1$ (near 1, default 7/8) improves robustness against noise and other perturbations if both coefficients have similar absolute values (φ is near to an odd multiple of $\pi/4$).

In other words, each minimum should follow the falling edge $a_i > 0$ and should be followed by the rising edge $a_i < 0$. Similarly, each maximum should follow a rising edge and should be followed by a falling edge.

3.3 Multi-scale issues

A *good peak of a certain width* produces *responses for several different SWD periods* as can be seen in Figure 3.3. In the case of the sine wave, the localization precision was better than 0.04 of the sample for $T \in \{8, 12, 16, 20, 24\}$). The strongest response (the amplitude of the stable wave) was at $T = 16$ (i.e., the natural period of the signal). Similar results were obtained for the Gaussian. The precision was better than 0.03 for $T \in \{8, 12, 16, 20, 24, 28, 32\}$. Also for the rectangle, the peak was detected at $T \in \{12, 16, 20, 24, 28, 32\}$.

Notice, that the coefficients are computed always from the window of the length T, the same as the period of the base function. This is the reason for a seemingly impossible result – both Algorithm 1 and Algorithm 2 with the period T can detect the extremes of the sine or cosine wave of the period of $2T$ or even any longer. Also, the isolated peak much more narrow than $T/2$ in a flat area wider than T can be detected and precisely localized. This paragraph says that the algorithm works over

the range specified in Section 3.1. However, there are limits. Naturally, e.g. neither the cosine with period $2T$ cannot be localized by SWD with period T, similarly, nor the group repeating with period $2T$. The short period has also limitations, e.g. the rectangle T samples wide cannot be localized by SWD with period T or shorter.

Even though the SWD *algorithm does not require* precise *knowledge of the peak width*, its *estimate must be provided*. Such estimate may not be available in many practical situations or the peak width may vary in a wide range. To address this problem, the multiscale algorithm was suggested which searches a hierarchy of peaks at *several periods choosing the best scale* by some criterion (similarly to the other multiscale detectors [28], [23]).

Considering the excellent multiscale property of SWD, the scale steps can vary by an *octave*, i.e. form the series 2^k, $k = 1 \dots n$, where n is the number of scales. It is more sparse than 1.4^n used by Mikolajczyk [28] and Lowe [23]. The integer scale allows a more efficient computation compared to a non-integer scale.

The *pyramid* can be obtained using a variable period or a fixed (the shortest) period and data-pyramid obtained by subsampling the original signal and using an anti-aliasing filter. A suitable choice of the fixed period is 8 as shown in Section 3.2.1.

Similarly to Mikolajczyk [28] or Lowe [23], the best scale can be chosen according to the strength of the response measured as the amplitude computed according to Equation 3.2. If the peak is near to the centre of the window then the amplitude may be replaced by the cosine coefficient b.

3.3.1 The multiscale stability and the scale

Having a peak found by SWD at the period T_0, the *quality of the peak* can be measured as a *variance of the locations* found at different nearby scales. The stability can be evaluated at approximately a half of an octave lower (T_L) and higher (T_H) than the original scale (T_0), in which the peak was detected. The word 'approximately' means that the period is always an integer.

The *multiscale stability* η is estimated as

$$\eta = \min(|x_i - x_L|, |x_i - x_H|), \tag{3.16}$$

where x_i is the initial estimate at the period T_0, x_L is estimated using the window of length T_L centred around x_i, and x_H is estimated using the window of length T_H centred around x_i. The minimum is used because the period can be refined to the value T^* from the range less sensitive to the scale change according to Equation 3.17.

The period T_0 may not be the correct scale of the peak. The approach to define the *best scale may be*

$$T^* = \arg\max_T A(x_0, T) \,,$$

where A is the amplitude found by SWD using the window of the length T centred around x_0. This idea is similar to the search of the correct scale for corners [28]. The amplitude can be used for a rough selection of the scale (period T_0) in the pyramid.

The scale may be further refined by

$$\begin{aligned} T^* &= (A_L\,T_L + A_0\,T_0)/(A_L + A_0) \quad \text{if } |\mathrm{x_i} - \mathrm{x_L}| < |\mathrm{x_i} - \mathrm{x_H}|\,, \\ T^* &= (A_H\,T_H + A_0\,T_0)/(A_H + A_0) \quad \text{if } |\mathrm{x_i} - \mathrm{x_L}| > |\mathrm{x_i} - \mathrm{x_H}|\,, \end{aligned} \tag{3.17}$$

where A_L, A_0, A_H are amplitudes computed using periods T_0, T_L, T_H.

3.3.2 The multiscale SW4-CNW-1D algorithm

The input is a vector $\mathbf{I}$ of the length N containing data and vector $\mathbf{T}$ of SWD periods provided by the user or adjusted automatically to the series

$$\mathbf{T}(i) = 2^{i+2}, i = 1 \ldots \mathrm{round}(\log_2(N)) - 2\,.$$

1. for *all scales* $\mathbf{T}(i)$ **do**
  Find peaks by Algorithm 2
end

2. for *all peaks* **do**
  if *a peak is found on two or more scales* **then**
    Choose the scale according to the highest amplitude.
  end

  Estimate the multiscale stability by Equation 3.16
  Refine the scale by Equation 3.17
end

Algorithm 3: The multiscale SW4-CNW-1D algorithm.

3.4 Experiments

The experiments should evaluate the performance of the single window SWD algorithm for the peak detection. A Gaussian peak

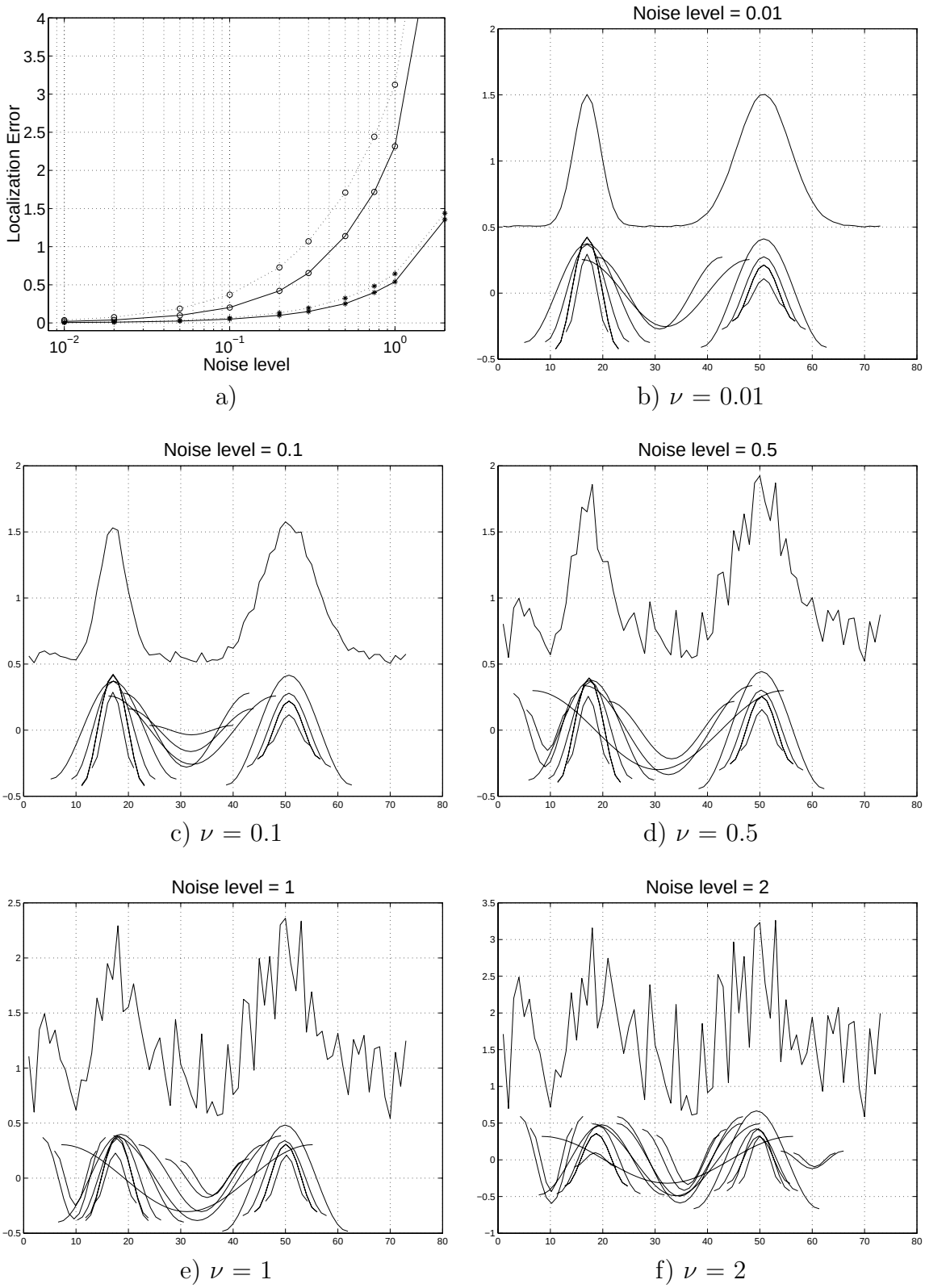
$$s \exp\left(\frac{-(x - x_0)^2}{\sigma^2}\right)\,,$$

where s is a scaling coefficient, x_0 is the ground truth location of the peak and σ controls the width of the peak, is used extensively in the experiments of this section.

3.4.1 Two peaks - robustness to noise

The robustness of the multiscale SWD algorithm to noise was tested on synthetic data. The signal containing two Gaussian peaks

$$\exp\left(\frac{-(x - 52.76)^2}{51.2}\right) \quad \text{and} \quad \exp\left(\frac{-(x - 17)^2}{12.8}\right)\,,$$



a) b) $\nu = 0.01$ c) $\nu = 0.1$ d) $\nu = 0.5$ e) $\nu = 1$ f) $\nu = 2$

Figure 3.6: Robustness to noise. The results are from 100 repeats. The uniform noise from the interval $\langle 0, \nu \rangle$ was added to the signal containing two Gaussian peaks of the unit amplitude. The signal was vertically shifted by adding the value 0.5 just for visualization purposes. a) Error versus noise level ν. b)...f) Examples of data.

ν	0.01	0.02	0.05	0.10	0.20	0.30	0.50	0.75	1.00	2.00
σ_1	0.005	0.010	0.025	0.050	0.099	0.150	0.252	0.400	0.540	1.356
σ_2	0.007	0.013	0.033	0.066	0.131	0.196	0.325	0.483	0.645	1.438
δ_1	0.020	0.039	0.100	0.203	0.420	0.654	1.140	1.718	2.316	5.952
δ_2	0.038	0.075	0.187	0.371	0.728	1.071	1.710	2.442	3.123	7.404

Table 3.3: Robustness to noise. The results are from 100 repeats. The uniform noise from the interval $\langle 0, \nu \rangle$ was added to the signal containing two Gaussian peaks of the unit amplitude. σ_i is the standard deviation of the estimate of the peak location and δ_i is the width of the interval, in which all the estimates from 100 repeats have fallen, $i = 1$ stands for narrow peak, $i = 2$ for wide peak.

$x \in \{1 \dots 81\}$ was degraded by the additive uniform noise. The uniform distribution was used as a good model of the quantization noise. Independent instances of uniform noise vectors Ψ_i, $i = 1 \dots 100$, $\Psi_i(j) \in [0, 1]$ were generated, scaled by the noise level ν and added to the signal S.

The results are summarized in Figure 3.6a and Table 3.3. The standard deviation of the estimate was smaller than one sample up to the noise level $\nu = 1$. Also, up to 30% noise level, all peak location estimates fell into the interval of one sample (pixel) width.

3.4.2 The effect of the period/width relation and the subpixel shift to the peak localization

This experiments studied achievable precisions and also how the precision is influenced by the choice of SWD period versus the width of the peak. The effect of the subpixel shift of the peak position relative to the sampling grid is observed too. Both phenomena are studied first without noise and later with noise.

The effect of the period/width relation and the shift to the peak localization, without noise

The Gaussian peak of the variable width and the position relative to SWD window is generated and localized by SWD to evaluate the effect of (1) the relation of the peak width and SWD period, and (2) the subpixel shift of the peak position relative to the window on the precision of the peak localization.

Figure 3.7 defines the shift of peak relatively to the base function. The cosine base of an even period has its minimum at the $T/2 + 1$ sample of the window.

Figure 3.8 shows the results of localization of a Gaussian peak. The localization error significantly falls with the rising SWD period and slightly grows with the rising peak width. The flat part of the error curve is almost the same (difference less than 1%) for all other tested symmetric peaks (data not shown), i.e., for the triangle, parabola and cosine wave.

Table 3.4 summarises the maximal localization error for the combinations of the

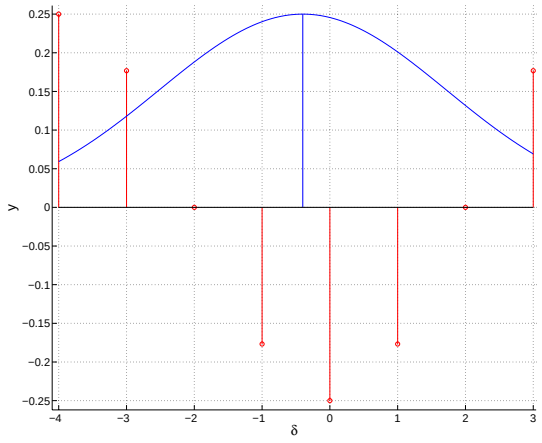


Figure 3.7: Continuous Gaussian peak (blue) and discrete cosine base function (red). The horizontal axis measures the shift δ of the Gaussian peak relative to the extreme of the base function. The location of the peak is marked by the vertical blue line.

T_{SWD} \ σ	3	6	12	24	48
8	0.003171	0.007760	0.0095064	0.009987	0.0101
12	0.000258	0.002366	0.0038168	0.004278	0.0044
16	0.000010	0.000779	0.0018900	0.002323	0.0024
24	0.000000	0.000066	0.0005845	0.000943	0.0010
32	0.000000	0.000002	0.0001940	0.000469	0.0005
48	0.000000	0.000000	0.0000168	0.000145	0.0002
64	0.000000	0.000000	0.0000007	0.000048	0.0001

Table 3.4: The maximum of the localization error Δ for the combinations of the SWD period T_{SWD} and the width of the Gaussians σ for a peak located between $T/2$ and $T/2+1$ of the window (i.e., shift $\in [-1, 0]$).

SWD period T_{SWD} and the width of Gaussians σ for the peak located between $T/2$ and $T/2+1$ of the window (i.e., shift $\delta \in \langle -1, 0 \rangle$). The maximum of Δ for $\delta \in \langle -1, 0 \rangle$ bounds the worst localization error achievable by Algorithm 1. The window after its last iteration is centred on the peak with the one sample precision. Almost the same results were obtained for the cosine, for the parabola ($y = x^2$) and for the triangle ($y = |x|$).

The localization error grows relatively fast with the shift of the peak out off the centre of the window. However, for the shift smaller than $T/4$, the error is smaller than $\delta/2$. For the shift smaller than $T/2$, the error is smaller than δ. It shows that Algorithm 1 will get better and better estimates as approaching to the correct location. The error in $T/4$ distance is an estimate of the maximal error of Algorithm 2, which moves the window by $T/4$.

Table 3.5 shows the relative localization error $\rho = \Delta/\delta$ for shift $\delta = T/4$. The

T_{SWD} \ σ	3	6	12	24	48
8	-0.2180	-0.3862	-0.4416	-0.4561	-0.4597
12	-0.0815	-0.2802	-0.3846	-0.4148	-0.4225
16	-0.0263	-0.1923	-0.3404	-0.3905	-0.4037
24	-0.0017	-0.0760	-0.2576	-0.3541	-0.3822
32	-0.0001	-0.0254	-0.1807	-0.3199	-0.3672
48	-0.0000	-0.0017	-0.0732	-0.2470	-0.3398
64	-0.0000	-0.0001	-0.0248	-0.1751	-0.3101

Table 3.5: The relative localization error $\rho = \Delta/\delta$ for the combinations of SWD the period T_{SWD} and the width of Gaussians σ for the peak located at $(T/2 + 0.5) + T/4$ sample of the window.

error growth becomes slower if the SWD period grows relatively to the width of the peak. This observation leads to the simple iterative algorithm which alternates two steps. In the first step, the peak location inside the given window is estimated. In the second step, the window is centred around the estimated peak location.

Figure 3.9 shows the dependence of the SWD amplitude on the location of the Gaussian peak relative to the window centre (more exactly, relative to the $T/2 + 1$ sample of the window). If the SWD period is shorter than or comparable to 4σ of a Gaussian then the amplitude has a significant minimum at the shift 0.5 of sample. It is caused by the fast fall of the amplitude of the sine component and only moderate rise of the cosine component in this interval. For longer windows, the curve is flat in the interval $\langle -2, 1\rangle$ (or even in a wider interval depending on the SWD period and σ). The maximum amplitude achieves the window 4σ long.

The effect of the period/width relation and the shift to the peak localization, with noise

The experiments from the previous unnumbered section were repeated with the presence of noise. Independent instances of uniform noise vectors Ψ_i, $i = 1 \dots 100$, $\Psi_i(j) \in \langle 0, 1\rangle$ were generated, scaled by the noise level ν and added to the signal.

It can be observed that the situation dramatically changed in the presence of noise. If the noise were not present then increasing of period would enhance precision.

In the presence of noise, the more narrow the peak and shorter period the better precision. The reason is obvious: noise contains a part of energy on the SWD period, i.e. noise contains the wave with random phase. This wave is added to the wave of the original signal and thus the resulting phase is shifted. The error of the phase φ is proportional to noise level and independent of the period. The error of the shift $\delta = T\varphi/(2\pi)$ is proportional to the period T too.

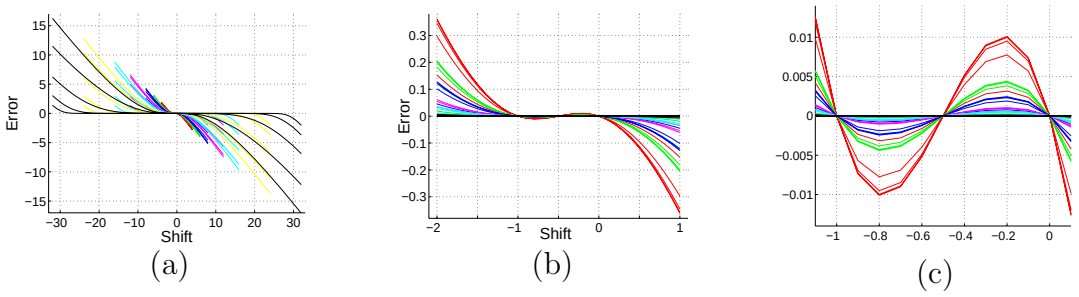


Figure 3.8: The localization error vs. the shift δ of the Gaussian. The width of Gaussians σ was 3, 6, 12, 24 and 48 (different lines with the same color). SWD period: 8 (red), 12 (green), 16 (blue), 24 (magenta), 32 (cyan), 48 (yellow) and 64 (black).

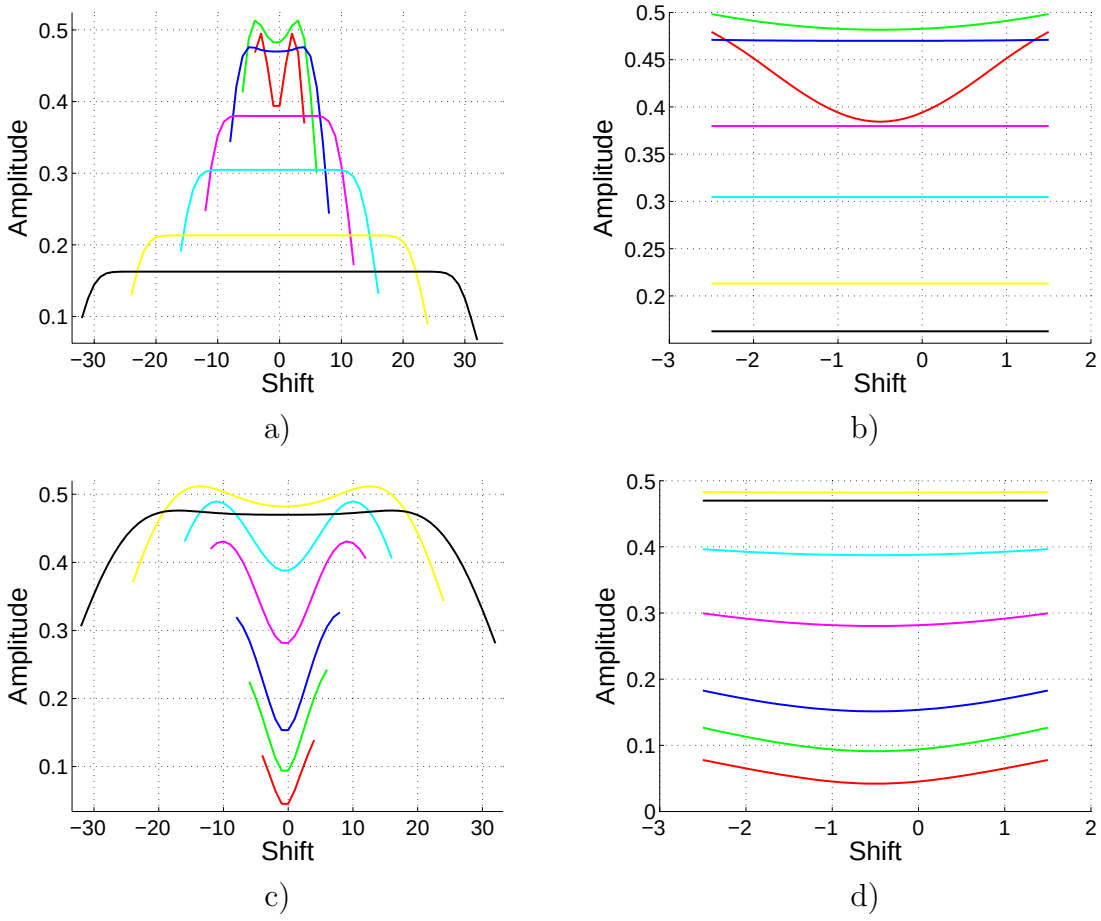
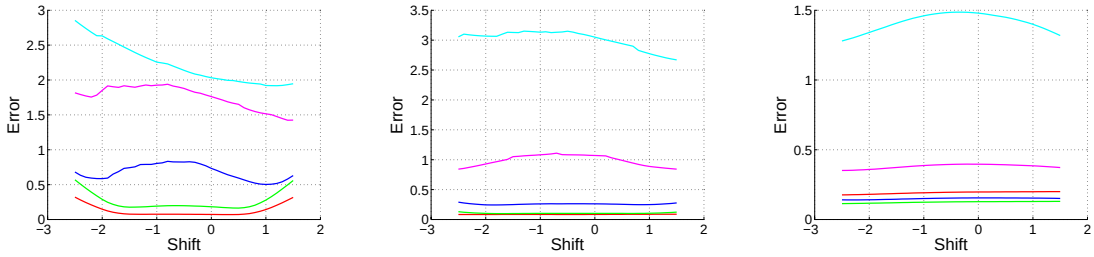


Figure 3.9: SWD amplitude vs. the shift δ of the Gaussian peak. a), b) $\sigma = 3$. c), d) $\sigma = 12$. SWD period was 8 (red), 12 (green), 16 (blue), 24 (magenta), 32 (cyan), 48 (yellow) and 64 (black).



a) $T_{SWD} = 8$ b) $T_{SWD} = 16$ c)$T_{SWD} = 32$

Figure 3.10: Error vs. the shift δ of the Gaussians. σ was 3 (red), 6 (green), 12 (blue), 24 (magenta) and 48 (cyan). Amplitude of noise was 0.2.

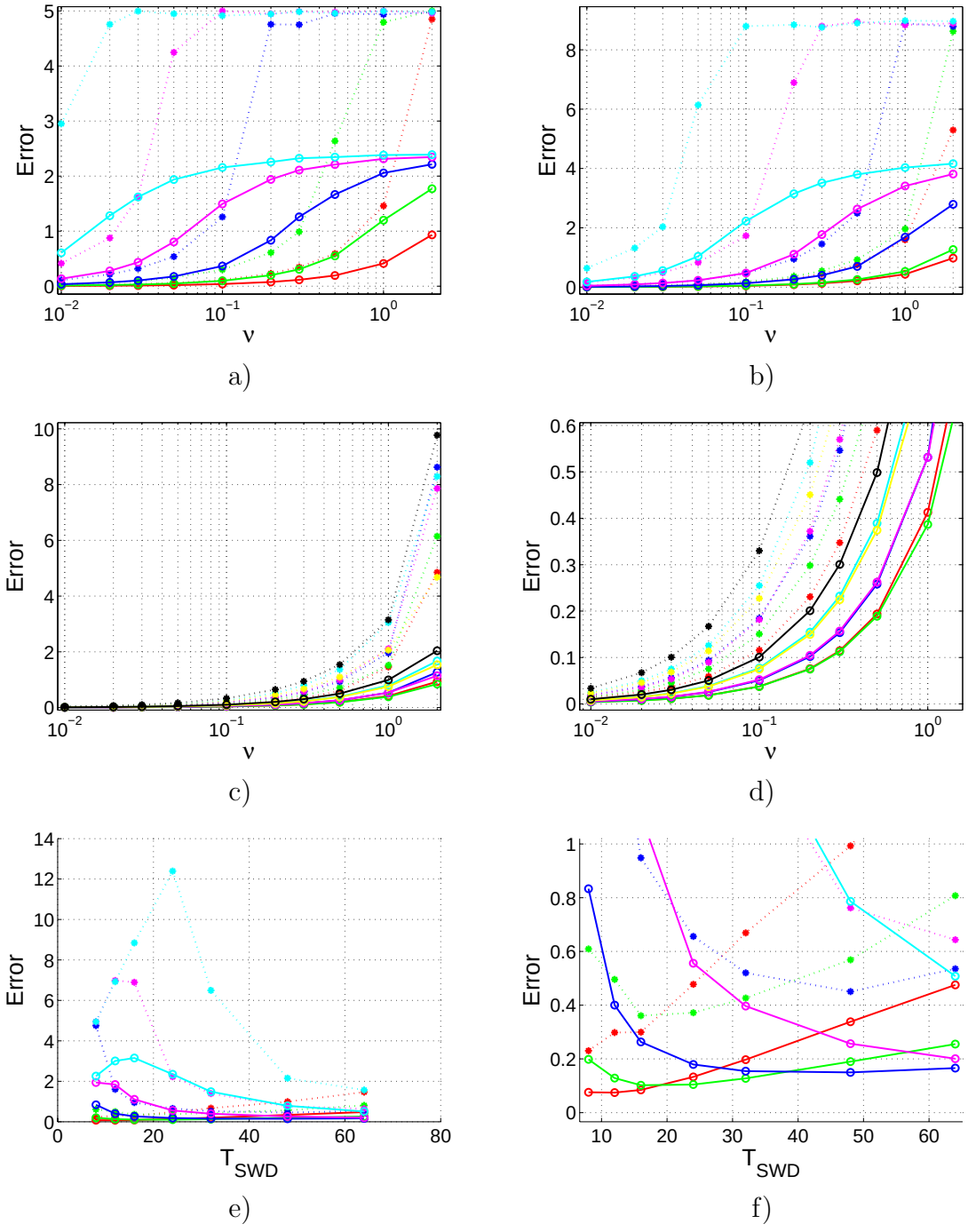


Figure 3.11: Error vs. the amplitude of noise. *Solid lines* denote the mean error, *dotted lines* denote the maximal error. The SWD period was 8 in a) and 16 in b). The Gaussian had σ 3 (red), 6 (green), 12 (blue), 24 (magenta), 48 (cyan). c)The localization error for SWD period which best fits σ; $[T_{SWD}, \sigma]$ was $[8, 3]$ - red, $[12, 3]$ - green, $[16, 6]$ - blue, $[24, 6]$ - magenta, $[32, 12]$ - cyan, $[48, 12]$ - yellow, $[64, 24]$ - black. d) is a detail of c). e)Error vs. SWD period. The width of the Gaussians σ was 3 (red), 6 (green), 12 (blue), 24 (magenta) and 48 (cyan). The amplitude of noise was 0.2. f) is a detail of e).

4 The Stable Wave detector in 2D

Let us observe the example of an *ideal blob in 2D* in Figure 4.1 similarly as we observed ideal peaks in 1D in the previous chapter. Assume an intensity profile along a line in the image domain intersecting a blob, e.g., a row or a column of the image. There is a peak on the 1D intensity profile.

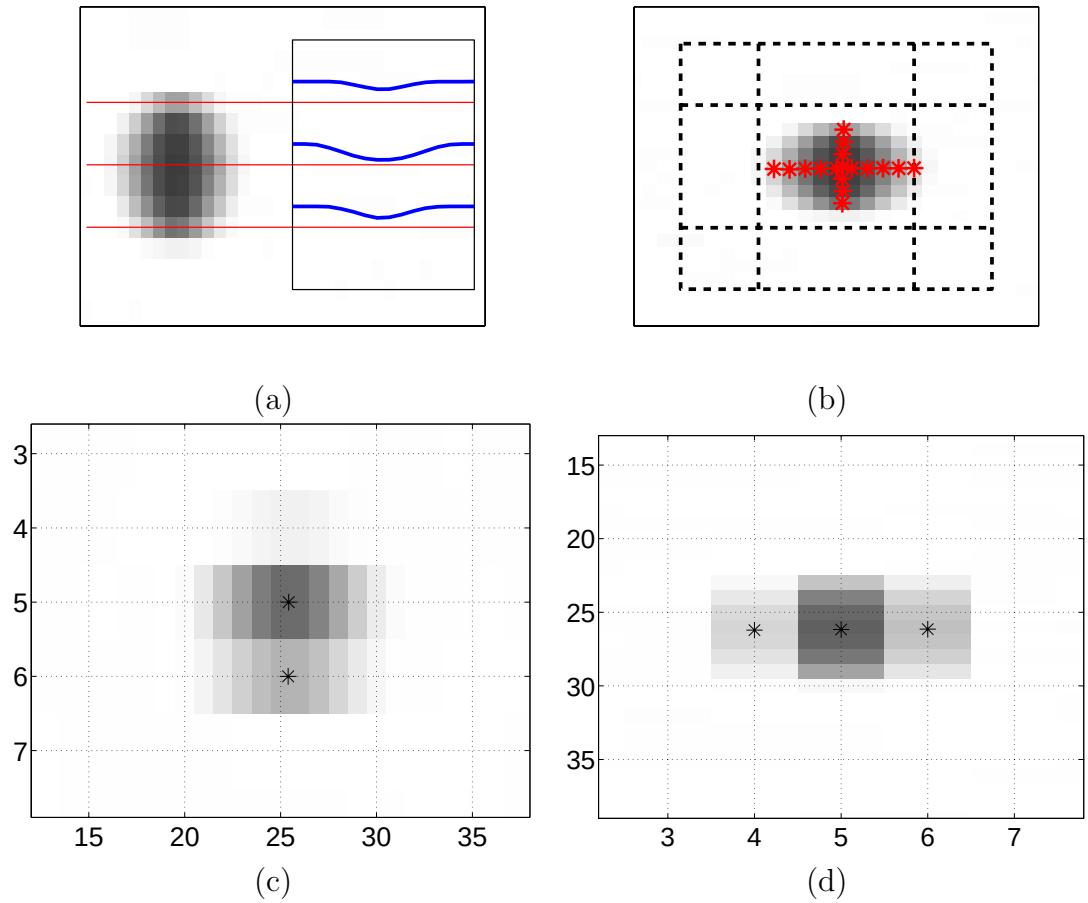


Figure 4.1: The image of blobs in a calibration pattern. a) The original image and the intensity profile along three rows intersecting the blob. b) The peak location of one blob from intensity profiles along rows and columns. c) The downsampled image in a vertical direction. d) The downsampled image in a horizontal direction. c), d) Black stars in the middle of the dark patch show the results of SWD-1D along rows and columns of the image.

The *peak in the intensity profile* along the line in the image is a necessary condition for the blob presence. This condition provides *two independent constraints* – one for rows and the second for columns. Assume that the SWD-1D algorithm from Chapter 3 is applied to the intensity profile along the image row. The SWD-1D localizes the peak, the representative point of the blob in the row with subpixel precision, see Figure 4.1 a). These points lay approximately on a vertical line, shown as red stars in Figure 4.1 b). Similarly, the results of SWD-1D on a column

passing the blob form a horizontal line. The two lines are almost perpendicular. The *2D location of the blob* can be estimated as the intersection of these two lines also with a subpixel precision.

4.1 2D Fourier transform

A more thorough mathematical explanation of the suggested Stable Wave approach can be derived from the 2D Fourier Transformation (FT) and phase-based methods of stereo matching. The 2D FT is a vector function yielding the phase and the amplitude as the vector $[f_h, f_v]$ (horizontal and vertical frequency). Two independent frequency vectors are needed to obtain a 2D phase information. For a better imagination, the vector $\mathbf{f}_1 = [f, 0]$ corresponds to the wave parallel to the axis x. $\mathbf{f}_2 = [0, f]$ is the wave parallel to the axis y, and $\mathbf{f}_3 = [f, f]$ is the wave parallel to the line $y = 1 - x$.

Similarly to 1D case in Chapter 3, only a small part of Fourier transform coefficients will be used to localize a blob, i.e. just the coefficients and the phase corresponding to the two frequency vectors $\mathbf{f}_1 = [f, 0]$ and $\mathbf{f}_2 = [0, f]$. The period $T = 1/f$ will be the size of the window analogically as in Chapter 3.

Let us look in detail to the computation of the Fourier coefficient a for the vector $\mathbf{f}_1$ in the part (square window) of the image (matrix) $\mathbf{I}$. The base function $\mathbf{S}^{2D}(r, c) = \mathbf{S}(c)$ depends on c only. The sums can be decomposed as

$$\begin{aligned} a = \sum_r \sum_c \mathbf{S}^{2D}(r,c)\,\mathbf{I}(r,c) \; &= \sum_c \mathbf{S}(c) \sum_r \mathbf{I}(r,c) = \sum_c \mathbf{S}(c)\,\mathbf{i}(c)\,, \\ \mathbf{i}(c) &= \sum_r \mathbf{I}(r,c)\,. \end{aligned} \tag{4.1}$$

The meaning is the following. First, perform summation of the intensity of image $\mathbf{I}$(row, column) over the rows to get the vector $\mathbf{i}$(column). Second, transform the vector $\mathbf{i}$(column) by the 1D FT to get the the coefficients and phase in the horizontal direction.

4.2 The single scale SWD algorithm in 2D

This section describes the *practical SWD algorithm in 2D*. The phases corresponding to frequency vectors $\mathbf{f}_1$, $\mathbf{f}_2$ can be used to localize the peak similarly as SWD-1D algorithm does. However this solution is not good. First, the application of the CNW criterion (Section 3.2.2) would be difficult. Second, the other problem is that the integrals for Fourier coefficients contain a large neighbourhood of the blob. The solid square in Figure 4.1a shows the optimal square window to detect the blob. The square areas in its corners bring just noise to the integrals.

Our *SWD-2D algorithm* combines the observation from Figure 4.1 with the mathematical derivation. In short, we *detect peaks in rows and fit the vertical line v through them*. Next, we *detect peaks in columns* and fit the horizontal line h through them. The *location of the blob is estimated as the intersection* of lines h and v.

The *outline of Stable Wave Detector* (SWD) in 2D is given in Algorithm 4. The input is an image represented by a matrix $\mathbf{I}$ with m rows and n columns and a prescribed SWD period T.

Algorithm 4 searches for peaks separately in both horizontal and vertical direction similarly as Algorithm 2. The two preprocessed images $\mathbf{I}^h$ (for horizontal direction, Figure 4.1c)) and $\mathbf{I}^v$ (for vertical direction, Figure 4.1d)) are used instead of the original image $\mathbf{I}$. Image $\mathbf{I}^h$ having $\text{floor}(4m/T)$ rows and n columns is computed from $\mathbf{I}$ by downsampling the image in the vertical direction by the factor $T/4$ according to

$$\mathbf{I}^h(r', c) = \frac{\sum_{i=r-k}^{r+k} \mathbf{I}(i, c)}{2k+1}, \quad r = r'\frac{T}{4}, \quad k = \text{floor}(T/8), \tag{4.2}$$

Similarly, $\mathbf{I}^v$ having m rows and $\text{floor}(4n/T)$ columns is formed from $\mathbf{I}$ by downsampling according to

$$\mathbf{I}^C(r, c') = \frac{\sum_{i=c-k}^{c+k} \mathbf{I}(r, i)}{2k+1}, \quad c = c'\frac{T}{4}, \quad k = \text{floor}(T/8). \tag{4.3}$$

Algorithm 4 uses four matrices $\mathbf{M}^{h+}$, $\mathbf{M}^{h-}$, $\mathbf{M}^{v+}$ and $\mathbf{M}^{v-}$ to mark the results of 1D search, the superscripts mean: 'h' horizontal, 'v' vertical, '$+$' positive coefficient b (corresponding to a minimum), '$-$' negative coefficient b (corresponding to a maximum). All these matrices have $\text{floor}(4m/T) + 2$ rows and $\text{floor}(4n/T) + 2$ columns. Each element of $\mathbf{M}^{*\#}$ (mark '$*$' stands for 'h' or 'v', '$\#$' for '$+$' or '$-$') corresponds to one of the overlapping windows (vectors) $\mathbf{W}_i^*, i = 1 \ldots \text{floor}(4q/T)$ of the length T. The number q is n or m, depending on the direction. The overlap of the windows is $(3/4)T$ (i.e., the same as in Algorithm 2). Further, the matrices $\mathbf{M}^+$, $\mathbf{M}^-$, having the same size as $\mathbf{M}^{*\#}$, are used to label candidate positions of the blobs, i.e., the places where the peak was detected in both directions.

A multiscale hierarchy can be obtained by repeating Algorithm 4 with different periods T. The series $T = 2^n$, $n \geq 3$ is a good compromise between the precision and the computational complexity. The other option is to downsample the image to get an image pyramid and run the algorithm on each image in the pyramid. The first option is a bit more precise, e.g. because the image pyramid is sensitive to rotation, etc. The second approach is more computationally efficient. The use of the image pyramid is described in Section 4.3.

4.3 SWD8 in 2D

This section describes the alternative way how the multiscale problem can be solved in 2D which yields the SWD8 algorithm. Instead of a variable period, the image pyramid is computed and the single period ($T = 8$) is used at each level of the pyramid. First, the formation of the image pyramid is described in Section 4.3.1. Next, the computation of coefficients a_i, b_i in both horizontal and vertical directions is explicated in Section 4.3.2.

1. Search candidate windows $\mathbf{W}^*$ in both directions.
for $[*, p, q] \in \{[h, r, j+2], [v, j+2, c]\}$ **do**
Find candidate windows $\mathbf{W}^*_{p,q}$ of $\mathbf{I}^*$ using the CNW criterion (Section 3.2.2). Mark candidates in a map $\mathbf{M}^{*\#}$, such that $\mathbf{M}^{*\#}(p, q) = 1$ iff $\mathbf{W}^*_{p,q}$ is a candidate. Mark '#' is '+' or '−', depending on the sign of coefficient b.
end
2. Search candidates for blobs.
for $\# \in \{+, -\}$ **do**
Compute $\mathbf{M}^{\#}(r, c) = \mathbf{M}^{h\#}(r, c)$ & $\mathbf{M}^{v\#}(r, c)$ for all r, c.
Find groups $G_i^{\#}$ of couples $[r, c]$ forming connected components of $\mathbf{M}^{\#}(r, c) = 1$. Each $G_i^{\#}$ represents a candidate for blob.
end
3. Localize blobs.
for *polarity* $\# \in \{+, -\}$ **do**
for *all* $G_i^{\#}$ **do**
Find all candidate windows $\mathbf{W}^h$ with polarity # that are adjacent to candidate windows from $G_i^{\#}$.
Localize peaks in $\mathbf{W}^h$ (in rows).
Fit the line v through peaks in rows. If the line v is not enough vertical, i.e., the absolute value of its slope is less than $1/\kappa$, then skip $G_i^{\#}$ because it is an unstable blob, $\kappa \leq 1$ is a user defined parameter (by default $\kappa = 0.7$).

Find all candidate windows $\mathbf{W}^v$ that are adjacent to candidate windows from $G_i^{\#}$.
Localize peaks in $\mathbf{W}^v$ (in columns).
Fit the line h through peaks in columns. If the line h is not enough horizontal, i.e. absolute value of its slope is greater than κ, then skip $G_i^{\#}$ because it is an unstable blob.

The location $\mathbf{x_i}$ of the blob in subpixel precision is the intersection of the lines v and h.
end
end

Algorithm 4: The single scale SWD algorithm in 2D

The SWD8 algorithm works on gray-scale images (natural gray-scale or other scalar intensity image, e.g, a color component). Color RGB images can be converted to the gray-scale simply by summing their color components for each pixel. The implementation does not normalize back to 8 bits and uses 16 bits (to increase the precision a little). Indices are used in the MATLAB fashion, i.e., they are counted from 1.

4.3.1 Scale space of images

The simplest way of downsampling the image by the factor two is just taking every second pixel horizontally and vertically

$$I_2(r, c) = I_1(2r, 2c). \tag{4.4}$$

Such approach would suffer from *aliasing effects*. A filter should be applied. One possibility is to sum 3×3 neighbourhood around even pixels in both the even row and the even column to yield the intensity value for the pixel on a more rough scale.

$$I_2(r, c) = \sum_{i=-1}^{1} \sum_{j=-1}^{1} I_1(2r + i, 2c + j). \tag{4.5}$$

Considering only *two scales it may be enough*. However, building the pyramid in such a way suffers from the following problem. The odd pixels in the first level propagate to the third level with higher weight than even pixels. The ratio is 1:2 for even-even:even-odd pixel and 1:4 for even-even:odd-odd. Therefore we use the following 3×3 filter

$$I_2(r, c) = \sum_{i=-1}^{1} \sum_{j=-1}^{1} \mathrm{Weight}(2 + i, 2 + j)\, I_1(2r + i, 2c + j), \quad \mathrm{Weight} = \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}. \tag{4.6}$$

This *filter propagates* each pixel with the same weight through the scales of the pyramid. Probably other filters may be found. However, the described filter seems to be a good choice. It is simple to implement and it yields a good performance. Another advantage of this filter is that its weights are powers of two. Also, the sum of its weights is $16 = 2^4$. Consequently, all computations can be performed in the integer arithmetics by simple bit shifts and additions which are core operations of all common processors.

4.3.2 Fourier coefficients

An efficient way to compute coefficients in 1D was derived in Section 3.2.1. The same idea will be applied here. Namely, the Equation 3.15 will be used. We can observe that for computing the partial sums for each pixel just a single multiplication and a single addition is necessary. Partial sum D^s for both the horizontal and the vertical directions becomes just the intensity values of the image. Therefore only the horizontal sum D_h^c and the vertical sum D_v^c need to be computed for each pixel.

4.3.3 SWD8 Algorithm

The *optimal width of the peak* to be found using period 8 is about 4 pixels (tip plus nearby slopes of the peak). If the window is centred around the peak of the optimal size then there is a sequence of pixels: 1 pixel background, 2 pixels slope, 2 pixels tip, 2 pixels slope, 1 pixel background. Considering a rounded shape blob of the optimal size, the presence of the blob induces the peaks in at least two neighbouring rows (Figure 4.1). Therefore, it is enough to test only every second row.

There are *two necessary conditions* on the presence of the blob: (1) the peak in a row and (2) the peak in a column, both passing the tip of the blob. Applying one of them independently on the image will yield only a small fraction of pixels as candidates for blobs. Such candidates will form strings (called ridges bellow). Tips of blobs lay only on ridges.

The algorithm consists of *three parts*: preprocessing, search for ridges, search for blobs on ridges.

The number of pyramid levels is defined by the user or set to the value

$$\text{floor}\left(\log_2\left(\frac{\min(m,n)}{16}\right)\right),$$

where m, n is the number of rows, columns, respectively. This choice gives a theoretical chance at least to find some blobs even at the top level of the pyramid.

Preprocessing

1. Create the image pyramid according to Section 4.3.1.
2. Prepare the skip matrix $S(r, c) = 0$ (8-bit per pixel) for each level of the pyramid.
3. Compute D_h^c and D_v^c for each pixel at each pyramid level.

Search for the ridges

Algorithm scans every second row r. First, it checks the skip matrix (if $S(r, c) = 0$ then the pixel is processed otherwise it is skipped). Next, the horizontal coefficients are computed and compared to detect peak $b > 2a$. If the peak is found in a column c then rows $r + 1$ and $r - 1$ are tested for the peak b' of the same polarity, i.e. verifying $\text{sign}(b) = \text{sign}(b')$, at columns $c - 1$, c, $c + 1$. If the peak candidate $[r, c]$ has no neighbour in rows $r + 1$ or $r - 1$ then it is discarded.

The scan of the row continues at $c+2$ as the other extreme cannot be more close. If the peak is found only in $r - 1$ then it is considered to be a short ridge, it is remembered, and the scan of the row continues at column $c + 2$. If the peak is found in $r + 1$ the algorithm tries to reveal the ridge in the vertical direction. All peaks belonging to the ridge are marked in the skip matrix (together with $\pm$ 2 pixel neighbourhood) to avoid multiple detection of the same ridge. The skip matrix can contain just flag skip/don't skip or the number of pixels to skip.

Search for blobs on ridges

Go along the ridge from one end (e.g. $\min(r)$) and test vertical coefficients if the peak of the same polarity is present. If the peak is found in the row r, the columns $c+1$ and $c-1$ at rows $r-1$, r, $r+1$ are tested for the peak of the same polarity (similarly as in the ridge search above). If the neighbouring peak is found then the pixel $[r, c]$ is remembered as the blob otherwise it is discarded. The search continues to the end of the ridge. Note that the ridge may contain more than one peak.

Blobs detected at several scales

Most of the blobs pop up at more than one scale of the pyramid. The detections of the same blob at different scales may be treated independently in some applications. In other applications, the unique blobs at the best scale should be found. There are two possible approaches to address the latter situation:

1. Find the blobs independently at all levels of the pyramid and explore the nearest neighbours. The detections of blobs closer each to other than some d_{min} shows that the single blob and the detection with the maximal amplitude can be selected to represent the blob.

2. Process the pyramid in a top-down manner. Find blobs at the highest level, propagate them to the lower level to test their presence. Continue the propagation until no blob is found or the lowest level of the pyramid is reached. Choose the blob with the maximal amplitude. It will avoid the nearest neighbour search in the list of candidates.

4.4 Experiments

4.4.1 Calibration pattern

This experiment investigates the multiscale stability of the detector on a real image of a calibration pattern. The robustness of the detector against noise is tested on images generated by adding an artificial uniform noise to the real image. The blobs were detected using the algorithm described in Section 4.2 applied with different periods T.

Data

The data comes from the procedure for calibrating a camera against radial distortion using a printed calibration pattern. The calibration pattern consists of round dots centred in a grid 4×4 cm which were printed on A0 size paper. The round dots will be observed as blobs. The paper with the calibration pattern was fixed on a flat wall. The diameter of dots was 1 cm except of the two dots in the centre which marked the origin of coordinates and the orientation. The first special dot had the diameter 2 cm. The second special dot had the diameter 0.5 cm. The pattern was captured by a cheap fix-focus camera with the resolution 640×480 pixels, focal length 4.3 mm, pixel size 5.6 μm.

The plane of the calibration pattern was almost perpendicular to the principal axis. The distance between the camera and the plane was about 1 m. The amplitude of the observed blobs (i.e. the difference between the background intensity and the minimal intensity of the blobs) was between 0.5 and 0.8 when all the intensity values were in the range $\langle 0, 1 \rangle$. The shape of the blobs is almost ideal from the point of view of SWD. The real image, shown in Figure 4.2a, was artificially degraded by additive uniform noise with amplitudes 0.1, 0.33, 0.5 and 1.

Ground truth

A crucial part in an experimental evaluation of the precision and stability is a good choice of the ground truth. The ideal ground truth would be the 'real' locations of the blobs. Unfortunately, considering the experimental setup, the real locations are not available.

Instead of the unknown blobs' locations, the model of the measurement is used. The dots of the calibration pattern lay on horizontal lines. If captured by the ideal camera, the blobs should also lie on lines fitted through them. In this case, the localization error can be estimated as the distance of the detected points from the lines. However, a real camera exhibits distortions transforming lines to curves. Therefore we measure the localization error as the distance of the points from the curves fitted through them in the horizontal direction. The curves are modeled by parabolas.

Results

Figure 4.2b shows the parabolas fitted into detected points to demonstrate the extent of the radial distortion. The parabolas were shifted in the vertical direction to start from the row 0 for better visualization.

Figures 4.2c,d,e,f show the central part of the image containing the two special dots in detail to demonstrate multiscale stability of SWD. The detail of the original image is Figures 4.2d. The stars in Figure 4.2g,h,i show the distances of the detected points from the horizontal parabolas fitted into them. The large errors in Figure 4.2g correspond to the poor fit of the parabola in the corners of the image due to the significant radial distortion. Most of the points fit their parabolas with uncertainty less than 0.05 of pixel.

All points except those in the corners of image fit with uncertainty below 0.1 of pixel. The standard deviation is 0.039. The additive noise up to the amplitude $\nu = 0.1$ did not disturb the SWD significantly. The standard deviation increased to 0.047. Up to the noise level $\nu = 0.5$, the threshold on the minimal strength of response can be found such that all markers are detected and no false detection occur. The standard deviation increased to 0.101 for $\nu = 0.33$ and to 0.151 for $\nu = 0.5$. At the noise level $\nu = 1$, SWD could still find most of the markers at the expense of some false detections.

Table 4.1 shows the results of the blob detection from Figures 4.2d using different scale levels. The example of the blob #4 shows an excellent multiscale stability of the detector and the relevance of our uncertainty measure.

	Blob #	1	2	3	4	5	6	7
	S	1	0.5	1	2	1	1	1
T=8	R	217.89	219.25	248.95	248.86	250.86	220.11	218.61
	C	389.68	450.97	389.01	419.84	480.63	481.25	420.45
	U	0.012	0.221	0.008	0.405	0.013	0.034	0.013
T=16	R	217.20	-	248.95	249.49	250.86	220.13	218.62
	C	389.66	-	389.01	419.86	480.61	481.25	420.45
	U	0.004	-	0.004	0.065	0.007	0.005	0.005
T=32	R	-	219.32	248.95	249.52	250.86	220.13	218.61
	C	-	450.97	389.01	419.83	480.61	481.24	420.44
	U	-	0.036	0.003	0.003	0.004	0.008	0.013

Table 4.1: The multiscale stability. The results of SWD at different scale levels. S is the diameter of the dot in the calibration pattern. R stands for the row coordinate, C for the column coordinate, U is the multiscale uncertainty.

4.4.2 Natural images

Let us show detected blobs on a real image of a garden rockery. The purpose of the experiment is to display visually where blobs are detected. The second aim is to

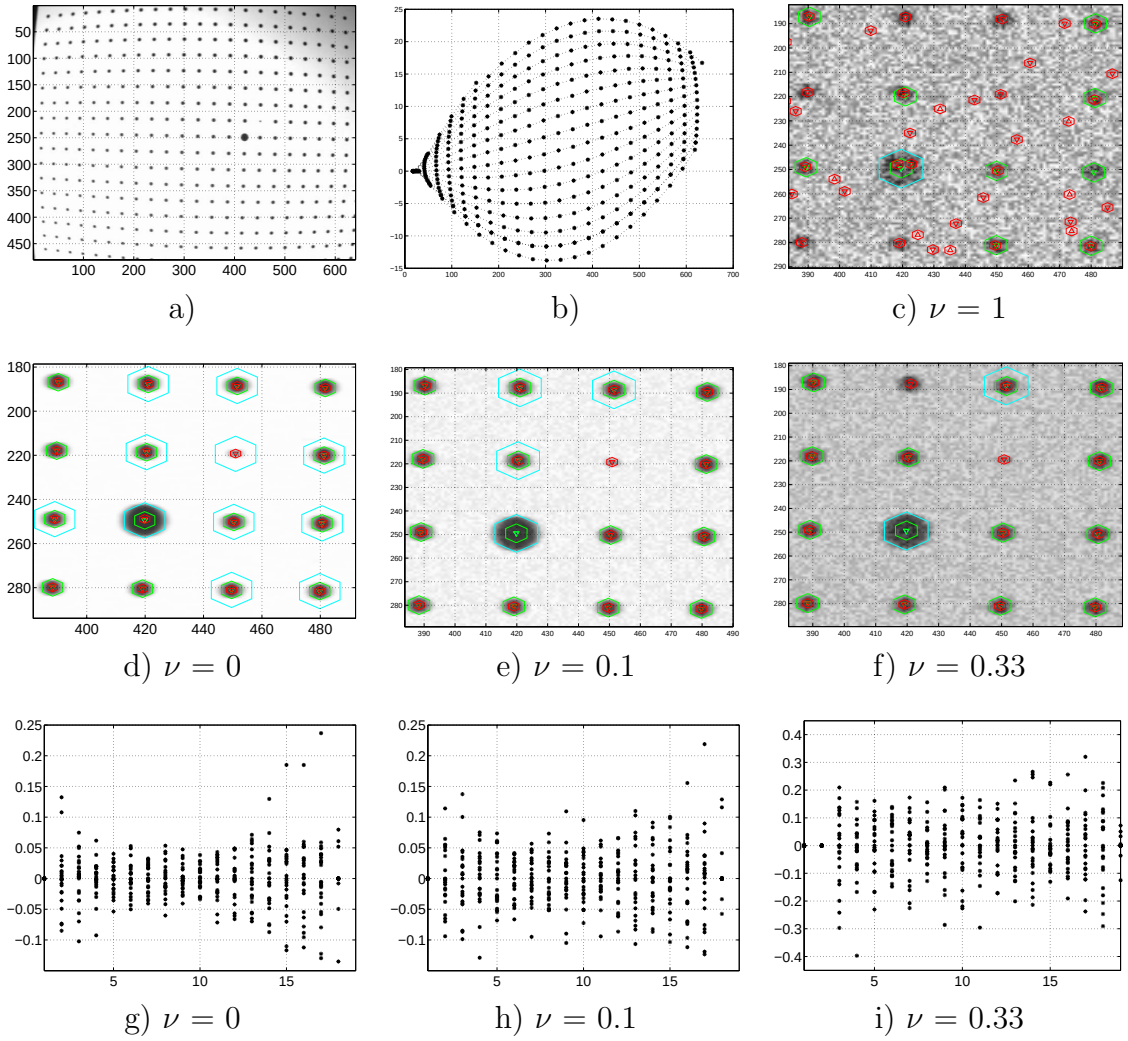


Figure 4.2: Robustness to noise. a) The whole calibration image. b) Distortion of horizontal lines, c), d), e), f) Details of the image with additive noise of amplitude ν. g),h), i) Deviation from parabolas fitted through points in the horizontal direction. Blob locations found by SWD-2D are depicted as triangles. The diameter of hexagons indicates the scale.

demonstrate the potential of the proposed method for stereo matching. The garden rockery was captured from three different view points. The reader can see that many blobs in the first image have a corresponding partner in the second and in the third image.

4.4.3 Comparison to the state-of-the-art

The diploma student Martin Pejčoch performed extensive experiments with SWD in 2D. He also created a visualization tool to demonstrate the behaviour of the algorithm from Section 4.3. He also performed the comparison with the state-of-the-art interest point detectors following the methodology of [29]. His work is described in the diploma thesis [34].

The results of the comparison with detectors were not very encouraging. Our Stable Wave-based blob detector had worse repeatability than most of the others. The positive result for the Stable Wave blob detector was that there is a small percentage (20-30%) of points detected by our detector which are more precise than the points detected by other detectors. Charts on pages 24–31 of the diploma thesis [34] demonstrate this phenomenon.

The experiment was conducted by Martin Pejčoch, in which the dependence of the repeatability on the allowed maximal localization error was evaluated. If the allowed localization error was up to 0.3 sometimes even up to 0.5 of the pixel width then our detector manifested a higher repeatability than other detectors. If a bigger localization error was allowed then our detector was overcome by other detectors in many cases. This was a rather disappointing result.

The niche, in which the Stable Wave blob detector performs better than its state-of-the-art competitors, was found in tracking in videosequences. Here, speed, precision and sparseness of the Stable Wave blob detector bring competitive advantages. In addition, the repeatability of our detector together with robustness to bigger deformations is comparable or even better than state-of-the-art methods have been providing lately. Use of the Stable Wave blob detector in tracking will be explicated in the next chapter.

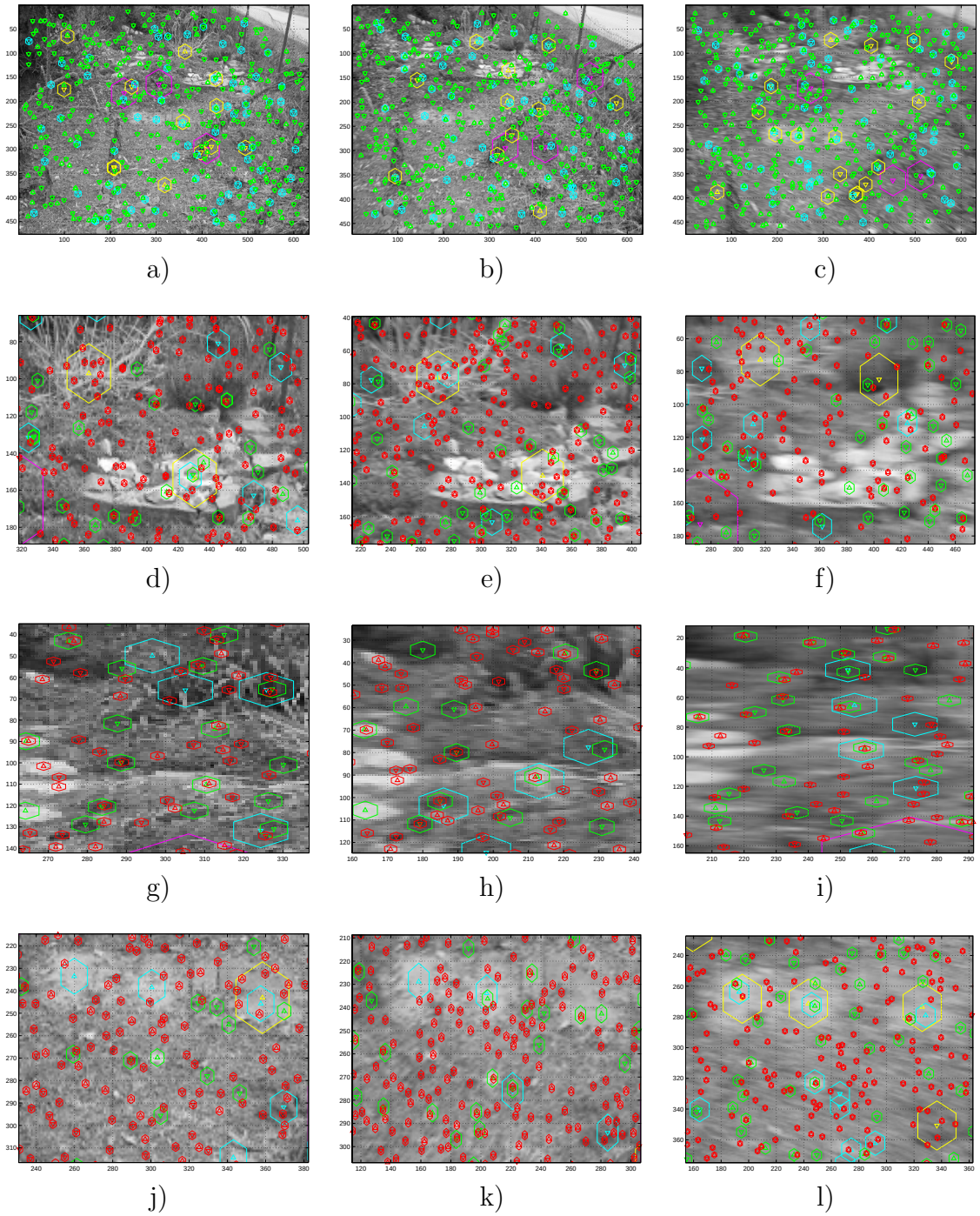


Figure 4.3: Images from the nature. The garden rockery.

4.4.4 A simple single view experiment in 3D

The aim of this experiment was to demonstrate usefulness of SWD in a simple navigation task. Namely, the goal was to measure a distance to an artificial label observed by a cheep camera. A detailed description of the camera is given Section 4.4.1. A very simple measurement method together with a simple experimental setup were chosen deliberately in order to minimize unknowns and to find ground-truth easily.

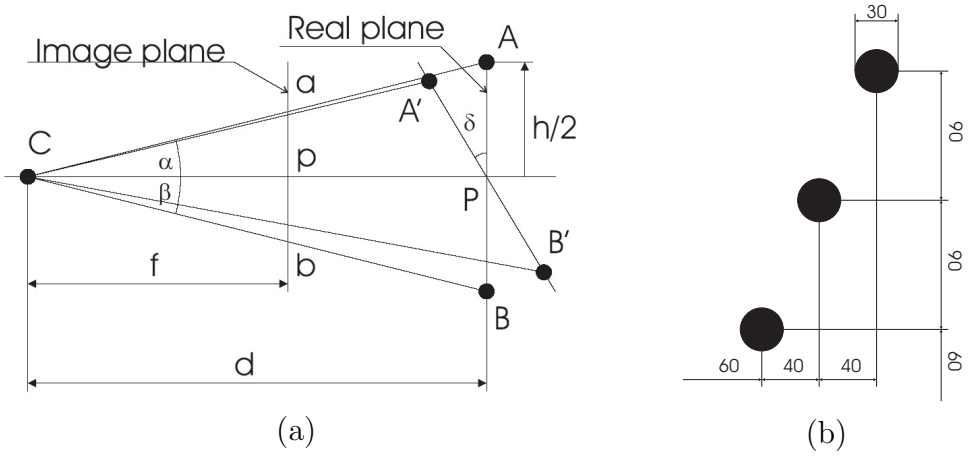


Figure 4.4: a) The setup of the distance measurement experiment from a single view. b) The target pattern used in the experiment.

Theory

A simple method for measuring the distance between the camera and the artificial label, shown in Figure 4.4b, is used as a benchmark for evaluating the impact of the localization error in 2D to the precision of spatial measurements, namely the depth. Figure 4.4 shows the principle of the method. The triangle similarity is used to compute the distance between the plane with two markers and the camera. The analysis can be carried out in 2D because in the real 3D world one can always find the plane containing the markers **A** and **B** and the camera centre **C**.

Considering the real plane with markers being parallel to the image plane, the distance d can be computed from the triangle similarity:

$$\frac{d}{|AB|} = \frac{f}{|ab|} \tag{4.7}$$

$$d = \frac{f|AB|}{|ab|} \tag{4.8}$$

In practical situations, it is difficult to keep the planes exactly parallel. However, if $d \gg |AB|$ and the deviation δ is reasonably small, this systematic error can be neglected as compared with the uncertainty of the localization of markers in the picture. An estimate of the bounds on the systematic error can be derived as follows:

First, assume the image plane parallel to the target plane, $|ab| = |ap| + |bp|$, $|ap| = f \tan\alpha = fk$,

$$k = \tan\alpha = \frac{h/2}{d}\,.$$

In the general case when the planes are not parallel, the point a would move to a' and the angle α would change to α', the distance $|a'p| = fk'$, where

$$k' \;=\; \tan\alpha' \;=\; \frac{(h/2)\cos\delta}{d - (h/2)\sin\delta} \;=\; \frac{h/2}{d}\;\frac{\cos\delta}{1 - \frac{(h/2)}{d}\sin\delta}.$$

The first fraction is equal to k in the parallel arrangement ($\delta = 0$). The second fraction is near to 1 if δ is small. In the typical situation, when $d \gg |AB|$ and δ is small, $\delta < 10^o$, the denominator turns almost 1. The denominator becomes practically insensitive to δ. The whole second fraction can be approximated well by $\cos\delta$ which is near to 1 up to a relatively large angle (e.g. 5^o correspond to 0.4% error, 10^o correspond to 1.5% error, 20^o correspond to 6% error).

Figure 4.4 shows the situation in which the middle point **P** between **A** and **B** lays on the principal axis. Generally, there is an angle φ between the line **CP** and the principal axis. The angle φ causes additional uncertainty because Equation 4.8 measures the perpendicular distance from the camera centre to the real plane, not to the point **P**. The angle φ can be easily measured from the image and used while computing the distance. Without knowledge of δ, the effect of φ cannot be fully compensated. However, the error can be significantly reduced if δ is small. The angle φ can enlarge or compensate the error caused by δ.

The results of numerical simulations which predicted the estimate error for various combinations of φ and δ values are summarized in Table 4.2 and in Figure 4.5.

The setup of the experiment

The label consists of three black circles printed on a white paper as shown in Figure 4.4 b. The marker was designed to fit A4 size paper (to be easily printed on a standard printer). The black circle on the white background is an ideal marker for SWD. The period of SWD should be near to the double of the marker size. The minimal distance between markers should be three times longer than the marker size to avoid their interference. The minimal distance of the marker from the border of the label should be longer than the double of the marker size in order to avoid interference of background patterns.

The plane containing labels was approximately parallel to the image plane of the camera. The angle δ slightly changed for individual snapshots but its value was always in the range $\langle -5^o, 5^o \rangle$. The distance of the label from the camera was adjusted with precision ±2cm. For each position, about 10 images were captured with slightly varying angles φ and δ.

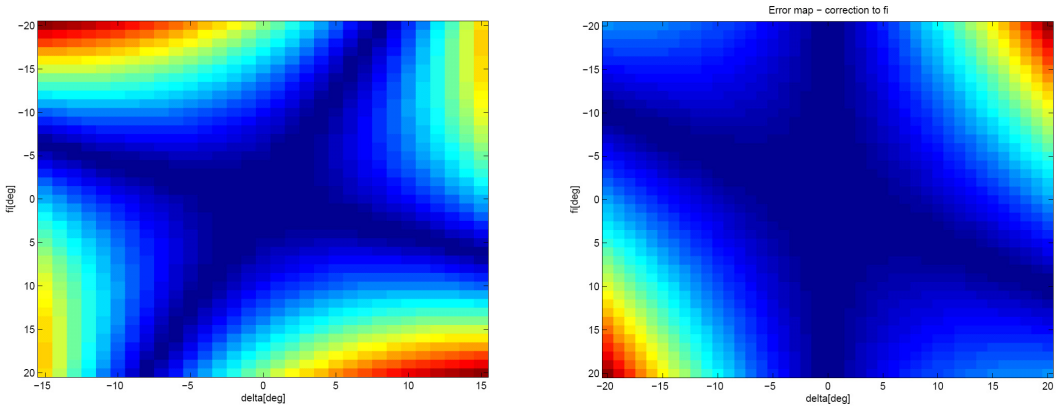
Error[%]		δ[deg]			
		<-5,5>	<-10,10>	<-15,15>	<-20,20>
φ[deg]	<-5,5>	0.7	2.7	5.6	9.5
	<-10,10>	2.7	3.2	7.0	12.0
	<-15,15>	5.2	6.3	7.7	14.0
	<-20,20>	8.6	10.3	11.3	15.3

a)

Error[%]		δ[deg]			
		<-5,5>	<-10,10>	<-15,15>	<-20,20>
φ[deg]	<-5,5>	1.1	3.1	6.0	10.0
	<-10,10>	2.0	4.8	8.7	13.7
	<-15,15>	2.8	6.6	11.5	18.9
	<-20,20>	3.7	8.5	14.7	22.7

b)

Table 4.2: Theoretical error. a) The maximal error of the simple method for various ranges of φ and δ. b) The maximal error after the correction to φ for various ranges of φ and δ.



a) b)

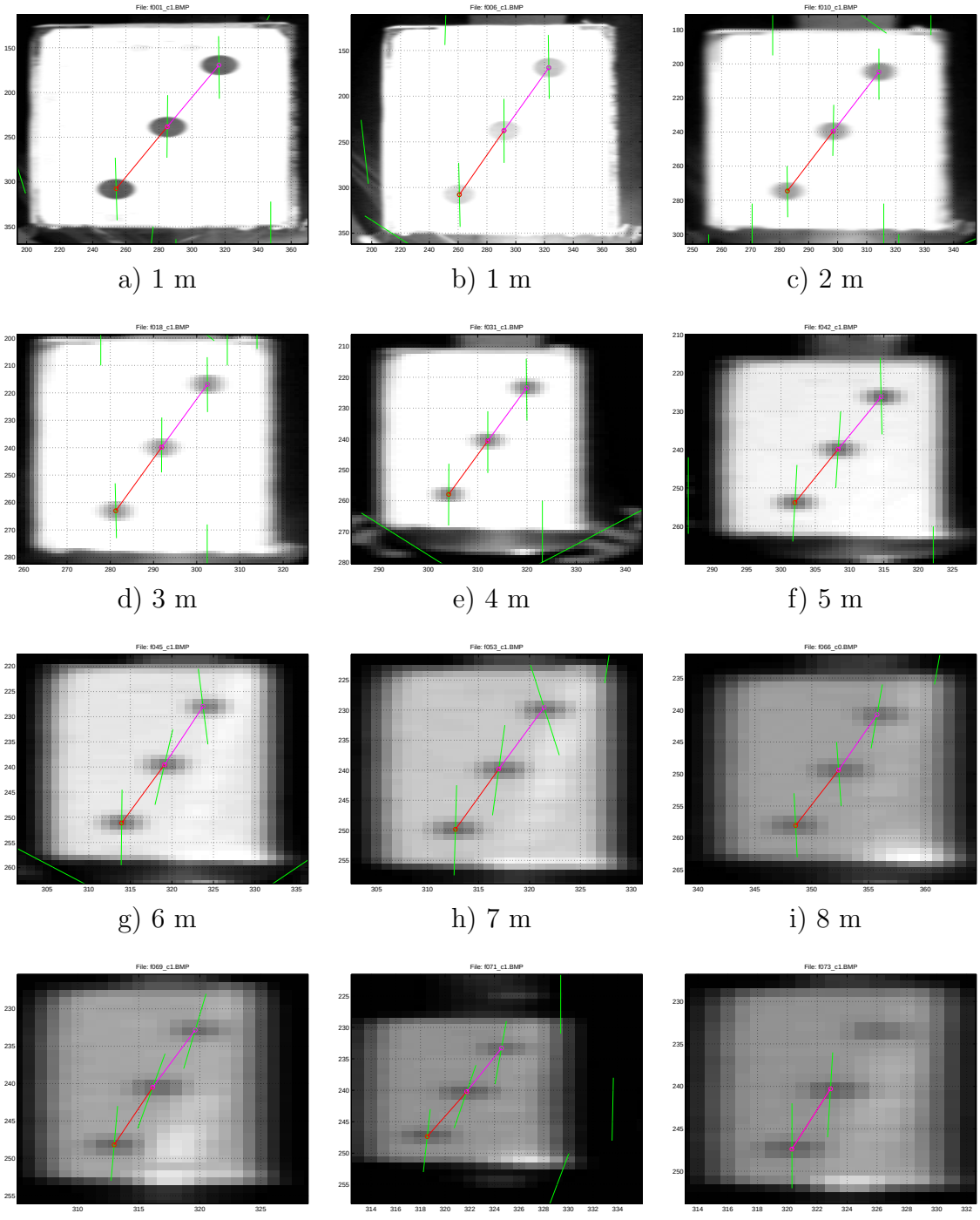
Figure 4.5: Theoretical error. b) The error map for the simple method. Warm colors correspond to the high relative error. d) The error map after the correction to φ. The warm colors correspond to the high relative error.

Real distance[m]	1.0	2.0	3.0	4.0	5.0	6.0	7.0	8.0	9.0	10.0
Mean est. dist.[m]	1.004	1.981	2.983	3.992	4.973	5.991	6.944	8.042	9.065	9.954
Mean abs. error [mm]	9.6	20.2	17.3	16.1	35.9	37.2	70.0	54.2	116.0	112.5
Mean abs. error [%]	1.0	1.0	0.6	0.4	0.7	0.6	1.0	0.7	1.3	1.1
Max. abs. error [mm]	32.6	39.2	36.8	40.2	82.2	91.6	207.5	199.8	364.4	318.5
Max abs. error [%]	3.3	2.0	1.2	1.0	1.6	1.5	3.0	2.5	4.0	3.2
Marker size [pix]	24.0	11.0	7.0	6.0	6.0	5.0	4.0	4.0	3.0	3.0
Marker distance [pix]	75.4	38.2	25.4	18.9	15.2	12.6	10.9	9.4	8.3	7.6

Table 4.3: The error of the measurement of the distance between the camera and the target plane.

Results

Table 4.3 summarizes results of the experiment. The distance between markers measured in the camera image ranged from 75 pixels for $d = 1$m to 7.6 pixels for $d = 10$m. The change in the image distance between the markers for $d > 8$m is less than 1 pixel. However, the localization error is still better than 6cm at $d = 8$m. It corresponds to 0.06 pixel precision in measurement of the marker-to-marker distance. Figure 4.6 shows examples of the captured images for different distance d of the target.



a) 1 m b) 1 m c) 2 m

d) 3 m e) 4 m f) 5 m

g) 6 m h) 7 m i) 8 m

j) 9 m k) 10 m l) 10 m

Figure 4.6: The image of the target observed from different distances (enlarged details). a), b) The effect of varying lighting conditions. The target became too small in 10m distance, one or more blobs were lost in some of the images e.g. l). However, if the target is detected then the localization precision was good k).

5 The ultrafast low-level tracker

This chapter deals with the algorithm establishing *point to point correspondences* between *two consecutive images* in a video sequence or any other pair of images. The displacement between images must be small or predicted in advance with reasonable precision[1].

The *low-level tracker* takes *two inputs*: (1) the *predicted location* of the key point, e.g. the key point detected in the previous frame of the sequence and (2) *the image* in which the point should be found which is the current frame of the sequence. The low-level tracker finds the new and a more precise position of the point in the current frame.

This *ultrafast tracker* was motivated by the original Stable Wave idea and namely by the iterative algorithm described in Section 3.1. Several competing trackers [31, 3] explore the following idea. The key points are detected in the previous and the current frame by testing all pixels in the $N \times N$ neighbourhood to find several good ('highly curved') pixels. This seek provides several potential correspondences in the neighbourhood linking one pixel from the previous frame to one from several pixels in the current frame. The best match is selected by evaluating another criterion assessing similarity of pixel neighbourhood in previous and current frames. Usual criteria for best match selection are correlation, sum of square differences, etc.

The proposed approach *needs to check much less pixels in the current frame*. The suggested tracker works similarly as the Newton's method used in optimization. Having the initial pixel position, the tracker computes (estimates) the needed shift (the direction and the distance) to the key point and makes this shift. This step is iterated. The estimates become more and more precise as the algorithm approaches to the key point. The point $\mathbf{x}_i$ in iteration i is considered as the key point if $\mathbf{x}_i$ is closer to $\mathbf{x}_{i-1}$ than some threshold. Typically as few as two to five iterations are necessary. At the key point, the *predicted shift is near to zero*, therefore the key points of the tracker are called *Zero-Shift-Points* (ZSP) in the rest of the chapter.

If the initial estimate is precise enough then it is not necessary to consider and compare multiple tentative correspondences, the proposed algorithm just finds the correct one.

5.1 The concept of Zero Shift Points

Since ZSPs are defined in terms of local shift vectors Δ, we first explain their computation. The idea behind is given is Section 4.1. The mapping f which maps pixel $\mathbf{y} \in \mathbb{Z}^{2+}$ to shift vector $\Delta^+ \in \mathbb{R}^2$ estimates the position of the maximum of the first harmonic wave of the window centred at the pixel $\mathbf{y} = [r_0, c_0]$. Similarly, Δ^- estimates the position of the minimum. The elements of the shift vectors $\Delta^+ = [\delta_h^+, \delta_v^+]$ and $\Delta^- = [\delta_h^-, \delta_v^-]$ are computed according to Equation (5.1). The subscript '$_\star$' in

[1] The alowable range of displacement is evaluated in Section 5.5.1.

$\delta^+_\star$, $\delta^-_\star$, $a_\star$, $b_\star$ is either 'h' (horizontal) or 'v' (vertical), respectively:

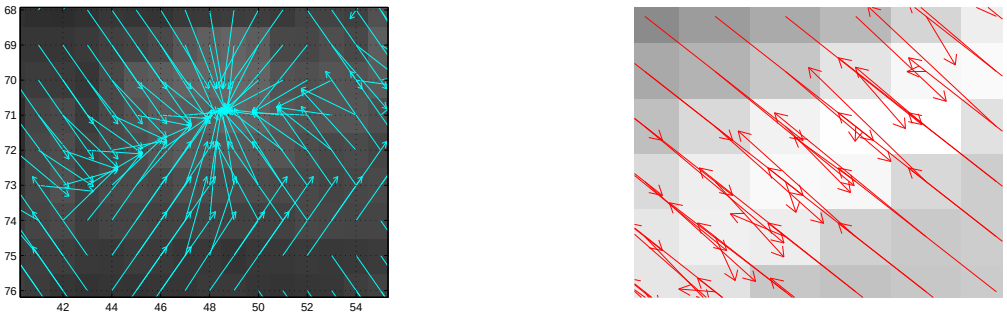
$$\begin{aligned}\delta^+_\star &= \begin{cases} T\arctan(a_\star/b_\star)/(2\pi) & \text{if} \quad (b<0) \\ -T\,\mathrm{sign}(a_\star)/4 & \text{otherwise} \end{cases} \\ \delta^-_\star &= \begin{cases} T\arctan(a_\star/b_\star)/(2\pi) & \text{if} \quad (b>0) \\ T\,\mathrm{sign}(a_\star)/4 & \text{otherwise} \end{cases}\end{aligned} \tag{5.1}$$

where $a_\star$ and $b_\star$ stand for sine and cosine coefficients. They are computed according to Equation (5.2) from rectangular windows of the length T and width W centred at the point $\mathbf{y} = [r_0, c_0]$. Both T and W are odd integers. Shift δ^* is limited to a fixed size when being far from the extreme of a desired polarity. Any other estimate would not be reasonably precise either. The goal is just to move away from the opposite extreme. The fixed step also reduces the computational cost.

The best results were obtained when W was the nearest odd integer to $T/2$. The coefficients are defined as

$$\begin{aligned} a_h &= \textstyle\sum_{i=-w}^{w}\sum_{j=-t}^{t} I(r_0+i, c_0+j)\,\mathbf{S}(j+t), \\ b_h &= \textstyle\sum_{i=-w}^{w}\sum_{j=-t}^{t} I(r_0+i, c_0+j)\,\mathbf{C}(j+t), \\ a_v &= \textstyle\sum_{i=-t}^{t}\sum_{j=-w}^{w} I(r_0+i, c_0+j)\,\mathbf{S}(i+t), \\ b_v &= \textstyle\sum_{i=-t}^{t}\sum_{j=-w}^{w} I(r_0+i, c_0+j)\,\mathbf{C}(i+t), \\ w &= (W-1)/2, \quad t = (T-1)/2, \\ \mathbf{C}(i) &= \cos(\phi_i), \quad \mathbf{S}(i) = \sin(\phi_i), \\ \phi_i &= 2\pi(i+0.5)/T, \quad i = 0, 1, \ldots, T-1. \end{aligned} \tag{5.2}$$

The point $\mathbf{z}$, in which Δ^+ or Δ^- becomes (approximately) the zero vector, is called a *zero shift point* (ZSP). There are two sets of ZSPs, ZSP^+ associated with maxima (and the Δ^+ field) and ZSP^- with minima (and Δ^-) of the intensity function, respectively. Since the processing of the two sets is identical and independent, we do drop the superscripts in the sequel. Such points are subpixel entities, i.e. they rarely appear at pixel centres. Vectors Δ at pixels around $\mathbf{z}$ are pointing close to $\mathbf{z}$ as depicted in Figure 5.1.



(a) (b)

Figure 5.1: Examples of typical shift fields (a) near a single ZSPs in a blob-like region and (b) near a ridge with multiple ZSPs.

Typical *locations of ZSPs are centres* of approximately *elliptical 'blobs'*, i.e. symmetric areas with higher/lower intensity then their neighbourhood (Figure 5.1a), narrow edge features (Figure 5.1b) and flat areas which are bigger than the length T of the moving window. ZSPs on ridge features and flat areas are not suitable for tracking since they are unstable and can be filtered out by simple rules, see Section 5.3.

5.2 The tracking algorithm

The following simple iterative algorithm tracks a single ZSP from its position $\mathbf{x}_0 \in \mathbb{R}^2$ the previous frame (or from a prediction of its position in the current frame) to a subpixel position $\mathbf{x} \in \mathbb{R}^2$ in the current image. The parameters of the algorithm are: the zero tolerance z (default $0.05\,T$), the maximum tracking distance d_m (default $T/2$), and the maximum number of iterations n (default 8). The default values were chosen experimentally.

1. $i = 1$, $\mathbf{y}_0 = \text{round}(\mathbf{x}_0)$

2. *Test the closeness of* $\mathbf{y}_{i-1}$ *to image margins.*
if $\mathbf{y}_{i-1}$ *is more near to the border than* $T/2 + 1$ **then**
| Fail.
end

3. *Compute the shift vector* Δ^* *in pixel* $\mathbf{y}_{i-1}$ *according to Equation (5.1). The superscript '*$\star$*' in* $\Delta^\star$ *is either '*$+$*' if the minimum is tracked or '*$-$*' if the maximum if tracked.*
The new position is $\mathbf{x}_i = \mathbf{y}_{i-1} + \Delta^*$.

4. *Test the convergence.*
if $||\mathbf{x}_i - \mathbf{x}_{i-1}||_\infty < z$ **then**
| $\mathbf{x} = \mathbf{x}_i$. Finish.
end

5. *Test the divergence.*
if $||\mathbf{x}_i - \mathbf{y}_0||_\infty > d_m$ **then**
| Fail.
end

6. *Prepare the next iteration.*
$\mathbf{y}_i = \text{round}(\mathbf{x}_i)$, $i = i + 1$.
if $i < n$ **then**
| Go to step 2
else
| Fail.
end

Algorithm 5: Tracks a single Zero Shift Point

The *choice of the maximum distance* d_m depends on the application. If the between-frames movement is small or well predictable then d_m may be $T/4$ or even

smaller. Smaller value of d_m saves time spent on the points which would be lost anyway. Increasing d_m over T will typically only increase the number of miss matches and increase the tracking time. Considering the shape and the size of a typical attraction basin (Section 5.5.1), the reasonable choice is $T/2$. If the predictions are precise then also the maximum number of iterations n can be a small number. The typical average number of iterations is about 2-5. Increasing n over 8 typically does not bring any effect. Avoiding the points near the image borders allows skipping step 2 of Algorithm 5. The points to track should be more far from the image borders than $d_m + T/2 + 1$.

5.3 Implementation issues

The use of *integral images* significantly speeds up the tracking of a large number of points. Tracking of only few (up to several tens) points with a small period may be faster without integral images. The integral images are cumulative sums along image rows and columns:

$$\begin{aligned} J^h(r,0) &= I(r,0), \quad J^h(r,c+1) = J^h(r,c) + I(r,c+1), \\ J^v(0,c) &= I(0,c), \quad J^h(r+1,c) = J^h(r,c) + I(r+1,c). \end{aligned} \tag{5.3}$$

The *2D convolutions* for sine and cosine coefficient evaluation (see Equation 5.2) can be efficiently computed if the intensity values are first summed along the width of the window to get vector $\mathbf{V}$ of length T. Then, the coefficients are computed as the dot products of the base vector $\mathbf{S}$ or $\mathbf{C}$ with $\mathbf{V}$. By using integral images, the sum of W values for each element of $\mathbf{V}$ can be replaced by a single subtraction. The coefficients related to the pixel $[r_0, c_0]$, $r_0 > T/2$, $r_0 < m - T/2$, $c_0 > T/2$, $c_0 < n - T/2$, can be efficiently computed as follows:

$$\begin{aligned} &\mathbf{V}^h(i+t) = \textstyle\sum_{j=-w}^{w} I(r_0 + j, c_0 + i) = \\ &\qquad = J^v(r_0 + w, c_0 + i) - J^v(r_0 - w - 1, c_0 + i), \\ &\mathbf{V}^v(i+t) = \textstyle\sum_{j=-w}^{w} I(r_0 + i, c_0 + j) = \\ &\qquad = J^h(r_0 + i, c_0 + w) - J^h(r_0 + i, c_0 - w - 1), \\ &w = (W-1)/2, \quad t = (T-1)/2, \quad i = -T/2 \ldots T/2, \\ &a_h = \mathbf{S} \cdot \mathbf{V}^h, b_h = \mathbf{C} \cdot \mathbf{V}^h, a_v = \mathbf{S} \cdot \mathbf{V}^v, b_v = \mathbf{C} \cdot \mathbf{V}^v. \end{aligned} \tag{5.4}$$

The computational cost of Algorithm 5 depends mainly on the following three operations:

1. *Preprocessing*, i.e., the integral image calculation: requires just two integer additions per pixel.

2. *Calculation of the coefficients*: requires T integer subtractions and T integer multiply-accumulate instruction per coefficient; four coefficients are necessary per each iteration of a single point.

3. *Computation of the shift vector* Δ: requires 2 floating point division, 2 floating point arctan operations and 2 floating point multiplications per each iteration of a single point. The low level tracker needs about 4 iterations for each point.

The low level tracker spends less than 10μs on each point per frame on the HP6540b notebook. The implementation runs in a single thread and some parts of the code are still far from being optimal.

5.4 Good points to track

A good point to track should be *unique in its neighbourhood* and the tracking algorithm should be able to follow it over a wide range of disturbances. There are 'simple' disturbances like translation, rotation and scale change. In these cases, it is possible to identify easily ZSPs, which will be sufficiently robust, just by testing the original image where the points were detected. Some ZSPs are resistant to the affine transform distortion up to some extend (when the ellipse becomes too elongated). The perspective distortion destroys the symmetry. Thus, it always reduces the precision of the estimated point location. If the perspective distortion is more severe then it causes disappearance of the ZSP.

The *best point to track* by Algorithm 5 is the *centre of an ideal blob* like the one shown in Figure 4.1. The *localization* of such point is completely rotation invariant and robust to the scale and affine change in a wide range. In real images, there are almost no ideal blobs. However, some patches are similar to them and can act as good targets for tracking.

On the other hand, there are some c*learly unsuitable ZSPs, which can be easily discarded*. These bad ZSPs are in flat areas (one or both of $b_\star$ coefficients are small) and on ridge features (Figure 5.1b). The unstable ZSPs $[r, c]$ on ridge features can be detect using criterion

$$|\delta_v(r, c \pm t)| > kt \quad \text{or} \quad |\delta_h(r \pm t, c)| > kt\,, \quad t = \text{ceil}(T/8)\,, k < 1\,, \tag{5.5}$$

where k is a tuning parameter (default 0.8).

5.4.1 Period refinement algorithm, rank

The *goal* of the period refinement is to *improve the stability* of ZSP in a scale-space. Even though good ZSPs are robust to scale change over an octave, it is beneficial to optimize the period which will be used for tracking. Placing the period to the middle of the range would increase robustness to unpredictable scale change. Sudden big scale changes are not much common in video sequences. More common is continuously changing scale as the target approaches or recedes. In such situation, the period minimizing $\delta\mathbf{x}/\delta T$ while keeping reasonable margin improve both precision and robustness.

Let $\mathbf{x}_T$ be ZSP detected at the period T and $\mathbf{y} = \text{round}(\mathbf{x}_T)$. Algorithm 6 searches $\mathbf{x}'_{T'}$ which is less sensitive to the scale change than $\mathbf{x}_T$ and evaluates the rank

$r \in \{0, 1, 2\}$ of ZSP. The rank is defined by the algorithm and measure the robustness of ZSP to the scale change.

Algorithm 6 is ad hoc, it does not try to find optimal period, just to improve the original guess. Experiments shows that the points which received higher rank have significantly higher chance to survive geometric transformation. The recommended strategy is to throw away points with small period and small rank because for a good precision of global movement estimation having less number (but still many) of more precise points is better. To maintain long range of displacement between frames as many as possible points with long period is useful. Even the point of low (zero) rank has reasonable chance to be tracked. There are several hundreds or thousands of points with the shortest period, however, just several points of the longest period. The number of detected point usually falls quadratically with period.

5.4.2 Search for good points to track

Good blobs to track are detected as ZSPs for periods in a wide range around of the optimal period. The locations of ZSPs localizing such blobs vary only slightly with the change of the period. This is important for two reasons. First, when the scale between consecutive images changes then the point can be localized using the same period with a small error. Second, when searching good ZSPs in the first image (or new ZSPs in a changing scene) then it is not necessary to scan all possible periods (all odd numbers from 5 to approximately a quarter of the size of the image) but only some selected periods in the exponential series (e.g., the series 9-19-39-79-159 was used in experiments of this chapter for typical images 640×480).

It is not necessary to test each pixel for the zero-shift condition since ZSPs suitable for tracking lie inside a basin of attraction with the size approximately equal to $T/2$ or bigger. Therefore it is enough to start tracking (Algorithm 5) from points on a regular grid with a period of $\approx T$.

The default choice of the maximal tracking distance is d_m^i. A recommended value for tracking is the minimal radius of a circle with the centre in ZSP inscribed into its attraction basin.

The choice of a grid of the size g allows the overlap of the grid fields. Smaller value reduces the risk of losing ZSP, however, the computation time grows quadratically with the reduction of g.

5.5 Experiments

The experiments presented below focus mainly on the performance of the low level tracker and examine its properties necessary for its successful integration into the higher level tracking algorithm. The most important properties are the range of the tracker (the maximal inter-frame disparity or the predictor error which the tracker can handle), the ability of the tracked ZSPs to survive photometric and geometric changes, and the precision of tracking.

The performance plots in Sections 5.5.1 and 5.5.2 were obtained mainly using the beginning of the 'Mouse pad' sequence [45], which is available online. The low

1. Compute $\Delta^p(y,T)$, $\Delta^p(y,T_+)$ and $\Delta^p(y,T_-)$ according to Section 5.1, where p is the polarity of $\mathbf{x}_T$, T_+ is the nearest odd integer to $T+T/4$ and T_- is the nearest odd integer to $T-T/4$.

2. Test the instability.
if $||\Delta^p(\mathbf{y},T_+)-\Delta^p(\mathbf{y},T)||_2 > T/8 \quad \& \quad ||\Delta^p(\mathbf{y},T_-)-\Delta^p(\mathbf{y},T_1)||_2 > T/8$ **then**
  Rank $r=0$, $\mathbf{x'}_{T'}=\mathbf{x}_T$.
  Fail.
end

3. Decide about the direction.
if $||\Delta^p(\mathbf{y},T_+)-\Delta^p(\mathbf{y},T)||_2 < ||\Delta^p(\mathbf{y},T_-)-\Delta^p(\mathbf{y},T)||_2$ **then**
  Set $T_0=T_-$, $T_1=T_+$ and T_2 to the nearest odd integer to $T_++T_+/4$.
else
  Set $T_0=T_+$, $T_1=T_-$ and T_2 to the nearest odd integer to $T_--T_-/4$.
end

4. Compute $\Delta^p(\mathbf{y},T_2)$.

5. Test the instability.
if $||\Delta^p(\mathbf{y},T_1)-\Delta^p(\mathbf{y},T)||_2 > T/8 \quad \& \quad ||\Delta^p(\mathbf{y},T_2)-\Delta^p(\mathbf{y}T_1)||_2 > T/8$ **then**
  Rank $r=0$, $\mathbf{x'}_{T'}=\mathbf{x}_T$.
  Fail.
end

6. Decide about the change of the period and the location.
if $||\Delta^p(\mathbf{y},T_0)-\Delta^p(\mathbf{y},T)||_2 < ||\Delta^p(\mathbf{y},T_2)-\Delta^p(\mathbf{y},T_1)||_2$ **then**
  the original period is the best, $\mathbf{x'}_{T'}=\mathbf{x}_T$, **if**
  $||\Delta^p(\mathbf{y},T_0)-\Delta^p(\mathbf{y},T)||_2 < T/8 \quad \& \quad ||\Delta^p(\mathbf{y},T_1)-\Delta^p(\mathbf{y},T)||_2 < T/8$ **then**
    Rank $r=2$
  else
    Rank $r=1$.
  end
else
  $\mathbf{x'}_{T'}=y+\Delta^p(\mathbf{y},T_1)$, $T'=T_1$.
  if $||\Delta^p(\mathbf{y},T_0)-\Delta^p(\mathbf{y},T)||_2 < T/8 \quad \& \quad ||\Delta^p(\mathbf{y},T_1)-\Delta^p(\mathbf{y},T)||_2 < T/8$ **then**
    Rank $r=2$.
  else
    Rank $r=1$.
  end
end
The refinement finished successfully.

Algorithm 6: Period refinement

1. i = 0.
2. Set the maximal tracking distance to $d_m^i = \text{floor}(T_i/2) + 1$.
3. Prepare start points $\mathbf{y}_j^i$ on a rectangular grid of the size $g = 2d_m^i - T/8 - 1$.
4. Track all points $\mathbf{y}_j^i$ to the nearest ZSP $\mathbf{x}_j^i$ using Algorithm 5.
5. Remove the points on oblique contours (Equation 5.5) and the points in flat areas (with small $|b_\star|$).
6. Refine the period according to Algorithm 6.
7. Remove the duplicate points, i.e., the points which have the same polarity and their mutual distance is below $T/2$.
Keep one of the neighbors with the highest rank. If more close points (more near than $T/2$) have the same rank choose the point with the higher energy $|b_h + b_v|$.
8. **if** $T_i >= \min(m, n)/4$ *($[m, n]$ is the size of image.)* **then**
| Finish.
end
9. $i = i + 1$, $T_i = 2\,T_{i-1} + 1$, go to Step 2

Algorithm 7: Search for good points to track.

level tracker was also tested on other publicly available sequences used in works of others [45, 17, 21]

- `http://cmp.felk.cvut.cz/demos/Tracking/linTrack/data/index.html`
- `http://info.ee.surrey.ac.uk/Personal/Z.Kalal/TLD/tld_dataset.ZIP`
- `http://www.metaio.com/research`

5.5.1 Attraction basins

The experiment evaluates the range of displacements between images (or the prediction error), which the ZSP tracker handles. The *attraction basin* of ZSP $\mathbf{x}$ is a set of pixels from which the tracking algorithm reaches $\mathbf{x}$ with some tolerance θ. Figure 5.2 shows the attraction basin of ZSP with polarity '+' and rank 2 grouped according to the period of ZSP. Notice that the basins are not symmetric and ZSP is usually located far from the centre. The density of ZSPs with '-' polarity and their basins are similar (not shown).

The *cumulative attraction basin* of all ZSPs of the period T is constructed by Algorithm 8.

Figure 5.3 shows the shapes of cumulative attraction basins (a, c, e) and the estimate of the tracker range (b, d, f) in the first image of the 'Mouse pad' sequence. The graphs were formed by individual pixel statistics of the image of the attraction basin, i.e. each blue point in the graph corresponds to a pixel in the image of cumulative attraction basin and depicts the relation between the radius of the pixel from ZSP and the number of basins covering the pixel. The red curve connects the minimal numbers of basins for a given radius. This curve predicts the minimum number

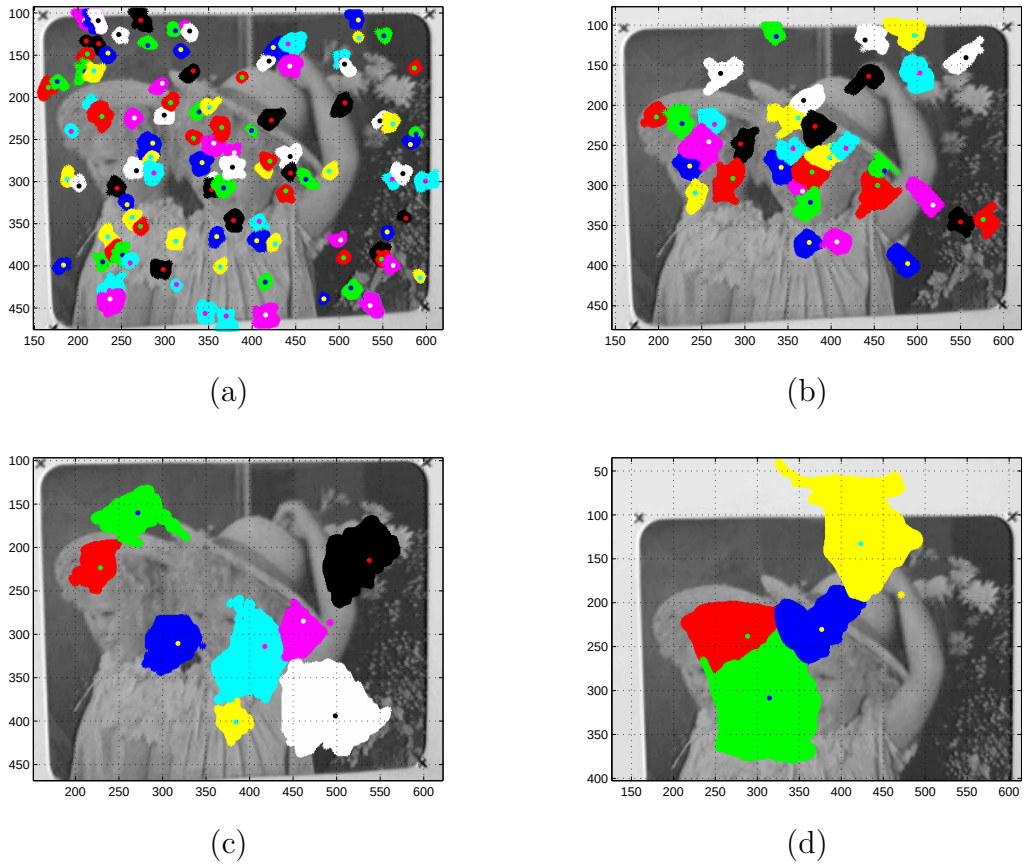


Figure 5.2: Attraction basins in the first image of the 'Mouse pad' sequence. ZSPs found with period a) 9, b) 19, c) 39, d) 79. Polarity of ZSPs is $+$. The dot of different color inside of each basin marks location of ZSP.

1. Find the stable ZSPs $\{\mathbf{x}(T)_i, i = 1 \dots N_T\}$ (where N_T is the number of ZSP of period T) in the image $\mathbf{I}$ according to Section 5.4.2.
2. Track each pixel $\mathbf{p}_j$, $j = 1 \dots M$ (where M is the number of pixels in the image $\mathbf{I}$) of the same image $\mathbf{I}$ to $\mathbf{p'}_j$ by the Algorithm 5 using the period T.
3. Find the attraction basin A_i of $\mathbf{x}_i$ as a set $\{\mathbf{p}_j\}$ such that $||\mathbf{x}_i - \mathbf{p'}_j||_2 < \theta$. (default $\theta = T/8$).
4. Compute the centred attraction basins $A'_i = \{\mathbf{p''}_{ji} = \mathbf{p}_j - \mathbf{x}_i\}$.
5. The cumulative attraction basin is a mapping $C(x, y)$ from $\mathbb{Z}^2$ to $\mathbb{Z}$. Values $C(x, y)$ are the numbers of occurrences of $[x, y]$ in centred attraction basins (e.g., $C(3, -2) = 5$ means that $\mathbf{p''} = [3, -2]$ occurs in 5 centred attraction basins).

Algorithm 8: Tracks a single Zero Shift Point.

of correct correspondences found by the low-level tracker for a displacement same as the radius. In other words, the curve estimates the relation between repeatability of ZSP of a given period over and displacement between images (or the error of predictor/higher level part of the tracking algorithm integrating the information from more coarse scales and motion/target model).

While comparing the curves estimating the repeatability for different images (Figures 5.4, 5.5, 5.6), the relation between the radius and the minimum number of correct correspondences can be observed: a longer period of ZSPs leads to larger basins of attraction. For shorter periods (up to approximately $T = 25$ in some images up to $T = 39$), the following rule can be deduced: the displacement smaller than $T/2$ guarantees over 50% of correct correspondences and the displacement smaller than $T/4$ guarantees over 80% of correct correspondence. For ZSPs with period $T > 25$, the relation longer period – lager attraction basin also holds, however, the number of ZSPs with long periods is limited to several (< 10) occurrences in an image, therefore, the statistic is not reliable. The attraction basins are usually asymmetric (see Figure 5.2) The consequence is an asymmetric global behavior. The tracker can handle long displacements in the some directions. However, its performance is much worse in some other directions. This problem can be solved by a higher level tracking algorithm (to be discussed in Section 6).

Figure 5.7 shows the locations of ZSPs and the tracking range estimate for all periods in the first image of 'Mouse pad' sequence and the some other image of this sequence differing by a significant scale change (the target is more far from the camera). A predictable phenomenon can be observed: As the target become smaller the blobs become smaller, and therefore, they are detected as ZSPs with shorter period and smaller attraction basins. Many blobs become too small to be detected. Consecutively, the tracking of a small target is more difficult. Of course, this issue is similar as in other approaches to tracking.

Figure 5.8 shows the effect of the blur of the attraction basin. The blur affects mainly small ZSPs. The number of small ZSPs is reduced approximately two times because the blur smears out small details. However, the ZSPs with period 29 or longer were only slightly affected by the blur, some of them were detected with a slightly changed optimal period. The range of displacement (which the tracker can handle) remains approximately the same. The effect of the blur will be studied in more details in Section 5.5.3.

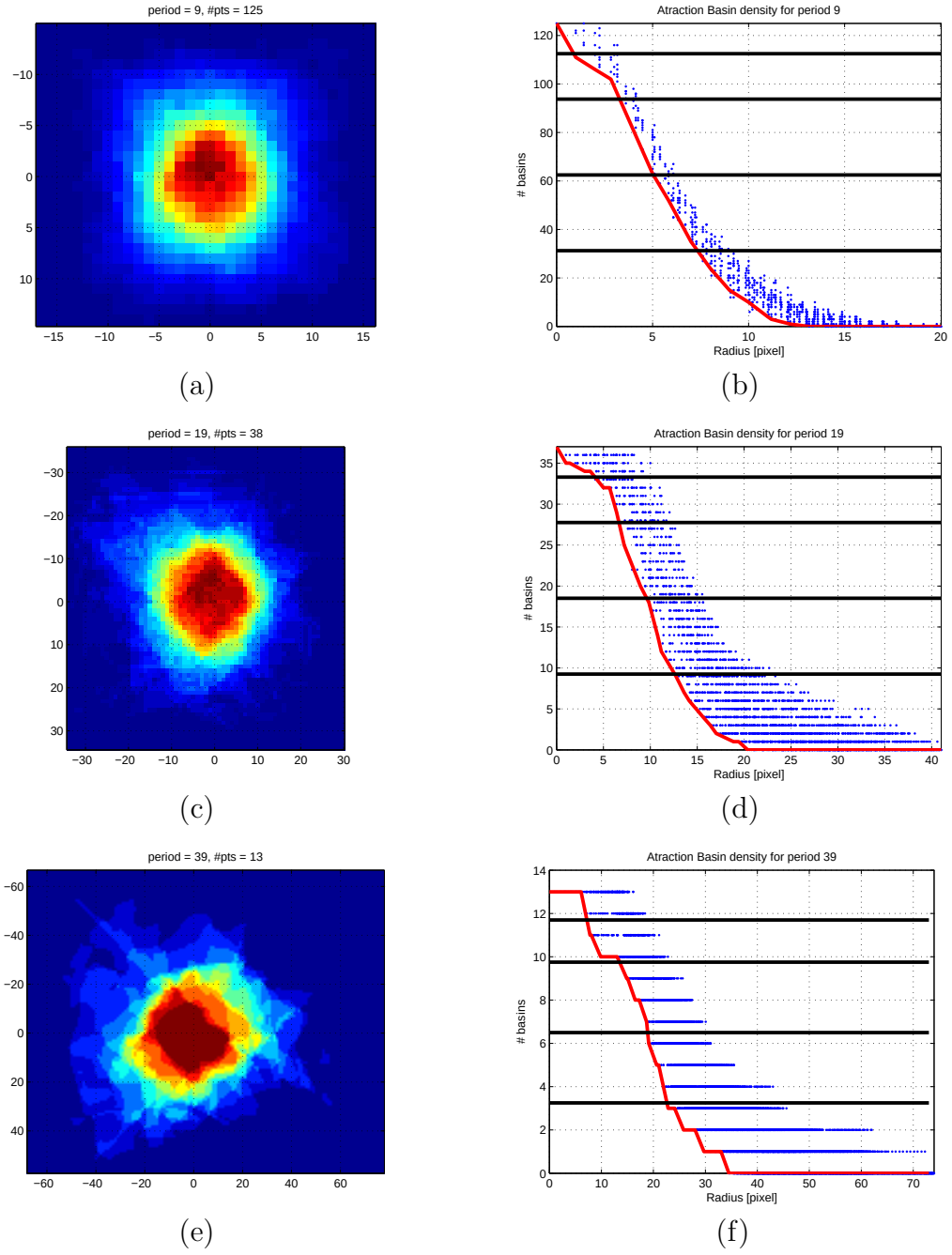
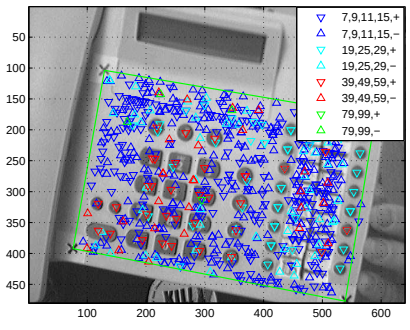
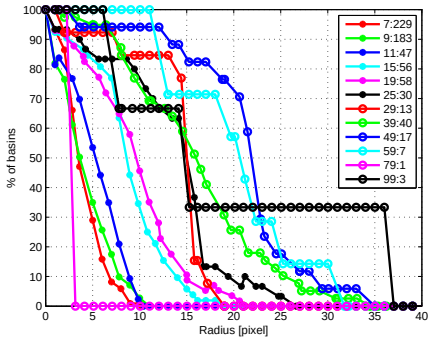


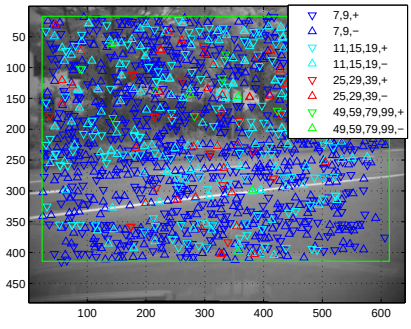
Figure 5.3: Cumulative attraction basins in the first image of the 'Mouse pad' sequence. a), c), e) Images of cumulative attraction basins, b), d), f) Each blue point corresponds to a pixel in the images of cumulative attraction basin and depicts the relation between the radius from ZSP and the number of basins covering the pixel. Black lines show 25%, 50%, 75% and 90% of the total number of basins. The red curve predicts the minimum number of correct correspondences found by the low-level tracker for a displacement of a given radius in an arbitrary direction.



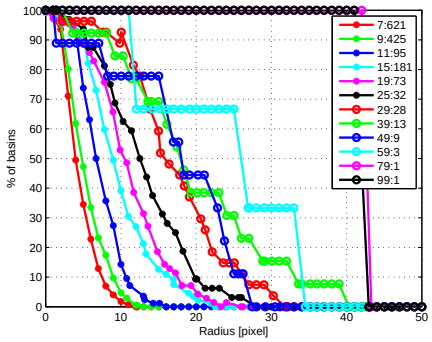
(a)



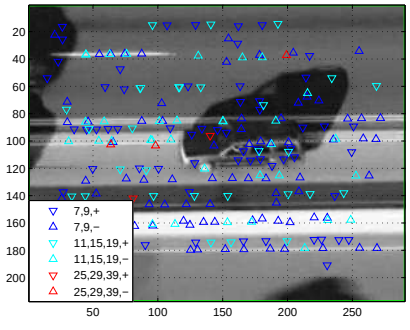
(b)



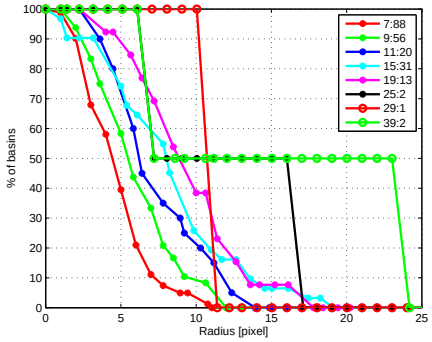
(c)



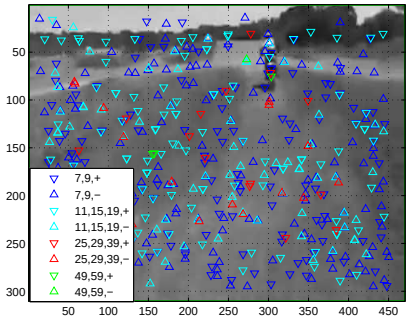
(d)



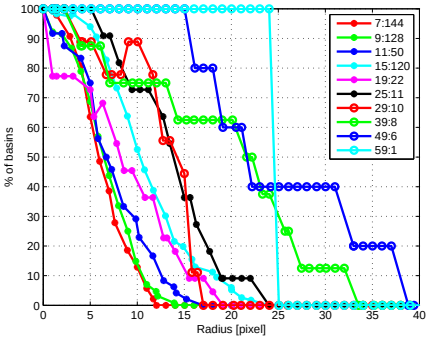
(e)



(f)

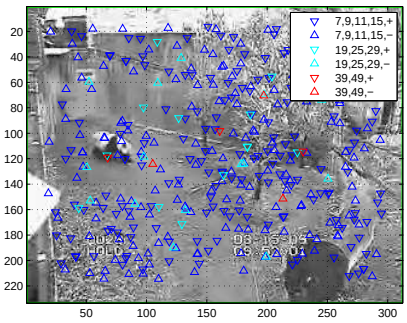


(g)

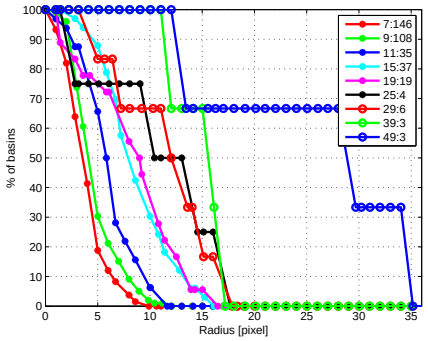


(h)

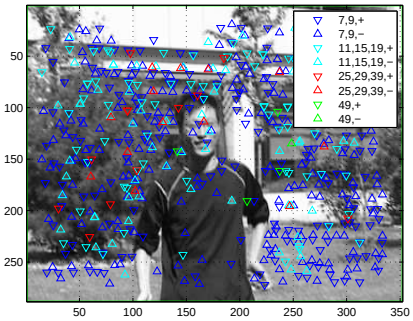
Figure 5.4: The attraction basin – other sequences. a) Locations of ZSP. The legend shows the period and the polarity of ZSP. b) Disparity vs. prediction of the minimum percentage of the successfully tracked points depending on the period. The legend shows the period : the number of ZSPs of a given period.



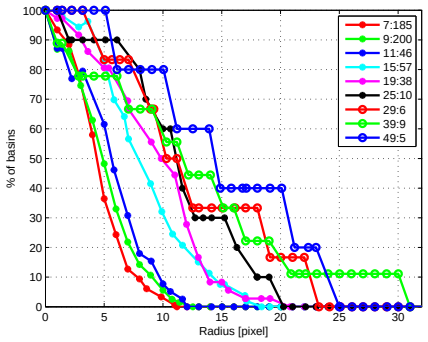
(a)



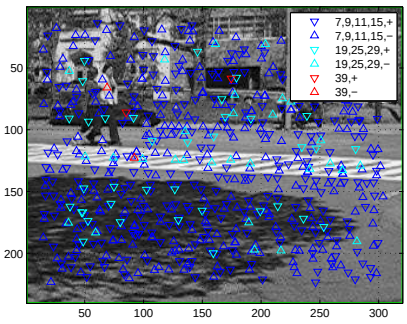
(b)



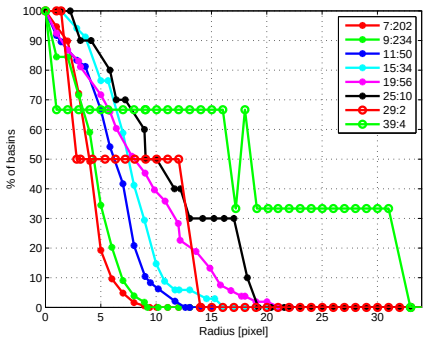
(c)



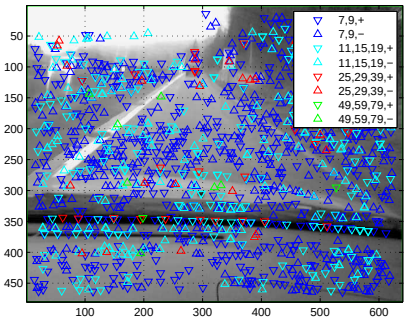
(d)



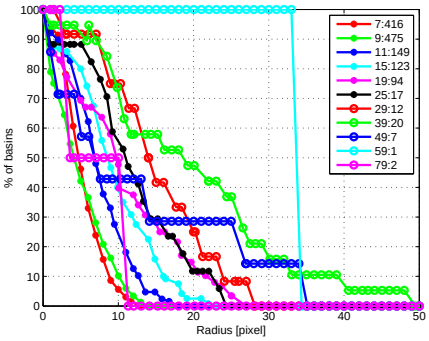
(e)



(f)

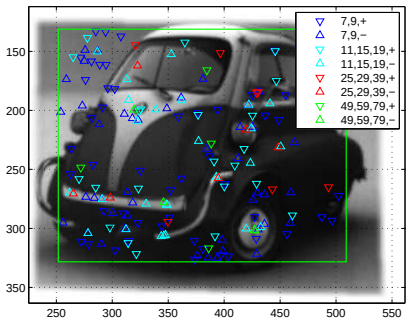


(g)

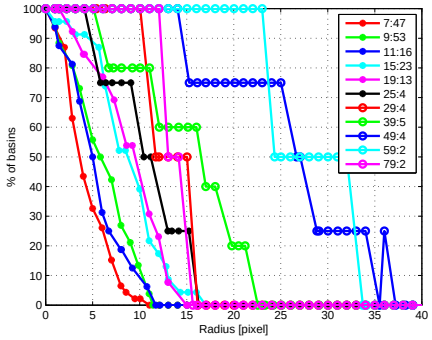


(h)

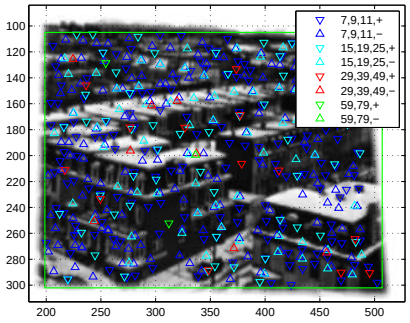
Figure 5.5: The attraction basin – other sequences. a), c), e) Locations of ZSP. The legend shows the period and the polarity of ZSP. b), d), f) Disparity vs. prediction of the minimum percentage of successfully tracked points depending on the period. The legend shows the period : the number of ZSPs of a given period.



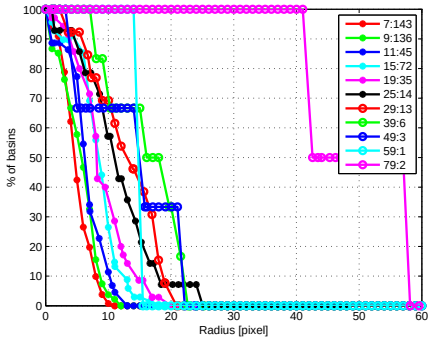
(a)



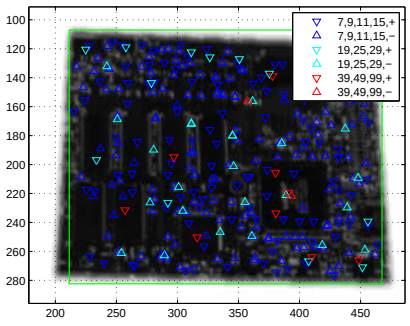
(b)



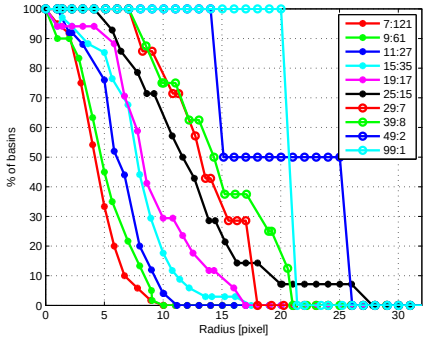
(c)



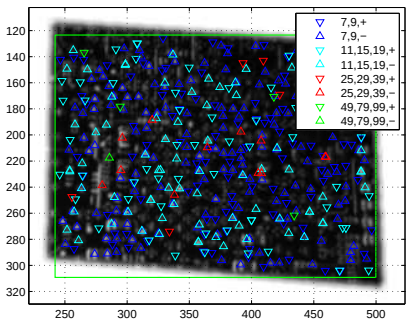
(d)



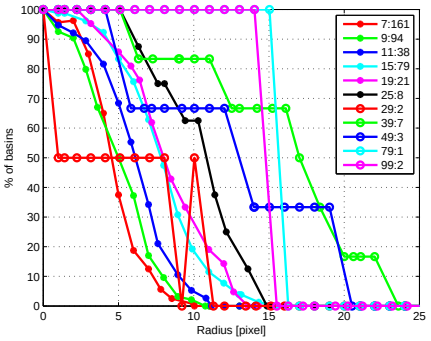
(e)



(f)



(g)



(h)

Figure 5.6: The attraction basin – other sequences. a), c), e) Locations of ZSP. The legend shows the period and the polarity of ZSP. b), d), f) Disparity vs. prediction of the minimum percentage of successfully tracked points depending on the period. The legend shows the period : the number of ZSPs of a given period.

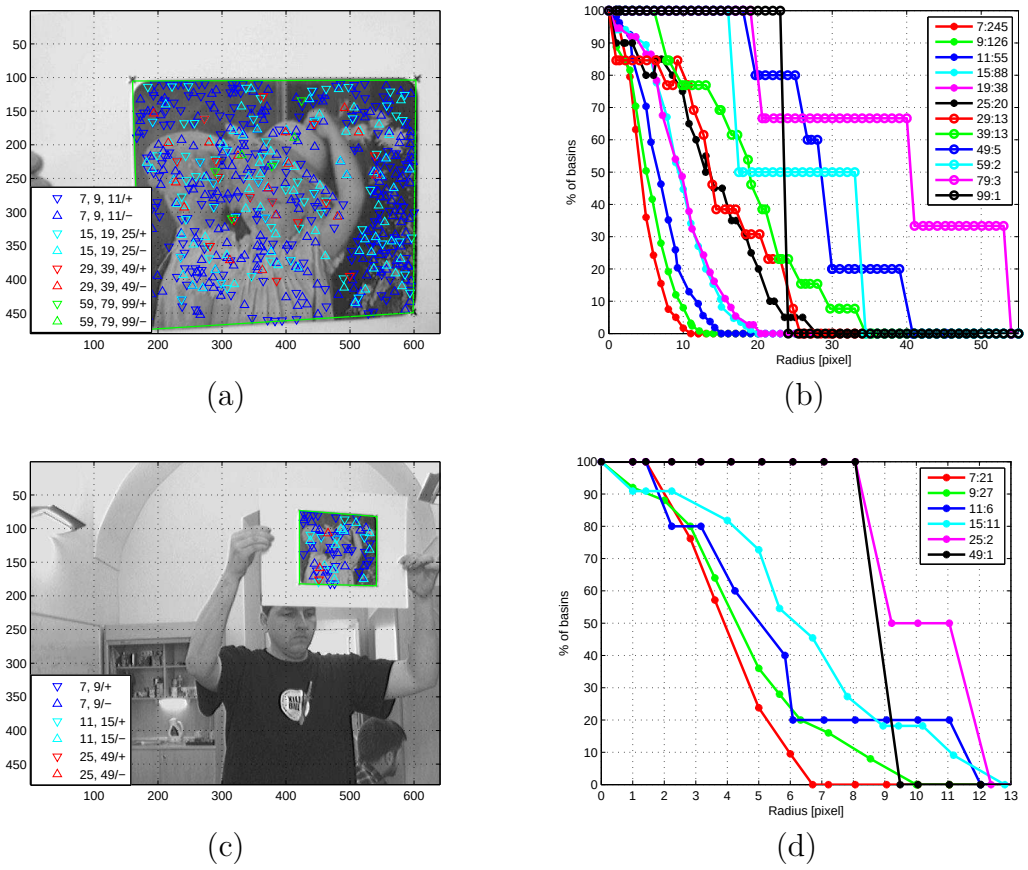


Figure 5.7: The attraction basin - effect of scale changes. a) Locations of ZSP. Image legend shows the period and the polarity of ZSP. b) Disparity vs. prediction of the minimum percentage of the successfully tracked point depending on the period. The legend shows the period : number of ZSPs of given period.

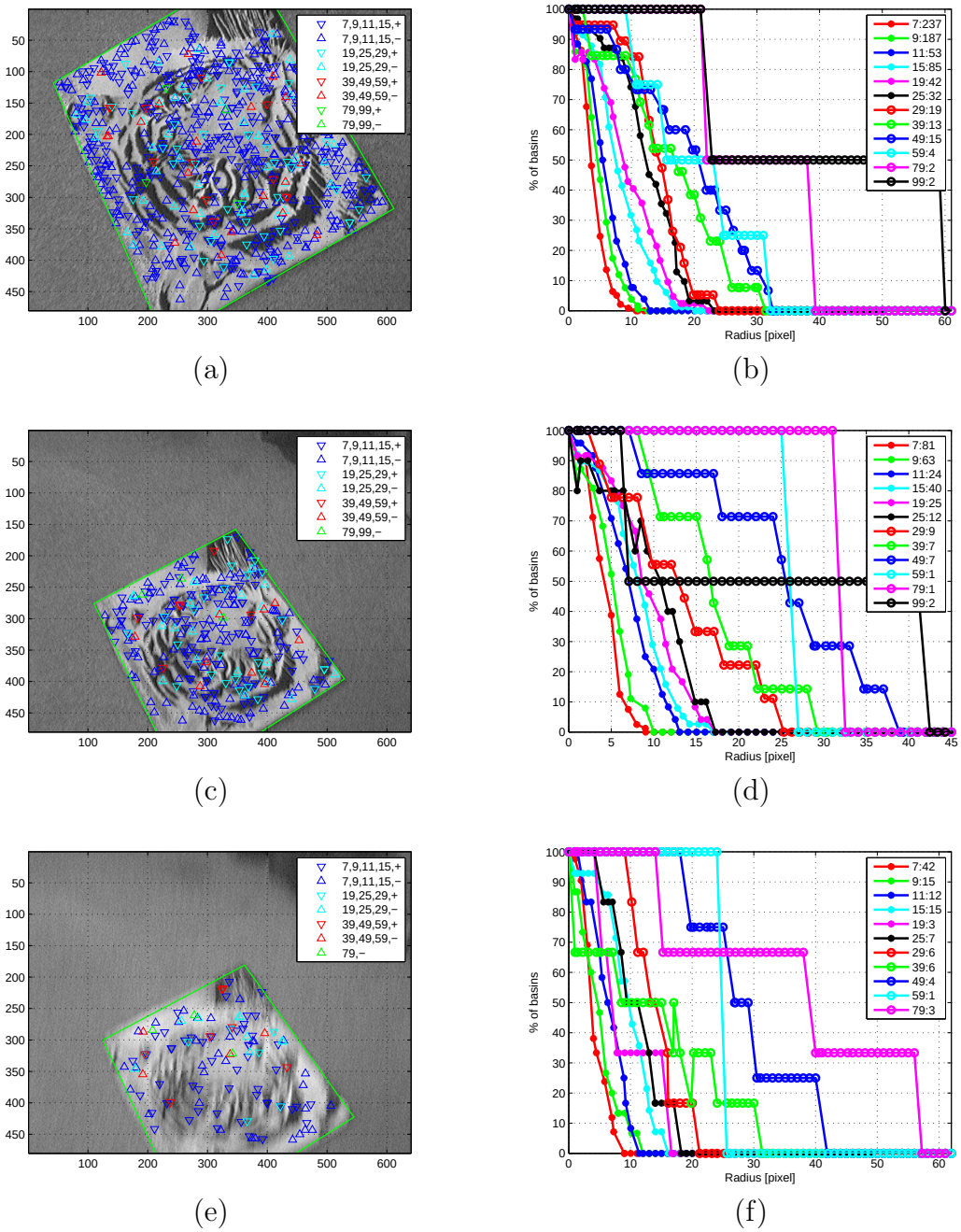


Figure 5.8: The attraction basin – the effect of scale and blur. a), c), e) Locations of ZSP. The legend shows the period and the polarity of ZSP. b), d), f) Disparity vs. prediction of minimum percentage of the successfully tracked points depending on the period. The legend shows the period : the number of ZSPs of a given period.

5.5.2 Synthetic data

The *experiment evaluates sensitivity of ZSPs to geometric distortions.* The first image of 'Mouse pad' sequence was enlarged by an artificial margin filled with zeros and used as a reference image. Without enlargement, some parts of the scene would disappear out off the warped image and some points from the original image would have corresponding locations out off the warped image.

First, the set of ZSPs $\{\mathbf{x}_i\}$ was detected by Algorithm 7 in the reference image. Second, the image was warped by a homography H and points $\mathbf{x}_i$ were projected to $\mathbf{x}'_i = H\mathbf{x}_i$. Finally, the tracker ran from $\mathbf{y}_i = \text{round}(\mathbf{x}'_i)$ while outputting $\mathbf{x}''_i$.

Two scores were evaluated: (1) *repeatability* – the percentage of correctly tracked points, (2) *the localization error* defined as the distance e between projected correct position $\mathbf{x}'_i$ and location $\mathbf{x}''_i$ found by the tracker. The points with the error $e > T/4$ were considered as failures of the tracker as well as the points where the tracking failed (marked by the algorithm itself).

The effect of the period of ZSP and the rank (given by Algorithm 6) was studied. As can be seen in Figures 5.9, ..., 5.14, the points with a higher rank have significantly better repeatability and better precision, especially for higher distortions. The period does not influence the repeatability much, however, the localization error scales with the period.

Figure 5.9 evaluates the effect of scale changes on the repeatability and precision in the range from 0.53 to 1.89. The scale change affects both the repeatability and precision. A bigger scale change leads to a bigger localization error and a lower repeatability. If the scale change is known in advance then it can be compensated by the same change of period as shown in Figures 5.15 and 5.16. The localization error becomes almost independent on the scale change and is similar to the localization error caused by the rotation, Figure 5.11.

Figures 5.10 and 5.11 evaluate the effect of a pure rotation and rotation with a small scale change on the repeatability and precision. The dependence of the tracking performance on the rotation angle is small, the worst is around 45^o. The selection of the points with the higher rank significantly improves on both the repeatability and the precision.

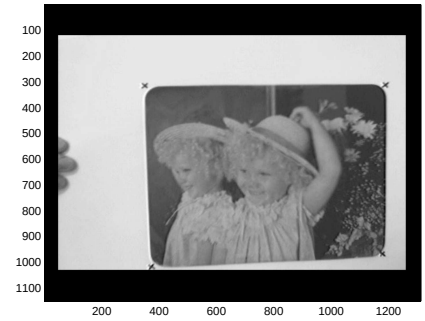
Figures 5.10 and 5.11 evaluate the effect of a tilt (anisotropic scale) on the repeatability and precision. The results are similar to the effect of the scale change and can be partially corrected by the period adjustment using a global scale change estimation (see Figures 5.15 and 5.16). The scale correction was computed from the tilt as

$$\sqrt{2}\sqrt{1 + \text{tilt}^2}\,.$$

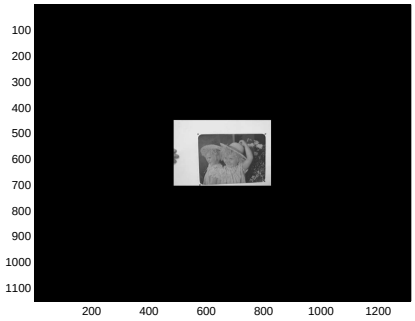
The tilt value will be probably unknown in a real situation. However, the scale change estimated while ignoring presence of the tilt may lead to similar results.

Finally, Figure 5.14 evaluates the effect of a perspective distortion, namely $H(3,1)$ on the repeatability and precision. The effect of $H(3,2)$ is very similar (not shown). The transformation contained also translation which compensated for the shift of the scene out of the image caused by changing $H(3,1)$. The effect of the perspective distortion is the worst from the studied transformations. No simple adjustment

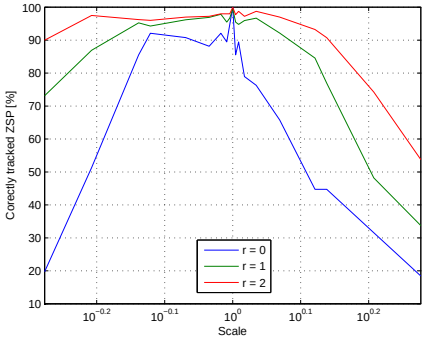
helps, probably. The perspective distortion acts locally both as the scale change and as the rotation which are different in different parts of the image. A solution to this problem will be suggested in Chapter 6.



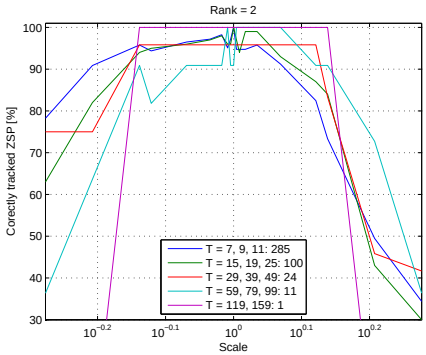
(a)



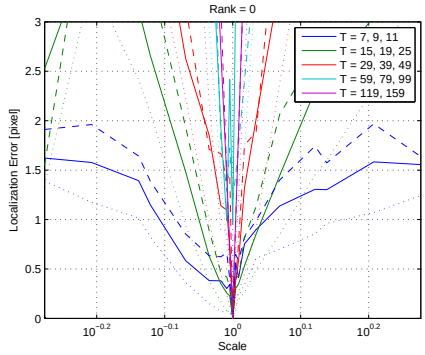
(b)



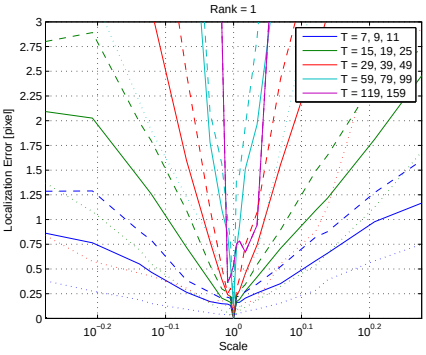
(c)



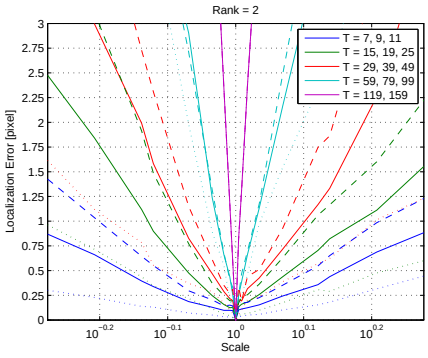
(d)



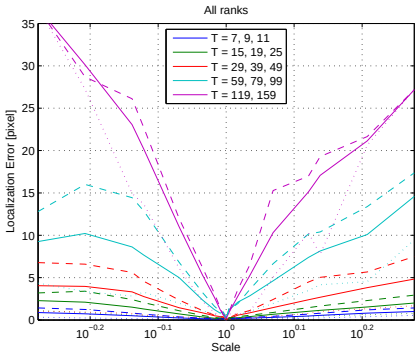
(e)



(f)



(g)



(h)

Figure 5.9: Synthetic images - scale change. Distorted images (border scales) a) scale = 0.53, b) scale = 1.89 Repeatability depending on the c) rank, d) period. e), f), g), h) Localization error depends on the rank and the period. Solid lines show average error, dotted line 20% percentile and dashed lines 80% percentile.

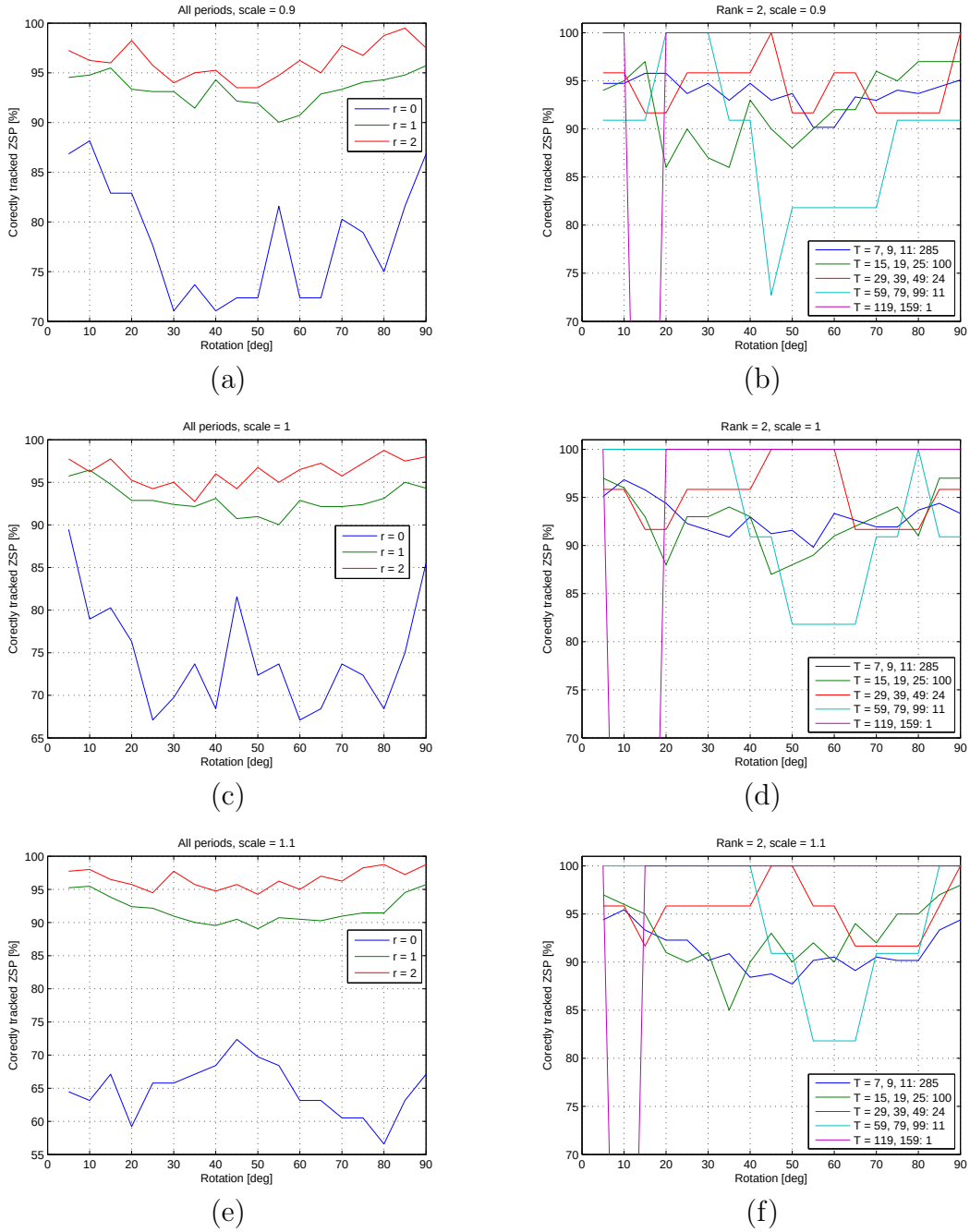


Figure 5.10: Synthetic images. Repeatability to the rotation with/without the small scale change a), b) scale = 0.9, c), d) scale = 1.0, e), f) scale = 1.1, depending on the a), c), e) rank, b), d), f) period.

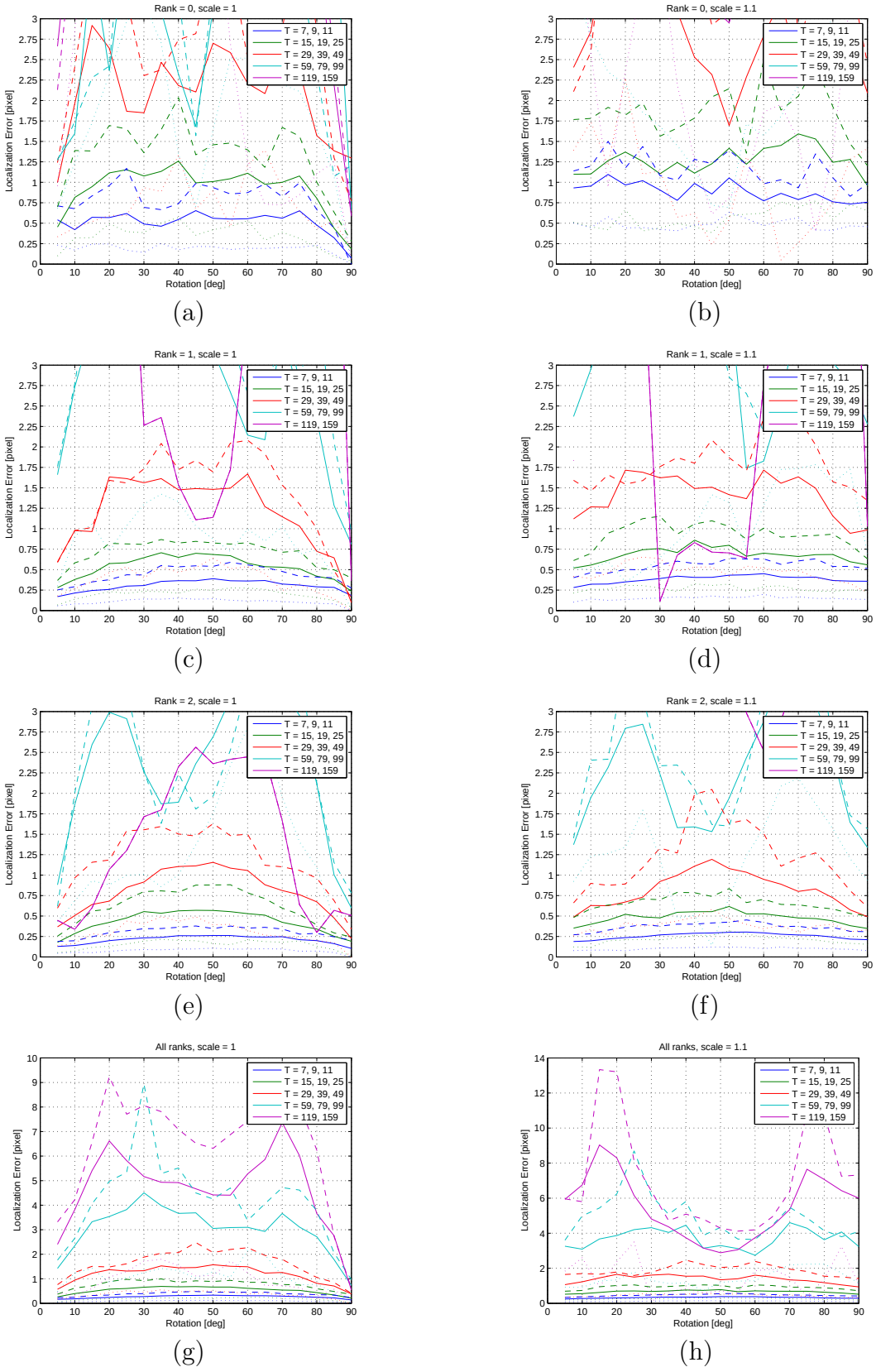


Figure 5.11: Synthetic images - rotation with/without small scale change. a), c), e), g) Scale = 1.0, b), d), f), h) Scale = 1.1. Solid lines show average error, dotted line 20% percentile and dashed lines 80% percentile.

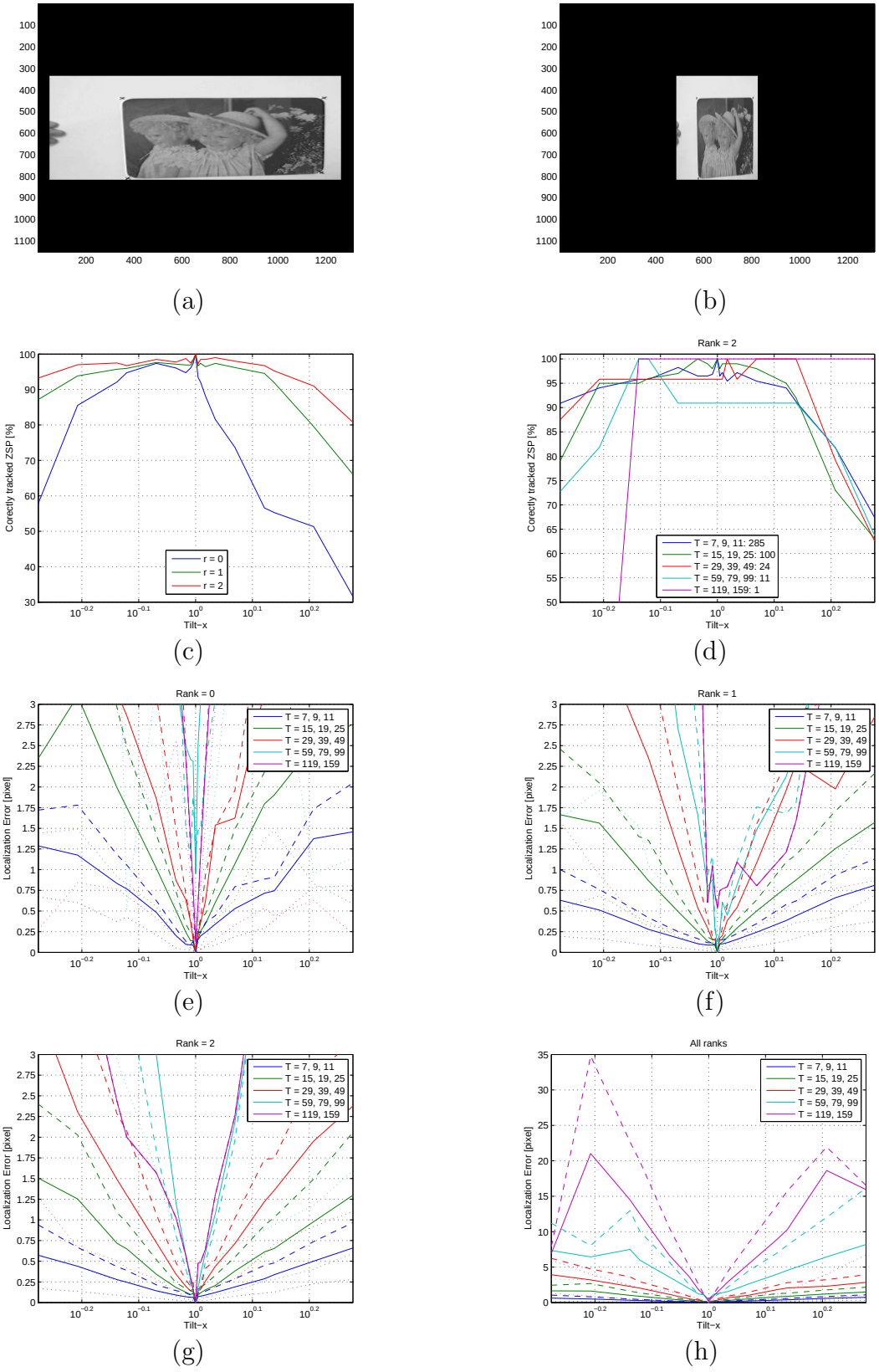
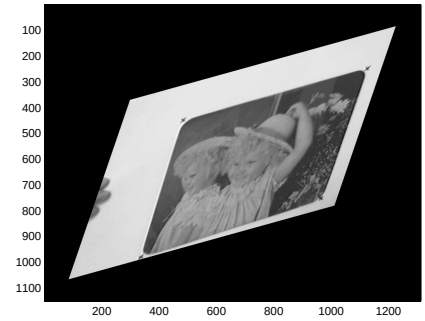
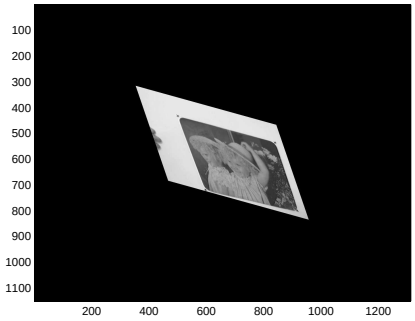


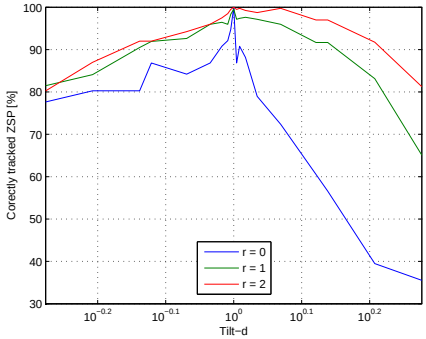
Figure 5.12: Synthetic images - tilt-horizontal . Distorted images a), b). Repeatability depending on the c) rank, d) period. e), f), g), h) Localization error depends on the rank and the period. Solid lines show average error, dotted line 20% percentile and dashed lines 80% percentile.



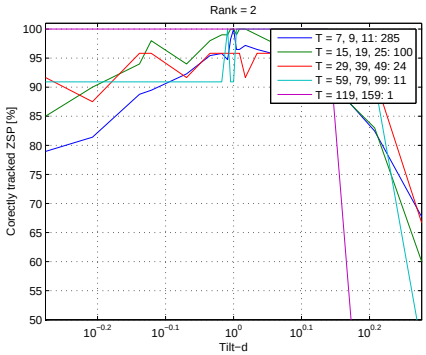
(a)



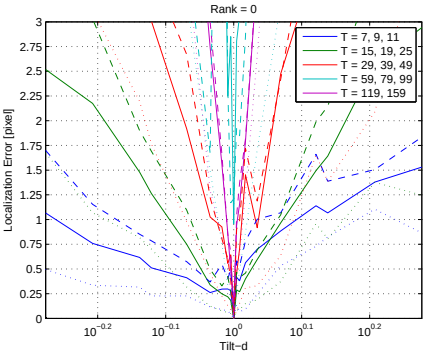
(b)



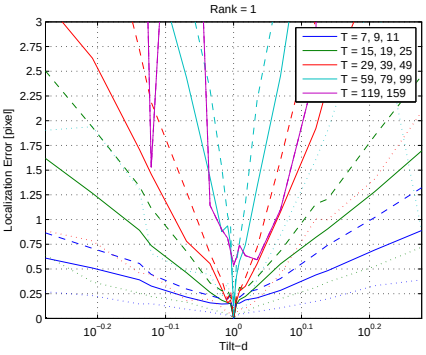
(c)



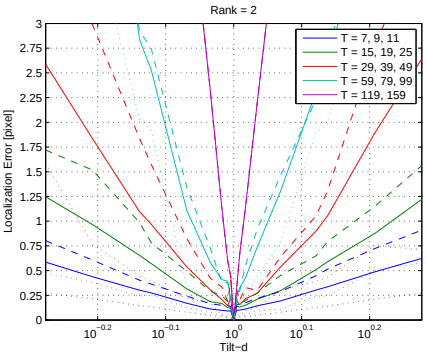
(d)



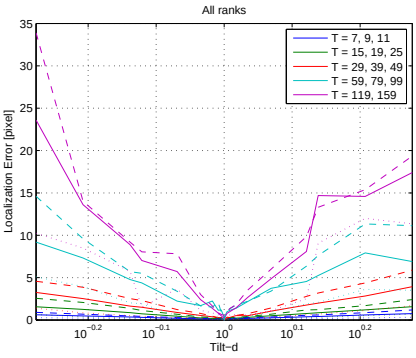
(e)



(f)



(g)



(h)

Figure 5.13: Synthetic images - Tilt-diagonal. Distorted images a), b). Repeatability depending on the c) rank, d) period. e), f), g), h) Localization error depends on the rank and the period. Solid lines show average error, dotted line 20% percentile and dashed lines 80% percentile.

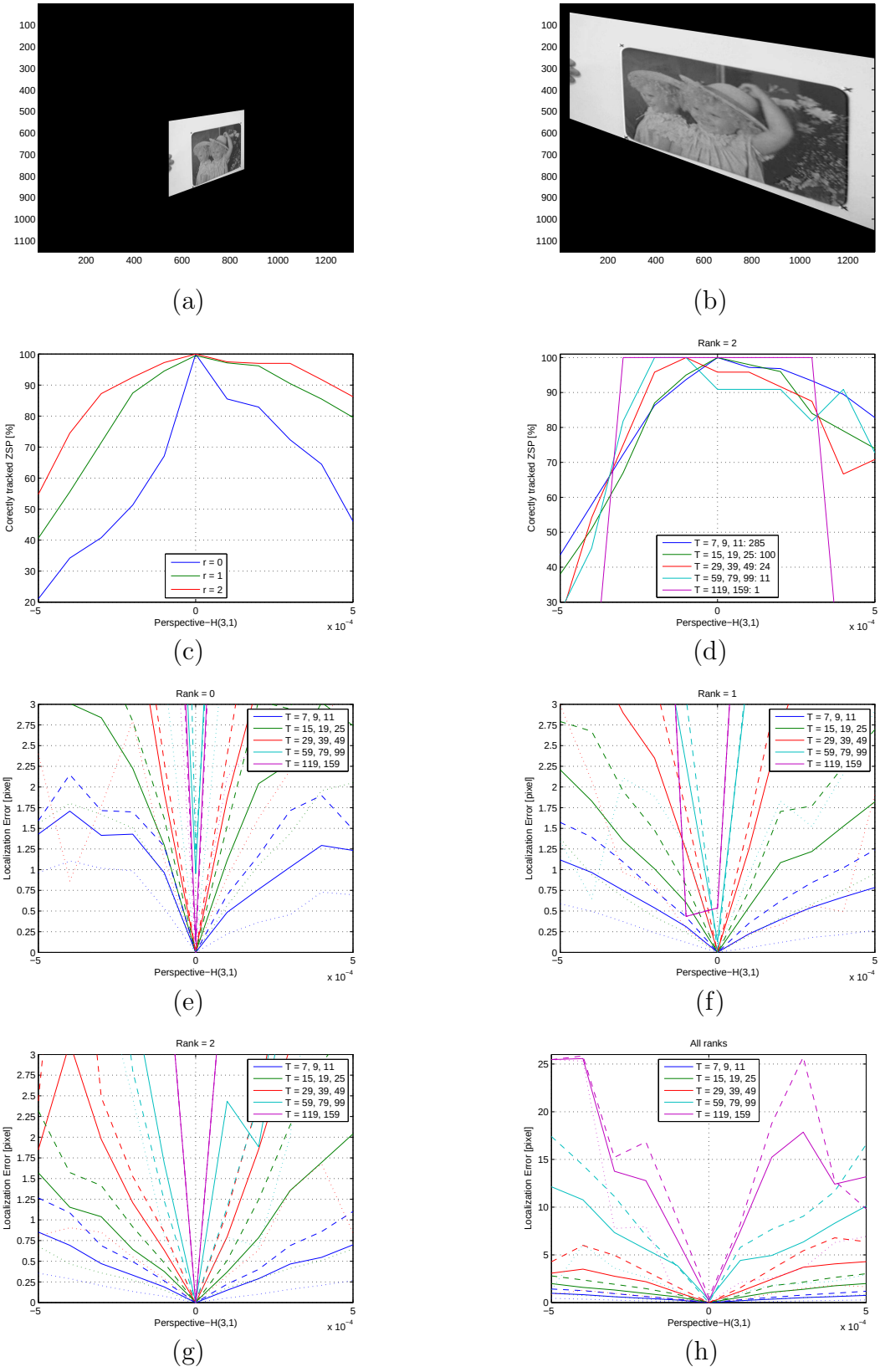


Figure 5.14: Synthetic images - perspective distortion. Distorted images a) $H(3,1) = -5 \cdot 10^{-4}$, b)$H(3,1) = 5 \cdot 10^{-4}$ Repeatability depending on the c) rank, d) period. e), f), g), h) Localization error depends on the rank and the period. Solid lines show average error, dotted line 20% percentile and dashed lines 80% percentile.

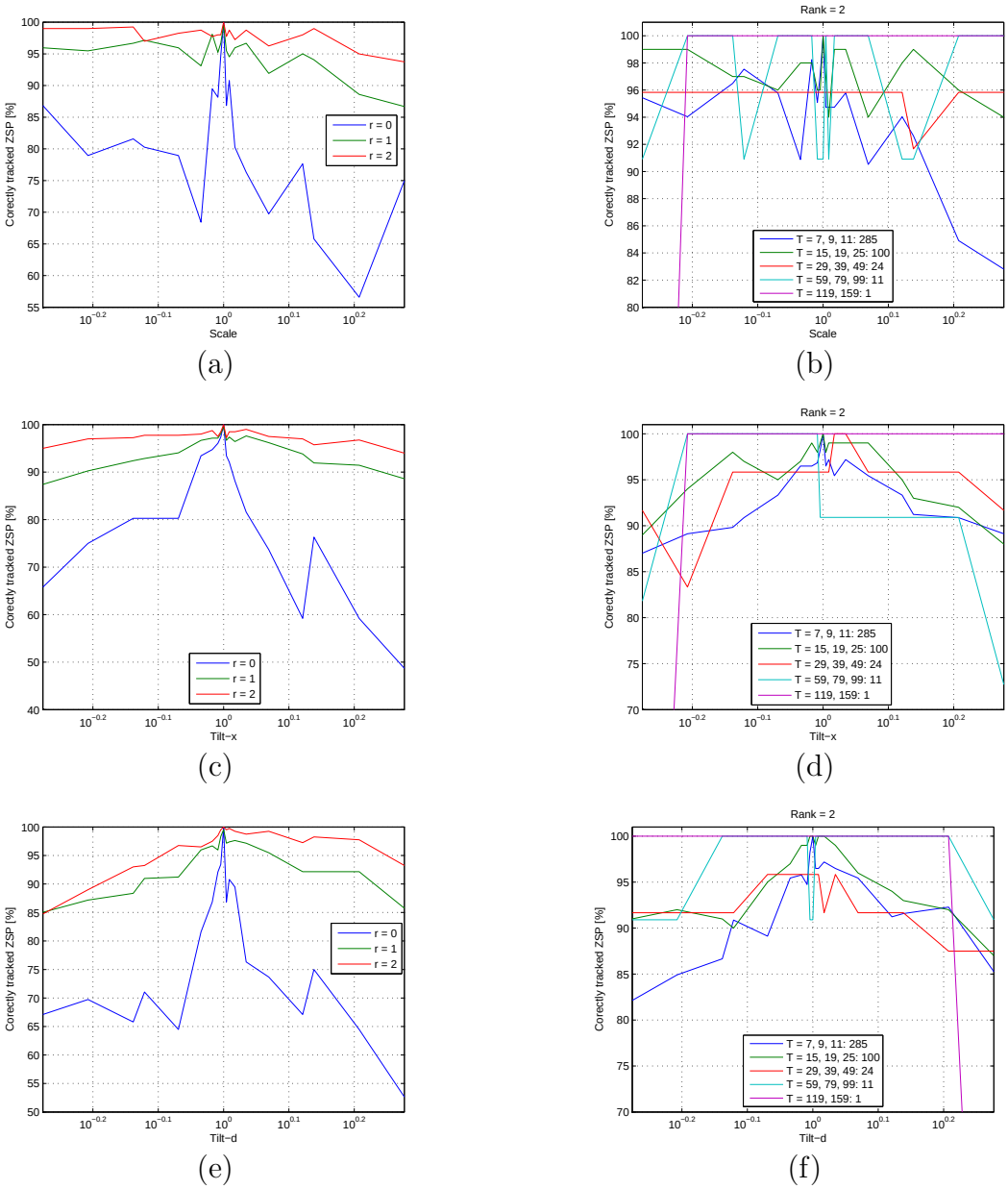


Figure 5.15: Synthetic images - the effect of the scale correction on repeatability.

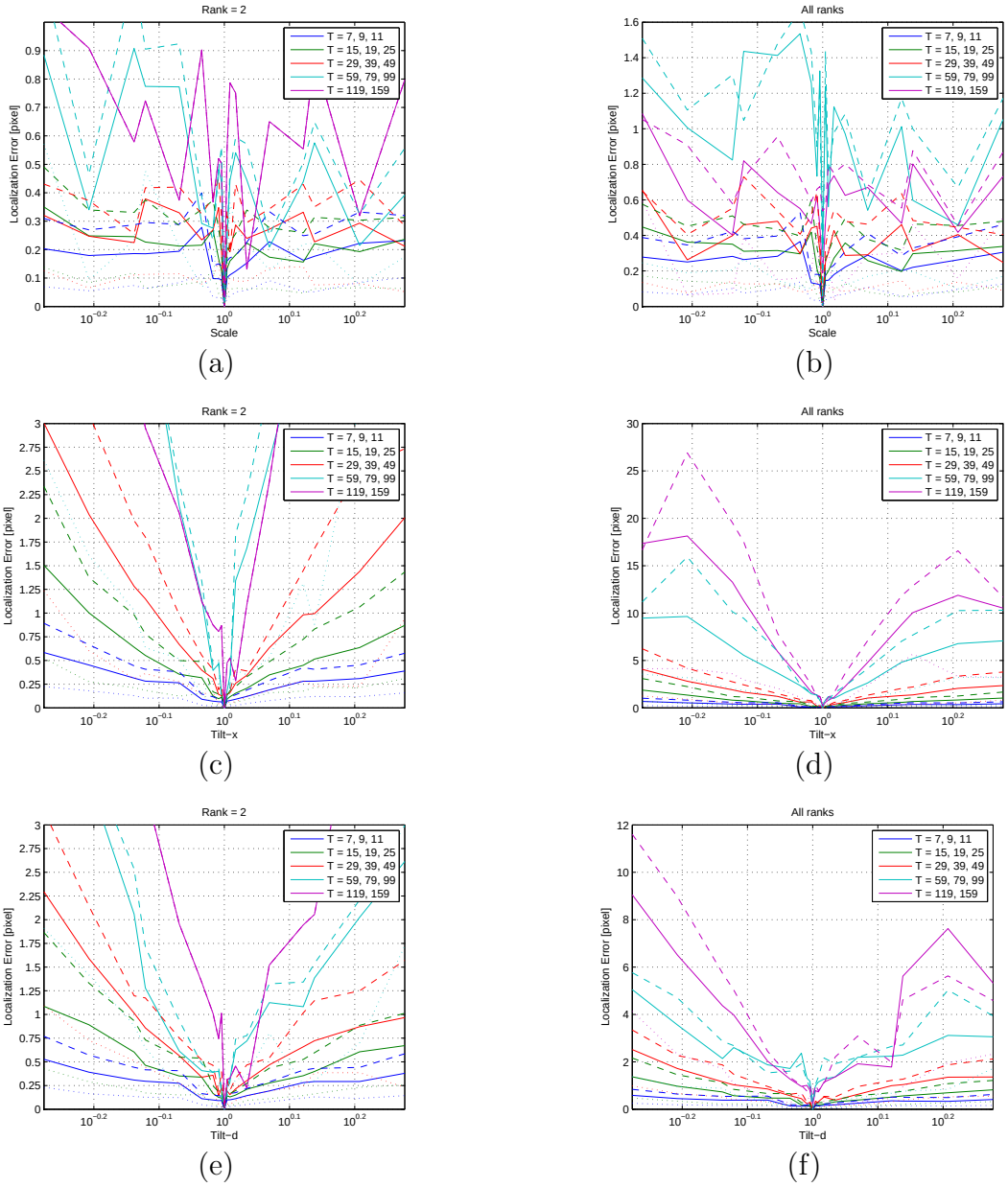


Figure 5.16: Synthetic images - the effect of the scale correction on the precision.

5.5.3 Guided tracking

The aim of the experiment is to evaluate the repeatability of Zero Shift Points (ZSPs) on real sequences. More precisely, the experiment tests the ability of ZSPs to survive a geometric distortion together with other disturbances present during the video sequence capture (e.g. blur, noise, illumination changes and JPEG compression). Except of geometry, the other distortions were not quantified either controlled during capture of the sequences. The repeatability evaluated by the experiments in this section measures the presence of a ZSP near the expected location not the range of displacement the tracker can handle.

The geometry was restored from ground truth data provided together with image sequence. The procedure (similar to Section 5.5.2) is described by Algorithm 9.

Detect the set of ZSPs $\{\mathbf{x}_i\}$ by Algorithm 7 in the reference (first) image of the sequence/subsequence.
for *second and all other images of the sequence* **do**
  Compute a transformation (homography) $\mathbf{H}$ between the reference image and the current image using ground truth data (positions of the target corners).
  Project ZSPs $\mathbf{x}_i$ from the reference image to $\mathbf{x}'_i = H\mathbf{x}_i$.
  Track $\mathbf{x}'_i$ to $\mathbf{x}''_i$ using Algorithm 5.
  Evaluate the localization error $e_i = ||\mathbf{x}''_i - \mathbf{x}'_i||_2$.
  Remove outliers, i.e points with $e_i > T_i/2$.
end

Algorithm 9: The guided tracking experiment

There are several publicly available sequences with some kind of a ground truth. Unfortunately, most of them are useless for evaluation of the low level tracker due to

- a very low precision (just rough rectangle around the target e.g. [17]) or
- the ground truth provided is too sparse, e.g. only for each 250th frame [21].

We found only one set composed of three sequences which provides ground truth for each frame with a reasonable precision [45]. This data was used in this section.

Unfortunately, the precision of the ground truth is insufficient (typically about 2 pixels, often 3 or 4 pixels) compared to the expected precision of the tracker (better than 1 pixel, see Section 5.5.2). The precision of the ground truth is just sufficient to the goal of the experiment – to evaluate the repeatability of ZSPs.

The experiment aims at discovering if and how many ZSPs exist in the correct position after the transformation. The ground truth is used twice: (1) it provides the correct position from which the localization error is measured, and (2) as a perfect predictor, which provides the initial point for tracking to the correct position in which ZSP should be found. The tracker should not ideally track but merely verify that the point is a ZSP or track it very near to the slightly shifted 'incorrect' position. Therefore, ZSPs with short periods (9,7,5) are affected twice by the imprecision of

the ground truth. First, the initial position may be insufficiently precise considering the small attraction basins (about 2-3 pixels). Second, the criterion distinguishing outliers depends on period. Therefore, some of these correctly tracked points may be regarded as lost just due to the imprecise ground truth. For this reason, the outlier threshold was enlarged from $T/4$ to $T/2$. This still guarantees that the mismatched points are rejected (the distance between ZSPs of the same period and the polarity is T or usually bigger).

Figure 5.17 shows the results on the first 150 frames of 'Mouse pad'. The first 75 images have only a small geometric distortion, just a small rotation up to 4^o which leads to an almost constant high repeatability. Similarly to results in Section 5.5.2, the higher rank leads to the higher repeatability. The lower repeatability of ZSPs with a shorter period may be due to the imprecise ground truth. The scale changed only slightly therefore the effect of the period adjustment was negligible in the first 75 frames.

The situation changed dramatically in the second half of the subsequence in which the target continuously moves away from the camera. As the target becomes more distant, the blobs become smaller and the appropriate period to detect ZSPs on blobs should be smaller. The repeatability decrease with the changing scale without the period adjustment.

The period adjustment was performed using the global scale estimated from the ground truth (the coordinates of the target corners) using Algorithm 13. The period adjustment has two effects. First, it allows tracking at the correct scale which improves repeatability – it keeps the repeatability high and almost independent of the target scale for reasonably sized ZSPs (i.e. big enough). Second, the period adjustment allows to identify in advance ZSPs with a very low chance to be tracked because they become very small.

The repeatability percentage is counted as $100n_i/n_1$ in the experiment without the period adjustment and as $100n_i/n'_1$ where n_i is the number of correctly tracked ZSPs in frame i. The value n_1 is the total number of ZSPs detected in the first image and n'_1 is the number of ZSPs with the adjusted period $T \geq 5$ (i.e. the minimal tracking period). The period adjustment works more precisely for longer periods because (naturally) the step of 2 between two neighbouring odd numbers presents a much bigger percentage of e.g. 7 than of e.g. 39. It may also be the reason for the lower repeatability of ZSPs with a shorter period together with the imprecision of the ground truth (as discussed above).

Figure 5.18 allows to see the effect of the scale changes and the period adjustment on the precision of the location of the tracked ZSPs. Two images with a different scale relative to the first image were chosen deliberately. The image 28 shows the target nearest to the camera with the scale 1.08. The image 150 shows the target farthest from the camera with the scale 0.52. If the period was not adjusted then many mismatched and imprecisely localized ZSPs can be observed. If the period adjustment is used then the localization errors are much smaller (and almost no mismatches occur as well) in the expense of some points do not have their counterparts (the crosses without a line connecting them to the tracked ZPSs in the figure), because their tracking even did not start knowing their insufficient size in advance.

Figures 5.21, 5.20 and 5.21 show the estimated parameters of geometrical changes and the number of correctly tracked ZSPs depending on the period in the three sequences. All available sequences with usable ground truth were explored. The sequences are slightly truncated in order to start with the reference image in which (1) the target is in a medium distance with respect to the whole sequence (i.e. there are some images with both bigger and smaller scale), and (2) the reference image is not affected significantly by the blur. The ZSPs are grouped according to their period in the reference image. The period adjustment was applied using the global scale estimated from the ground truth. The results show that ZSPs with a longer period survive within almost a whole sequence. The exceptions are 'unfair' parts of the sequences in which the part of the target is out off the image. It leads to moving some ZSPs out off the image or too close to the image border with respect to their size.

Figure 5.22 shows the effect of the blur. Small ZSPs survive the blur badly. On the contrary, big ZSPs ($T \geq 29$) are unaffected by the blur. It is understandable because the blur smears out the details. Probably, the precision of the ground truth is also affected by the blur because it was manually reconstructed from the images [45]. The observation is similar to the results of Section 5.5.1 where ZSPs were detected independently in two similar images (with a negligible geometric difference) where the first image was not blurred and the second image was significantly blurred. The blurred image contained much less ZSPs with short periods than the image without blur. The number of ZSPs with long periods was almost the same in both images.

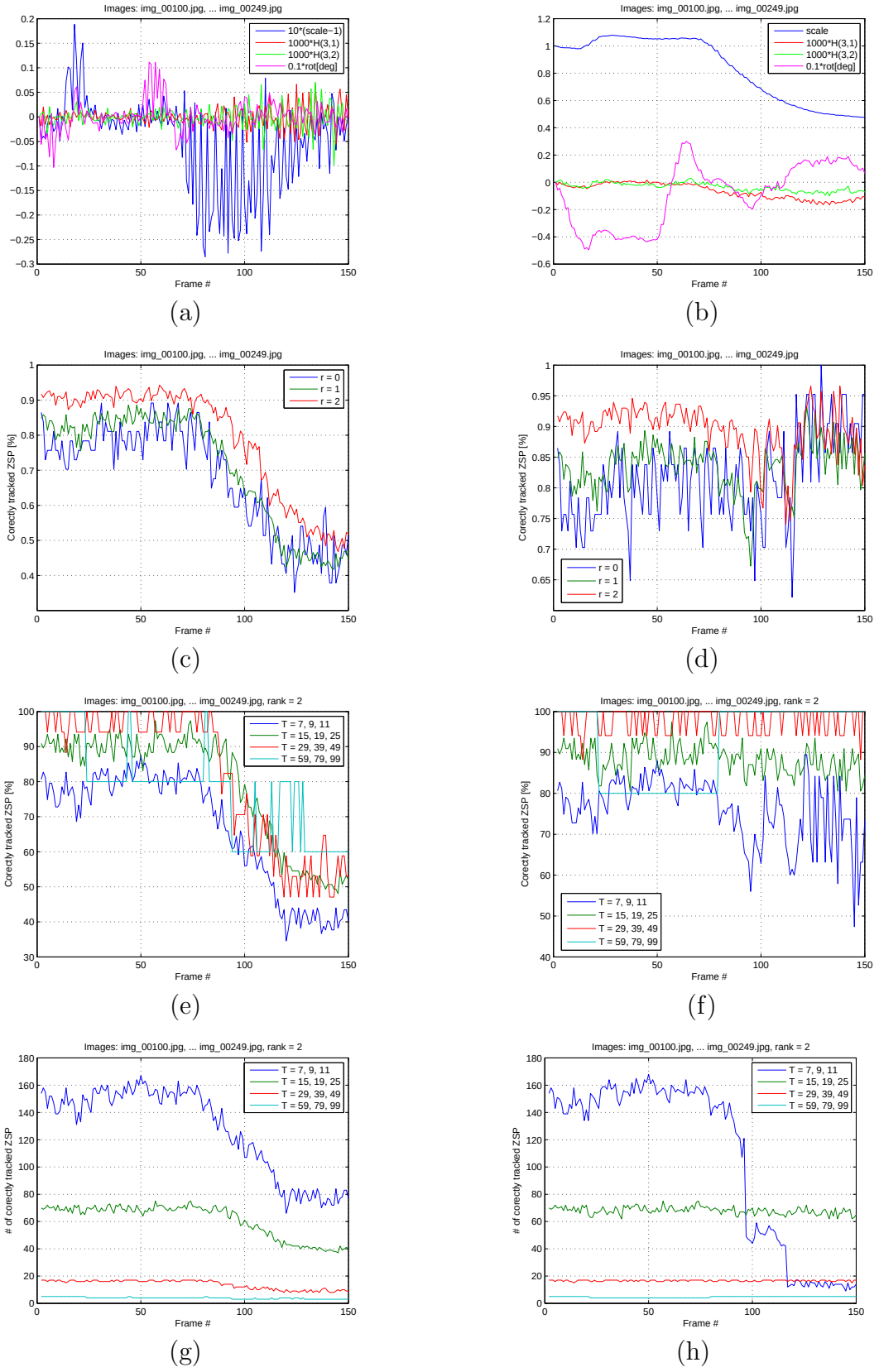


Figure 5.17: Guided tracking – the beginning of 'Mouse pad' sequence. a) Change of geometry between neighbouring frames. b) Change of geometry between the first and current frame. c), e), g) No period adjustment. d), f), h) With the period adjustment.

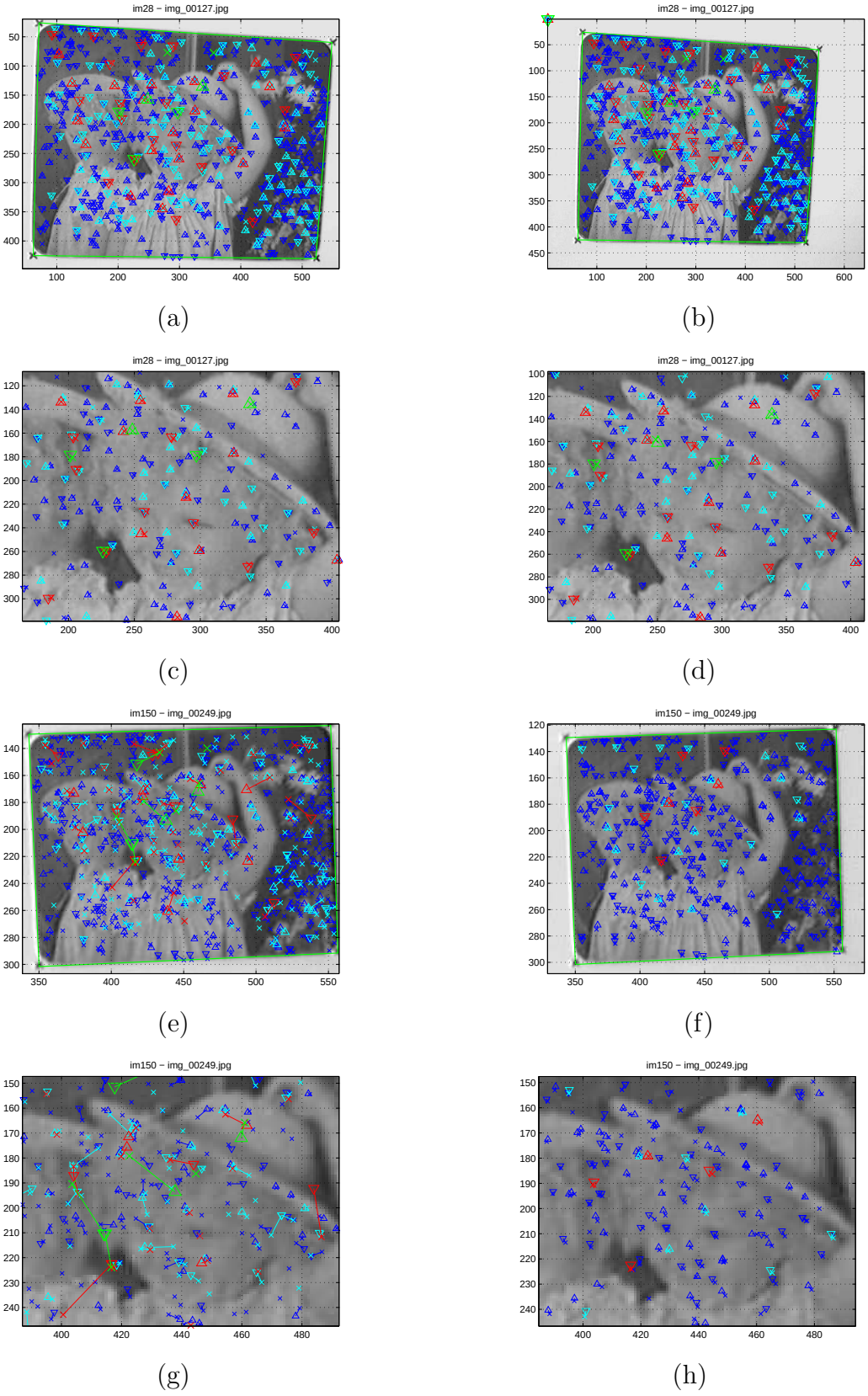


Figure 5.18: Guided tracking – effects of the period adjustment on the precision. 'Mouse pad' sequence. a)...d) Scale = 1.08. e)...h) Scale = 0.48. Crosses mark projected locations of ZSPs $\mathbf{x}'_i$, triangles mark locations found by tracker $\mathbf{x}''_i$. The color groups ZSPs according to their period. The lines of the same color connect corresponding projected and tracked locations.

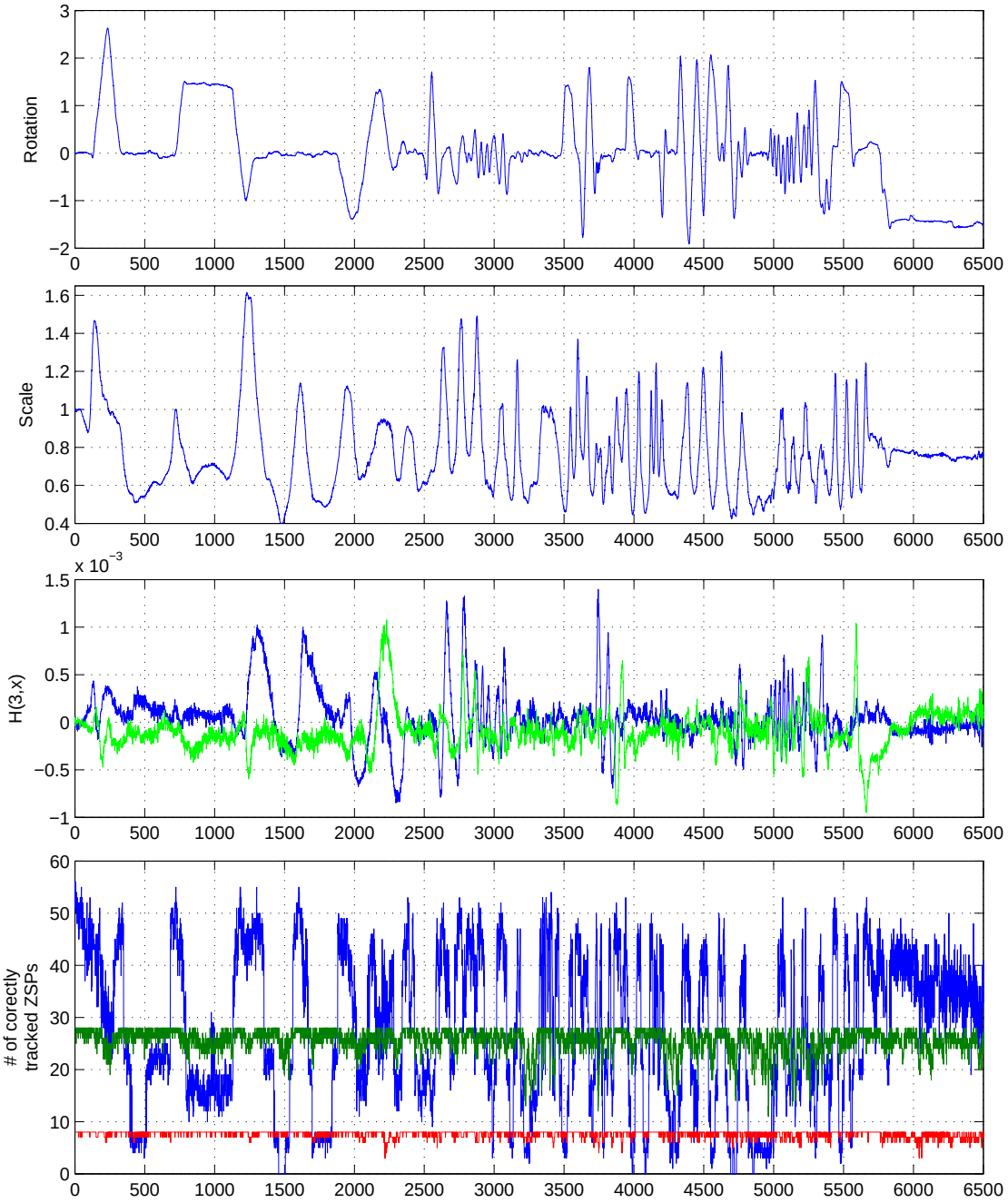


Figure 5.19: Guided tracking, 'Mouse pad' sequence. The top three charts show the estimated parameters of geometrical change of current image relatively to the first image. The most bottom chart shows the number of correctly tracked ZSPs depending on period in the first frame (7, 9, 11 – blue, 15, 19, 25 – green, 29, 39, 49 – red).

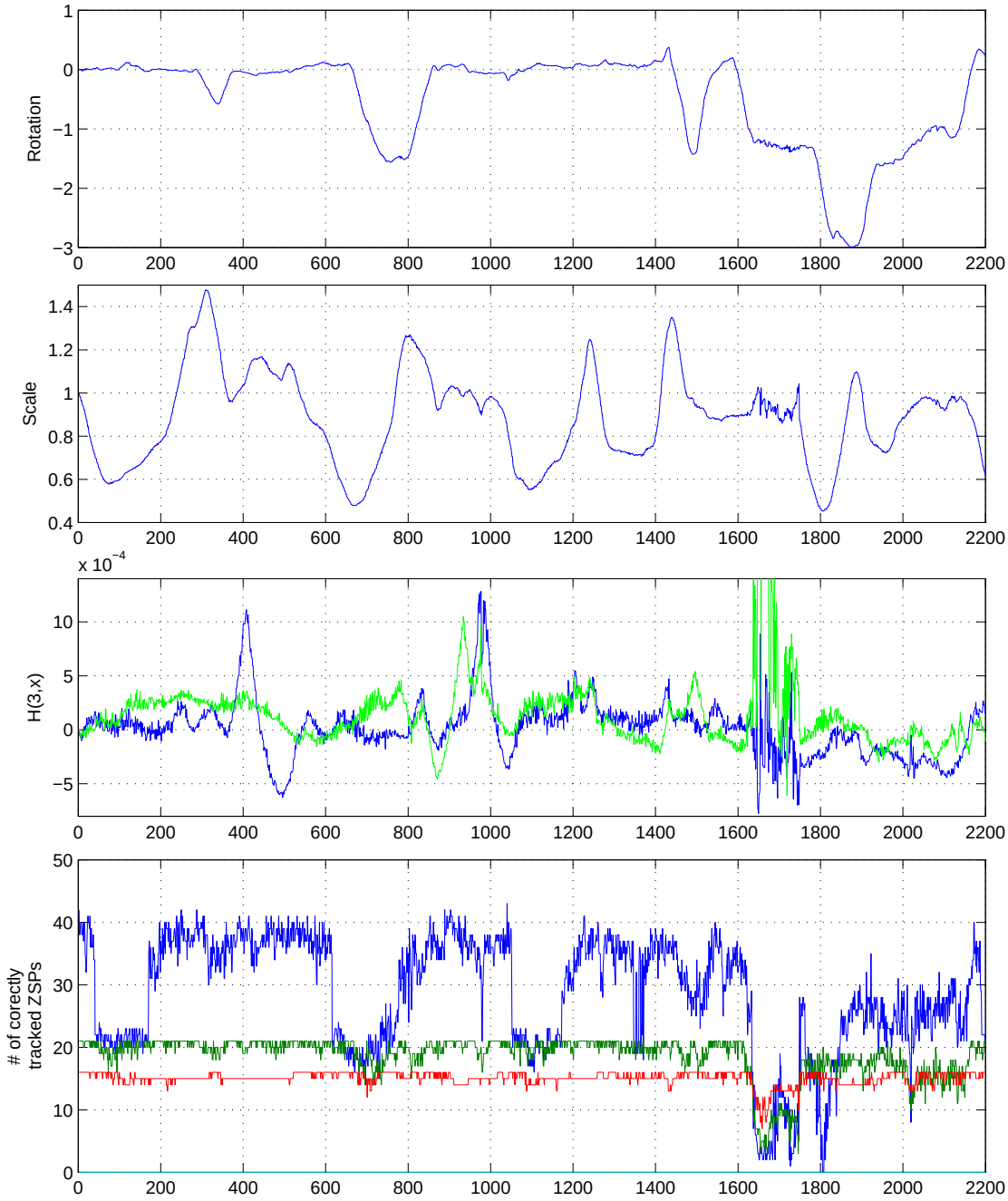


Figure 5.20: Guided tracking, 'Phone' sequence. The top three charts show the estimated parameters of geometrical change of current image relatively to the first image. The most bottom chart shows the number of correctly tracked ZSPs depending on period in the first frame (7, 9, 11 – blue, 15, 19, 25 – green, 29, 39, 49 – red).

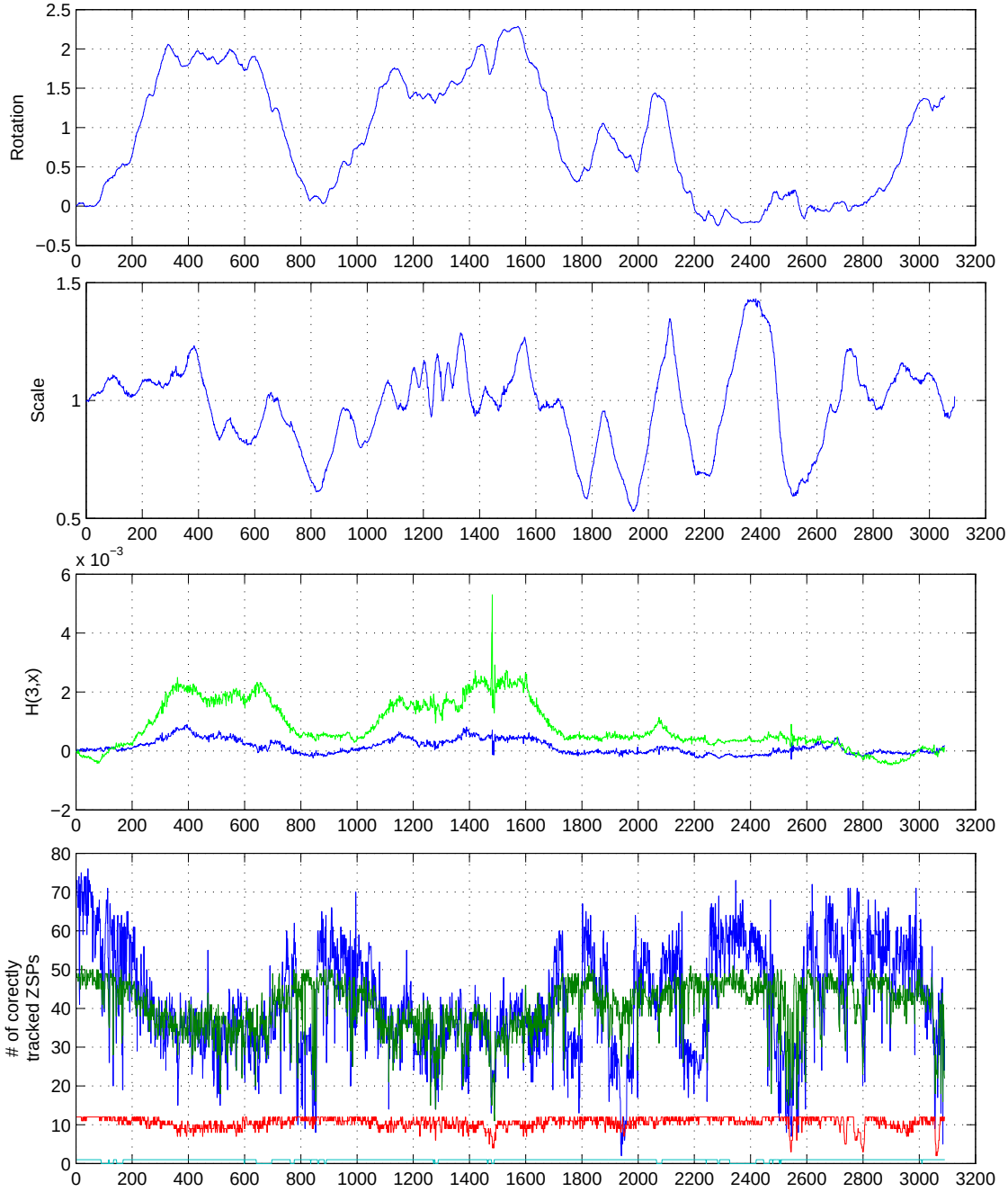


Figure 5.21: Guided tracking, 'Towel' sequence. The top three charts show the estimated parameters of geometrical change of current image relatively to the first image. The most bottom chart shows the number of correctly tracked ZSPs depending on period in the first frame (7, 9, 11 – blue, 15, 19, 25 – green, 29, 39, 49 – red).

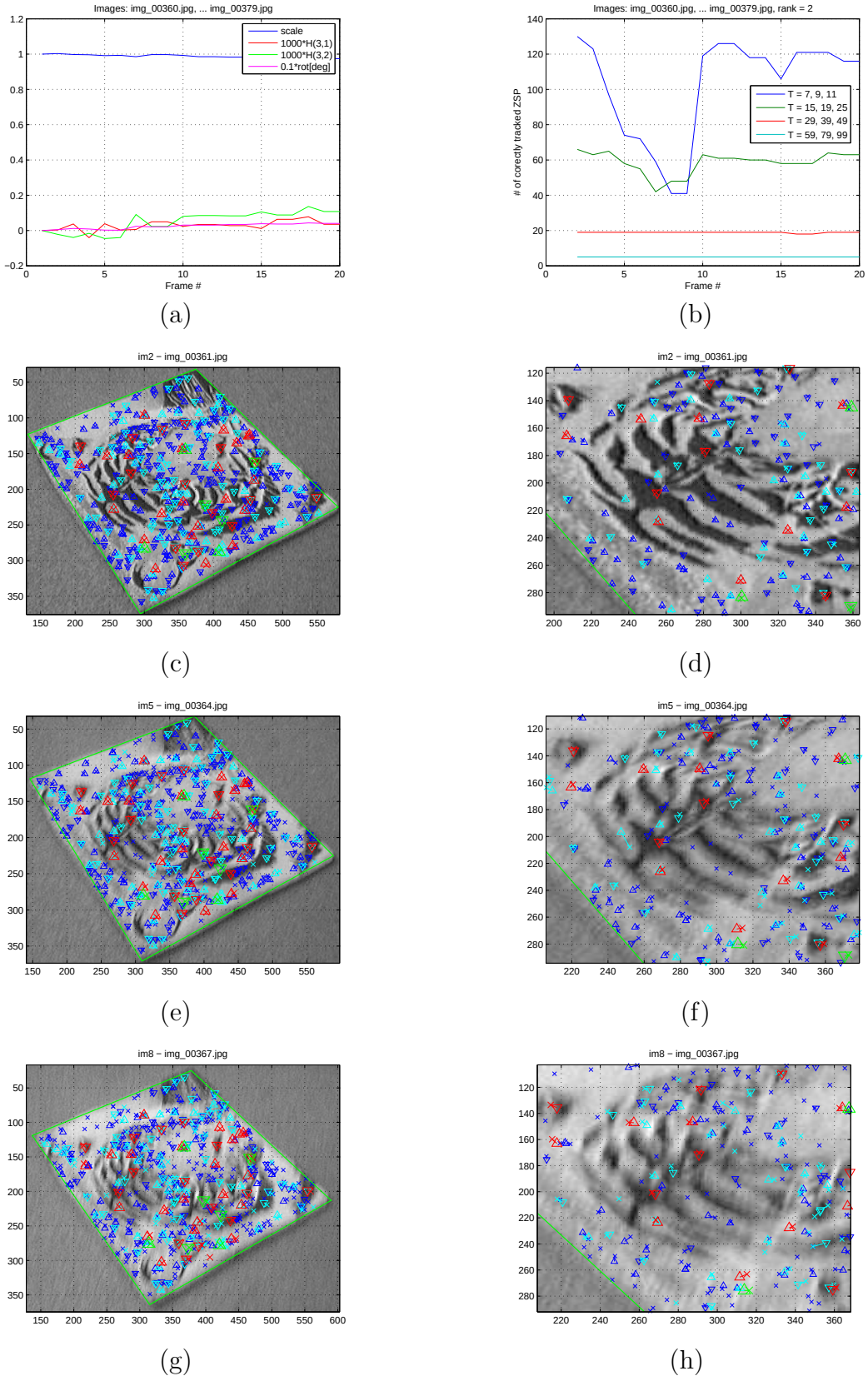


Figure 5.22: Guided tracking – effect of the blur. Crosses mark projected locations of ZSPs $\mathbf{x}'_i$, triangles mark locations found by tracker $\mathbf{x}''_i$. The color groups ZSPs according to their period. The lines of the same color connect corresponding projected and tracked locations.

6 The high-level tracker

This chapter deals with the i*ntegration of the low level tracker* from Chapter 5 into a *tracking system* aiming to track a more complex object than mutually independent points. One of the desired applications is to provide data for the camera pose estimation or/and for the mapping of the environment in which a human or a robot is moving. The high level tracker is called simply tracker inside this chapter.

6.1 General components and functions of the tracker

The general *problems solved by a tracker* are:

1. To discriminate the target against background.
2. To estimate a global motion.
3. To detect malfunctions of the tracking.
4. (Optionally) to predict a global motion in the next frame.

The *target may be represented as* a :

1. Segmented object against background. One or more such prospective targets may be segmented in the new image and compared with the reference target in the previous image or its global model.
2. Hypothesis about a global motion may be generated and verified.
3. A cloud of trackable points (or regions) having a similar (model dependent) behaviour. Tracking the target means to track these points and to estimate the global motion from individual local motions of individual points.

Only the systems based on point tracks are considered in this chapter. Such *tracker must solve the following problems*:

1. Tradeoff between the range and precision of tracked points.
2. Prediction of the local movement of individual points.
3. Detect outliers – mismatches.
4. Handle lost points – no corresponding point found in new image which is detected at the lower level.
5. Estimation of the global motion from the individual local motions of tracked points.
6. Short range of available ZSPs (Zero Shift Points) when tracking a small and fast target.

The low-level tracker produces some lost correspondences and some mismatches (which are usually worse than revealed lost correspondence). The percentage of such failures depends on the complexity of the sequence, e.g. on the scale, perspective changes, blur, unpredicted long displacement when the attraction basin is missed entirely, etc. Section 5.5 deals with the effects of such disturbances of the low level tracker in more detail.

6.1.1 Coarse to fine approach

The crucial part of the tracking is to solve the *tradeoff between the range and precision of the tracked points at different scales* properly. This problem is common to all trackers based on point/region correspondences [45, 3]. This problem is solved in the low level tracker sometimes, e.g. in the pyramidal KLT tracker in OpenCV library [44, 2]. The points corresponding to bigger regions (a courser scale) have a longer range and they can handle a longer displacement between consecutive images. However, the localization precision is worse compared to points corresponding to smaller regions (a finer scale). And vice versa, the points from the finer scale are more precise, however, they have the shorter range. The scale corresponds to period T in the case of Zero Shift Points (ZSPs).

Generally, the system should work in *coarse to fine manner*. At coarse scales the rough estimates of movement should be made and this information should be propagated into finer scales to achieve good precision of resulting global model (between frame homography or 3D pose estimation and environment map). The proper *propagation of information from course scales to fine scales* is essential. It is necessary conditions for tracking the points corresponding to small regions (which have small range). It is crucial to identify outliers at each scale to avoid propagated mismatches at lower levels which may cause the resulting global model completely invalid. The point to track have no chance to correct bigger prediction error than its range. The low level tracker may just say 'I am lost' by the comparison of surrounding regions in the current image and the previous image or if tracking distance (from initial position) is too long.

6.1.2 Zero Shift Points (ZSPs) and the image pyramid versus the pyramid of ZSPs (and integral image)

In a coarse to fine approach, there are two ways how to handle the scale. The first possibility is to build an image pyramid and use regions of the same size (e.g. the same smoothing σ for Harris detector or the same $N \times N$ mask for KLT tracker [2]). The second possibility is to work with the original image and change the size of regions. The first approach includes a costly preprocessing (an anti-aliasing filter should be applied) bringing the advantage of a scale independent complexity when tracking individual points. The second approach does not require the preprocessing step, however, the complexity of tracking each point depends (usually linearly) on the area of the region (i.e. it typically depends quadratically on the range).

In the case of ZSPs, a very simple integral image (see Section 5.3) can be used to reduce the complexity while tracking each point. The computation of this integral image is much faster than building the image pyramid. Using the integral image, the complexity of tracking each point depends linearly on its range.

Next, the number of ZSPs with period T decreases quadratically with growing T. As a result, the time spent on a more course level of the point pyramid is shorter than the time spent on a more fine level. Changing the period is more flexible/cheaper than changing the scales of individual levels of image pyramid. It also allows to follow the scale change of target more precisely and independently if several target are tracked in a single image. Therefore, the point pyramid instead of the image pyramid will be used in the rest of this chapter, even though both approaches are possible when ZSPs are used.

Another option is to combine the image pyramid and the integral image, i.e. to build a pyramid of integral images. The pyramid may change its resolution in an octave step. Several short periods (e.g. 7, 9, 11) may be used to improve the scale adaptation. Even for short periods, the integral image speeds up the computation.

6.2 Point to target models

This section describes several multiscale models imposing some geometric assumptions on the target. All of them use the ZSP pyramid. ZSPs are divided into the levels of the pyramid according to their period. The pyramid my be organized in various ways. One of the possible solution is described in Algorithm 10. This construction of the pyramid was used in the experiments of this chapter.

1. Find ZSPs in the first image using Algorithm 7.
2. (Optionally) Discard all ZSPs with low rank (0,1) or discard selectively ZSPs with short periods and low rank to reduce the computational cost.
3. Sort ZSPs in a descending order according to the period.
4. Find the border period T_0^b for the highest level of the pyramid. Given n_{min}^h (the required minimum number of ZSPs in the highest level of the pyramid), $T_0^b = T_k - 1$, where $k = n_{min}^h$ and T_k is the period of the k-th ZSP sorted according to Step 3.
5. Compute other border periods $T_1^b = 0.75T_0$, $T_i^b = 2T_{i-1}^b, i = 1 \dots n$, where n is the biggest integer so that $T_n^b < 2T_0^b$.
6. ZSPs with the period T in the range $T_i^b <= T < T_{i+1}^b$ form the level i of the pyramid. (The first level contains ZSPs with shortest periods.)
7. The top level of the pyramid contains ZSPs with $T > T_0^b$.
8. The two highest levels of the pyramid may have a nonempty intersection.
if *the second highest level is included entirely in the top level* **then**
| reduce the number of levels by removing the second highest level.
end

Algorithm 10: Construction of the ZSP (Zero Shift Point) pyramid.

ZSPs found by Algorithm 7 are already pre-sorted to a natural pyramid during the search implicitly. The natural pyramid levels differ in the period by the octave because the searched periods are $T_0^s = T_0$, $T_{i+i}^s = 2T_i^s + 1$. Some of ZSPs changed the period to the nearest odd integer of $\{T_i^s - T_i^s/4, T_i^s + T_i^s/4\}$ in the refinement step. Algorithm 10 respects the natural levels given by Algorithm 7 because the attraction basins of the points at the single level of the natural pyramid do not overlap and cover the image uniformly, i.e. one blob corresponds to one ZSP at the level. This natural pyramid is then adjusted by forcing the top level to contain required minimum number of points which enables detecting outliers. The two highest levels of the pyramid may have a nonempty intersection which allows to obtain just ZSPs with the longest period from the second highest level if the natural highest level had few points and the points taken from the second highest level will be still available for outlier detection in the second highest level.

6.2.1 Local coherence without an explicit geometric model of the target

This model imposes probably *minimal assumptions on the target and its movement.* Only a *local coherence* is assumed as close points move similarly. The local movement is approximated by the translation. Small scale changes, rotation, and perspective deformations of the target can be approximated by the different local translation in different parts of the target. The global motion is not evaluated by the tracker itself. The target is a cloud of points covering either the whole reference (first) image or its part.

The *locally coherent model*:

- Groups ZSPs into the levels of pyramid defined by Algorithm 10
- Describes the neighborhood of each ZSP $\mathbf{x}_i$ on its level of pyramid, i.e. finds the set of m nearest points or the set of m_i points which are closer than some maximal distance. The choice of the neighborhood set may depend on the application. In the experiments presented, the m nearest points are preferred to guarantee enough number of points for the outlier identification.
- Assigns the predictor to ZSPs on all pyramid levels except of the highest level. The predictor of ZSP at the level l is the nearest ZSP at the level $l + 1$.

The *tracking* is performed by Algorithm 11. Figure 6.1 shows the locally coherent model on the detail of the first image of 'Mouse pad' sequence. There are blobs of different size and quality. Most of them are small blobs detected just on the lowest level. Bigger blobs are usually (not always) detected on several levels (2-3) it means they are robust to scale change of more than 3-4 octaves.

Well symmetric blobs (e.g. eyes – two most top-right cyan/blue triangles or the most top-right group of red, cyan, blue triangles) are localized in almost the single point at several levels. Less symmetric blobs are localized on only single level or the locations at different levels are shifted (e.g. the biggest dark blob with green and

1. Set level l to the highest level of the pyramid.
2. Track individual ZSPs at the level l using Algorithm 5.
3. Detect outliers – points whose disparity differs from the disparity of neighbors more than δ_m.
4. (Optional.) Remove points that are outliers for more than k consecutive frames. The neighborhood sets and predictors should be rearranged.
5. Correct the outlier disparity – replace the disparity of outliers by median/mean disparity of their neighbours.
6. Correct outlier position for the next frame to its initial position plus disparity.
7. (Optional.) Retrack outliers from their corrected positions. Consider ZSPs shifted more than δ_m as outliers and return them back to their position before retracking.
8. **if** $l = 1$ **then**
  Finish.
else
  $l = l - 1$
end

9. Propagate the disparity to ZSPs on level l from $l + 1$ – add disparity of the nearest point from the higher level to the initial position of the ZSP to be tracked. Go to Step 2.

Algorithm 11: The tracking of the locally coherent model.

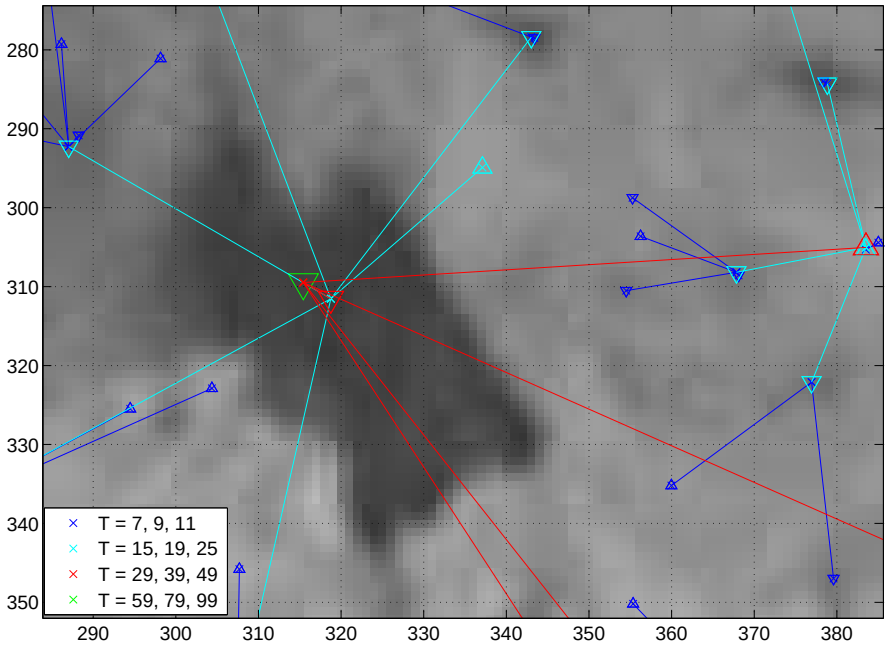


Figure 6.1: A locally coherent model illustrated on the detail of the 'Mouse pad' sequence. The color distinguishes the levels of the pyramid (blue – lowest, cyan, red, green – highest). Crosses show locations of ZSPs, the orientation of triangles (the same color/location as the cross) shows the polarity of the wave, more precisely of the cosine coefficient. The line of the same color as the cross/triangle connects ZSP at level l to its predictor ZSP at the level $l+1$.

red triangles or the most top-left blob with cyan and blue triangles). Single ZSP al level l predicts displacement for typically 2-6 ZSPs at level $l-1$. The two ZSPs at the same level of different polarity cannot be mixed up even being near (the basic property of low-level tracker Section 5.1). The correspondences of ZSPs at different levels are established separately, therefore, neither ZSPs from different levels cannot be mixed up if the consequent images have similar scale.

6.2.2 Extensions of the locally coherent model

The *basic locally coherent model* may be *insufficient for long tracks*, in which the change of the global scale (moving target) or the local scales (moving camera – the near regions change their scale more fast than more far regions) is significant. Therefore, this tracker may be considered more as a middle-level than a high-level tracker (equivalent approximately to the pyramidal KLT tracker in the OpenCV library [44, 2]).

The *performance can be improved* if the *tracker cooperates with the application*, which uses the tracked points. It means if the tracker can use information revealed by the application such as global or local scale, prediction of global movement or further outlier detection criterion and outlier correction mechanism. Probably the most important information is the *scale estimate for the period adjustment* of the low-

level tracker. The importance of the period adjustment was proved in Section 5.5.2 and Section 5.5.3. The example of the use of locally coherent model to track planar target is studied in Section 6.3.1. The tracker with the locally coherent model can be used also for measuring of 3D positions of points relative to the camera and the estimation of the camera movement. Here, the change of local scale of each point can be estimated from the change of the distance between the point and the camera. Outlier detection criterion may be epipolar geometry or reprojection error.

Locally coherent model and RANSAC estimating planar homography

The connection of the tracker with the locally coherent model to RANSAC estimating homography is used as a model example how to create the application exploring the advantages of tracking ZSPs. Suggested system tracks the planar target and estimates homography between the reference (e.g. the first) frame and the current frame.

The *homography* may be used for *several purposes* inside the tracker:

1. The *estimate of the scale change for the period adjustment* (essential for successful tracking, see Section 5.5.2 and Section 5.5.3). The scale can be obtain from homography or independently using the first two steps of Algorithm 13. Because RANSAC may be time consuming to be run every frame the independent fast scale estimate may be better in applications were the knowledge of the homography of every frame is not necessary. The scale estimate by Algorithm 13 may be more reliable then the estimated homography (as the experiments suggested).

2. *Outlier detection* according to reprojection error.

3. *Refreshment.* The points from the reference image may be time to time (e.g. each 5th – 100th frame) projected by homography to the current image, retracked to eliminate imprecision of homography and to adopt to possible changes of target and then used to track in the new image instead of possibly lost or drifted points tracked frame by frame. After retracking, the points that moved too far from their projected positions should be removed (or alternatively their position may be adjusted in the reference image by the inverse of the homography).

4. *Renew.* The deformation of the targed or significant scale or perspective change may cause ZSPs (which are continuously tracked from the reference image) to disappear or become unstable. If such problem is detected or expected (e.g scale changed significantly) it is better to set the current image as reference, find a new set of ZSP and construct new model. Next tracking will be performed with new model and homographies will be estimated between new reference image and current image. If the homography between the first and the current image is necessary then it can be computed by multiplying the homography matrices of individual subsequences. It is not recommended to perform renew too often, because each renew potentially introduce drift due

to the break of feedback between the current and the first image formed by point correspondences. Each estimated homography has some imprecisions and they are accumulated.

5. *Warp.* The new image may be warped using homography between reference and last tracked image to minimize geometric distortion. It eliminates the need for period adjustment and reduce effect of perspective on ZSP localization. It seemingly saves ZSPs with short periods from dissapearing, however, blob formed from single pixel by warp may be very unstable. Alliassing effecs may occure when 'zooming out' by warp.

Range extension

Small target may contain only small amount of ZSPs with long period and even their periods may be too short to have sufficient range for fastly moving target or fast rotation around principal axis. Both phenomena occurs often in the sequences used for experiments. Both the *range and resistance to rotation can be easily extended* using approach which generate and test hypothesis about global motion. Algorithm 12 describes generally range extension for any tracker **T** based on point tracks which is able to evaluate the quality of tracking result.

1. Track points $\{x_i^0\}$ using tracker **T** to $\{x_i^1\}$.
2. **if** *tracking quality is sufficient* **then**
  Finish, $\{x_i^1\}$ are tracked points.
end

3. **for** *all available global motion hypothesis* **H** **do**
  Transform $\{x_i^0\}$ to $\{x_i^{0H}\}$ according to **H**.
  Track points $\{x_i^{0H}\}$ using tracker **T** to $\{x_i^{1H}\}$.
  if *tracking quality is sufficient* **then**
    Finish, $\{x_i^{1H}\}$ are tracked points.
  end
end
4. **if** *all available hypothesis* **H** *were tested* **then**
  Select $\{x_i^{1H}\}$ with the highest quality or $\{x_i^1\}$ if its quality is better than all $\{x_i^{1H}\}$
end

Algorithm 12: Range extension.

There are obvious global motion hypotheses:

1. Rotation around the principal axis. It can be preformed fastly by camera motion and it causes big displacements of individual points. It generates basically two hypothesis (or multiples of two if symmetry is assumed).

2. Translation. Basically at least four directions should be tested.

3. The combinations of the two previous.

Application dependent hypothesis may exist which is more probable in some circumstances.

Considering tracking of ZSPs, both mentioned *global motion hypotheses are relevant*. The size of step (rotation angle or the length of translation) should create compact area as an union of attraction basins of x_i^{0H} at the top level. A diameter of the inscribed circle centred in ZSP of a typical attraction basin is about $T/2$. Therefore, digonal translations of length $\delta = T_x/\sqrt{2}$ are recommended, T_x is the shortest period on the top level of the pyramid. Diagonal translation means that the vectors of translation are $[\pm\delta, \pm\delta]$. This is an intuitive recommendation successfully used in the experiment of Section 6.3.3 without any tuning. Similarly, the rotation step was set to $\pm 10^o$ by intuition. A further analysis may reveal more exact rule.

6.2.3 The planar target

The *planarity* or the *local planarity of the target* is a common assumption in tracking. This approximation may by useful also for nonplanar target which are more distant from the camera than their depth and do not change the orientation with respect to the camera significantly. The global motion may be approximated by some of the 2D to 2D transformations, which are sorted by their complexity:

1. Translation.

2. Translation and scale.

3. Translation and rotation (probably not useful, not common).

4. Similarity – translation, rotation and scale.

5. Affinity.

6. Homography.

The transformation can be *estimated using RANSAC* [5, 4] or some other robust method or using a simple least-squares method available in easy to use OpenCV library [44]. Unfortunately, the OpenCV implementation of RANSAC is not the best (in both speed and performance). The similarity and simpler transformations may be robustly estimated more easily and more quickly than homography or affinity as described in Section 6.2.3 and in some situations may work better.

The robust estimation of the similarity

Translation, scale and rotation can be estimated separately and integrated to a similarity matrix. Such estimation is very simple and fast. The method was inspired by the work [18] which uses median to estimate translation and scale change. The

presented method extends to rotation and it adds minimum distance requirement on the point pairs used for the estimation. This requirement increases 'signal to noise ratio' using a simple observation, namely, that the difference of two close noisy points is unstable. The robust similarity estimation is described in Algorithm 13.

1. Find the corresponding pairs $\{[\mathbf{x}_i, \mathbf{x}_j], [\mathbf{x}'_i, \mathbf{x}'_j]\}$ of points with both mutual distances $d_{i,j} = ||\mathbf{x}_i - \mathbf{x}_j||_2$ and $d'_{i,j} = ||\mathbf{x}'_i - \mathbf{x}'_j||_2$ longer than d_{min}. The points $\mathbf{x}_i$ and $\mathbf{x}_j$ are from the first image, the points $\mathbf{x}'_i$ and $\mathbf{x}'_j$ are from the second image.
2. Estimate scale as $s = \text{median}(d'_{i,j}/d_{i,j})$.
3. Compute angles φ_{ij} and φ'_{ij} of the abscissae $[\mathbf{x}_i, \mathbf{x}_j]$ and $[\mathbf{x}'_i, \mathbf{x}'_j]$ and their difference δ_{ij} using Equation 6.1.
4. Estimate the rotation as $\delta = \text{median}(\delta_{ij})$.
5. Compose the similarity matrix $\mathbf{S}'$ from the zero translation, scale s and rotation δ.
6. Project all $\mathbf{x}_i$ to $\mathbf{x}''_i = \mathbf{S}'\mathbf{x}_i$.
7. Estimate the translation vector $\mathbf{t} = \text{median}(\mathbf{x}'_i - \mathbf{x}''_i)$.
8. Integrate the translation into $\mathbf{S}'$ to obtain the transformation $\mathbf{S}$ from the first image to the second image.
$\mathbf{S} = \mathbf{S}'$
$\mathbf{S}(1,3) = \mathbf{t}(1)$
$\mathbf{S}(2,3) = \mathbf{t}(2)$

Algorithm 13: The estimation of the similarity from point correspondences.

The *estimation of the translation* and scale is straightforward. However, the rotation is a bit tricky. The problem is in both the difference and the median operation over the angles which must be carefully treated with respect to the values of the individual angles and their relations. The essence of the problem is that the angle $-\pi$ and π is the same point in this context and when the angles are near but on opposite sides of this points they must be treated differently than, e.g., the near zero case. The computation of the difference of angles is described by Equation 6.1 (symbols are defined in Algorithm 13).

$$\delta_{ij} = \begin{cases} \varphi'_{ij} + \varphi_{ij} & \text{if} \quad \varphi'_{ij} - \varphi_{ij} > \pi \\ -(\varphi'_{ij} - \varphi_{ij} + \pi) & \text{if} \quad \varphi'_{ij} - \varphi_{ij} < -\pi \\ \varphi'_{ij} - \varphi_{ij} & \text{otherwise.} \end{cases} \tag{6.1}$$

The problem with the median is that the point $-\pi \sim \pi$ should be continuous and, e.g. the set of angles $\{-3, -3.1, 3.1, 3\}$ must have this order or the order $\{3, 3.1, -3.1, -3\}$ and not $\{-3.1, -3, 3, 3.1\}$ as the usual mathematical order. At the first sight it could be solved by using the angles in the range $\langle 0, 2\pi \rangle$. However this range has a discontinuity at zero with the same effect. The solution is to choose the range of angles depending on their values.

1. Set initial positions of ZSPs $\{\mathbf{x}_i^0\}$ to the positions found in the previous frame.
2. Adjust periods of ZSPs according to the scale change estimated in the previous frame. Discard ZSPs with $T < 5$.
3. (Optional.) Apply predicted global motion (e.g., a displacement predicted from the known target velocity).
4. Set level l to the highest level of the pyramid. Set $\mathbf{H}_{l+1}$ to identity.
5. Project $\mathbf{x}_i^{0l}$ on level l to $\mathbf{x}_i^{1l} = \mathbf{H}_l \mathbf{x}_i^{0l}$, $i = 1 \dots m_l$, where m_l is the number of points on the level l.
6. Track $\mathbf{x}_i^{1l}$ using Algorithm 5 to $\mathbf{x}_i^{2l}$.
7. Estimate the transformation $\mathbf{H}_l$ from $\mathbf{x}_i^{0l}$ to $\mathbf{x}_i^{2l}$.
8. Detect outliers, i.e. points the tracked location $\mathbf{x}_i^{2l}$ of which is shifted from the predicted position $\mathbf{x}_i^{1l}$ more than αT_i, $\alpha \leq 1/2$.
9. (Optional.) The local optimization. (Give the second chance to outliers: use $\mathbf{H}_l$ to predict their start positions $\mathbf{x}_i^{1l}$, track them again and then refine $\mathbf{H}_l$.)

10. **if** $l > 1$ **then**
| $l = l - 1$, go to Step 5
end
11. Estimate the transformation $\mathbf{H}_G$ between the reference image and the current image using points on the lowest level of the pyramid.
12. Detect outliers with respect to $\mathbf{H}_G$ and replace their positions by $\mathbf{H}_G \mathbf{x}_i^0$.
13. (Optional.) Discard points that are outliers for more than k consecutive frames from the list of candidates for tracking in next frames.

Algorithm 14: The general algorithm to track a planar target.

Tracking a planar target – the general algorithm

The estimated planar transformation is useful not only as a possible output of tracking. It is also useful in the propagation of information from course to fine scales to provide a more precise starting point for ZSPs with a shorter period (and therefore a shorter range). The transformation can be also used as a constraint to detect outliers. Finally, it may be used to compute starting points for tracking the next frame. It is valuable to keep the same set of points (or at least its trackable subset) as long as possible to reduce the drift caused by the cumulated imprecision of between frame estimates.

Algorithm 14 describes the general algorithm for tracking a planar target. The model for tracking consists of the ZSP pyramid (Section 6.2) and a set of transformation matrices $\{\mathbf{H}_i\}$. The transformation $\mathbf{H}_i$ can be any of the transformations listed at the beginning of Section 6.2.3. A different transformation may be used, at different levels of the pyramid and for different functions during tracking. For the rough information, a simpler transformation is better as less degrees of freedom imply a lower chance of overfitting. E.g., the transformation which propagates information from the highest level of the pyramid to the second highest level may be just a translation. To propagate information into other levels and to identify outliers, the similarity may be precise enough and still fast and robust. The most general homography may be necessary to estimate transformation between the reference frame and the current frame and to project points from the reference frame to initial tracking positions in the new frame.

6.3 Experiments

6.3.1 The model of the local coherence

This section describes tracking experiments using the locally coherent model and several of its improvements (period adjustment, re-tracking outliers, and the connection with the geometrical model of the target). The periods of ZSPs to track in the current image $\mathbf{I}_i$ are adjusted using the estimate of the global scale change computed from point correspondences between the reference image $\mathbf{I}_0$ and the previous image $\mathbf{I}_{i-1}$ found by the tracker. A change in the scale is estimated according to the first two steps of Algorithm 13.

The effect of the tightness of the outlier detection criterion and the period adjustment

The first experiment examines the performance of the basic locally coherent model and the effect of the tightness in the outlier detection criterion. The tightness is controlled by the threshold δ_m of Algorithm 11. The basic model is compared with its essential extension – the period adjustment. Figure 6.2a shows the model. The reference image serves as a background. There are five ZSPs at the top level of the pyramid with periods 99, 79 and 59 pixels giving the range of the displacement over

30 pixels which the tracker still handles. The target is well covered by points at all scales of the pyramid.

Figure 6.2b shows parameters of geometric changes computed from the ground truth and the scale change estimated from the tracked points. The latter fits well to the scale computed from the ground truth. The first half of the sequence exhibits only a small scale change. In the second half of the sequence, the target moves continuously away from the camera and the scale drops to 1/2 at the end of the sequence. The first 20 frames have almost a constant scale but some degree of the perspective distortion can be observed which leads to decrease of the repeatability shown in Figure 6.3.

The criterion for distinguishing outliers should be a rather loose one. If the criterion is too tight then many points are lost due to the model inability to follow the changing target as can be seen in Figure 6.3a and Figure 6.3b. Relaxing the outlier detection threshold δ_m leads to a significant improvement of the repeatability. The advantage of a more flexible model is even more pronounced in the second half of the sequence, where the scale was changing fast and the tracker with the overly tight model failed soon. It is not due to the localization error of the low level tracker which is much smaller compared to $\delta_m = T/2$. E.g. $\delta_m = 3.5$ pixel for the shortest period was used and 80% of inliers ($T = 7$) were localized with the error smaller than 2 pixels. For longer periods, it is even more evident. E.g., for the longest period $T = 25$ on the second level of the pyramid, the tolerance $\delta_m = 12.5$, However, 80% of inliers were localized with the error relative to the ground truth which is smaller than 4 pixels, see Figure 6.2 d).

The relaxed model was able to deal with a much bigger scale change even without the period adjustment. It managed to track over 30 points to the end of the sequence with the error less than 3 pixels. These good points allowed RANSAC to estimate the homography between the first and the last image of the sequence with a reasonable error (the error of the location of the target corners was [0.40, 1.63, 1.36, 3.15]% of the target size (see Figure 6.4). Notice that RANSAC was used only to evaluate the quality of the tracked points and not to support the tracker.

The model is able to carry a substantial percentage of small points over difficult frames and lock them to the correct positions again later. It can be seen in (Figure 6.4 c), e) at the end of the sequence. There are groups of frames in which RANSAC is not able to reveal the correct homography and some other groups of frames where the homography was estimated correctly. The period adjustment improves significantly the performance of the tracker as can be seen in (Figure 6.4b), d), f) – RANSAC could reveal the correct homography in all frames of the sequence with the error mostly less than 0.5% in the first half of sequence (with a small geometric distortion) and less than 1% in the most of the frames in the second half of the sequence (with a significant scale change).

The similar effect as δ_m has the number of neighbours used to identify outliers. More neighbours do not perform better in the presence of geometric distortions other than a simple displacement. The bigger required number of neighbours leads to the longer distance between them and the assumption that the local motion can be explained just as a translation (even a different one in different parts of image) is

violated. The experiments in which the neighbourhood was tuned are not presented here even though they were performed. It was observed that the best number of neighbours was the minimal one, i.e. the required number of neighbours was set to three and it was used in all experiments presented.

Re-tracking the outliers and the connection with the geometrical model of target (homography estimated by RANSAC)

Figure 6.5 shows a much more challenging part of the 'Mouse pad' sequence in which the geometric distortion changes fast in a wide range, both in the scale and in the perspective. The best setting from the previous experiment was used (period adjustment and $\delta_m = T/2$, see Figure 6.6) and compared with two other improvements – (1) re-tracking the outliers and (2) the connection with the geometrical model of the target (see Figure 6.7).

Re-tracking of the outlier is cheap. The low level tracker is run from the corrected position. The distance which the point moved during the re-tracking is tested and the points that moved a short distance are returned to the set of inliers. Re-tracking significantly increases the number of the smallest correctly tracked points. It can be seen again in Figure 6.7 e) that small points are important because they are the most precise ones. The effect is most pronounced between the frames 120 and 160th in which the used scale returned near to the value it had at the beginning of the sequence. The points which were not tracked in the previous part of the sequence (they were only transferred according to the movement of their predictors) were tracked again and some of them locked to their correct positions.

A more powerful and at the same time a more demanding is the connection to the geometric model of the target. The planar homography is used as a geometric model here. RANSAC was employed to estimate the homography between the reference (the first) frame and the current frame. Only inliers identified by the tracker were passed over to RANSAC.

The homography estimated by RANSAC was used to refresh the model. All points detected in the reference image were projected to the current image and tracked again. The points which moved further from its start positions during tracking were removed. The remaining points were used as the starting positions for the next frame. The refreshment was performed every 5 frames in the experiment in Figure 6.7. The period of refreshment can be observed well on the results as the step improvement of repeatability followed by its decrease. There is a big step in the number of correctly tracked points in the frame 110 caused by the change of the scale back to the value approximately equal to one. This allowed to use small points in tracking again which would not be possible due to their short period in the previous frames.

The homography projected them near to their correct positions so they could have been tracked again. The percentage of 'returned' points was significantly higher compared to the case when their positions were just transferred according to their predictors. Compare Figure 6.6 c) period adjustment and transfer, Figure 6.7 a) period adjustment, re-track and transfer and Figure 6.7 b) period adjustment, re-

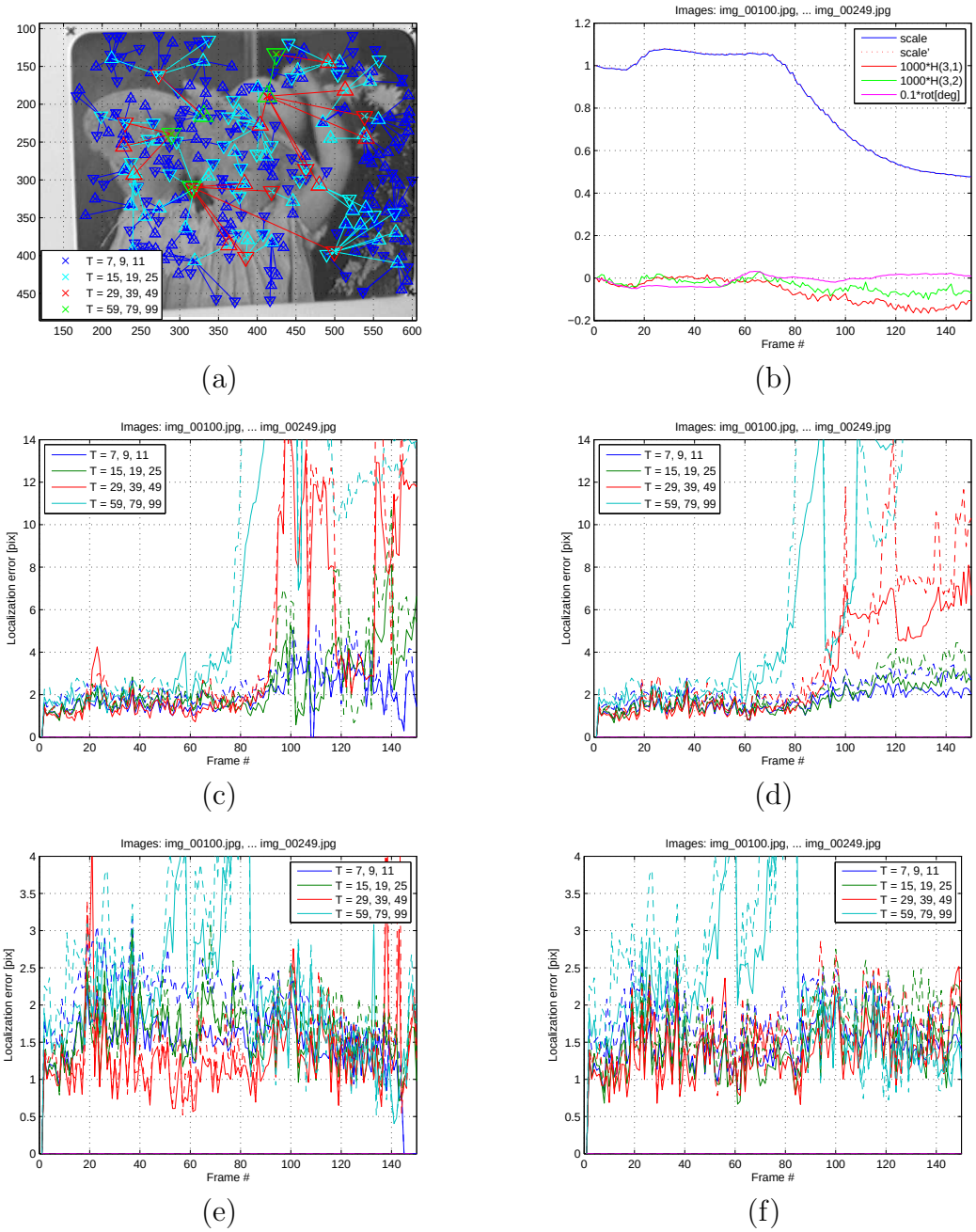


Figure 6.2: The 'Mouse pad' sequence. Tracking with the model of the local coherence. a) The model on the reference image (see the caption of Figure 6.1 for colours and symbols explanation). b) The parameters of the geometric distortion estimated from the ground truth and scale (marked with ' in the legend) estimated from the tracked data (almost no difference of the two scale estimates). c), d), e), f) The localization error of inliers (identified by the tracker itself), solid lines – average error, dashed lines – 80% percentile, c), d) Without period adjustment. e), f) With period adjustment. The effect of the internal outlier detection threshold δ_m c), e) $\delta_m = T/4$, d), f) $\delta_m = T/2$.

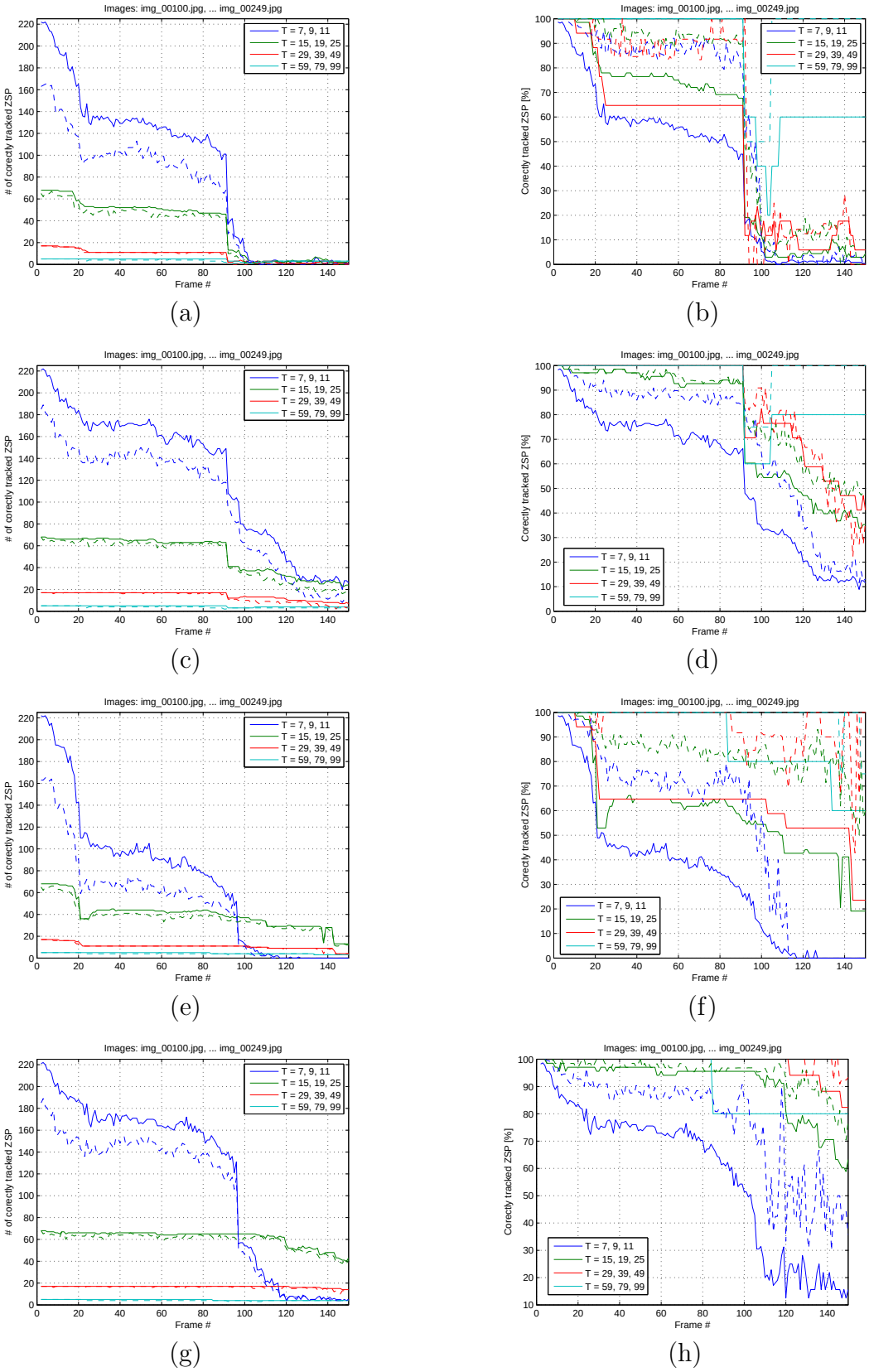


Figure 6.3: The 'Mouse pad' sequence. Tracking with the model of the local coherence – repeatability. Solid lines denote all points consistent with the ground truth. These are both the points recognized by the tracker as inliers and the outliers corrected by the tracker. Their corrected position is evaluated. Dashed lines represent points which were recognized by the trackers as inliers and which were inliers in ground truth information too.

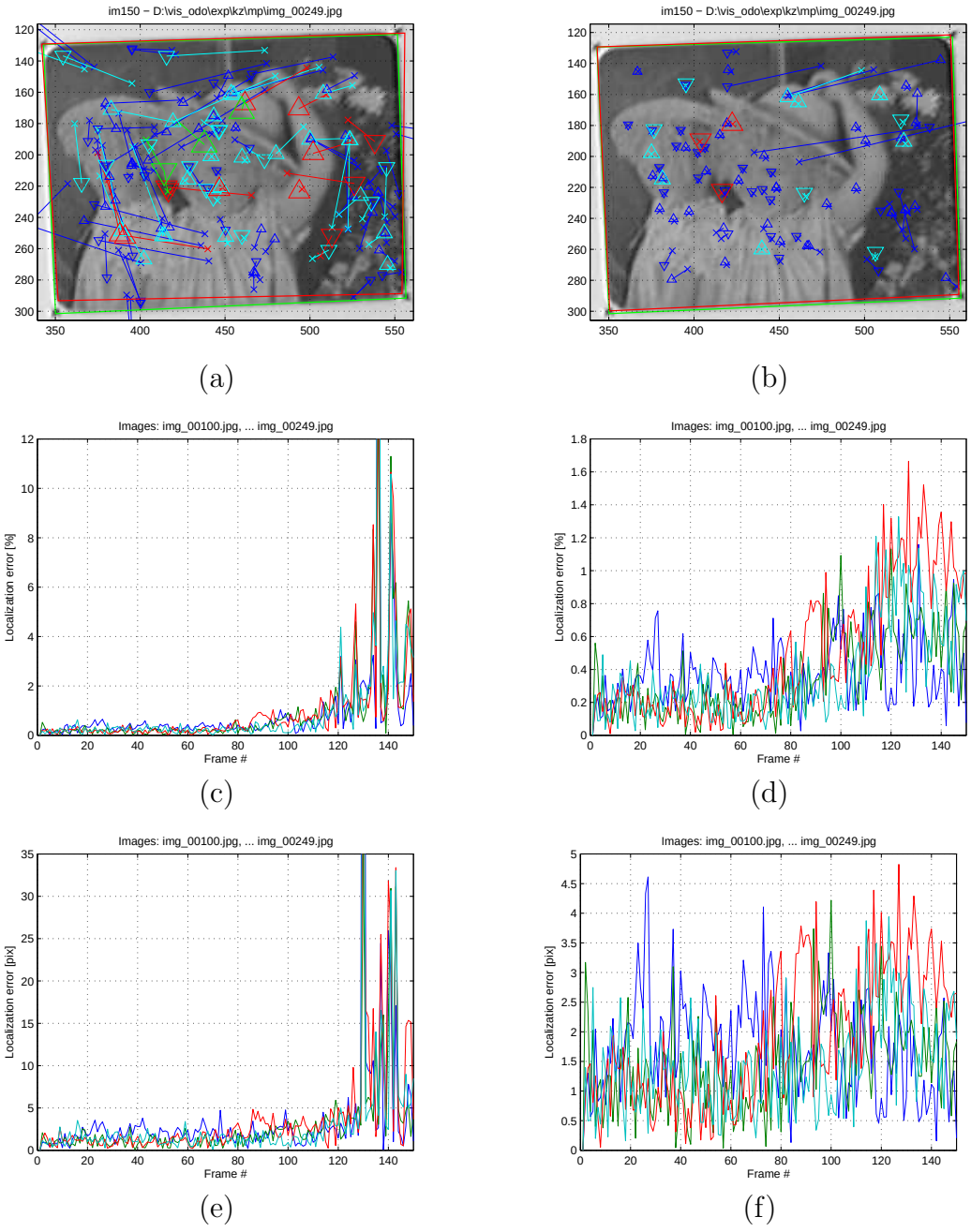


Figure 6.4: The 'Mouse pad' sequence. Tracking with the model of the local coherence. The localization error of target corners relative to ground truth. The percentage error is relative to the size of the target in the current frame (the length of the longer target diagonal). a), c), d) Without period adjustment. b), e), f) With period adjustment. Each color corresponds to a target corner.

track, RANSAC and projection.

6.3.2 Comparison with the state-of-the-art

This section compares the results of tracking with the improved locally coherent model (period adjustment, refreshment by RANSAC estimating homography) and the state of the art reported by [45], where their "tracker formed by Number of Sequences of Learned Linear Predictors (NoSLLiP)" is compared with (1) the tracker based on SIFT [24], (2) KLT tracker [25], and (3) another tracker based on Sequences of Learned Linear Predictors (SLLip) [15]. Unfortunately, the results of other trackers than NoSLLiP are provided only for 'Mouse pad' sequence. The criterion for comparison is the number of loss-of-locks. The loss-of-lock occurs when the location of the target corners estimated by the tracker differs from the ground truth more than 25% of the length of the top edge of the target [45]. At that frame, the tracker is restarted to continue from the correct position.

The tracker with the improved locally coherent model (ILCM) uses homography to refresh every 5 frames and to renew if the scale changed out of range $\langle 0.6, 1.5 \rangle$ or too many points were lost. For meaning of 'refresh' and 'renew', see Section 6.2.2). The scale change was estimated independently on homography using Algorithm 13.

The numbers of loss-of-locks on 'Mouse pad' sequence are 13 (NoSLLiP), 281 (SIFT), 398 (KLT), 1083 (SLLip), 93 (SLLip, half range). The ILCM tracker had 92 loss-of-locks. The number of frames in the sequence was 6935. The numbers of loss-of-locks on the 'Towel' sequence (3229 frames) was 5 (NoSLLiP) versus 8 (ILCM). The 'Phone' sequence contains 2300 frames provided on the web. However the results published in [45] are from the first 1800 frames as the author confirmed on our request without specifying the reason. The numbers of loss-of-locks on the first 1800 frames was was 20 (NoSLLiP) versus 9 (ILCM). The last 500 frames are the most difficult ones: ILCM tracker has another 8 loss-of-locks there, i.e. 17 in all 2300 frames.

The reasons for loss-of-locks of ILCM tracker was the same in all three sequences: the fast movement of the small target ($\approx$50%) or a big change of the geometrical distortion ($\approx$30%) and the failure of RANSAC due to a small number of points in an unlucky geometric relation each to other (even though that points were correctly tracked). Some of the frames of the sequence have a completely wrong ground truth. One or two corners were out off the image, the reconstruction of the ground truth would be possible only by the projection according to homography from some other image, in which all corners are visible, or by someone's manual guess. We did not want to manipulate with the ground truth so this frames were also counted as failures (probably the same approach as in [45], nothing was mentioned about incorrect ground truth there).

6.3.3 The model of the local coherence with the range extension

This experiment evaluates efficiency of the range extension method described in Section 6.2.2. The tracker with locally coherent model works inside range extending

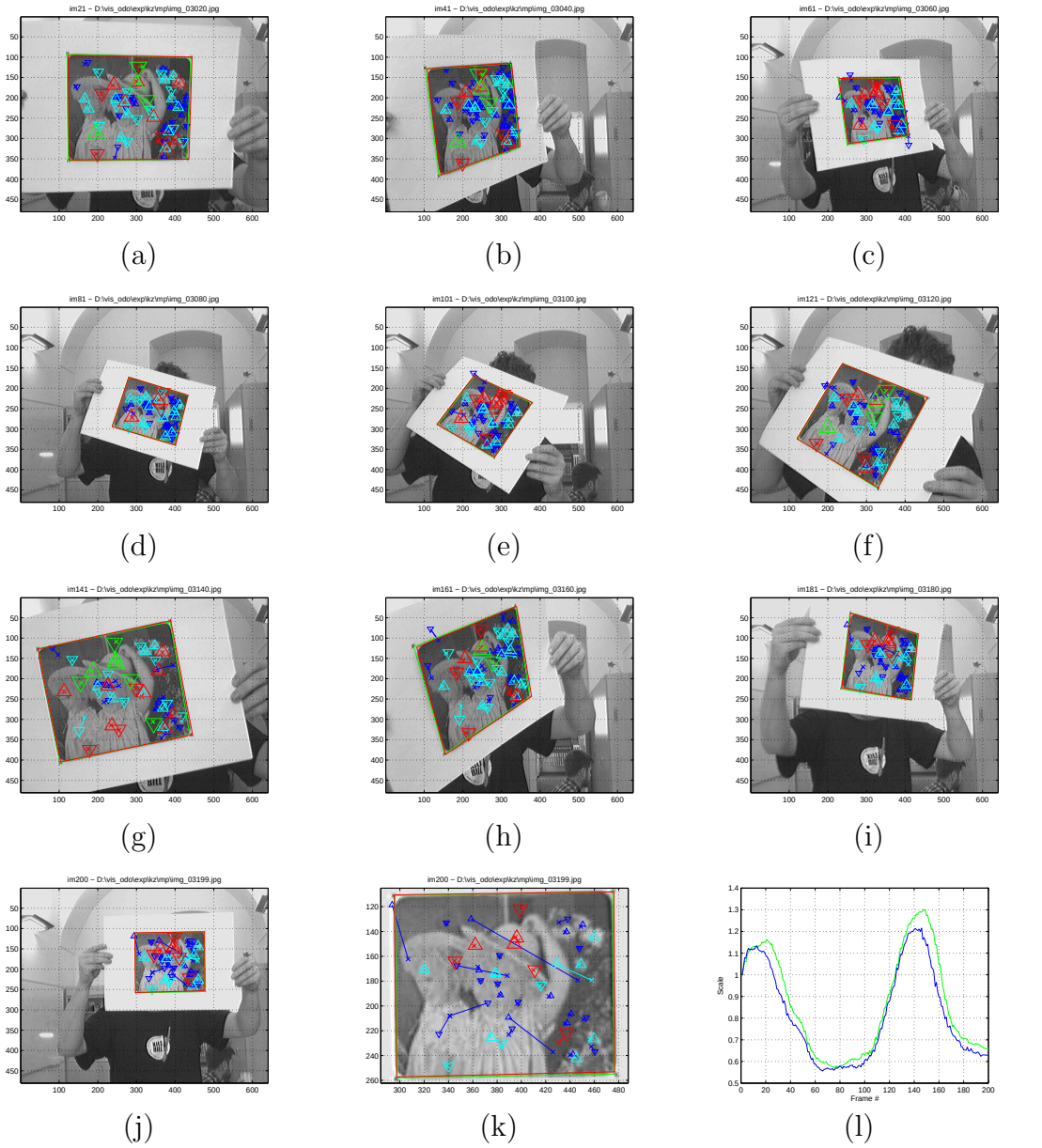


Figure 6.5: The 'Mouse pad' sequence. Tracking with the model of the local coherence with period adjustment. Examples of images from tracked sequence. The green quadrilateral shows the ground truth for the current image, the red quadrilateral is the projection of the ground truth quadrilateral in the first image to the current image. l) The scale change estimated from the ground truth (green) and tracked points (blue).

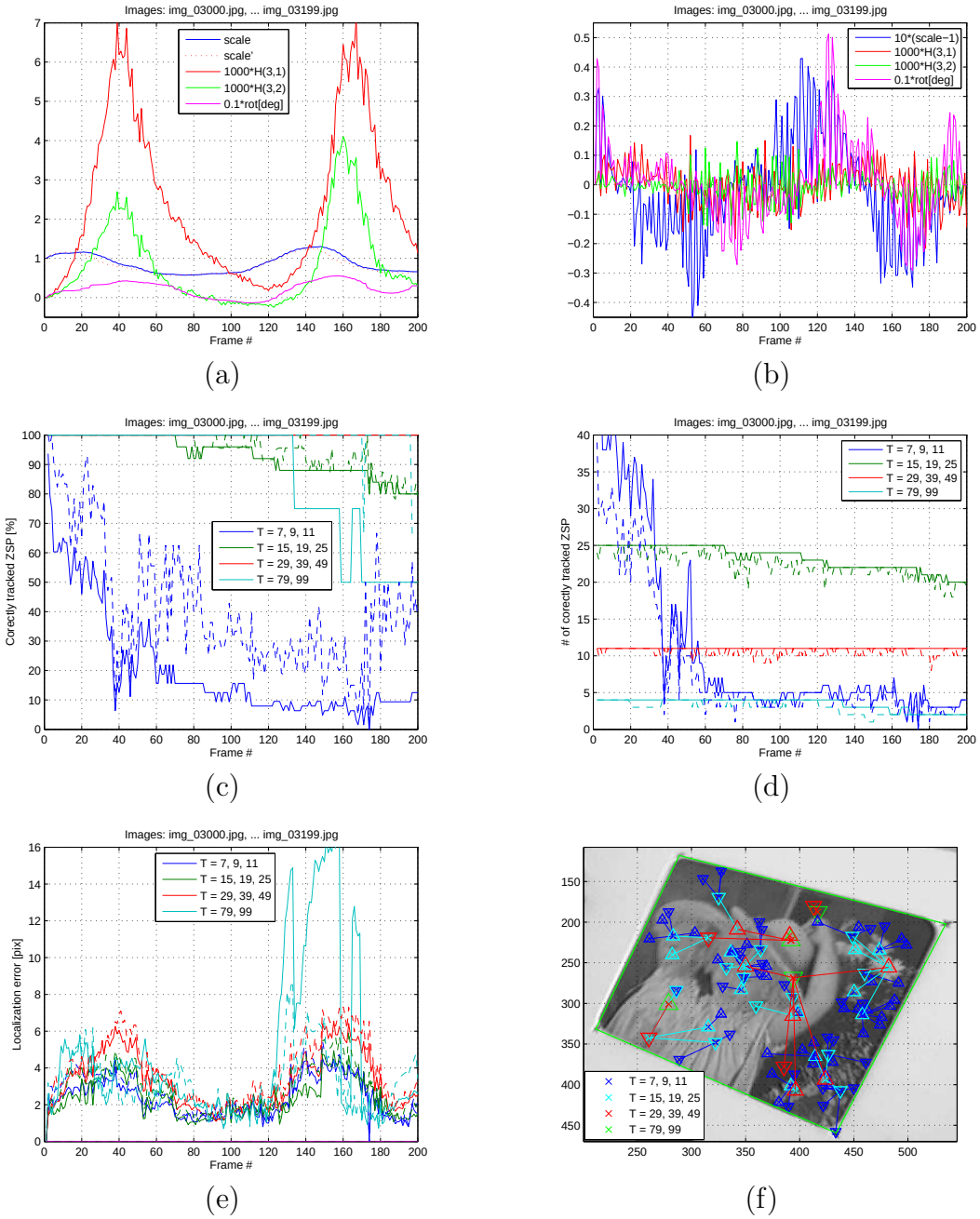


Figure 6.6: The 'Mouse pad' sequence. Tracking with the model of the local coherence with the period adjustment, $\delta_m = T/2$. a) Geometric distortion parameters (current to reference image) estimated from the ground truth and the scale (marked with ' in the legend) estimated from the tracked point. b) Geometric distortion parameters (current to previous image) estimated from the ground truth. c) The percentage, d) the number of correctly tracked points consistent with the ground truth. The meaning of lines is same as in Figure 6.3 e) The localization error of inliers (identified by the tracker itself), solid lines – average error, dashed lines – 80% percentile, f) ZSPs in the reference image (see the caption of Figure 6.1 for details).

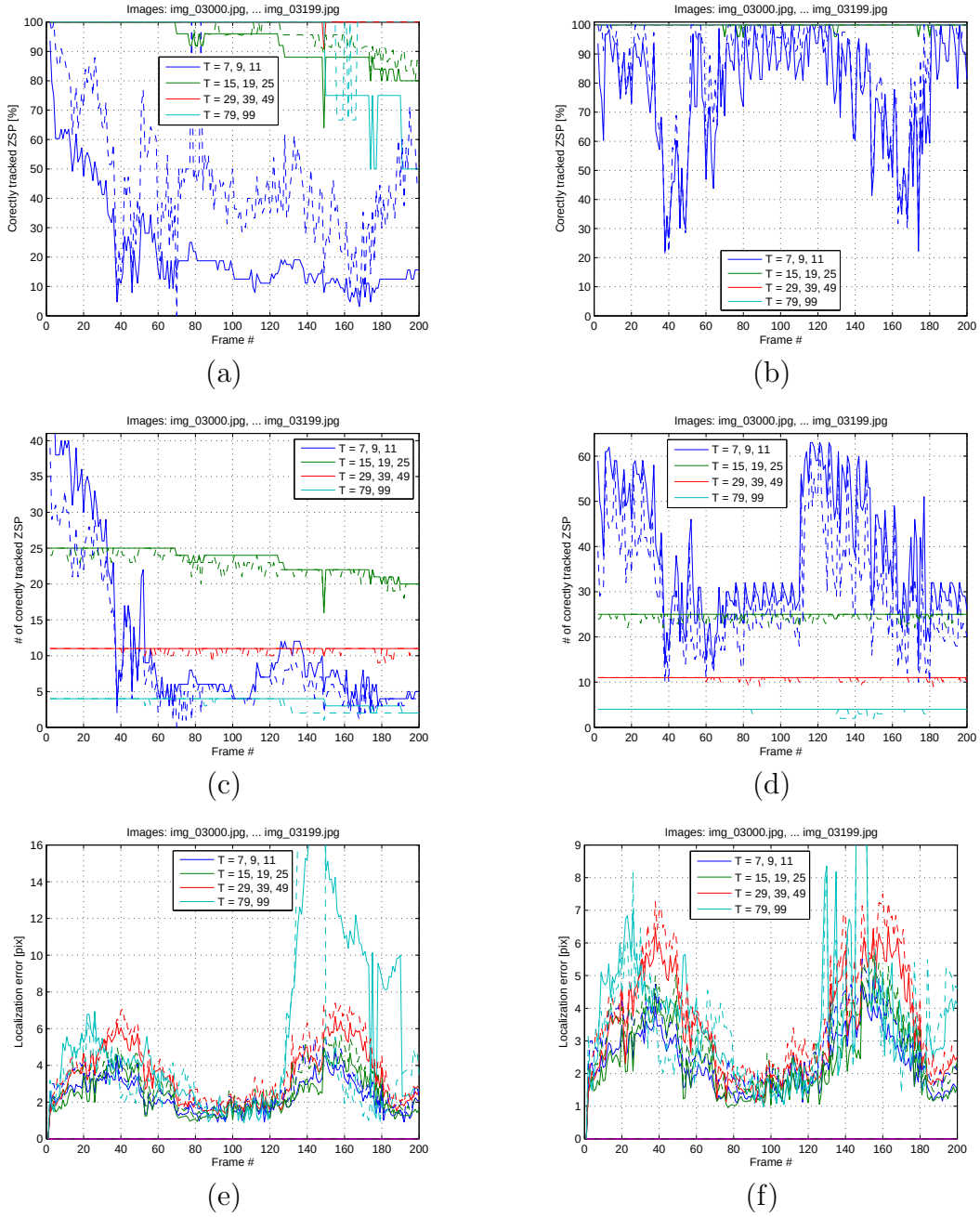


Figure 6.7: The 'Mouse pad' sequence. Tracking with the model of the local coherence improved by a), c), e) the re-tracking of outliers and b), d), f) the geometric model of the target (the homography estimated by RANSAC). The meaning of the lines is the same as in Figure 6.6.

algorithm. This system is connected with RANSAC estimating homography (for refresh and renew) and with independent scale estimator (Algorithm 13) for period adjustment.

The hypothesis of global movement were rotation $\pm 10^o$ (2 hypothesis), diagonal translation (4 hypothesis) and their combinations (2x4 hypothesis) – total up to 14 additional (to the zero global movement) hypothesis, and therefore 1 to 15 runs of tracker on single frame. The length of translation δ was the shortest period of ZSPs on the top level of the pyramid. Diagonal translation means that the vectors of translation were $[\pm\delta, \pm\delta]$. The hypothesis generation was triggered by the decrease of the number of inliers at any level of pyramid under 2/3 compared to the previous frame. If a hypothesis just tested earned more than 3/4 of inliers compared to the previous frame then it was considered correct and no other hypothesis was tested. If all hypotheses were tested, the hypothesis with the highest number of inliers was chosen. Table 6.1 shows the numbers of frames where a certain number of hypotheses were tested in individual sequences. The range extension method tracked each frame 2.3 times on average in the case of 'Mouse pad' sequence. In the worst observed case, each frame was tracked 14 times.

# hypothesis	0	1-2	3-6	7-13	14
'Mouse pad'	6103 (88%)	193 (2.8%)	50 (0.7%)	24 (0.4%)	577(8.3%)
'Phone'	2116 (92%)	2 (0.1%)	33 (1.4%)	1 (0.04%)	147(6.4%)
'Towel'	2674 (83%)	0 (0%)	2 (0.06%)	1 (0.03%)	552(17%)

Table 6.1: The numbers of frames where certain number of hypothesis were tested in individual sequences.

Figure 6.8 shows the parameters of geometric distortion of the current frame relative to the reference frame of the subsequence and the error of the target localization. The sequence was divided into subsequences by new reference frames of renew operations. The renew operation was triggered when the scale changed out of range $< 0.8, 1.3 >$ or too many points were lost. The reference frames are marked by black vertical lines in the top chart of Figure 6.8. The second chart compares the scale change estimate based on tracked points which is computed from the ground truth. The scale estimate mostly matches well the real one and is never completely wrong. On the other hand homography estimates by RANSAC are often imprecise and these errors are accumulated by renew operations as can be seen in the bottom-most chart of Figure 6.8. If the imprecision occurs between renew operation it is usually corrected in the next frame (the spikes on the error plot). The thick black vertical lines marks the loss-of-lock frames. The number of loss-of-lock was 11 – big improvement compared to 92 of ILCM tracker without range extension and slightly better than 13 of NoSLLiP tracker [45] (the difference is too small to tell that presented approach is more robust).

Figure 6.9 shows the examples of loss-of-lock frames (the couple of frames – the frame just before the fail and the failed frame). In most of the failed frames, the most of inliers (identified by the tracker itself) fit their ground truth positions, however,

their configuration is probably badly conditioned for homography estimation. The similarity may be more suitable than homography in these cases even though it cannot explain the distortion of the target fully. All these loss-of-lock frames occurred when the target became very small compared to the first frame of the sequence.

Figures 6.10 and 6.11 show the results on the the 'Phone' sequence. The target is more difficult than in 'Mouse pad' sequence. The difficulty is in periodically repeated patterns (the buttons). This may be the reason for a relatively high number of loss-of-locks reported in [45]. The presented tracker is probably less sensitive to this phenomenon for two reasons: (1) ZSPs on the top level of the pyramid have the period much longer than the period of repeated patterns and hence are insensitive to them; (2) the score testing the hypothesis about the global motion easily distinguishes a badly locked position (a high number of lost correspondences if one line of phone buttons is lost). However, this sequence is difficult even for the presented tracker. The biggest ZSPs do not have good quality and disappear in some frames. The repeated patterns play their game and the tracker locks in an incorrect position. The last loss-of-locks occurred much earlier than it could be caught by the experiment evaluation criterion, however, the error was smaller than the threshold for many frames (frames 1975-2257). Here, mostly the locally coherent tracker failed and not RANSAC. This is different as compared to 'Mouse pad' sequencein which the main reason for failures was the imprecise homography.

Figures 6.12 and 6.13 show the results on the 'Towel' sequence. The situation is similar to 'Mouse pad' sequence. The main reason for failures is the imprecise homography estimated by RANSAC, probably due to bad configuration of inliers.

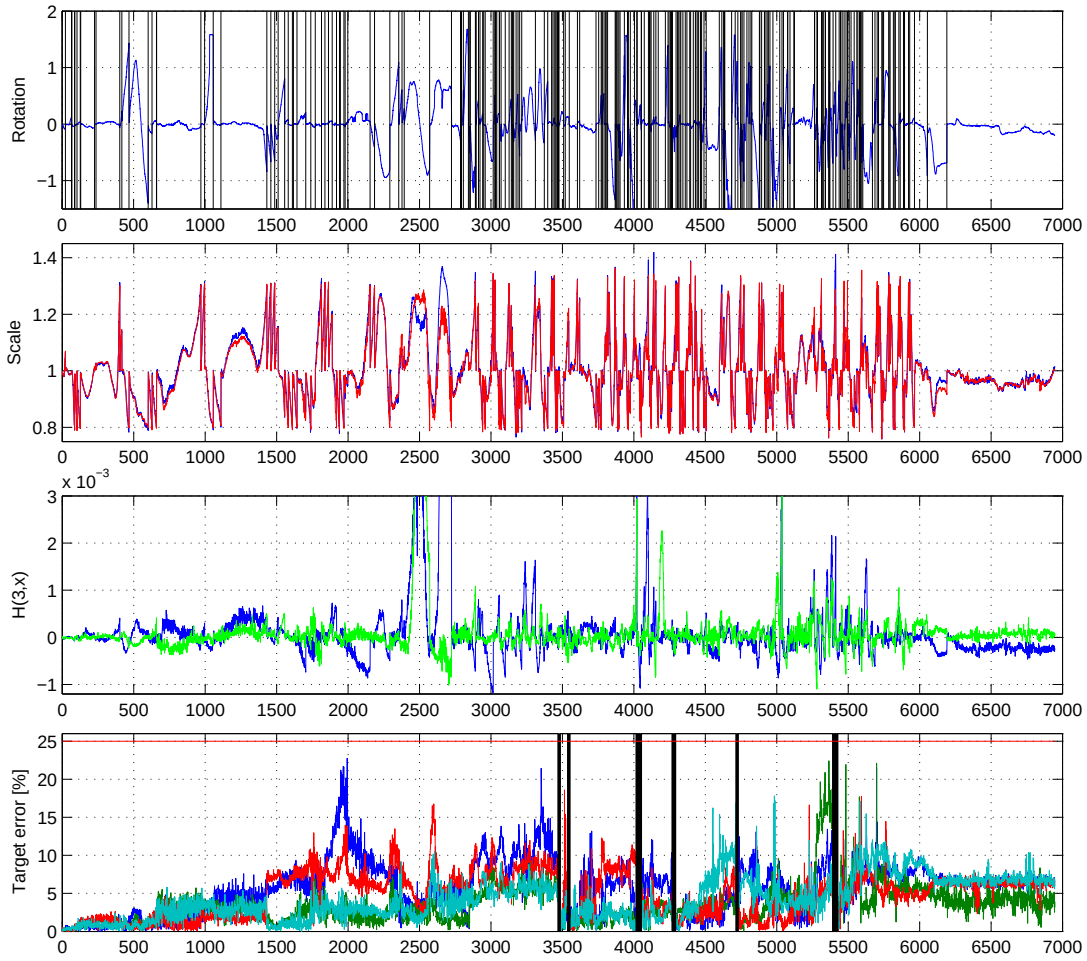


Figure 6.8: 'Mouse pad' sequence. The top three charts show the estimated parameters of geometrical change of current image relatively to the reference image (marked by black vertical lines in the most top chart). The second chart compares the scale change estimates based on tracked points (red) and computed from ground truth (blue). The most bottom chart shows the localization error of the target corners (relative to target size). Each corner drawn in different color. The thick black vertical lines marks the loss-of-lock frames. The red horizontal line shows the threshold of loss-of-lock decision.

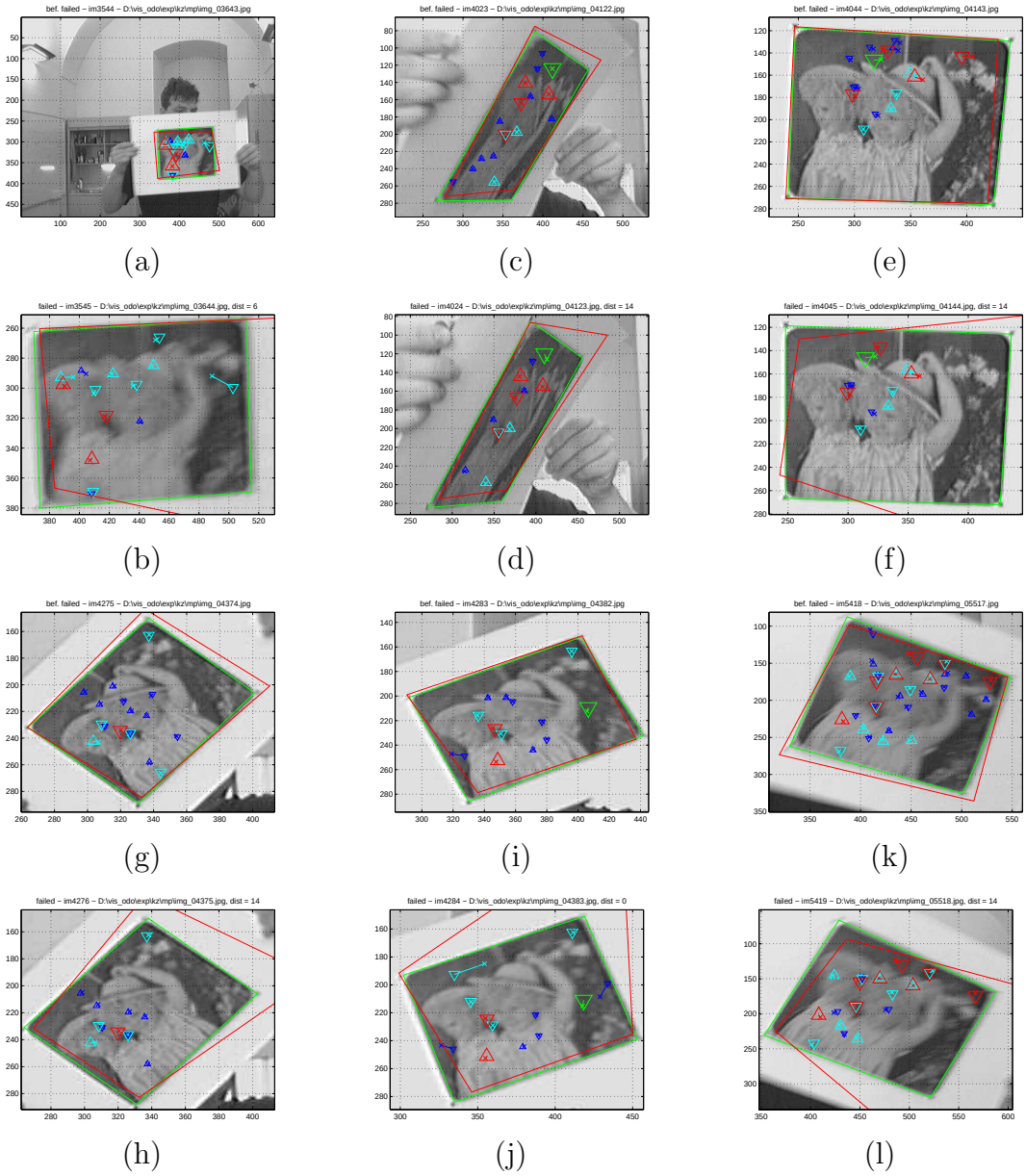


Figure 6.9: 'Mouse pad' sequence. Failures (frame couples – frame before fail and failed frame) of ILCM tracking with range extension. The green tetragons shows the ground truth for the current image, the red tetragons is the projection of the ground truth tetragons in the first image to the current image. The triangles marks the positions of inlier ZSPs (identified by tracker itself) and the crosses marks their ground truth positions.

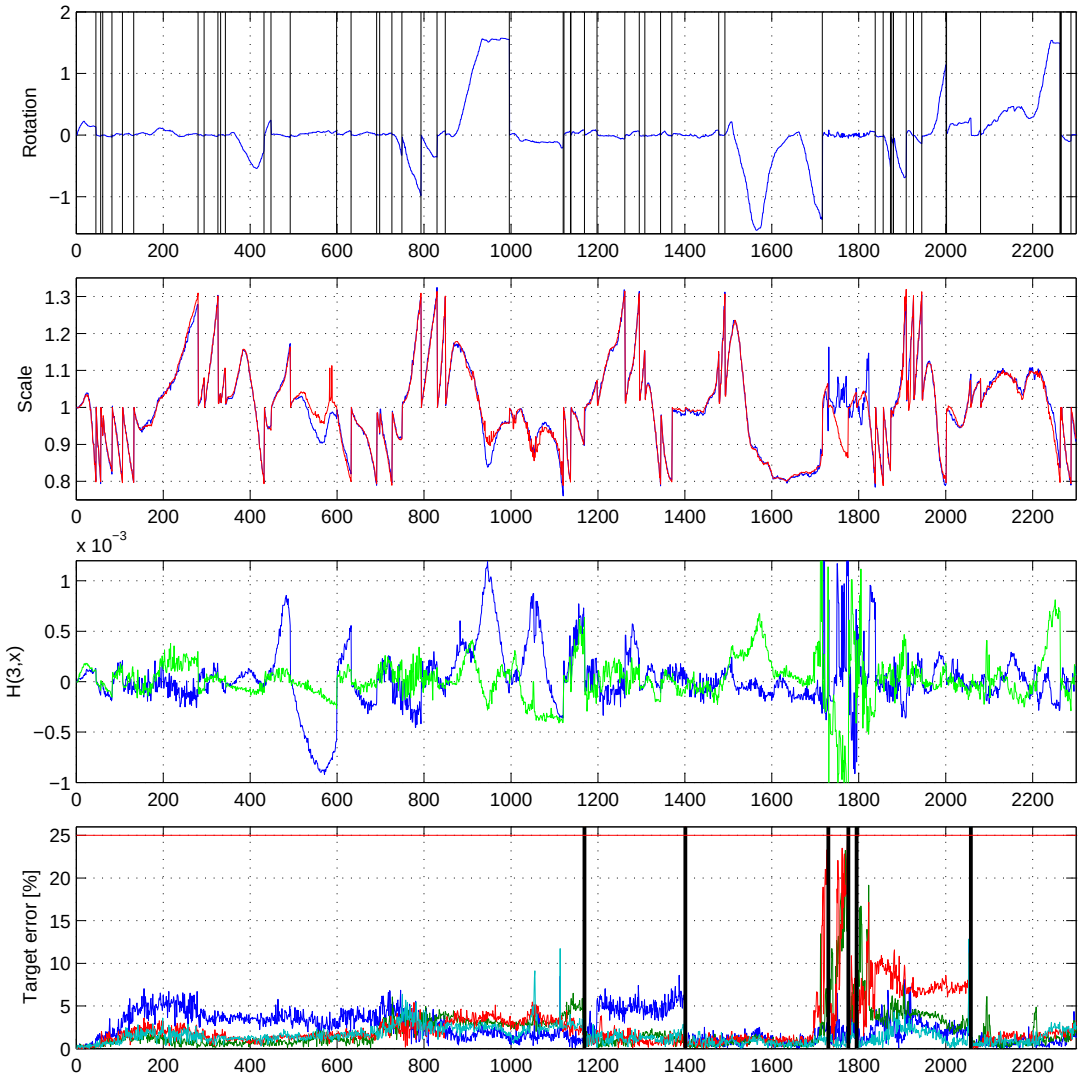


Figure 6.10: 'Phone' sequence. The top three charts show the estimated parameters of geometrical change of current image relatively to the reference image (marked by black vertical lines in the most top chart). The second chart compares the scale change estimates based on tracked points (red) and computed from ground truth (blue). The most bottom chart shows the localization error of the target corners (relative to target size). Each corner drawn in different color. The thick black vertical lines marks the loss-of-lock frames. The red horizontal line shows the treshold of loss-of-lock decision.

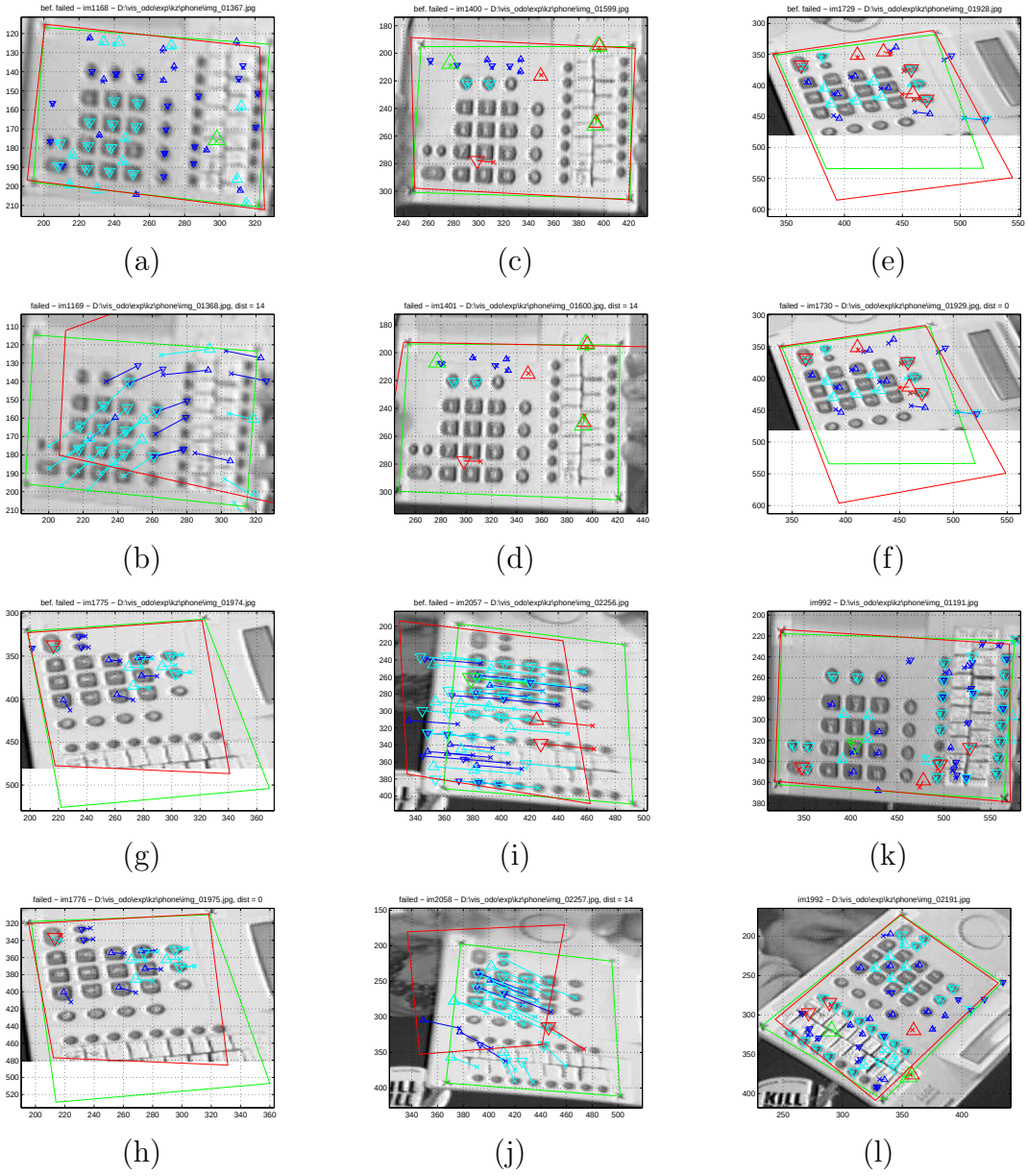


Figure 6.11: 'Phone' sequence. Failures of ILCM tracking with range extension. The green tetragons shows the ground truth for the current image, the red tetragons is the projection of the ground truth tetragons in the first image to the current image. The triangles marks the positions of inlier ZSPs (identified by tracker itself) and the crosses marks their ground truth positions. k), l) examples of correctly tracked frames (just to fill available space)

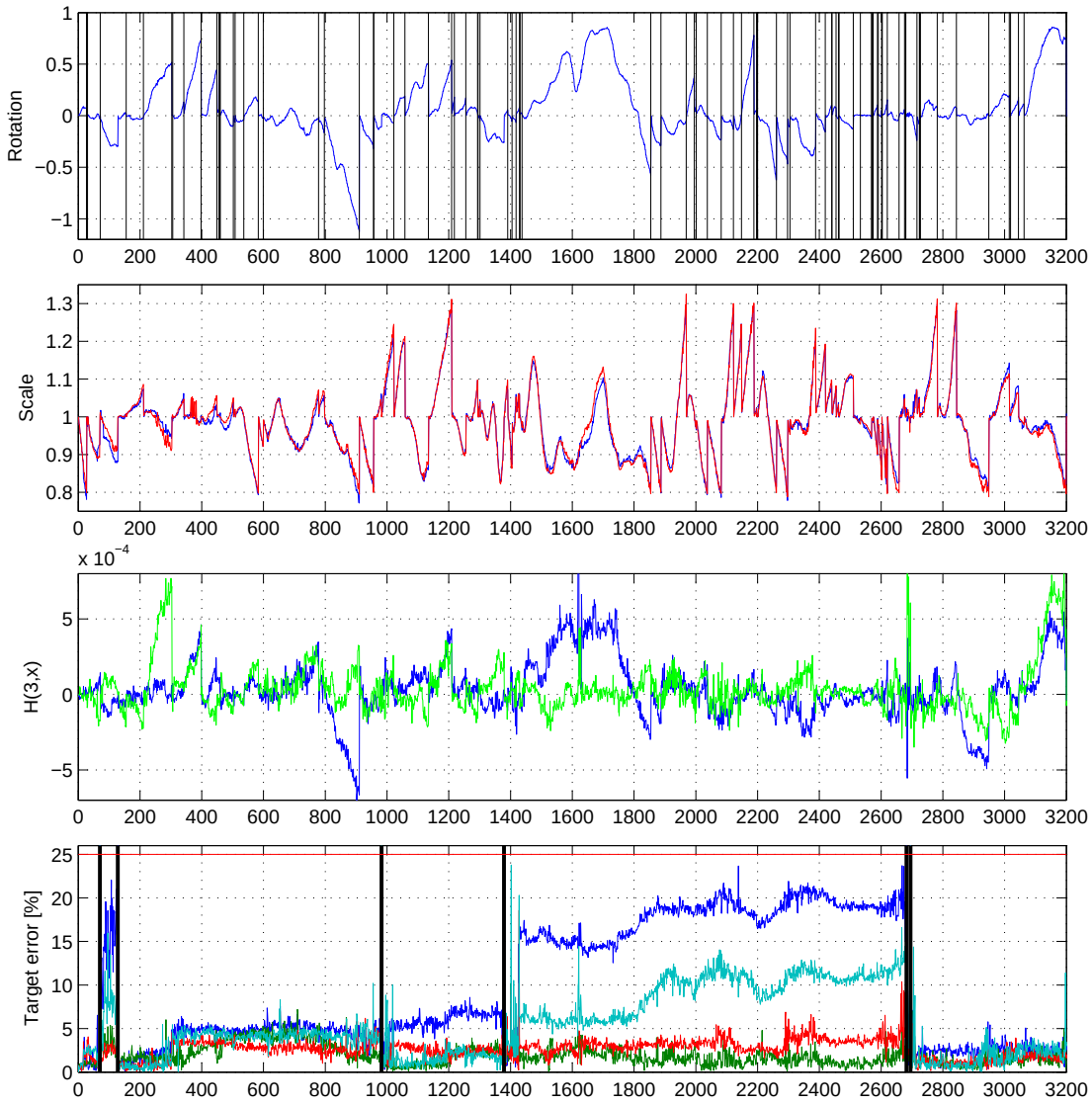


Figure 6.12: 'Towel' sequence. The top three charts show the estimated parameters of geometrical change of current image relatively to the reference image (marked by black vertical lines in the most top chart). The second chart compares the scale change estimates based on tracked points (red) and computed from ground truth (blue). The most bottom chart shows the localization error of the target corners (relative to target size). Each corner drawn in different color. The thick black vertical lines marks the loss-of-lock frames. The red horizontal line shows the treshold of loss-of-lock decision.

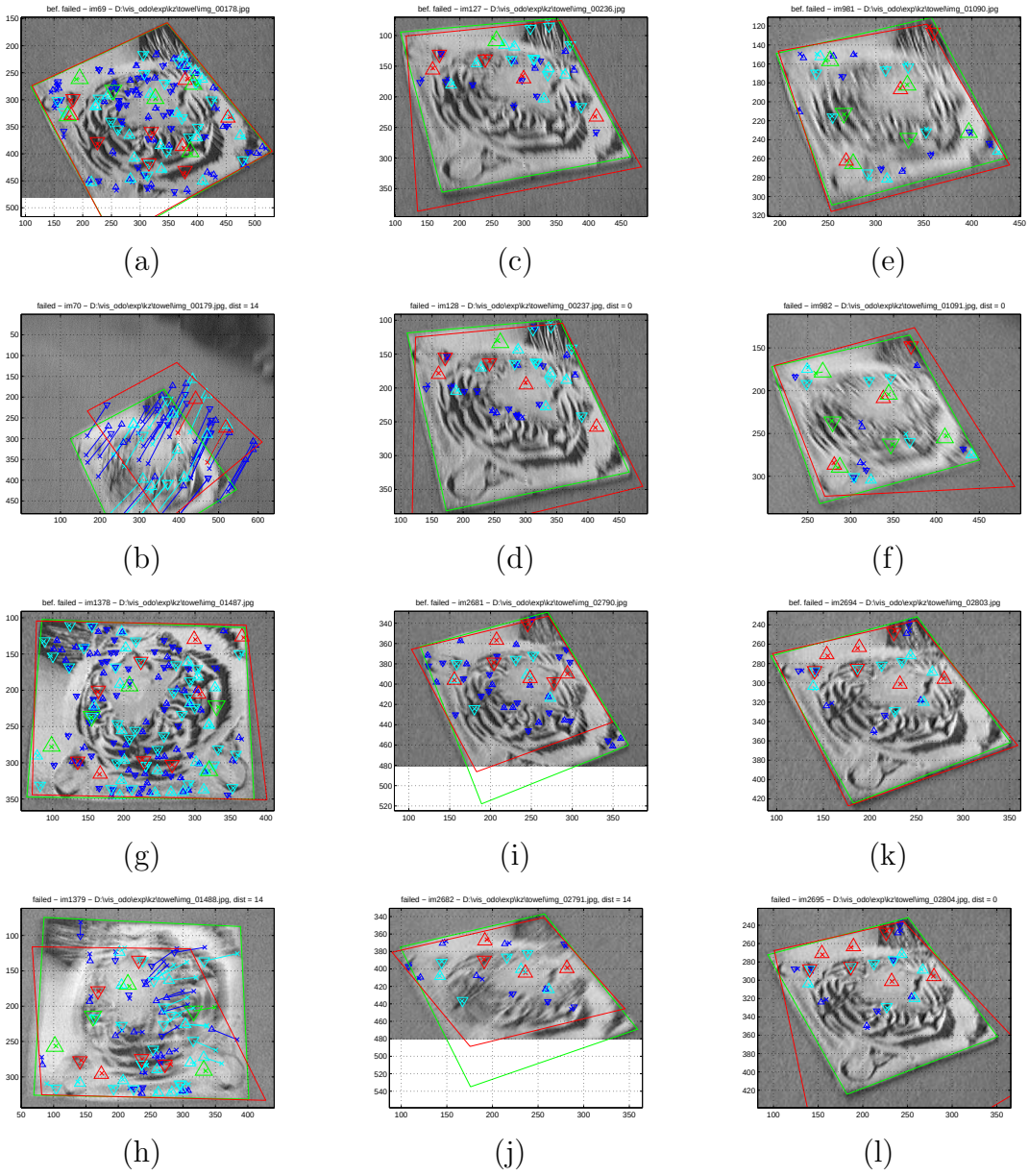


Figure 6.13: 'Towel' sequence. Failures (frame couples – frame before fail and failed frame) of ILCM tracking with range extension. The green tetragons shows the ground truth for the current image, the red tetragons is the projection of the ground truth tetragons in the first image to the current image. The triangles marks the positions of inlier ZSPs (identified by tracker itself) and the crosses marks their ground truth positions.

7 Conclusions

7.1 Thesis contributions

The main contributions (sorted by their importance) of the work presented are:

1. The *ultra fast low level tracker* was proposed and described in Chapter 5. The 'low level' means that the tracker establishes point-to-point correspondences between two consecutive frames of the videosequence. The local patches surrounding the points to be tracked are roughly rotationally symmetric blobs which are brighter or darker than their background. The tracking time of a single point is in the order of microseconds.

2. The *criteria to select good blobs* to track and the fast method to search them were explicated in Section 5.4. We demonstrated the long endurance of these points in tracking on real sequences in Section 5.5.3, their repeatability and precision on synthetic data in Section 5.5.2.

3. The *Zero Shift Point* (ZSP) pyramid was proposed, i.e. the pyramid grouping the points depending on their scale for the multiscale tracking on a higher level of abstraction (tracking not individual points but a more complex pattern connecting the points). A simple integral image idea is used while working with the ZSP pyramid and replaces the traditional image pyramid.

4. The *locally coherent model* was designed which imposes only a week assumption on the target and its movement, i.e. that near points move similarly. The local motion is approximated by a pure translation. It was demonstrated that this simple model is able to cope even with relatively big scale changes and perspective deformations on real data in Section 6.3.1. The tracker based on a locally coherent model is able to identify and correct outliers of the low level tracker (which are probably incorrect matches). The tracker self-evaluates the quality of tracking based on the number of outliers.

5. *Examples were provided how to use the tracker* with the locally coherent model in the application using point-to-point correspondences established by the tracker. The testing application was the RANSAC procedure estimating homography between the current and reference image. It was showed how to use the information gathered by application as a feedback to the tracker to significantly improve the performance of the whole system.

   The *method extending the range of movements* that the tracker can handle was suggested. The method is based on the generation and test of global motion hypotheses and on the ability of the tracker to self-evaluate the quality of tracking. This method in not applicable in general for combinatorial complexity reasons as it is dependent on the number of hypotheses. However, the method is useful in the case of the tracker with the locally coherent model. In

such a case the tracker is able to self-detect a failure and ask for a new hypothesis. The failures occur rarely and 14 basic hypotheses significantly reduced the number of loss-of-locks. It costs 2.3 times more in average if compared to the situation when this mechanism is not used.

The method *solves the problem of a small fast target* which hurts all trackers utilizing the attraction basin as KLT or linear predictors. The trackers which detect and match have this method built in naturally in some limited extend, in the limited neighbourhood around the expected position, in which all key points are considered as tentative correspondences.

Less important contributions:

1. The *Stable Wave Detector* (SWD) was designed for a wide baseline or the detect-and-match approach in Chapter 4. The experiments showed the sub-pixel precision in the localization of 'good' blobs, e.g. for the camera pose estimation or for designing fiducial markers for ground truth computations.

   The circle as the fiducial marker seems better than crosses. A colleague from our department Karel Zimmermann used crosses as fiducial markers while calculating ground truth on a 'Phone' and 'Mouse pad' sequences. It was observed that crosses suffer by decreasing size and blur while circles of the same size did not manifest this unwanted behaviour. The crosses after decreasing resolution and blur look more like circles. The corner detector seeking ground truth locations likely does not find what it is designed for. The outcome is an error of several pixels in the calculated ground truth. The experiments with SWD have shown that this undesirable effect is much less pronounced with circles.

2. *Peak detector* was proposed, i.e. SWD in 1D. Both SWD blob detector and the ultra fast low level tracker are based on its general idea of the localization based on the phase of the first harmonic wave. We did not do any comparison to other peak detectors, however due to its precision, robustness to noise and a very low computation complexity it may be ideal for practical application. The peak detector was already used in a hand held explosives analyzer produced by RS Dynamics. Here the low power consumption was important issue. The details cannot be mentioned here because of company confidentiality.

7.2 Ideas for the future work

The Stable Wave-based blob detector and tracker was described extensively in this thesis. Two previous conference publications [11, 12] on the topic were only sketchy. The thesis makes detailed thoughts available to the research community.

The *implementation* of the Stable Wave related algorithms exists in C language and in MATLAB under the *open software licence for non-commercial purposes*. It is available `http://cmp.felk.cvut.cz/cmp/software/StableWave/`. There is a hope

that the community will use the Stable Wave and will provide us with user experience comments. The intention is to improve the Stable Wave software gradually.

There is also an intention to use the tracker with the newly developed stabilized camera platform within the project SVICE conducted at the Department of Cybernetics, Czech Technical University Prague. The platform is intended to be mounted on airplanes or helicopters. A quick inner feedback loop from the gyroscope stabilizes the platform to cope with vibrations and movement of the aerial vehicle. The slower feedback loop tracks the target using a camera in both the visible and far infrared spectrum. Stable Wave tracker will be tested on the stabilized platform data and, if test are successful, used in the real device.

The experience with the Stable Wave blob detector and tracker revealed one prospective direction for improvement. When failures in attempts to extend the range of the detector were analyzed, the incorrectly estimated homography (perspective projection) was the main culprit. The configuration of correct points gets wrong some times. Some outliers are missed as well.

This is an instance of a general problem in the RANSAC-like estimation. Having a model of inliers does not guarantee that it is correct. This problem is not solvable in general. However, there could be ways how to constrain effects of this problem. The model has to be flexible enough. It is not difficult to handle two neighbouring frames. The challenge is to have the model which survives the whole sequence.

The hope is in combining a good model selection with constraining admissible homographes. Neither similarity nor affine transformation suffice as a model for longer sequences. The co-play of RANSAC and homography causes that the uncertainty is explained by a meaningless scale, rotation, translation. These partial transformations contributing to homography could be constraint. The similarity (i.e., rotation, translation and scale) can be estimated and interpreted as admissible or not admissible. There are ideas how to extend the method in this direction.

Bibliography

[1] S. Avidan. Ensemble tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(2):261–271, February 2007.

[2] J.-Y. Bouguet. Pyramidal implementation of the Lucas Kanade feature tracker, description of the algorithm, 2000. Intel Corporation, Microprocessor Research Lab, http://robots.stanford.edu/cs223b04/algo_tracking.pdf.

[3] R. O. Castle, G. Klein, and D. W. Murray. Video-rate localization in multiple maps for wearable augmented reality. In *Proc 12th IEEE International Symposium on Wearable Computers*, pages 15–22, 2008.

[4] O. Chum and J. Matas. Optimal randomized RANSAC. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30:1472–1482, 2008.

[5] O. Chum, J. Matas, and J. Kittler. Locally optimized ransac. In *Pattern Recognition, 25th DAGM Symposium*, pages 236–243, 2003.

[6] D. Comaniciu, V. Ramesh, and P. Meer. Real-time tracking of non-rigid objects using mean shift. In *Proceedings of the IEEE conference Computer Vision and Pattern Recognition*, pages 142–149. IEEE Computer Society, Los Alamitos, USA, 2000.

[7] T. F. Cootes, G. J. Edwards, and C. J. Taylor. Active appearance models. *IEEE Trans. Pattern Anal. Mach. Intell.*, pages 681–685, 2001.

[8] R. Deriche and G. Giraudon. A computational approach for corner and vertex detection. *IJCV*, 10(2):101–124, 1993.

[9] T. Dickscheid, F. Schindler, and W. Förstner. Coding images with local features. *International Journal of Computer Vision*, pages 1–21, 2010. published on-line, April 2010.

[10] M. Donoser, H. Bischof, and M. Wiltsche. Color blob segmentation by MSER analysis. In *Proceedings of the International Conference on Image Processing*, pages 757–760, Atlanta, USA, 2006. IEEE, Los Alamitos, USA.

[11] J. Dupač and V. Hlaváč. Stable wave detector of blobs in images. In Katrin Franke, Klaus-Robert Müller, Bertram Nickolay, and Ralf Schäfer, editors, *Proceedings of the 28th DAGM (German Pattern Recognition Society) Symposium*, volume 4174 of *Lecture Notes in Computer Science*, pages 760–769, Heidelberg, Germany, September 2006. DAGM, Springer.

[12] J. Dupač and J. Matas. Ultra-fast tracking based on zero-shift points. In *Proceedings of the 36th International Conference on Acoustics, Speech and Signal Processing*. IEEE Signal Processing Society, May 2011. Venue, Prague, Czech Republic, accepted.

[13] F. Fraundorfer and H. Bischof. Evaluation of local detectors on non-planar scenes. In *Proc. of 28th Workshop of the Austrian Association for Pattern Recognition (AGM/AAPR)*, pages 125–132, Osterreichische Computer Gesellschaft 3-85403-179-3, Hagenberg, 2004.

[14] C. Harris and M. Stephens. A combined corner and edge detector. In *Proc. of 4th Alvey Vision Conference*, pages 147–151, March 1988.

[15] F. Jurie and M. Dhome. Real time robust template matching. In *Proceedings of the British Machine Vision Conference (BMVC2002)*, 2002.

[16] T. Kadir, A. Zisserman, and M.Brady. An affine invariant salient region detector. In *Proceedings of the 8th European Conference on Computer Vision*, volume 1 of *LNCS 3021*, pages 228–241, Prague, May 2004. Springer-Verlag.

[17] Z. Kalal, J. Matas, and K. Mikolajczyk. P-N Learning: Bootstrapping Binary Classifiers by Structural Constraints. *CVPR*, 2010.

[18] Z. Kalal, K. Mikolajczyk, and J. Matas. Forward-Backward Error: Automatic Detection of Tracking Failures. *International Conference on Pattern Recognition*, 2010.

[19] G. Klein and D. Murray. Parallel tracking and mapping for small AR workspaces. In *Proc. Sixth IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR'07)*, 2007.

[20] Y. Li, H. Ai, T. Yamashita, S. Lao, , and M. Kawade. Tracking in low frame rate video: A cascade particle filter with discriminative observers of different lifespans. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*. IEEE Computer Society, Los Alamitos, USA, 2007.

[21] S. Lieberknecht, S. Benhimane, P. Meier, and N. Navab. A dataset and evaluation methodology for template-based tracking algorithms. In *8th IEEE International Symposium on Mixed and Augmented Reality*, pages 145–151, Los Alamitos, CA, USA, 2009. IEEE Computer Society.

[22] M. Loog and F. Lauze. The improbability of Harris interest points. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(06):1141 –1147, June 2010.

[23] D. G. Lowe. Object recognition from local scale-invariant features. In *Proc. of IEEE International Conference on Computer Vision (ICCV1999)*, pages 1150–1157. IEEE Computer Society, 1999.

[24] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[25] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the 7th International Joint Conference on Artificial Intelligence (IJCAI '81)*, pages 674–679, 1981.

[26] J. Matas, O. Chum, Urban M., and T. Pajdla. Robust wide baseline stereo from maximally stable extremal regions. In Paul L. Rosin and David Marshall, editors, *Proc. of the British Machine Vision Conference*, volume 1, pages 384–393, London, UK, September 2002. BMVA.

[27] J. Maver. Self-similarity and points of interest. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(07):1211 –1226, July 2010.

[28] K. Mikolajczyk and C. Schmid. Scale & affine invariant point detector. *International Journal of Computer Vision*, 60(1):63–86, 2004.

[29] K Mikolajczyk, T Tuytelaars, C Schmid, A Zisserman, J Matas, F Schaffalitzky, T Kadir, and L van Gool. A comparison of affine region detectors. *International Journal of Computer Vision*, 65(7):43 – 72, November 2005.

[30] H. P. Moravec. Towards automatic visual obstacle avoidance. In *Proc. of The 5th International Joint Conference on Artificial Intelligence, MIT, Cambridge, Massachusetts*, page 584. IJCAI, August 1977.

[31] D. Nistér, O. Naroditsky, and J. R. Bergen. Visual odometry. In *Proc. of IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'04)*, pages 652–659. IEEE Computer Society, 2004.

[32] M. Ozuysal, P. Fua, and V. Lepetit. Fast keypoint recognition in ten lines of code. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*. IEEE Computer Society, Los Alamitos, USA, 2007.

[33] T. Pajdla and V. Hlaváč. Zero phase representation of panoramic images for image based localization. In Franc Solina and Aleš Leonardis, editors, *Proc. of 8-th International Conference on Computer Analysis of Images and Patterns*, number 1689 in Lecture Notes in Computer Scinence, pages 550–557, Tržaška 25, Ljubljana, Slovenia, September 1999. Springer Verlag.

[34] M. Pejčoch. Tracking in video sequence using stable wave method. Master thesis, Czech Technical University in Prague, Faculty of Electrical Engineering, Prague, Czech Republic, January 2009.

[35] E. Rosten and T. Drummond. Fusing points and lines for high performance tracking. In *IEEE International Conference on Computer Vision*, volume 2, pages 1508–1511, October 2005.

[36] E. Rosten and T. Drummond. Machine learning for high-speed corner detection. In *European Conference on Computer Vision*, volume 1, pages 430–443, May 2006.

[37] F. Schaffalitzky and A. Zisserman. Multi-view matching for unordered image sets, or 'how do i organize my holiday snaps?'. In *Proceedings of the 7th European Conference on Computer Vision*, volume 1 of *LNCS 2350*, pages 414–431. Springer-Verlag, 2002.

[38] C. Schmid, R. Mohr, and C. Bauckhage. Evaluation of interest point detectors. *International Journal of Computer Vision*, 37(2):151–172, 2000.

[39] J. Shi and C. Tomasi. Good features to track. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 593–600. IEEE Computer Society, Los Alamitos, USA, 1994.

[40] C. Tomasi and T. Kanade. Detection and tracking of point features. Technical report CMU-CS-91-132, Carnegie Melon University, April 1991.

[41] T. Tuytelaars and L. Van Gool. Content-based image retrieval based on local affinely invariant regions. In *International Conference on Visual Information Systems*, pages 493–500, 1999.

[42] T. Tuytelaars and L. Van Gool. Matching widely separated views based on affine invariant regions. *International Journal on Computer Vision*, 59(1):61–85, 2004.

[43] T. Tuytelaars and K. Mikolajczyk. Local invariant feature detectors: A survey. *Foundations and Trends in Computer Graphics and Vision*, 3(3):177–280, 2008.

[44] www.willowgarage.com. OpenCV (open source computer vision) library.

[45] K. Zimmermann, J. Matas, and T. Svoboda. Tracking by an optimal sequence of linear predictors. *IEEE Trans. Pattern Anal. Mach. Intell.*, pages 677–692, 2009.

[46] O. Zuniga and R.M. Haralick. Corner detection using the facet model. In *Computer Vision and Pattern Recognition*, pages 30–37, Los Alamitos, CA, 1983. IEEE.