Unsupervised Learning of Correspondence Relations in Image Streams

Dissertation
zur Erlangung des Doktorgrades
der Naturwissenschaften

vorgelegt beim Fachbereich Informatik und Mathematik der
Johann Wolfgang Goethe – Universität
in Frankfurt am Main

von

Christian Conrad

aus Künzelsau

Frankfurt 2017
(D30)

vom Fachbereich Informatik und Mathematik
der Johann Wolfgang Goethe – Universität
als Dissertation angenommen.

Dekan:	Prof. Dr. rer. nat. Uwe Brinkschulte
Gutachter:	Prof. Dr.-Ing. Rudolf Mester Prof. Roland Memisevic, Ph.D.
Datum der Disputation:	18. Januar 2017

für Andrea, Paula, Julian und Maximilian

© 2017 CHRISTIAN CONRAD ALL RIGHTS RESERVED

Zusammenfassung

Die vorliegende Arbeit stellt Ansätze zum automatisierten und nicht überwachten Lernen von Korrespondenzbeziehungen zwischen Paaren von langen Bildströmen vor. Die Grundlage bildet dabei das sogenannte Korrespondenzproblem, d.h. die Identifizierung von 2-D Bildpunkten, die die Projektion desselben 3-D Weltpunktes darstellen. Das Korrespondenzproblem kann sowohl *lokal* als auch *global* betrachtet werden. Im lokalen Fall ist das Ziel, einzelne Paare von korrespondierenden Bildpunkten (Pixel) zu bestimmen. Im globalen Fall wird eine globale Transformation gesucht, die das Korrespondenzproblem für alle Pixel gleichzeitig löst.

Das Korrespondenzproblem ist das grundlegende Problem vieler Aufgaben im Bereich der Low-Level Vision, insbesondere in der Stereo- und Bewegungsanalyse. Wie in biologischen Systemen, ist auch in technischen Systemen die Fähigkeit zur Tiefen- und Bewegungswahrnehmung essentiell. Sie erlaubt u.a. Rückschlüsse auf die Distanz zu Objekten und deren Größe zu ziehen, erlaubt die Bewegung von Objekten zu erkennen und diese zu verfolgen.

In den klassischen technischen Ansätzen zur Lösung des Korrespondenzproblems werden Verfahren eingesetzt, die die gesuchten Korrespondenzen auf Basis einer Rechenvorschrift bestimmen. Sie erklären aber nicht, wie Korrespondenzbeziehungen automatisch und nicht überwacht gelernt werden können. Das automatisierte Lernen von Korrespondenzen ist nicht nur aus theoretischer Sicht interessant. In größer und komplexer werdenden Bildverarbeitungssystemen wird es zunehmend wichtig, dass sich einzelne Module eigenständig kalibrieren können und lernen, welche Eingabedaten als typisch und atypisch zu betrachten sind. Beispielsweise sind Kameranetzwerke besonders aufwändig zu kalibrieren und durch Menschen zu überwachen. Deshalb werden Lernansätze angestrebt, die z.B. automatisch erkennen welche Kameras ein, in der 3-D Welt, überlappendes Sichtfeld zeigen oder atypische Ereignisse erkennen, die von einem menschlichen Operator näher untersucht werden sollten. Eine weitere wichtige Eigenschaft ist, dass die Systeme erkennen und signalisieren sollen, wann ihre Ausgabedaten auf Grund hoher Unsicherheiten nicht vertrauenswürdig sind.

Lokale Korrespondenzbeziehungen Im ersten Teil der Arbeit entwickle ich den *Temporal Coincidence Analysis* (TCA) Algorithmus, der es erlaubt, lokale Korrespondenzbeziehungen in Paaren von langen Bildströmen zu lernen. Der klassische Ansatz um Pixel Korrespondenzen in Paaren von Bildern zu bestimmen, beruht auf der sogenannten *spatial feature matching pipeline*. Dies ist ein dreistufiger Prozess, in dem zunächst

markante Stellen (*Keypoints*) in beiden Bildern bestimmt werden. Dies geschieht auf Basis sogenannter Keypoint Detektoren. Hierbei ist das Ziel Bildstellen zu identifizieren, die sich mit hoher Wahrscheinlichkeit im zweiten Bild wiederfinden lassen. Im zweiten Schritt werden die Keypoints auf Basis von Deskriptoren beschrieben, die aus einer kleinen räumlichen Nachbarschaft um den Keypoint herum extrahiert werden. Im dritten Schritt werden dann Pixel Korrespondenzen bestimmt, indem die Deskriptoren aus beiden Bildern verglichen (gematched) werden und zueinander ähnliche Deskriptoren eine Korrespondenz definieren. In der Literatur wurde eine Vielzahl von unterschiedlichen Keypoint Detektoren und Deskriptoren vorgestellt. Heute werden diese in einer konkreten Applikation als Werkzeuge verwendet, um Korrespondenzen zu berechnen. Das Vorgehen erklärt allerdings nicht, wie Korrespondenzen autonom gelernt werden können. Zu beachten ist, dass die Pipeline immer auf einzelne Paare von Bildern angewendet wird, auch wenn ein Paar von Bildströmen vorliegt.

Im Gegensatz zu den klassischen rein räumlichen Verfahren, betrachte ich in einem Paar von Bildströmen nur die *zeitliche* Information einzelner Pixel. Wie beim räumlichen Ansatz, definiere ich Keypoints bzw. *Events* im Zeitsignal, die dann auf Basis eines Deskriptors gematched werden. Für eine ausgewählte Bildstelle (Seed Pixel) im ersten Bildstrom werden Events zwischen aufeinander folgenden Zeitschritten berechnet. Sobald ein Event detektiert wurde, werden Events im zweiten Bildstrom bestimmt und mit dem Event am Seed Pixel verglichen. Hierbei lässt sich typischerweise noch keine Korrespondenz bestimmen, da eine Vielzahl von Events im zweiten Bildstrom mit dem am Seed Pixel kompatibel sind. D.h. wir erhalten lediglich eine Menge von Korrespondenzkandidaten. Die Idee ist nun, diese Korrespondenzkandidaten über viele Zeitschritte zu sammeln. Unter der Hypothese, dass die Events am Seed Pixel und dessen wahrer Korrespondenz fast immer gematched werden und dass alle anderen Matches rein zufällig sind, lässt sich die gesuchte Korrespondenz nach genügend langem Lernen identifizieren. Das TCA Verfahren beruht auf dem eben skizzierten Vorgehen.

Dabei ist TCA keine Heuristik: Ich zeige wie TCA auf Basis eines statistischen Modells hergeleitet werden kann. In diesem Modell entspricht die Detektion und das Matchen von zeitlichen Events einem zeitlichen Update Schema für eine Posteriori Verteilung. Diese Verteilung nenne ich *Korrespondenzverteilung*. Die Korrespondenzverteilung beschreibt dabei die *mittlere* Korrespondenzbeziehung, die in den gegebenen Bildströmen beobachtet wurde. Eine Korrespondenzverteilung und eine klassische Pixel-zu-Pixel Korrespondenz kodieren dieselbe Information, wenn die Tiefenstruktur der betrachteten Szene näherungsweise statisch ist. In diesem Fall enthält die Korrespondenzverteilung einen Peak an den Koordinaten der wahren Korrespondenz, die über die Zeit konstant ist. Ändert sich die Tiefenstruktur in der betrachteten Szene, dann variiert die gesuchte Korrespondenz entlang der sogenannten Epipolarlinie. In diesem Fall kann die wahre Korrespondenz zu jedem Zeitpunkt verschieden sein und aus der gelernten mittleren Korrespondenz kann nicht direkt auf die aktuelle Korrespondenz geschlossen werden. In diesem Fall entspricht die gelernte Korrespondenzverteilung also einem Prior über dem Raum der möglichen Korrespondenzen.

Ich beschreibe Korrespondenzverteilungen auf Basis verschiedener Attribute. Diese umfassen beispielsweise die Statistiken der Verteilung erster und zweiter Ordnung, sowie deren Entropie. Diese Attribute erlauben es den Lernprozess zu überwachen und zu entscheiden, wann eine Korrespondenz mit hoher Wahrscheinlichkeit gelernt werden konnte. Dies ist dann der Fall, wenn die Entropie der Verteilung klein ist. Weiterhin erlauben die Attribute, die Unsicherheiten in der gelernten Korrespondenz explizit in Form einer Kovarianzmatrix zu repräsentieren und an Verfahren weiterzureichen, die auf Basis gelernter Korrespondenzen arbeiten.

Ich zeige weiterhin, dass TCA ohne weitere Modifikationen im Skalenraum angewendet werden kann. Dies erlaubt Korrespondenzverteilungen zunächst auf einer groben Auflösungsstufe zu lernen und diese dann als Prior für die nächst feinere Auflösungsstufe zu verwenden. Dieses Vorgehen erlaubt eine wesentliche Einsparung von Speicherplatz, der zur Bestimmung einer Korrespondenzverteilung benötigt wird.

Da zeitliche Events auf Basis von zeitlich aufeinander folgenden Grauwerten bestimmt werden, müssen wir Helligkeitsunterschiede zwischen den betrachteten Ansichten explizit in Betracht ziehen. Hierfür führe ich die *Grey Value Transfer Function* (GVTF) ein, die einen beobachteten Grauwert in einer Ansicht, in den zugehörigen Grauwert in einer zweiten Ansicht überführt. Ich approximiere die GVTF als Polynom niedriger Ordnung und lerne dessen Parameter auf Basis von gelernten Korrespondenzen. Hierfür bilde ich ein 2-D Histogramm (auch Komparagramm genannt) von Paaren von Grauwerten, welche an den Stellen korrespondierender Pixel extrahiert werden. Ich zeige, dass dieses Histogramm aber nicht explizit bestimmt werden muss. Stattdessen kann das Histogramm implizit über einen Satz sogenannter *sufficient statistics* repräsentiert werden.

Im Allgemeinen müssen wir davon ausgehen, dass eine gelernte Korrespondenz mit einer räumlichen Unsicherheit behaftet ist. Schon für kleine räumliche Unsicherheiten kann dies das Komparagramm stark verfälschen. Deshalb betrachte ich das lokale räumliche Signal an den Koordinaten der betrachteten Pixel und füge nur diejenigen Grauwertpaare zum Komparagramm hinzu, wenn die lokalen Signale näherungsweise homogen sind. Dies entspricht im Prinzip dem zur Keypoint Detektion entgegengesetzten Vorgehen: Es werden nur diejenigen Grauwertpaare berücksichtigt, die nicht an Keypoint Positionen liegen. Die Parameter der GVTF werden dann als Lösung eines Regressionsproblems im Sinne der kleinsten Fehlerquadrate bestimmt.

Auf Basis von Simulationen belege ich zunächst die Anwendbarkeit von TCA zum Lernen von Korrespondenzverteilungen und des GVTF Lernverfahrens. Weiterhin zeige ich in einer Reihe von Experimenten, dass TCA in der Lage ist, lokale Korrespondenzbeziehungen in realen Multi-Kamera Daten zu lernen. Die betrachteten Daten decken dabei ein weites Spektrum von Multi-Kamera Szenarien ab: Dies sind sowohl statische Setups, in denen die Kameras aus jeweils fester Position eine bewegte Szene betrachten, als auch dynamische Setups, in denen die Kameras selbst bewegt werden. Hierbei wird aber angenommen, dass die relative Orientierung zwischen den Kameras gleich bleibt. In den betrachteten Szenarien können die Kameraansichten stark gegeneinander rotiert und verschoben sein, große Skalierungsunterschiede aufweisen und deutliche Beleuchtungsunterschiede zeigen.

TCA kann nicht nur in Stereo-Szenarien eingesetzt werden. In einer Reihe von Experimenten zeige ich, dass TCA in der Lage ist, Korrespondenzverteilungen in monokularen Szenarien zu lernen, die den mittleren beobachteten Bewegungsfeldern entsprechen. Hierfür wende ich TCA auf zeitlich versetzte Bildströme an, die aus einer einzelnen Kamera stammen. Unter anderem zeige ich, dass mein Verfahren dünnbesetzte mittlere Flussfelder lernt, die man für Bewegungen wie Geradeausfahren erwarten würde. TCA kann ebenfalls verwendet werden, um globale Bewegungsfeldparameter wie die Roll-, Nick- und Gierrate zu schätzen. Hierfür sammele (*poole*) ich im aktuellen Zeitschritt alle Korrespondenzkandidaten der Seed Pixel, die näherungsweise im Unendlichen liegen.

Ich erweitere das Basismodell zu einem Mischverteilungsmodell, welches aus 3 TCA Experten und einer latenten Variablen besteht. Dabei modelliert die latente Variable die aktuelle Bewegung (links, rechts, geradeaus). In der Trainingsphase wird die latente Variable mithilfe des Phasenkorrelationsverfahrens (Kuglin und Hines 1975a) bestimmt. Der aktuelle Wert der latenten Variable bestimmt dann, welcher TCA Experte trainiert wird. Schlussendlich ergeben sich drei charakteristische mittlere Flussfelder, wie man sie für die einzelnen Bewegungsklassen erwartet.

Ich zeige weiterhin, wie auf Basis der dünnbesetzten Flussfelder, der mittlere Fluss an beliebigen Bildpositionen interpoliert werden kann. Hierfür definiere ich ein biquadratisches Modell, dessen Parameter mithilfe der gelernten Korrespondenzverteilungen geschätzt werden. Dabei berücksichtige ich explizit die gegebenen Unsicherheiten in Form von Kovarianzmatrizen.

Globale Korrespondenzbeziehungen Im zweiten Teil der Arbeit stelle ich ein Verfahren zum autonomen Lernen von globalen Transformationen zwischen Paaren von langen Bildströmen vor. Globale Transformationen zwischen Paaren von Bildern werden üblicherweise mit einem parametrisierten funktionalen Modell beschrieben. Die Parameter des Modells werden dann auf Basis von Pixel Korrespondenzen geschätzt. Wie schon zuvor beschrieben, werden hierfür typischerweise Keypoints berechnet und mittels Deskriptoren gematched.

Im Gegensatz hierzu stelle ich ein Verfahren vor, welches eine globale Transformation in nicht parametrischer Form autonom bestimmen kann. Voraussetzung hierfür ist allerdings, dass viele Bildpaare zur Verfügung stehen, die alle über dieselbe globale Transformation verknüpft sind. Mein Ansatz basiert auf der Anwendung der *Canonical Correlation Analysis* (CCA) (Hotelling 1936). Dabei extrahiert die CCA zwei Mengen von Basisvektoren, die die gesuchte Transformation implizit kodieren. Es ist besonders hervorzuheben, dass mein Verfahren nicht nur die Parameter einer bekannten Transformationsklasse lernt, sondern die Transformation an sich.

Obwohl die durch die CCA gelernte Transformation nur implizit vorliegt, kann sie auf neue, d.h. auf nicht zum Training benutzte Daten angewendet werden. Hierfür wende ich den MMSE Schätzer aus (A. Pezeshki u. a. 2006) an. Die Anwendung der Transformation entspricht einer linearen Transformation, allerdings können die Basisvektoren beliebige, insbesondere auch nicht lineare Transformationen repräsentieren.

Da die gelernte Transformation nur implizit gegeben ist, lässt sich das gemeinsame Signal, d.h. der Überlappungsbereich beider Ansichten nicht direkt ablesen. Ich zeige, dass das gemeinsame Signal jedoch auf Basis der Canonical Correlations bestimmt werden kann. Hierfür betrachte ich den Verlauf der Canonical Correlation (CC). Die CC gibt für jedes Paar von Basisvektoren an, wie hoch die (empirische) Korrelation der auf dieses Basisvektorpaar projizierten Daten ist. Die CC ist groß für Basisvektoren, die im überlappenden Bereich Energie (von 0 verschiedene Faktoren) aufwenden. Die CC bricht für diejenigen Basisvektorpaare ein, die Energie im nicht überlappenden Bereich aufwenden.

In einer Reihe von Experimenten demonstriere ich, dass die CCA in der Lage ist, verschiedene Klassen von Transformationen zu lernen. Dies sind neben Rotation, Translation, Skalierung und allgemeinen nicht linearen Transformationen, insbesondere auch allgemeine Permutationen. Dies ist möglich, da die CCA keine Annahmen über die Topologie der Daten macht. Die Experimente zeigen auch, dass die CCA bei der Prädiktion in den nicht überlappenden Bereichen extrapoliert.

In der Praxis kann die CCA über die Singulärwertzerlegung berechnet werden. Dies bedeutet gleichzeitig, dass die CCA in Abhängigkeit der Rechenresourcen typischerweise nicht auf Basis voll aufgelöster Bilder berechnet werden kann, sondern nur auf kleinen Patches. Ich zeige, dass die CCA dennoch verwendet werden kann, um Korrespondenz Priors für reale Multi-Kamera Szenarien (mit hoher Auflösung) zu bestimmen. Hierfür wende ich die CCA auf einer grob aufgelösten Version zweier Bildströme an. Für eine bestimmte Position in der ersten Ansicht bestimme ich dann ein Binärbild in grober Auflösung, in der nur eben dieser Position der Wert 1 zugeordnet wird. Dieses Bild wird dann per Prädiktion in die zweite Ansicht transformiert und auf die feine Auflösungsstufe projiziert. Dabei ergibt sich dann ein Bereich in der zweiten Ansicht, in der sich die wahre Korrespondenz mit hoher Wahrscheinlichkeit befindet (der Korrespondenz Prior).

Fazit Die in der vorliegenden Arbeit vorgestellten Verfahren zeigen, dass Korrespondenzbeziehungen automatisiert und unüberwacht gelernt werden können. Aus meiner Sicht ist die Verknüpfung der vorgestellten Ansätze mit klassischen berechnenden Verfahren von besonderem Interesse. Dabei kann zunächst Vorwissen über die zu erwartenden Korrespondenzbeziehungen gesammelt werden, um dann in jedem Zeitschritt die wahre Korrespondenz zu bestimmen.

Abstract

In this dissertation, I investigate unsupervised learning of correspondence relations in long streams of visual data. Specifically, I regard the so called *correspondence problem* which defines the task of identifying pairs of 2-D pixel positions, which are the images of the same 3-D world point. The correspondence problem is a fundamental problem to be solved in many low-level computer vision algorithms. Prominent examples include stereo and motion estimation, where pixel correspondences among two different images (taken at the same point in time with two cameras (stereo) or with the same camera at two different points in time (motion)) are sought.

In the first part of the dissertation, I will introduce the method of *Temporal Coincidence Analysis* (TCA) which is an algorithm to learn the geometric and photometric relationships in binocular camera setups in an unsupervised manner. The fundamental difference to classic spatial matching techniques is that the search for similar spatial patterns is replaced by an analysis of *temporal coincidences* of single pixels. In TCA, a correspondence is represented by means of a *correspondence distribution.* This correspondence distribution is estimated based on the repeated detection and matching of strong temporal grey value changes (=events) among the regarded views. Correspondences are never computed explicitly, only the evidence for a correspondence relation by means of matched events is collected over time.

I will show that TCA is theoretically justified by means of a statistical model for matching pairs of independent signal channels. In this model, the basic principle of detecting and matching temporal events in the grey value signal is cast as a temporal update scheme of an associated posterior distribution. The posterior distribution may also be approximated by replacing the posterior update with a simple threshold test. The correspondence distribution then turns into an accumulator array. It is very important to note that the correspondence distribution encodes the average correspondence relation and not the correspondence at a specific point in time. This is a conceptual difference to classical spatial feature matching, both with respect to the processing structure but more importantly to the results obtained: a spatial approach considers a single pair of images and outputs a pixel-to-pixel correspondence. Compared to this, I process sequences of images and learn the average correspondence relation. Only under specific conditions, which will be explained in the thesis, the results of both approaches are identical.

I define a set of correspondence distribution attributes which allow to monitor the progress of the learning scheme over time and allow to assess the uncertainty of the cor-

respondence estimate. These are important entities and allow to propagate uncertainties to higher level processes which operate on the learnt correspondences.

As TCA regards grey value differences among different cameras, we have to take photometric differences among the regarded views into account. Therefore, besides learning the geometrical relationships, I propose a method to learn the photometric relationships among the views. To this end, I define the *Grey Value Transfer Function* which maps grey values observed in the first camera to its corresponding grey value in the second camera. I will present an algorithm to estimate the GVTF based on pairs of grey values observed at corresponding pairs of pixels learnt via TCA.

I will illustrate the applicability of TCA in a series of simulations as well as in a series of experiments based on real world camera setups. TCA can handle setups in which the camera views are translated and/or rotated w.r.t. each other or show a large scale difference. The cameras may be static or moving. The only assumption that is made is that the relative orientation of the cameras is fixed and that the video streams are synchronised.

Besides learning correspondence distributions in a stereo setup, I will show that TCA can also be used to learn the average optical flow in a monocular video stream without explicitly estimating optical flow vectors. As will be seen, TCA is applied as for the stereo case without changing the learning process in any way. Experimental results validate that TCA can learn motion correspondences for various camera setups, including static and moving cameras.

I will also show that TCA can be used to infer global hidden motion variables, e.g., the yaw rate of a moving camera. To this end, I pool the set of matched events over many pixels lying approximately at infinity. The matched events will then form a cluster, encoding the true yaw rate.

I will also present several straight forward extensions to TCA, such as applying TCA in a coarse to fine approach, based on a scale space representation. Furthermore, I show how a sparse set of learnt optical flow vectors may be used to estimate the parameters of a bi-quadratic model which allows to interpolate the optical flow at arbitrary pixel locations. Based on a mixture of TCA experts, I show how the method can be used to learn multiple different dominant types of motion.

In the second part of the thesis, I present an approach to learn global image transformations in an unsupervised manner. My approach is based on a method known as *Canonical Correlation Analysis* (CCA). Given a large set of image pairs, which are related by a single fixed transformation, I show that CCA extracts pairs of basis vectors which encode the sought transformation implicitly. Note that this work is not about estimating the *parameters* of a specific type of transformation, but to learn *the* transformation itself. In contrast to a classic approach, I never compute spatial features and a transformation is not encoded by means of a parametric model but by a mapping tensor which may represent arbitrary nonlinear transformations. The learnt transformation can be applied to previously unseen data based on a MMSE estimator. While this is a linear operation, the learnt basis vectors may encode arbitrary nonlinear transformations.

As will be discussed in the thesis, a limitation of classic CCA is that only one transformation at a time may be learnt and that it is not possible to directly infer the shared signal, i.e., the parts of the signal which are visible within both views irrespective of the transformation. In order to extract the spatial footprint of the shared signal, I propose to regard the *summed energy* filters.

In a somewhat more practical application, I show that CCA can be used to generate correspondence priors in real-world binocular camera setups. To this end, I learn a global transformation between the input views on a coarse scale. Then, for a given pixel location in a first view, a region within a second view may be determined in which the true correspondence is located with high probability.

To conclude, in this dissertation I will present two learning based approaches addressing the important correspondence problem in computer vision. Both approaches are unsupervised and show that local and global correspondence relations can be *learnt* instead of being *computed*.

Contents

List of Figures **xix**

1 Introduction **1**
1.1 Motivation and Thesis Statement 1
1.2 Contributions and Outline 3
1.3 Scientific Dissemination 5

2 Background **7**
2.1 Notation 7
2.2 Geometric and Photometric Relationships in Multi-Camera Scenarios . 9
2.3 The Correspondence Problem in Low-Level Vision 12
2.3.1 Correspondence Relations and the Spatial Feature Matching Pipeline 14
2.3.2 Learning Correspondence Relations 18
2.4 Summary 18

I Learning Pixel Correspondences 19

3 Temporal Coincidence Analysis **21**
3.1 Introduction 21
3.1.1 Correspondence Distribution 27
3.1.2 Basic Temporal Coincidence Algorithm 30
3.2 Related Work 34
3.3 A Theory of Matching 37
3.3.1 Definitions and Problem Statement 38
3.3.2 Temporal Update Scheme 43
3.3.3 Posterior Entropy and Conditional Update Scheme 45
3.3.4 Instantiating the Model - Gaussian Case 47
3.3.5 Simulation and Comparison to Correlation 51
3.4 TCA Algorithm 66
3.4.1 Temporal Event Signal and Event Detection 66
3.4.2 Grey Value Transfer Function 68
3.4.3 Event Matching and Estimation of Correspondence Distributions 69
3.5 On the Analysis of Correspondence Distributions 70

3.6 Learning the Grey Value Transfer Function 74
  3.6.1 Related Work 75
  3.6.2 GVTF Model 78
  3.6.3 Observation Model 78
  3.6.4 Filling the Comparagram 79
  3.6.5 Estimating the GVTF 82
    3.6.5.1 Online Update Scheme 84
    3.6.5.2 Deriving a Functional Form 84
    3.6.5.3 Incorporating the GVTF 85
  3.6.6 Simulation 86
3.7 TCA on the Pyramid / In Scale Space 89
3.8 Summary and Conclusion 93

4 Learning Stereo Correspondences 95
4.1 Introduction 95
4.2 Evaluation 95
  4.2.1 Evaluation on Synthetic Data 95
  4.2.2 Selecting Model Parameters 97
  4.2.3 Evaluation on Real World Data 106
  4.2.4 Experiments on `GUBo1616` 106
  4.2.5 Experiments on `GUBo1606` 119
  4.2.6 Experiments on `GUCar` 126
4.3 Summary and Conclusion 134

5 Learning Motion Correspondences 135
5.1 Introduction 135
  5.1.1 Related Work 135
5.2 TCA for Motion (TCAM) 137
  5.2.1 Approach 137
  5.2.2 Parametric Flow Model 139
    5.2.2.1 Computational Complexity 140
5.3 Experiments 140
  5.3.1 Learning Average Motion Maps 141
  5.3.2 Interim Conclusion 154
  5.3.3 Learning Multiple Motions 154
5.4 Motion as a Hidden Variable 155
  5.4.1 Optical Flow Structure 155
  5.4.2 Learning Hidden Flow Field Variables 157
  5.4.3 Motion as a Latent Variable 159
5.5 Summary and Conclusion 163

II Learning Global Transformations 165

6 Learning Global Mappings 167
6.1 Introduction 167
6.1.1 Mapping as a Linear Projection 169
6.2 Related Work 172
6.3 Learning Transformations with CCA 174
6.3.1 Observation Model 174
6.3.2 Canonical Correlation Analysis 176
6.4 Application of CCA to Transformation Learning 178
6.5 Learning Global Transformations on Natural Images 179
6.5.1 CCA Basis Vectors 180
6.5.2 Behaviour of the Canonical Correlations 187
6.5.3 Inferring Shared Dimensions 192
6.5.4 Predictions 197
6.6 Correspondence Priors for Binocular Camera Setups 202
6.6.1 Method 202
6.6.2 Learning the Inter-Image Transformation 202
6.6.2.1 Prediction 203
6.7 Learning Multiple Transformations 209
6.7.1 Approach 209
6.8 Summary and Conclusion 214

7 Summary and Outlook 215
7.1 Summary 215
7.2 Outlook 217

A Video Sequences Overview 219
A.0.1 Binocular Sequences 219
A.0.2 Monocular Sequences 222

Bibliography 225

List of Figures

2.1 Spatiotemporal domain 8
2.2 Epipolar geometry 10
2.3 Stereo and motion correspondence problem 13
2.4 Dense and sparse optical flow 14
2.5 Local spatial features 16

3.1 Temporal signal of single pixels 22
3.2 Large temporal signal changes 23
3.3 Temporal signal of pixels in two views 25
3.4 Events in the temporal signal of single pixels 26
3.5 Stereo correspondence distributions 29
3.6 Lab setup experiment 32
3.7 Evolving correspondence distributions for the lab setup 33
3.8 Bunch of signals 38
3.9 Joint distributions under the Gaussian model 50
3.10 Exemplary distributions under the Gaussian model 51
3.11 Sample posterior and approximations 53
3.12 Simulation results, noise level 10% 56
3.13 Simulation results, noise level 20% 57
3.14 Simulation results, noise level 30% 58
3.15 Simulation results, noise level 40% 59
3.16 Simulation results, noise level 50% 60
3.17 Simulation results, noise level 60% 61
3.18 Simulation results, noise level 70% 62
3.19 Simulation results, noise level 80% 63
3.20 Simulation results, noise level 90% 64
3.21 Simulation results, noise level 100% 65
3.22 Statistics of temporally subsequent grey values in natural video 67
3.23 Joint histograms for sequences `GUBo1616` and `GUCar` 68
3.24 Photometric error due to spatial error 80
3.25 GVTF measuring points and initialisation 86
3.26 Learning the GVTF via the proposed method 88
3.27 Proposed vs naïve learning scheme 91

3.28 Functional GVTF fitting 92

4.1 Test setup 96
4.2 Exemplary results for sequence `Forrest` 99
4.3 Correspondence distribution's attributes 102
4.4 Average energy consumption 103
4.5 TCA on the pyramid of sequence `Forrest` 105
4.6 Exemplary results for `GUBo1616` 107
4.7 Correspondence distribution's attributes for `GUBo1616` 109
4.8 Exemplary results for `GUBo1616` 111
4.9 Correspondence distribution's attributes for `GUBo1616` 112
4.10 Learning the GVTF 115
4.11 Difference images and error histogram for `GUBo1616` 116
4.12 Difference images and error histogram for `GUBo1616` (2) 117
4.13 Learning the GVTF 118
4.14 Exemplary results for `GUBo1606` 121
4.15 Correspondence distribution's attributes for `GUBo1606` 122
4.16 Learning the GVTF 123
4.17 Difference images and error histogram for `GUBo1606` 124
4.18 Difference images and error histogram for `GUBo1606` (2) 125
4.19 Exemplary results for `GUCar` 127
4.20 Schematic description of a structure match 128
4.21 Correspondence distribution's attributes for `GUCar` 130
4.22 Correspondence distribution's attributes for `GUCar` (doubled event threshold) 131
4.23 Exemplary results for `GUCar` 133

5.1 Sparse flow for `GUCar` 142
5.2 Colorwheel 143
5.3 Dense flow and confidence maps for `GUCar` 144
5.4 Interpolated sparse flow for `GUCar` 145
5.5 Sparse flow for `GUCar` 147
5.6 Dense flow and confidence maps for `Kitti-Odo` 148
5.7 Interpolated sparse flow for `GUCar` 149
5.8 Dense flow and confidence maps for `GUOmni` 151
5.9 Dense flow and confidence maps for `Virat` 153
5.10 Learning multiple motions 155
5.11 Inferring current yaw rate via TCA 158
5.12 Motion as a latent variable 160
5.13 Learnt and interpolated sparse flow for `Kitti-Odo` (left turn) 161
5.14 Learnt and interpolated sparse flow for `Kitti-Odo` (right turn) 162

6.1 2-D spatial transformations . . . 168
6.2 Representation of transformations . . . 170
6.3 Learning hidden transformations . . . 172
6.4 Observation model . . . 175
6.5 Learning a 90 degree rotation . . . 182
6.6 Learning a 90 degree rotation (2) . . . 183
6.7 Learning affine transformations . . . 184
6.8 Learning permutations . . . 185
6.9 Learning nonlinear transformations . . . 186
6.10 Canonical correlations for translated and rotated views . . . 189
6.11 Canonical correlations for full permutations . . . 190
6.12 Canonical correlations for nonlinear transformations . . . 191
6.13 Energy distribution among dimensions for the noise free case . . . 193
6.14 Approximated first and second order derivative of the canonical correlation curve . . . 194
6.15 Approximated first and second order derivative of the canonical correlation curve . . . 195
6.16 Summed energy filters up to estimated signal rank . . . 196
6.17 Learnt rotations applied to previously unseen data . . . 198
6.18 Learnt translations and rotations applied to previously unseen data . . . 199
6.19 Learnt permutations applied to previously unseen data . . . 200
6.20 Learnt nonlinear transformations applied to previously unseen data . . . 201
6.21 Learning transformations in real world setups . . . 203
6.22 Generation of correspondence priors . . . 203
6.23 Canonical correlation curves . . . 204
6.24 Summed energy filters . . . 205
6.25 Correspondence priors for real world setups . . . 207
6.26 Correspondence maps for real world setups . . . 208
6.27 Transformation invariant basis . . . 210
6.28 Splitting transformations . . . 211
6.29 Splitting transformations (2) . . . 212
6.30 Splitting transformations (3) . . . 213

A.1 Exemplary frames for binocular sequences . . . 221
A.2 Exemplary frames for monocular sequences . . . 223
A.3 Exemplary frames . . . 223

1 Introduction

In this thesis, I investigate learning of correspondence relations in long streams of visual data. In the first part of the thesis, I regard learning of local correspondence relations, i.e., pixel correspondences, which are at the core of stereo vision and motion estimation. My primary goal is not to determine pixel-to-pixel correspondences at a specific time instant, but to learn the *distribution* of the correspondence relation by only regarding the *temporal* course of single pixels. In contrast to classic approaches, I show that stereo and motion perception may be learnt without explicitly computing stereo disparity or motion vectors.

In the second part of the thesis, I regard learning of global correspondence relations, i.e., global mappings or transformations between pairs of image streams. In contrast to classic approaches, I do not fit the parameters of a parametric model based on spatial image features. Instead, I determine sets of new basis vectors, which implicitly encode the underlying transformation. This is done without the need of computing spatial features, but requires pairs of long image streams in which the global transformation is held fixed.

1.1 Motivation and Thesis Statement

Since the advent of computer vision roughly 50 years ago (Papert 1966), we may ask whether the vision problem is solved? In order to answer this question, we first need to define the *vision problem.* According to Marr (Marr 1982, p. 31), the vision problem can be described as the process, which generates a description of the world based on images of it. This description has to be useful to the viewer (=a specific application) and should not contain irrelevant information. This is a very general description of the vision problem and depending on the description sought, we may answer the introductory question by *yes, partially and no.*

Up to now, computer vision technology is successfully applied in various fields, in which the underlying vision problem is at least satisfactorily solved. These range from consumer electronics such as handheld cameras, which are able to detect faces, to advanced driver assistance systems, which (among others) automatically keep a car on track [1], through to high tech systems like NASA's Mars Exploration Rover (MER), which made use of stereo vision for navigation (Matthies et al. 2007). In all of these

---

[1] cf. pioneering work by (Dickmanns et al. 1994), see (McCall and Trivedi 2006) for a review article

applications, rather specialised vision problems are addressed, which are solved based on a carefully designed and engineered system.

In contrast to isolated vision problems, we are interested in more general vision systems, which are able to *learn* and *adapt* to their environment with almost no supervision and are able to solve various tasks. As such, the replication of the human visual system can be considered the holy grail of computer vision. However, up to now it is largely unknown *how* the human visual system really works. Nevertheless, it is an accepted view that the ability to see *and* understand and interpret the environment is *not* a vision problem only. Instead, the output of many subsystems (vision, memory etc.) are integrated to form a hypothesis of what is actually seen. These generated hypotheses not necessarily represent the true physical world. For example, various optical illusions show to some extent that the brain sees what it wants to see, based on past experience.

As we have seen in the definition of the vision problem, an autonomous system needs to be able to extract the relevant information about the environment in which it operates in order to solve a specific task. Among others, very important sources of information or *cues* are given by depth and motion perception. For example, depth perception allows to infer distances to objects, their size etc. Motion perception allows to infer ego-motion, to track objects etc.

At the core of both depth and motion perception is the so called *correspondence problem* in which we have to identify the projection of the same 3-D world point in two different images (taken at the same point in time with two cameras (stereo) or with the same camera at two different points in time (motion)). Many different approaches to solve the correspondence problem have been proposed. Most of them are based on *computing* correspondences based on an engineered recipe, but they do not explain how correspondences can be *learnt*. In contrast to this, in this thesis I address the problem of *learning* correspondence relations from long image sequences in an autonomous and unsupervised manner. My motivation lies in the question, how processing architectures can evolve almost automatically that successfully unveil the inherent stereo or motion structure in streams of visual data.

Learning of correspondence relations is not of theoretical interest only; in order to build more general or *intelligent* vision systems, it is important that systems are able to adapt to the environment in which they operate. This includes self-calibration, i.e., learning of (probably time varying) model parameters and learning of what is *typical* and *abnormal*. Furthermore, the systems need to be able to assess their confidence or uncertainty about the learnt entities and need to know when they cannot provide reliable output (Förstner 1991). In turn, the learnt knowledge may serve as a prior to constrain a specific problem; e.g., if the typical disparity or optical flow range is known, we may restrict or guide a stereo or motion estimation scheme.

Learning and self-adapting vision systems become more and more important for industrial vision applications. As an example, large scale multi-camera surveillance networks are especially tedious to calibrate and to monitor by human personnel. Here, the goal is that a vision system learns which cameras show an overlap or learn typical and suspicious object movement and to generate alerts, which should be inspected by staff. For example, the company *Snap*[2] markets video surveillance software, which is able to automatically detect pairs of cameras which are very likely to show an overlap. Learning is done based on the method proposed in (van den Hengel et al. 2007a) (also see Sec. 2.3).

The approaches developed in this thesis can directly be used as a module in existing systems. They may serve as low level processes, which can generate priors for local and global correspondence relations. Especially the approach to learn local correspondence relations can adapt to given computational/energy constraints and explicitly models uncertainties in its estimates.

1.2 Contributions and Outline

In Chapter 2, I will introduce mathematical notation and background material, which is relevant for an understanding of the material presented in the main part of the thesis.

The main body of the thesis is organised in two major parts, in which I investigate learning of local and global correspondence relations in long streams of visual data, respectively. Throughout the thesis, I will review and introduce relevant related work on a per chapter basis.

In Chapter 3, I propose the method of *Temporal Coincidence Analysis* (TCA), which allows to learn local correspondence relations (=on the pixel level) between the views of binocular camera setups. The only assumption that is made is that the relative orientation of the cameras is fixed and that the cameras are temporally synchronised. I will begin with a rather informal introduction and motivation of TCA before I turn to a theoretical derivation. We will see that the essential difference to standard spatial matching techniques is that the search for similar spatial patterns is replaced by an analysis of *temporal coincidences* of single pixels. In this scheme, we collect evidence for a correspondence by the detection and matching of temporal events in long image sequences. This approach yields correspondence distributions, which will encode the average correspondence observed. Depending on the scene structure, the average correspondence may encode the true pixel-to-pixel correspondence at every time step or will otherwise be a prior on where the correspondence is to be expected. I propose a set of attributes of the correspondence distribution, such as its first and second order statistics and its entropy, in order to monitor the learning process and assess uncertainties in the correspondence estimate. Furthermore, I show that TCA may naturally be applied within a multi-scale framework and allows coarse to fine correspondence learning.

In TCA, temporal events are extracted from the temporal course of a pixel's grey value signal. Thus, the matching of events will depend on the photometric differences

---

[2] http://www.snapsurveillance.com/

among the regarded views. I therefore introduce the concept of the *Grey Value Transfer Function* (GVTF) which maps grey values between pairs of cameras and show that the GVTF can be learnt from a set of learnt correspondences. Both TCA and the GVTF model will be supported and validated by means of simulation results.

In Chapter 4, I apply TCA to real-world binocular (stereo) camera setups. We will see that TCA is able to capture and represent the correspondence relations found for rotated and twisted orientations of the cameras, arbitrary imaging geometry and under large tolerance for photometric differences in the image sensors. Furthermore, the cameras may be static or moving.

In Chapter 5, I apply TCA to monocular camera setups and learn correspondence relations which are induced by the optical flow. We will see that TCA is able to learn the average flow field, which a moving platform observes during forward motion. Typically, these will be sparse flow fields and I present a bi-quadratic model which allows to interpolate these. I extend the basic model and propose a latent variable model, which consists of a mixture of TCA experts. Besides forward motion, this model can also learn the average motion maps observed during left and right motion. During training, I infer the platform motion by a method known as *phase correlation* (Kuglin and Hines 1975a). Furthermore, by pooling the matched events over a subset of pixels lying at infinity, I show that TCA can also be used to infer instantaneous motion. As for the stereo case, the method is not restricted to a specific camera model.

In the second part of the thesis, starting in Chapter 6, I present an approach to learn global image transformations in an unsupervised manner. The proposed method builds on a model known as *Canonical Correlation Analysis* (CCA) (Hotelling 1936). Given a large set of image pairs, which are related by a single transformation, a transformation is learnt and represented by means of two sets of basis vectors, obtained via CCA. This is in contrast to a functional and parametric approach to model/infer global transformations, in which only the parameters of a transformation are determined. My goal is to learn the transformation itself.

While the transformation obtained via CCA is only given implicitly, I show that it can be applied to previously unseen data by means of an MMSE estimator. While this is a linear operation, the basis vectors may represent nonlinear transformations of the input data as well. I show how the rank of the signal that is shared among the views may be determined from canonical correlations and how the overlapping (=shared) dimensions among the views may be inferred. Experimental results validate the presented approach and show that various kinds of transformations can be learnt unsupervised. While I assume that the training data is related by a single and fixed transformation, I present a straightforward mixture model of CCA experts, which allows to split multiple fixed transformations.

In Chapter 7, I give a brief summary of the thesis and conclude with an outlook for extensions and future work.

In Appendix A, I present an overview of the real-world image streams used throughout the thesis for evaluation and illustration purposes.

1.3 Scientific Dissemination

The material presented in this thesis has been published or is in the process of submission. The learning schemes for local and global correspondence relations have been published in (Conrad, Guevara, et al. 2011) and (Conrad and Mester 2012, 2016). The TCA approach also is the basis for (Conrad and Mester 2013) and (Eisenbach, Conrad, et al. 2013). The learning of the GVTF has been published in (Conrad and Mester 2015). Some of the technical ideas presented in the GVTF estimation and the interpolation of sparse motion fields are inspired by our work presented in (Conrad, Mertz, et al. 2013), (Guevara et al. 2012) and (Guevara et al. 2011).

2 Background

In the following, I will introduce background material, relevant for an understanding of the material presented in subsequent chapters. I will shortly review the common approach to describe the geometric relationship between pairs of views by means of epipolar geometry. This leads to the so called *fundamental matrix*, which can be estimated from spatial *keypoints*. I will review the classic approach to establish spatial keypoints by means of the spatial feature matching pipeline. Further related work will then be reviewed on a per chapter basis. For an in depth introduction to the material presented in the following, I refer to standard textbooks such as (Forsyth and Ponce 2002; Hartley and Zisserman 2004; Szeliski 2010).

Within the main part of the thesis, I will then show that geometric and photometric relationships within general binocular camera setups can be learnt unsupervised and without making use of epipolar geometry and spatial keypoints in the classic sense.

First of all, I will next introduce the notation used throughout this thesis.

2.1 Notation

Before introducing the thesis specific notation, let us define the basic mathematical entities used throughout the thesis:

Scalars	$a, b, c, ...$
Vectors	$\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}, ...$
i-th vector element	$\boldsymbol{a}(i)$
Matrices	$\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}, ...$
i-th matrix row	$\boldsymbol{A}_{i,:}$
i-th matrix column	$\boldsymbol{A}_{:,i}$
Vector/Matrix transpose	$\boldsymbol{a}^T$, $\boldsymbol{B}^T$
Vector of zeros	$\mathbf{0}$
Vector of n zeros	$\mathbf{0}_n$
Constants	A, B, C
Scalar valued function	$f(a)$, $f(\boldsymbol{b})$, $f(\boldsymbol{C})$
Vector valued function	$\boldsymbol{f}(a)$, $\boldsymbol{f}(\boldsymbol{b})$, $\boldsymbol{f}(\boldsymbol{C})$

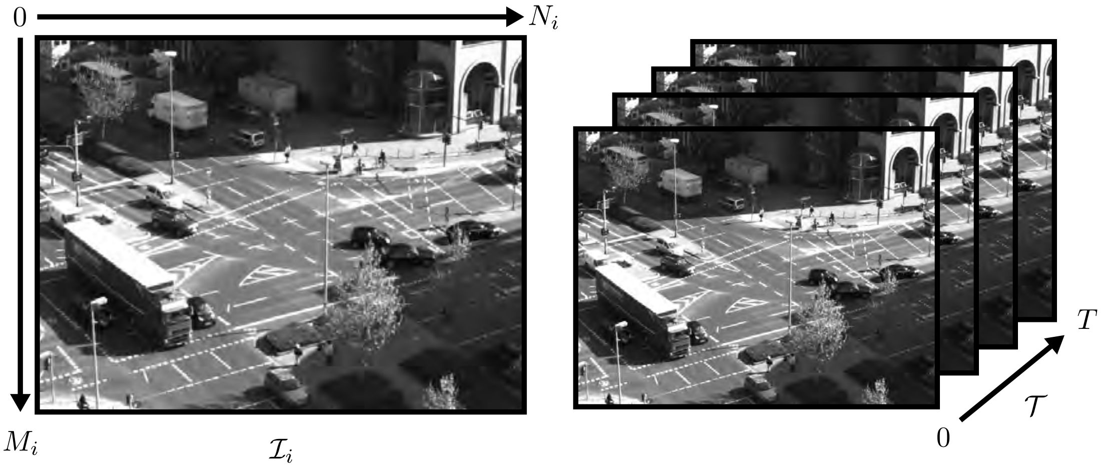


Figure 2.1: **Spatiotemporal domain:** (left) Spatial domain $\mathcal{I}$, (right) spatiotemporal domain $\mathcal{D}$.

Based on these definitions, we proceed as follows. Let $\mathcal{C}_i$ be the i-th camera, also referred to as *view*. With the i-th camera, a spatiotemporal image function s_i is associated as follows. Let $M_i, N_i, T \in \mathbb{N}$, then the spatial domain $\mathcal{I}_i$ of the image function is defined as (cf. Fig. 2.1 (left)):

$$\mathcal{I}_i := [0, M_i] \times [0, N_i]. \tag{2.1}$$

The temporal domain $\mathcal{T}$ of the image function is defined as (cf. Fig. 2.1 (right)):

$$\mathcal{T} := [0, T]. \tag{2.2}$$

Then the spatiotemporal domain $\mathcal{D}_i$ of the image function is given by:

$$\mathcal{D}_i = \mathcal{I}_i \times \mathcal{T}. \tag{2.3}$$

Let $\mathcal{G}$ be the set of observable grey values, the image function s_i is then defined as:

$$s_i : \mathcal{D}_i \to \mathcal{G}. \tag{2.4}$$

If we regard several cameras, the spatial resolution of their image functions may be different, but the temporal resolution is assumed to be the same for all cameras.

A pixel is now defined as a point in the spatial image domain with an associated grey value. Specifically, let $\boldsymbol{x}_i \in \mathcal{I}_i$ be the spatial coordinates of a pixel in camera $\mathcal{C}_i$:

$$\boldsymbol{x}_i = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}_i , \tag{2.5}$$

where its grey value at time $t \in \mathcal{T}$ is given by the image function $s_i(\boldsymbol{x_i}, t)$. I may refer to $\boldsymbol{x}_i$ also simply as *pixel* in the following. When it is clear from the context, I may also drop the index of the spatial coordinates $\boldsymbol{x}_i$ and simply write $\boldsymbol{x}$. Sometimes I refer to $\boldsymbol{x}_i$ in its *homogeneous* coordinates, in which a third coordinate $x_3 = 1$ is attached to $\boldsymbol{x}_i$.

For some camera $\mathcal{C}_i$, its *image* at time $t \in \mathcal{T}$ is given by the respective image function $s_i(\cdot, t)$. I will use the notation $I_{t,i}$ to denote the 2d array (=image) of signal values of camera $\mathcal{C}_i$ at time t, with $I_{t,i}(\boldsymbol{x}_i) = s_i(\boldsymbol{x}_i, t)$. An image *stream* is then given by a temporally ordered set of images.

Two specific camera setups are considered throughout the thesis, these are i) the binocular (camera) setup and ii) the monocular (camera) setup. The binocular setup consists of two cameras $\mathcal{C}_i$ and $\mathcal{C}_j$ and their associated image functions s_i and s_j defined on the spatiotemporal domains $\mathcal{D}_i$ and $\mathcal{D}_j$, respectively. The monocular setup is defined similarly, but for a single camera.

Based on the previous definitions, I may now introduce the concept of a spatial correspondence. Regard a binocular camera setup as visualised in Fig. 2.2 (left). A spatial location $\boldsymbol{x}_i \in \mathcal{I}_i$ in camera $\mathcal{C}_i$ and a spatial location $\boldsymbol{y_j} \in \mathcal{I}_j$ in camera $\mathcal{C}_j$ form a *spatial correspondence*, given that they are the images of the same 3-D point (cf. Fig. 2.2 (left)). I will use the shorthand notation $\boldsymbol{x}_i \leftrightarrow \boldsymbol{y}_j$ to denote this relation.

For a fixed spatial location $\boldsymbol{x}_i$, there exists a *point-to-point* correspondence to a spatial location $\boldsymbol{y}_j$, if $\boldsymbol{x}_i \leftrightarrow \boldsymbol{y}_j$ holds for all $t \in \mathcal{T}$. For a fixed spatial location $\boldsymbol{x}_i$, there exists a *point-to-many* correspondence, if there exist at least two different points in time where $\boldsymbol{x}_i \leftrightarrow \boldsymbol{y}_j$ holds for $t_1 \in \mathcal{T}$ and $\boldsymbol{x}_i \leftrightarrow \boldsymbol{z}_j$ holds for $t_2 \in \mathcal{T}, t_1 \neq t_2$.

For the monocular setup, I define a *temporal correspondence* as the pair of pixel locations $\boldsymbol{x}_i$ and $\boldsymbol{y}_i$ in camera $\mathcal{C}_i$, which are the images of the same 3-D point at two consecutive points in time $t, t+1 \in \mathcal{T}$. As for the binocular setup, point-to-point and point-to-many correspondences are defined.

I will use the notation $\{\mathcal{X}\}_N$ to denote the set of N instances of $\mathcal{X}$. For example, $\{\boldsymbol{x}_i\}_N$ denotes the set of N pixel coordinates in $\mathcal{C}_i$. I use the notation $\boldsymbol{x}_{i;n}$ to denote the n-th pixel coordinates out of the given set.

2.2 Geometric and Photometric Relationships in Multi-Camera Scenarios

We regard the *geometric* and *photometric* relationship of a pair of cameras $\mathcal{C}_i$ and $\mathcal{C}_j$. The following discussion concentrates on the two camera case, however, the presented theory also holds for two images of the same camera, taken at two different time steps.

The geometric relationship describes *corresponding* pixel locations $\boldsymbol{x}_i \leftrightarrow \boldsymbol{y}_j$ among cameras $\mathcal{C}_i$ and $\mathcal{C}_j$. We may represent the geometric relationship in a functional form by means of a mapping function $\boldsymbol{f}$, which maps a pixel location $\boldsymbol{x}_i$ to its corresponding pixel location $\boldsymbol{y}_j$ according to $\boldsymbol{y_j} = \boldsymbol{f}(\boldsymbol{x}_i)$. Likewise, we may describe the geometric

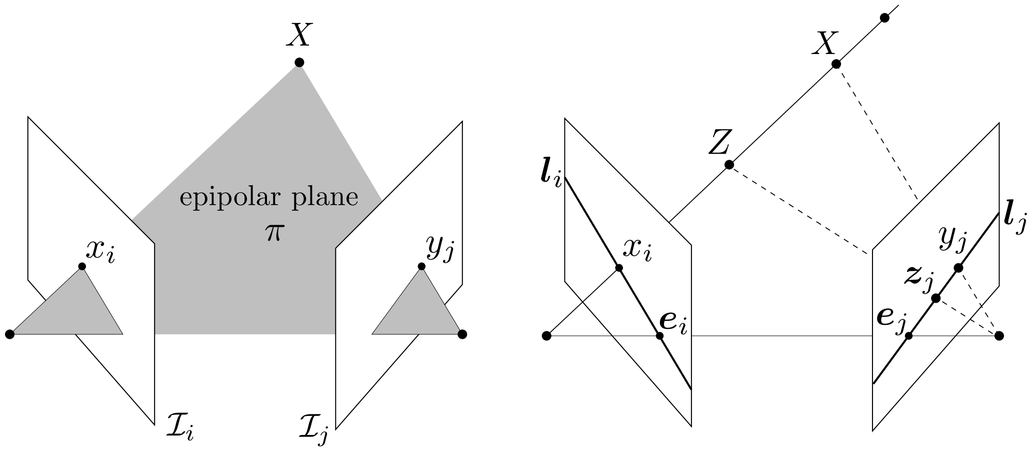


Figure 2.2: **Epipolar geometry:** (left) A 3-D (world) point X is projected onto the image plane of camera $\mathcal{C}_i$ and $\mathcal{C}_j$. The projected 2-D points form a spatial correspondence and lie in the epipolar plane $\boldsymbol{\pi}$, which is defined by the triangle given by the camera centres and the 3-D world point. (right) The true correspondence for a pixel location $\boldsymbol{x}_i$ in $\mathcal{C}_j$, lies along the epipolar line $\boldsymbol{l}_j$. If $\boldsymbol{x}_i$ is the projection of a different 3-D point the true correspondence in $\mathcal{C}_j$ will move along the epipolar line. Illustrations adapted from (Hartley and Zisserman 2004).

relationship by enumerating all pixel correspondences among cameras $\mathcal{C}_i$ and $\mathcal{C}_j$. As will be described in the following, depending on the scene structure, the mapping function or the set of corresponding pixels may vary over time. For the moment being, we regard the case in which the mapping may change over time. In the second part of the thesis (cf. Sec. 6) we regard the case of global / static mapping functions.

Based on the theory of *epipolar geometry*, the geometric relationship of a pair of cameras may be described in algebraic form by means of the *fundamental matrix* $\boldsymbol{F}$, or the *essential matrix* $\boldsymbol{E}$ if the calibration of the cameras is known (Hartley and Zisserman 2004).

The fundamental matrix is a 3 × 3 matrix of rank 2 and for a pair of corresponding pixels $\boldsymbol{x}_i \leftrightarrow \boldsymbol{y}_j$ (here assumed to be given in *homogeneous* coordinates) it holds:

$$\boldsymbol{y}_j^T \boldsymbol{F} \boldsymbol{x}_i = 0. \tag{2.6}$$

For each pixel location $\boldsymbol{x}_i$ in $\mathcal{C}_i$, the fundamental matrix defines a corresponding *epipolar line* $\boldsymbol{l}_j$ in $\mathcal{C}_j$ along which the true correspondence $\boldsymbol{y}_j$ must lie according to:

$$\boldsymbol{l}_j = \boldsymbol{F} \boldsymbol{x}_i. \tag{2.7}$$

Similarly, for $\boldsymbol{y}_j$ we have:

$$\boldsymbol{l}_i = \boldsymbol{F}^T \boldsymbol{y}_j. \tag{2.8}$$

Every epipolar line in $\mathcal{C}_i$ passes through the epipole $\boldsymbol{e}_i$ and likewise every epipolar line in $\mathcal{C}_j$ passes through the epipole $\boldsymbol{e}_j$. The epipole in $\mathcal{C}_i$ is the projection of the camera centre of $\mathcal{C}_j$ in $\mathcal{C}_i$ (and does not need to lie within the visible part of the image plane, cf. Fig. 2.2).

We see that if $\boldsymbol{F}$ is known, the search space for a correspondence in either view is reduced to a 1-D space, i.e., a line in 2-D space. It is important to note that the relations defined by the fundamental matrix are independent of the scene structure, i.e., scene depth. If the scene depth at pixel location $\boldsymbol{x}_i$ changes, its correspondence in $\mathcal{C}_j$ will simply move along the epipolar line. However, we also note that the fundamental matrix does not fully define the geometric relationship, it merely serves to restrict the search space for pixel correspondences among the views.

As can be seen from Eq. 2.6, the fundamental matrix can be estimated from a set of K known correspondences $\{\boldsymbol{x}_i \leftrightarrow \boldsymbol{y}_j\}_K$. Expanding and rearranging Eq. 2.6 we obtain:

$$\begin{aligned} &[(y_1)_j \cdot (x_1)_i \quad (y_1)_j \cdot (x_2)_i \quad (y_1)_j \quad (y_2)_j \cdot (x_1)_i \quad ... && (2.9)\\ &\quad ... \quad (y_2)_j \cdot (x_2)_i \quad (y_2)_j \quad (x_1)_i \quad (x_2)_i \quad 1] \;\; \boldsymbol{f} = 0, \\ &\Leftrightarrow \quad \boldsymbol{a}_{ij}^T \boldsymbol{f} = 0, && (2.10) \end{aligned}$$

with $\boldsymbol{f} = (f_{11}, f_{12}, f_{13}, f_{21}, f_{22}, f_{23}, f_{31}, f_{32}, f_{33})^T$ being the elements of $\boldsymbol{F}$. By stacking $\boldsymbol{a}_{ij}^T$ for the K (at least 8) correspondences in a matrix $\boldsymbol{A}$ we arrive at:

$$\begin{bmatrix} (\boldsymbol{a}_{ij;1})^T \\ (\boldsymbol{a}_{ij;2})^T \\ \vdots \\ (\boldsymbol{a}_{ij;K})^T \end{bmatrix} \boldsymbol{f} = \boldsymbol{A}\boldsymbol{f} = \boldsymbol{0}. \tag{2.11}$$

Equation 2.11 defines a homogeneous equation system which can be solved in a least squares sense via singular value decomposition (SVD) (Golub and Loan 2012). Once the fundamental matrix is determined, the rank constraint (rank 2) needs to be enforced, e.g., via the SVD (Hartley and Zisserman 2004, p. 279). It should be noted that the entries of matrix $\boldsymbol{A}$ should be normalised before solving for $\boldsymbol{f}$, as the entries in $\boldsymbol{A}$ largely differ in magnitude (cf. (ibid.) and further analysis in (Mühlich and Mester 2004)).

The fundamental matrix may also be deduced from only 7 correspondences, using the constraint $\det(\boldsymbol{F}) = 0$ (due to its rank deficiency). However, typically the fundamental matrix is robustly estimated from many more than 7 correspondences using a RANSAC style approach (Fischler and Bolles 1981; Hartley and Zisserman 2004).

As can be seen from the previous derivations, once a set of correspondences is available the estimation of $\boldsymbol{F}$ is straightforward. The challenge merely lies in the estimation of a suitable number of pixel correspondences, which will be discussed in the following section.

Besides the geometric relationship of a pair of cameras, their photometric relationship is of interest as well. Regard a pixel correspondence $\boldsymbol{x}_i \leftrightarrow \boldsymbol{y}_j$ at some point in time t. The respective grey values $s_i(\boldsymbol{x}_i, t)$ and $s_j(\boldsymbol{y}_j, t)$ will only rarely be identical, due to signal noise, but more importantly due to different photometric characteristics of the views. The photometric relationship defines the mappings of grey values:

$$s_i(\boldsymbol{x}_i, t) \rightarrow s_j(\boldsymbol{y}_j, t), \tag{2.12}$$

$$s_i(\boldsymbol{x}_i, t) \leftarrow s_j(\boldsymbol{y}_j, t), \tag{2.13}$$

and explain the illumination differences among the views.

Whenever information from multiple images is to be extracted or compared/matched, in general, illumination differences among the views need to be taken into account. We will return to the learning of photometric relationships including a review of relevant related work in Sec. 3.6.

Let us next regard the correspondence problem.

2.3 The Correspondence Problem in Low-Level Vision

In (Marr and Poggio 1979), Marr defines the correspondence problem for a pair of images as i) selecting a particular location in one of the images and ii) identify this location in the second view. While this is a simple, yet complete task description, it turns out that the correspondence problem is a challenging task in practice.

The basic principle of identifying correspondences appears in many forms in computer vision, both in low-level and high-level vision: In this thesis, I regard the correspondence problem in low-level vision, i.e., on the pixel level. However, on a more abstract level, the high-level task of, say, object recognition may also be interpreted as a correspondence problem.

The perhaps most widely studied instances of the correspondence problem in low-level vision appear in *stereo* vision and *motion* estimation. In stereo vision, we regard two images taken at the same point in time by two cameras $\mathcal{C}_i$ and $\mathcal{C}_j$. In the motion case, we regard two images taken at two different points in time by a single camera $\mathcal{C}_i$.

Estimation of stereo and motion correspondences play an important role in almost every high-level computer vision task. One reason for this is that motion and depth cues allow to form strong hypothesis about the sensed environment (e.g., distances to objects, moving direction of objects etc.) and hence disambiguate the visual input. Furthermore, based on a set of pixel correspondences we may compute an algebraic representation of the geometric relationships of a pair of cameras (i.e., the *fundamental matrix*, cf. Sec. 2.2). Identifying correspondences in multiple views is also a core problem in methods to 3-D reconstruction, such as structure from motion[1] and bundle adjustment[2], which

---

[1] see (Torr and Zisserman 2000) for a review of methods

[2] see (Triggs et al. 2000) for a review of methods

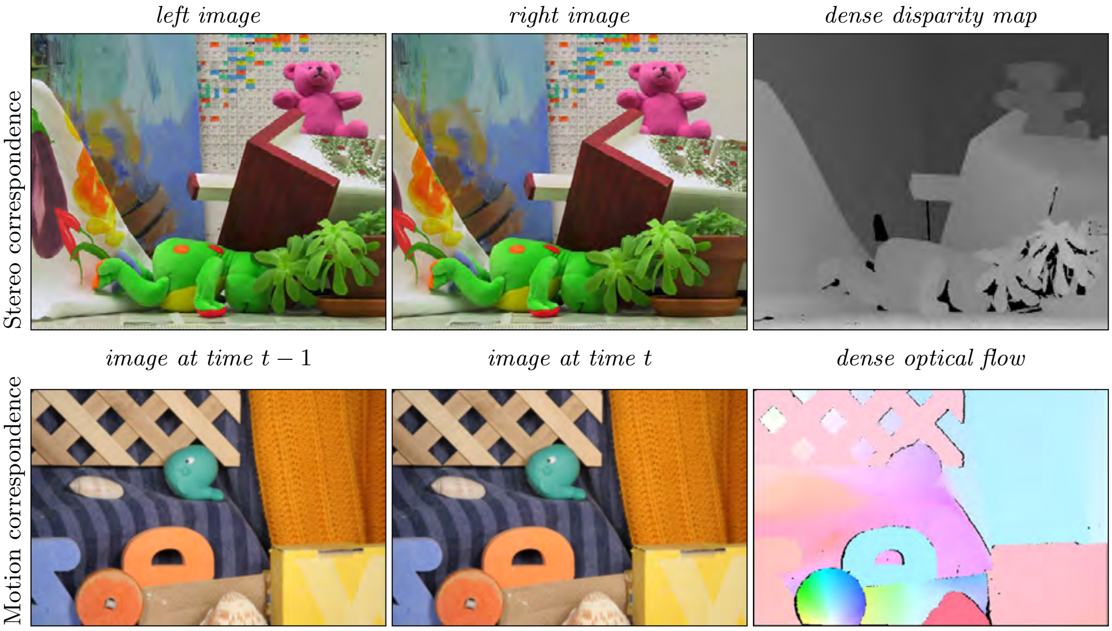


Figure 2.3: **Stereo and motion correspondence problem:** (top) Image pair `Teddy` related by disparity (stereo), (bottom) image pair `Rubber Whale` related by optical flow (motion). Images taken from (Baker et al. 2011). Compare with Fig. 2.4. Best viewed in colour.

is, e.g., used to reconstruct 3-D scenes from internet photo collections (Snavely et al. 2008, 2006).

Conceptually, the correspondence problem in stereo vision and motion estimation is the same; pairs of pixel locations which are the images of the same 3-D point are sought. The difference between a stereo and a motion correspondence merely lies in the *cause* or explanation why a particular correspondence exists. In stereo, a correspondence is induced by a specific scene depth and is encoded by the disparity vector. In the motion case, a correspondence is due to apparent object motion (which itself depends on the depth of the regarded object) and is encoded by the optical flow vector.

One refers to the *dense* or *sparse* correspondence problem if the correspondence problem is to be solved for every pixel (dense) or only a subset of pixels (sparse). Figure 2.3 visualises example image pairs for the stereo and motion problem, as well as a dense disparity map and a dense optical flow map. As shown in Fig. 2.3, a dense representation of disparity or optical flow is typically visualised as a grey or colour image. For the stereo case, if the images are rectified (correspondence known to have the same y coordinate) a grey value encodes the scalar disparity value. For the motion case, the optical flow vectors are typically colour coded in HSV space. The orientation of the vector determines the hue value, while its magnitude determines the saturation. A sparse set

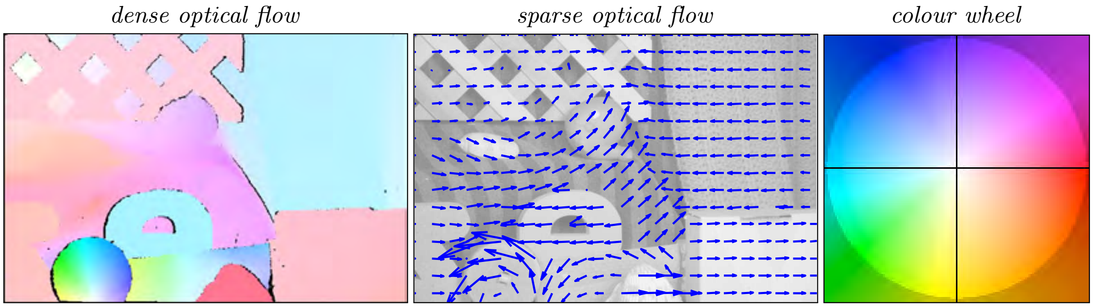


Figure 2.4: **Dense and sparse optical flow:** (left) Dense and (middle) sparse visualisation of optical flow for image pair `Rubber Whale` (Baker et al. 2011), (right) colour wheel by which the optical flow vector is colour coded. See text for details. Best viewed in colour.

of disparity or flow vectors is often visualised by means of a 2-D vector field (see Fig. 2.4 for examples).

Let us now turn to the problem of establishing spatial correspondences. As before, we may differentiate between sparse and dense algorithms. It is merely up to the final application (and the available computational resources) whether a dense or sparse representation is sought. If the goal is to generate accurate 3-D models of some real world object, a dense approach is to be preferred. If the goal is to track an object trough an image sequence, one would rather extract prominent object *features* and only compute the optical flow for these. It is beyond this thesis to give an in depth review of all the existing methods for stereo and motion correspondences, but I refer to (Baker et al. 2011; D. Scharstein and Szeliski 2002; Szeliski 2010) for an overview.

The efficiency and robustness of a correspondence algorithm may be increased by the incorporation of *prior knowledge* about the scene geometry, object motion etc. This prior knowledge allows to restrict the search space among which the true correspondence is to be expected. For example, if the fundamental matrix for a stereo image pair is known, the search space may be restricted to a line as opposed to the whole 2-D space. Similarly, a motion estimation scheme may be guided if the expected range of the flow vectors is known. This is of special interest when the optical flow is large.

In the following, I will review the standard approach to the estimation of (sparse) spatial correspondences, which is based on the spatial feature matching pipeline.

2.3.1 Correspondence Relations and the Spatial Feature Matching Pipeline

In the following, we regard a binocular camera setup with cameras $\mathcal{C}_i$ and $\mathcal{C}_j$. From both cameras, we obtain a stream of T images (cf. Sec. 2.1). We assume that the camera's fields of view overlap at least partially. Our goal is to establish a set of P correspondences

$\{\boldsymbol{x}_i \leftrightarrow \boldsymbol{y}_j\}_P$ across the images. We may study the correspondence problem now at some point in time t for the pair of images $I_{i,t}$ and $I_{j,t}$. As has been described already, it is important to note that the correspondences established at time t will usually only exist during short time intervals. If we regard a pixel location $\boldsymbol{x}_i$ in camera $\mathcal{C}_i$ as fixed, its corresponding location $\boldsymbol{y}_j$ in $\mathcal{C}_j$ may vary over time, depending on the scene structure and camera setup. I will discuss this in greater detail in Sec. 3, however, for the moment being, let us assume that our task is to establish correspondences among the pair of images $I_{i,t}$ and $I_{j,t}$.

The conventional spatial approach to establish a sparse set of correspondences is based on the spatial feature matching pipeline. This pipeline consists of three stages (Szeliski 2010):

i) the detection of H spatial *keypoints* or local features at positions $\{\boldsymbol{x}_i\}_H$ in image $I_{i,t}$ and K keypoints at locations $\{\boldsymbol{y}_j\}_K$ in image $I_{j,t}$. Usually we have $H \neq K$, i.e., we detect different numbers of keypoints across the images,

ii) each keypoint is described by means of a *descriptor*, based on a local neighbourhood centred on the keypoint,

iii) a set $\{\boldsymbol{x}_i \leftrightarrow \boldsymbol{y}_j\}_P$ of P correspondences is established by matching the descriptors across the images.

A keypoint may be regarded as a spatial location in an image with a distinctive local neighbourhood, which encodes a large amount of information that can be utilised to match keypoints across images. While keypoints are often considered to be of a corner type shape, it should be noted that keypoints may also be given by blobs, edges, etc. (cf. (ibid.)).

One of the early and perhaps most classic (corner) keypoint detectors is the *Harris* detector (Harris and Stephens 1988), which itself is an improved version of the detector proposed in (Moravec 1980). A keypoint is detected, given that the local autocorrelation surface of a pixel location is peaked (Szeliski 2010). This means, when a small window around the regarded pixel is shifted in any direction, this will lead to a large dissimilarity (in terms of the pixel wise SSD) with the original (non-shifted) patch. The detection itself is then based on the analysis of the local autocorrelation function, or the *structure tensor* (which itself is an estimate of the Hessian of the local autocorrelation, see Sec.3.6 for details). Other detectors, which are based on the analysis of the structure tensor have been proposed, e.g., (Förstner and Gülch 1987; Shi and Tomasi 1994). Other well known detectors are the FAST (Rosten and Drummond 2006) and FASTER (Rosten, Porter, et al. 2010) detectors, which are tailored for real time processing. See (Schmid et al. 2000; Szeliski 2010; Triggs 2004; Tuytelaars and Mikolajczyk 2008) for a general review of other detectors and further details.

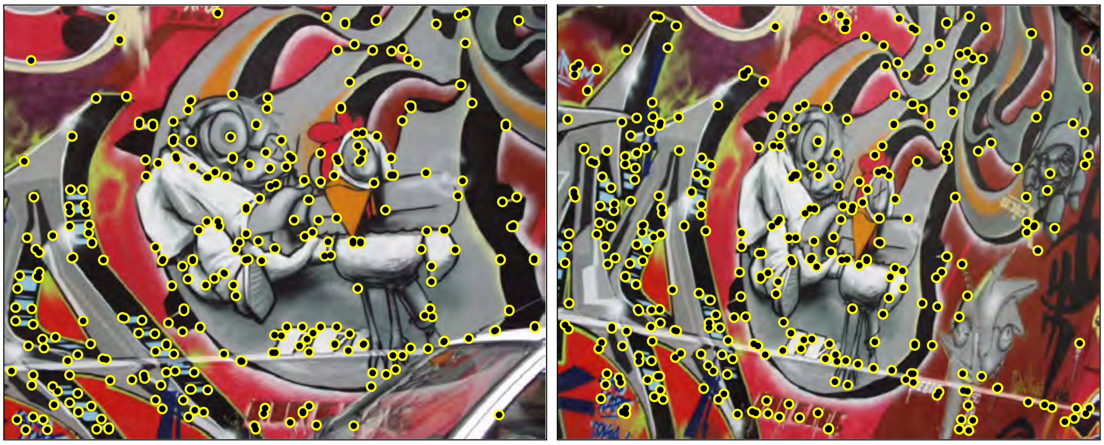


Figure 2.5: **Local spatial features:** For two images of the `Graf` data set (Mikolajczyk 2014), I computed *Harris* corners, which are marked by yellow/black dots. Best viewed in colour.

According to (Förstner 1986, 1991), important properties of keypoint detectors are (among others) distinctness, invariance and stability. *Distinctness* refers to the definition of a keypoint itself. *Invariance* means that keypoints, or rather the measure based on which they are computed should be invariant to expected geometric distortions. *Stability* refers to a property, which today is often termed repeatability (due to (Schmid et al. 2000)) and means that a keypoint detected in image $I_{i,t}$ should also be detected in image $I_{j,t}$ (if visible). Clearly, spatial transformations of the input images are a challenge to keypoint detectors, as they work on a spatial support region around the regarded pixel. If the image is then viewed under, say, an affine transform, the descriptor may change or no keypoint may be detectable at all. Similar arguments hold when the illumination among the images is different. This is often handled via some form of normalisation of the grey values (cf. Sec. 3.6 for further discussion).

Figure 2.5 shows Harris keypoints, which I have computed for two images of the *Graf* dataset (Mikolajczyk 2014). It can be seen that keypoints are detected at corner like structures, but not in homogeneous areas or line like structures. While the images show the same content to a large extent, there is a considerable difference in the viewpoint. By visual inspection, it can be seen that many keypoints found in the first image are also detected in the second image. However, there is a considerable amount of non-repeated keypoints.

Once sets of keypoints $\{\boldsymbol{x}_i\}_H$ and $\{\boldsymbol{y}_j\}_K$ in images $I_{i,t}$ and $I_{j,t}$ are detected, they have to be matched based on some form of descriptors. As for keypoint detectors, many different descriptors have been proposed. Design criteria for descriptors are invariance to certain geometric and photometric transforms, or covariance such that the descriptor commutes with these transforms (Tuytelaars and Mikolajczyk 2008). The perhaps best known descriptor (including a detection stage) is the *Scale Invariant Feature Transform* (SIFT) (Lowe 2004, 1999). While it is proven to produce stable results, it suffers from high computational demands. Since then, an ever growing number of descriptors have been proposed. *Speeded Up Robust Features* (SURF) (Bay et al. 2006) (including a detection stage as well) which may be seen as a faster extension of SIFT. The *Binary Robust Independent Elementary Features* (BRIEF) descriptor (Calonder et al. 2010) is tailored to be efficiently computable and matchable. The descriptor builds on intensity differences which are mapped to a binary representation. Matching is then performed by computing the Hamming distance between descriptors (Rublee et al. 2011). For further details on keypoint descriptors and comparative evaluations of these see, e.g., (Mikolajczyk and Schmid 2005).

Besides the purely spatial approaches described previously, several spatiotemporal detectors and descriptors have been proposed. Many of these are extensions of classic spatial approaches like the Harris detector to the temporal domain (Laptev 2005).

Spatiotemporal detectors and descriptors are mainly used in event and action recognition (see (H. Wang et al. 2009) for an overview). However, the spatiotemporal features may not directly be utilised to determine pixel correspondences across images, as the correspondence relation may change over time and hence *within* the spatiotemporal feature. See Sec. 3.1 for a detailed discussion of this issue.

While the correspondence problem may be studied for a pair of static images $I_{i,t}$ and $I_{j,t}$ as described previously, today it becomes increasingly important to establish correspondences in streams of images $\{I_i\}_T$ and $\{I_j\}_T$. For example, in multi-camera networks, driver assistance scenarios (Franke et al. 2005; Müller et al. 2011) or simultaneous localisation and mapping applications (Thrun et al. 2005), where long image sequences are available.

Most approaches to estimate the geometric relations between multiple views of a dynamic scene are based on processing information that is either purely spatial, or a compound of spatial and temporal information. Usually correspondences are sought to estimate the fundamental matrix, trifocal tensor, etc. Often, knowledge about epipolar geometry is used as a constraint during estimation. Examples of purely spatial methods are the ones described previously. However, it seems to be rather inefficient not to make use of the temporal information.

In the field of multi-camera networks, several spatiotemporal approaches have been proposed. Wang et al. (X. Wang et al. 2010) describe learning of correspondences by matching trajectories that belong to the same activity, in order to connect non-overlapping viewpoints over entry and exit zones.

Tracks in multiple cameras are also generated in (L. Lee et al. 2000) and used to estimate an overall ground plane, which is aligned to the different camera views.

In (Sinha et al. 2004), the epipolar geometry for a set of cameras is estimated by dynamic object silhouettes. Here, they make explicit use of constraints derived from epipolar geometry.

In order to learn the topology of a camera network, Ellis et al. (Ellis et al. 2003) identify camera entry and exit zones between different views by accumulating events of disappearing and appearing objects within these zones.

There are also semi-automatic methods, e.g. in (Svoboda et al. 2005), in which fiducial markers/laser pointers are manually moved through the camera network. These markers are easily tracked within the images and allow for recovery of the epipolar geometry.

In (van den Hengel et al. 2007b; van den Hengel, Dick, and Hill 2006) a method to determine camera overlap in large surveillance networks is presented. The approach is based on the principle that non-overlapping image regions show distinct activity patterns at a specific time instant.

2.3.2 Learning Correspondence Relations

While we have seen how correspondences can be *computed* based on local features, in this thesis I am interested in *learning* correspondences within long streams of images *without* explicitly solving the correspondence problem and without using knowledge about epipolar geometry at all. As we have seen, learning of the geometric relationship may be studied locally or globally. In a local approach, individual pixel correspondences are sought, while in a global approach a holistic mapping function is sought. In the first part of the thesis, I will address unsupervised learning of individual pixel correspondences while the second part of the thesis is concerned with learning global mappings/transformations. Results presented in this thesis show that correspondence relations can evolve autonomously, both for the stereo and motion case.

2.4 Summary

To summarise, this chapter introduced the principles of correspondence estimation in low-level vision. The common approach to establish pixel correspondences is based on the detection and matching of local spatial keypoints. These correspondences are then the precursor to estimate the fundamental matrix. The fundamental matrix is an algebraic representation of the geometric relationship of a pair of views. The fundamental matrix is independent of the scene structure observed by the cameras and reduces the search space for correspondences to a 1-D space.

Part I

Learning Pixel Correspondences

3 Temporal Coincidence Analysis

In Sec. 2, we have seen that the predominant approach to establish spatial correspondences is based on the detection of i) spatial keypoints and ii) matching of these keypoints based on spatial descriptors. In fact, there is a large 'toolbox' of keypoint detectors and descriptors, which allows to actually engineer a system that *computes* spatial correspondences. In contrast, I am interested in discovering principles that let a system *learn* correspondences autonomously. In the following, I will motivate and develop such a principle, which only regards the temporal information of single pixels and never explicitly computes disparity or flow vectors and does not assume any knowledge about epipolar geometry. For reasons that become clear shortly, I denote this approach *Temporal Coincidence Analysis* (TCA).

3.1 Introduction

In the following, I will develop and analyse the method of *Temporal Coincidence Analysis* (TCA), which is an algorithm to learn the geometric and photometric relationships among pairs of views. The key motivation of the proposed method is the question: What can we learn about the relationships between pairs of views by only looking at single pixels over extended periods of time and simultaneously making almost no assumptions about camera models or scene geometry? While the mathematical structure of the geometric relationship between two cameras is well known and explained by epipolar geometry (cf. Sec. 2.2 and (Hartley and Zisserman 2004)), this does not explain how correspondences may evolve over time and can be learnt in an autonomous way.

In contrast to the spatial feature matching pipeline (cf. Sec. 2.3), I only regard the temporal pixel grey value process and show that comparing single pixels, but doing so for a long time, allows to solve the task of finding geometric correspondences between images. The principle of the method is to collect evidence over extended periods of time: Two pixels in two different views of the same scene *may* correspond to the same location in the observed (3-D) space, if their grey value signals show a similar behaviour over time. Compared to a spatial approach, TCA shares a similar processing structure, but instead of spatial pixel neighbourhoods we identify, describe and match single pixel *temporal* neighbourhoods. Next, I will give an intuitive and rather informal overview of the approach before I present a formal treatment in Sec. 3.3.

First, let me motivate that the temporal signal of single pixels carries enough information to establish spatial correspondences. To this end, we detect temporal keypoints,

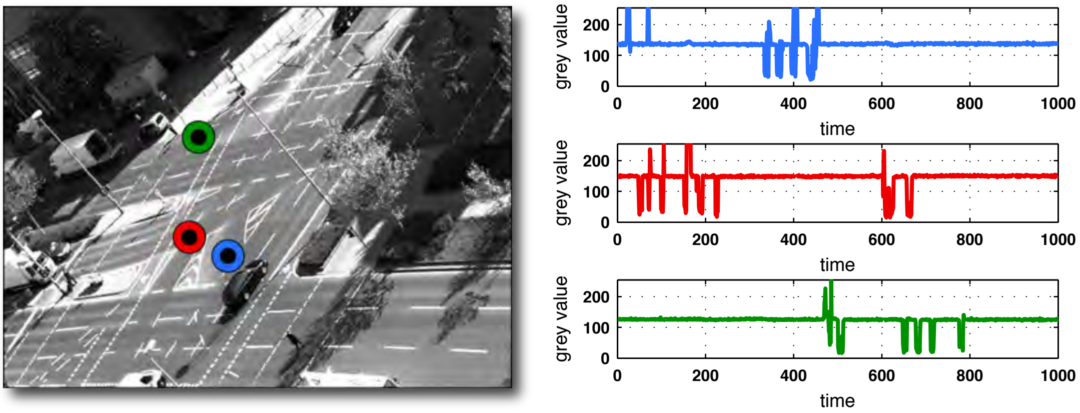


Figure 3.1: **Temporal signal of single pixels:** (left) For each selected pixel, its temporal grey value signal over 1000 frames of view $\mathcal{C}_1$ of sequence `GUBo1616` (right) is plotted. The temporal signal of each pixel may be considered a fingerprint. TCA solely builds on these temporal signals to establish correspondences. See text for details.

summarise them by a suitable descriptor and finally match them. Regard Fig. 3.1. For three arbitrarily selected pixels in view $\mathcal{C}_1$ of sequence `GUBo1616` (c.f. App. A), their temporal grey value signal is plotted over 1000 frames. It can be seen that the temporal signals are most of the time unique and characteristic for each of the selected pixels. Furthermore, it seems that the temporal signals switch between two states: i) a *resting* state, where the signal is merely constant but afflicted with noise, and ii) an *active* state, where the signal changes suddenly and abruptly. Figure 3.2 shows for each of the previously selected pixels the frame at time t, for which the signal change between time $t-1$ and t was maximum within the 1000 regarded frames. It can be seen that in this scene, strong signal changes are due to object motion, caused by cars and pedestrians at the regarded pixel. Besides object motion, signal changes may be caused due to self (camera) motion, illumination changes etc. The detection and matching of these signal changes are at the core of TCA. We also need to define a descriptor, which summarises the signal change over a specific *length* of the signal. In the previous example, signal changes were described by the grey value difference within consecutive time steps. Other choices are of course possible, e.g., the absolute difference or a simple binary descriptor. However, the descriptors I consider are only determined within consecutive time steps. While this could be considered an arbitrary choice, the theoretical justification for this is that a pixel-to-pixel correspondence may change over time when the scene depth at the regarded pixel changes (cf. Sec. 3.4 for further details).

In the following, I am especially interested in strong signal changes above the signal noise level and denote these as *events* or keypoints in the temporal signal. Events may be considered the temporal counterpart to spatial keypoints; both are assumed

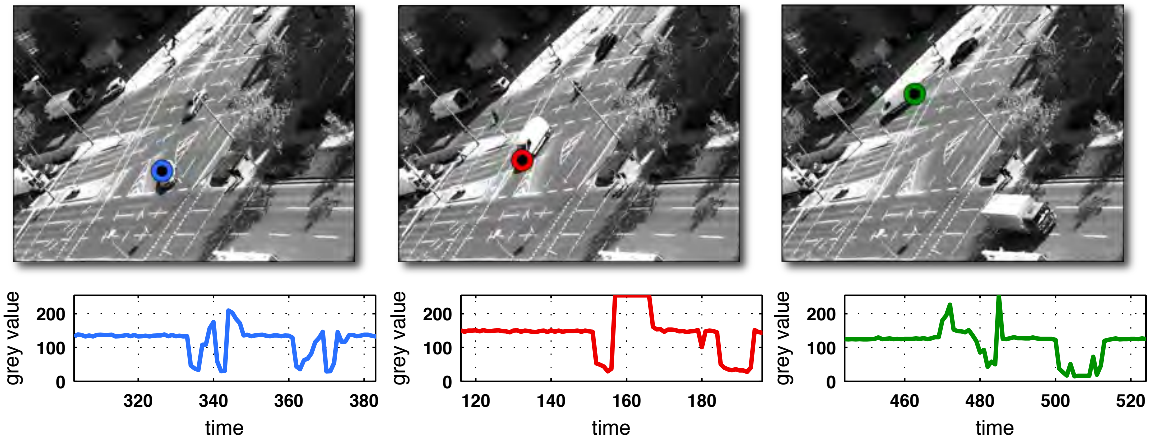
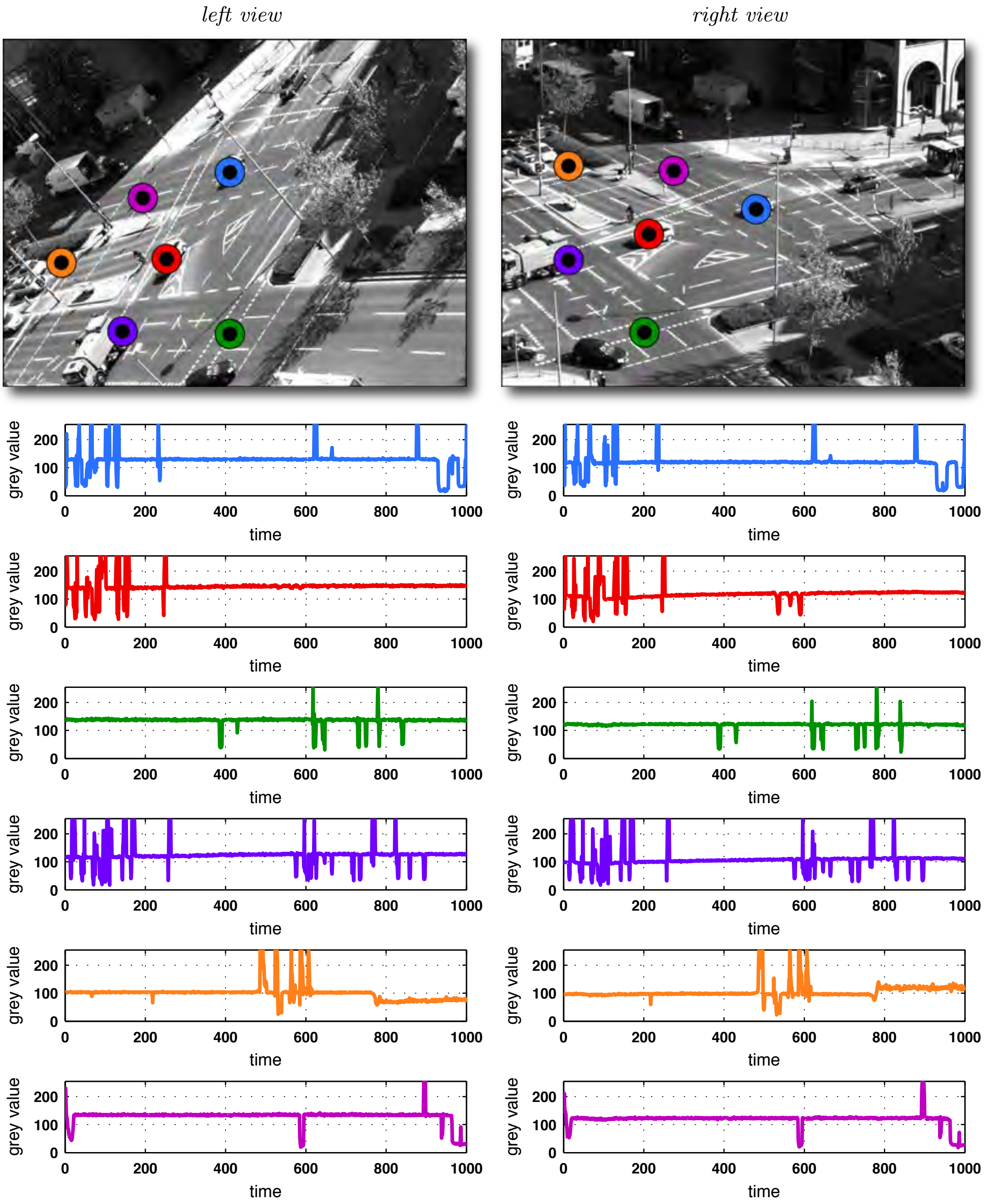


Figure 3.2: **Large temporal signal changes:** For each selected pixel, the frame at time t, for which the signal change between time $t-1$ and t was maximum is visualised. For this sequence, signal changes above the noise level are predominantly due to object motion. See text for details.

to be unique or distinctive within the temporal and spatial domain, respectively, and thus carry information, which allows them to be matched. As will be shown later, the uniqueness or distinctiveness of an event is inversely proportional to the probability of observing the event.

Let us now turn to the problem of identifying corresponding pixels among two views. Intuitively, a pair of corresponding pixels should obey a similar temporal grey value signal (assuming mainly lambertian surfaces). Figure 3.3 visualises the temporal grey value signal for six hand selected correspondences in sequence `GUBo1616`. Again, the individual temporal signals switch between a resting and an active state. As all selected pixels lie on the road surface, their grey values within the resting state are very similar, and we may not expect to identify a correspondence based on single grey values. Note that in practice, even corresponding signals may differ significantly by means of their grey values due to noise, illumination differences, differing camera transfer functions etc.. A single pixel in one view may also be distributed over a set of pixels in a second view, due to perspective differences of the views. All these effects can also be seen from the signals shown in Fig. 3.3. Hence, similar grey values are only a very weak cue for a correspondence. However, really decisive are strong *coincident* signal changes, i.e., *events* in the grey value signal. Figure 3.4 visualises the same signals as Fig. 3.3, but now with constant signal parts masked out. Regard the first (blue marked) signal and its first non-masked signal chunk, where events occur. By only considering events in the signal, the number of possibly corresponding signals gets reduced (green and orange marked signals), but we are still not able to uniquely identify the true correspondence. Obviously, non-corresponding signals may obey an event at the same point in time (red and purple marked signals). However, what is possible is to identify a correspondence based on the *repeated* coincidence of events. The idea is then to accumulate for each signal and each possible correspondence the number of similar events over time. Then, the true corresponding signal should have the highest count of coincident events and we may learn the true correspondence if a sufficient number of frames has been processed.

It is very important to note that by accumulating coincident events, we learn a *distribution* over the spatial domain, which encodes the average correspondence relation. This is a conceptual difference to classical spatial feature matching, both with respect to the processing structure but more importantly to the results obtained: a spatial approach considers a single pair of images and outputs a pixel-to-pixel correspondence. Compared to this, I process sequences of images and output a *correspondence distribution.* Only under specific conditions, which will be explained in the following, the results of both approaches are identical.



left view *right view*

Figure 3.3: **Temporal signal of pixels in two views:** For each selected pixel within the views of `GUBo1616`, its temporal grey value signal within 1,000 frames is plotted. Each grey value signal contains large and abrupt changes. See text for details. Figure only interpretable when viewed in colour.

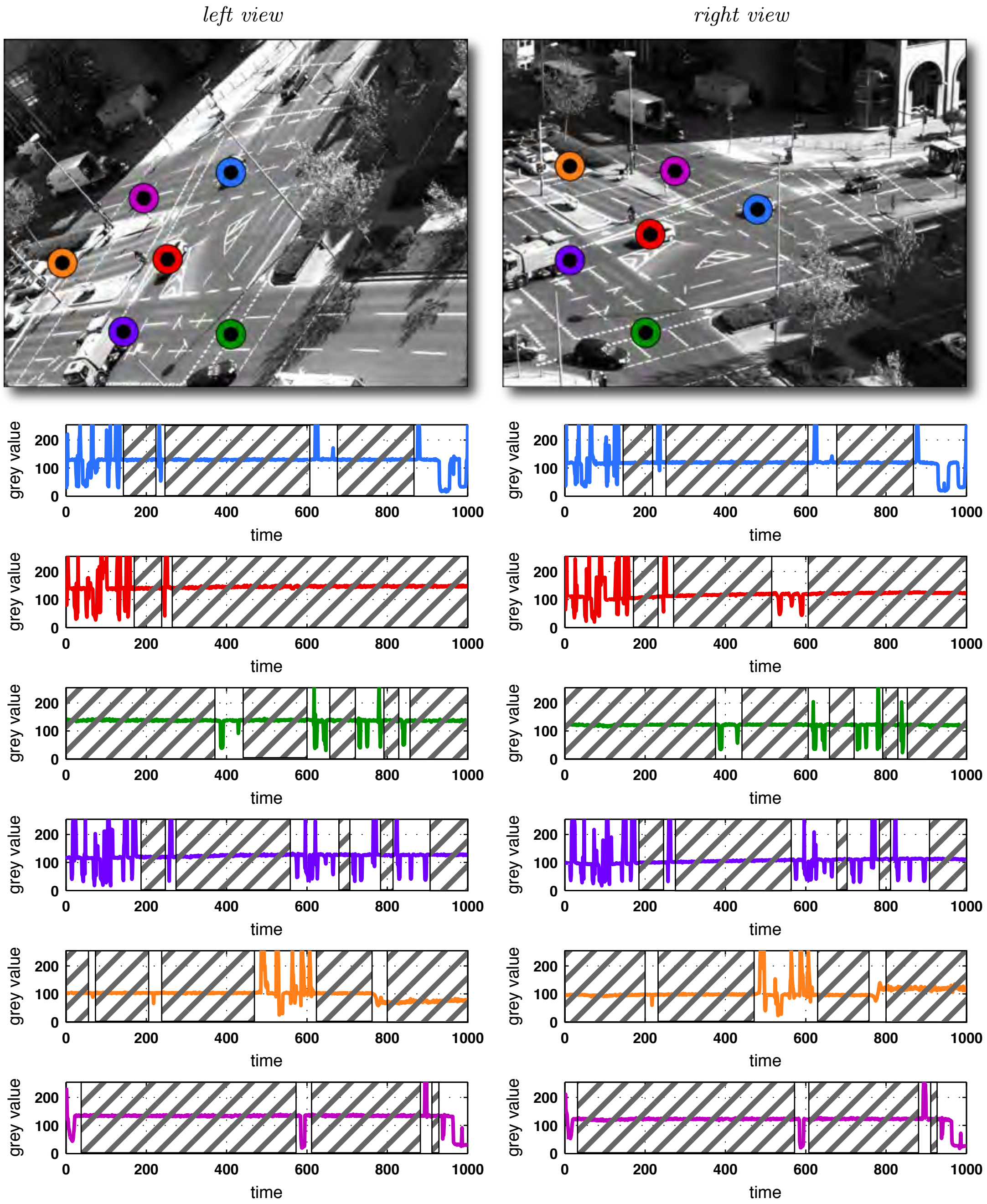


Figure 3.4: **Events in the temporal signal of single pixels:** For each selected pixel, its temporal grey value signal within 1,000 frames is plotted. Constant signal parts carry only little information for a correspondence and are masked out. Decisive are strong signal changes, i.e., *events* in the grey value signal. See text for details. Figure only interpretable when viewed in colour.

I stress again that TCA is not meant as a competitor to classic stereo or flow algorithms tailored to return instantaneous (per frame) correspondences, but may be seen as a prior generator. There are good reasons to make use of well-engineered algorithms to estimate highly accurate disparity or optical flow, but my objective is to demonstrate that correspondence relations may be learnt and updated over time without any supervision and minimal assumptions about the given data; all that is needed are intensity changes above the sensor noise level. Observe that the average correspondence relations still contain valuable information, e.g., in order to guide a higher level process, restrict search areas, or generate confidence information.

Next, let me introduce the concept of a correspondence distribution.

3.1.1 Correspondence Distribution

Regard a binocular camera setup with cameras $\mathcal{C}_i$ and $\mathcal{C}_j$. For a pixel $\boldsymbol{x}_i$ in the image plane of the first camera, its associated correspondence distribution is a discrete bivariate probability distribution over the (discrete) spatial domain of view $\mathcal{C}_j$. The distribution assigns non zero probability to those locations, where the true correspondence *may* lie.

Let $\boldsymbol{Y}_j = (Y_1, Y_2)_j^T$ be a random vector, defined on the two dimensional coordinates of the spatial domain $\mathcal{I}_j$ of camera $\mathcal{C}_j$. Random vector $\boldsymbol{Y}_j$ is distributed according to a probability mass function $f_{\boldsymbol{Y}}$ and the probability of observing the event $\boldsymbol{Y} = \boldsymbol{y}_j$ is given as:

$$f_{\boldsymbol{Y}}\left[\boldsymbol{y}_j\right] = Pr[\boldsymbol{Y} = \boldsymbol{y}_j]. \tag{3.1}$$

The distribution depends on the camera model, the scene geometry and the actual correspondence relation (stereo or motion). In the examples and explanations given in the following, I will usually refer to stereo correspondence distributions. However, they are equally defined for the motion case.

I will make no assumption about the shape of the distribution, specifically, I do not restrict the distribution to be of a certain parametric form. However, in practice, the correspondence distributions will not be arbitrary, but fall into a small number of classes as discussed in the following.

Assume that the scene depth at pixel $\boldsymbol{x}_i$ is arbitrary but fixed over time, and that there is a corresponding pixel $\boldsymbol{z}_j \in \mathcal{I}_j$ in the spatial domain of camera $\mathcal{C}_j$, such that $\boldsymbol{x}_i \leftrightarrow \boldsymbol{z}_j$ holds. Then, the correspondence distribution of pixel $\boldsymbol{x}_i$ in camera $\mathcal{C}_j$ is given by a unit impulse δ at the true corresponding location in $\mathcal{I}_j$:

$$f_{\boldsymbol{Y}}\left[\boldsymbol{y}_j\right] = Pr[\boldsymbol{Y} = \boldsymbol{y}_j] = \delta(\boldsymbol{y}_j - \boldsymbol{z}_j). \tag{3.2}$$

Note that a correspondence distribution of the form given in Eq. 3.2 only exists when the scene depth at pixel $\boldsymbol{x}_i$ is constant over time. This is for example the case for the correspondences in sequence `GUBo1616`, shown in Fig. 3.3. As the cameras observe a traffic junction from a distance much larger than the expected depth changes due to moving cars and pedestrians, the scene depth at any pixel will at least approximately be constant. Therefore, every correspondence that exists between the views will be valid for any time instant and as long as the relative orientation of the cameras does not change. Compare this to correspondences in sequence `GUCar`. As the cameras are moved through a traffic scene, we expect the scene depth to vary. Regard Fig. 3.5, which shows for a selected pixel $\boldsymbol{x}_1$ in $\mathcal{C}_1$, its true corresponding pixel in $\mathcal{C}_2$, at two different points in time. Observe how the corresponding pixel in $\mathcal{C}_2$ has moved between the regarded time steps. Clearly, this is due to the fact that the scene depth at pixel $\boldsymbol{x}_1$ has changed. While at time t the pixel lies on the road surface, at time $t+1$ the pixel lies on the car, which in fact is closer to the camera than the road surface. Hence, the location of the true correspondence has to change as well. However, the location of the true correspondence will not be arbitrary, but according to the theorems of epipolar geometry (cf. Sec. 2.2), will be restricted to lie on the *epipolar line.*

If we assume that the scene depth at a pixel $\boldsymbol{x}_i$ is sampled equally often, the correspondence distribution will be given by a uniform distribution along a line in the spatial coordinate space, that is, a uniform distribution along a 1D structure. Depending on the actual camera model, this might well be a curve, e.g., for fisheye optics. In natural scenes, the scene depth at a specific pixel will not be sampled uniformly, but will be limited to a small range of possible depth values. Therefore, we only observe parts of the true correspondence distribution, which have to be understood as the *average* correspondence relation.

As noted previously, if nothing is known about the camera setup, the actual shape of the correspondence distribution may be arbitrary. However, for many real world camera setups, the type of correspondence distributions being observed will fall into a small number of classes. The classes of correspondence distributions which I consider in the following are:

- i) point-to-point correspondences, if only small depth variations occur,
- ii) point-to-line correspondences, if large depth variations occur,
- iii) no correspondence, if no correspondence exists.

Note that these classes are not meant to be exclusive to a certain camera setup, but only depend on the observed scene depth. For example, in sequences similar to `GUCar`, the pixels at the horizon are expected to obey a point-to-point correspondence, as the scene depth will stay almost the same. Pixels in the centre part of the image are expected to obey a point-to-line correspondence distribution, as the scene depth changes often (cf. Fig. 3.5).

While a classic spatial feature approach addresses the problem of estimating point-to-point correspondences for single pairs of images, my goal is to estimate the correspondence distribution. It is important to note that we may not directly read off the

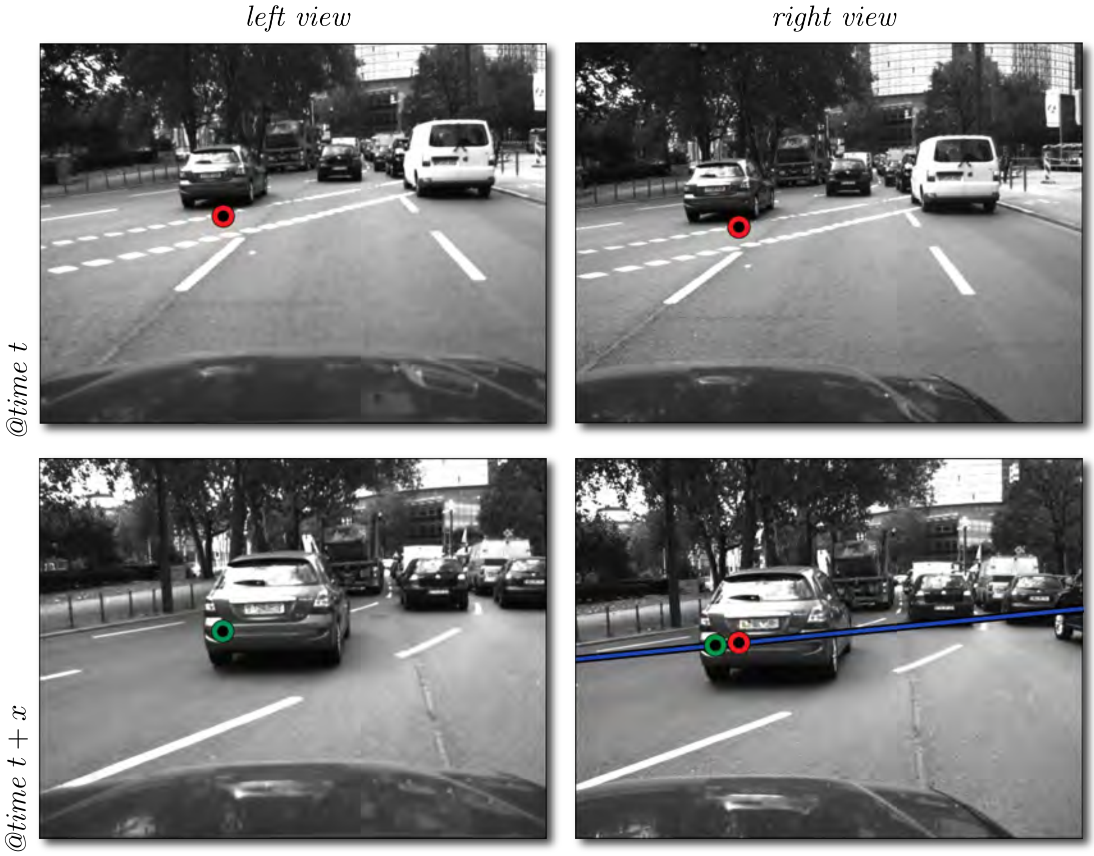


Figure 3.5: **Stereo correspondence distributions:** (top) A correspondence at time t is visualised (red marking). (bottom) When the scene depth at time $t + x$ varies, the true correspondence (green marking) varies along the epipolar ray (blue line). Best viewed in colour. See text for details.

true correspondence at a specific point in time. From a point-to-line correspondence distribution, we may only determine a region of high probability containing the true correspondence. Clearly, this information could be used as a prior term in a spatial approach.

3.1.2 Basic Temporal Coincidence Algorithm

From the previous examples, we have seen that corresponding pixels obey a similar temporal signal. While single grey values are only a very weak cue for a correspondence, really decisive are the repeated detection and matching of coincident events in the temporal signal.

Temporal Coincidence Analysis builds on this idea and is an algorithm to learn correspondence distributions. TCA is split into two phase, which are repeated over time: 1) event detection, 2) matching of events based on a suitable descriptor. Matched events are then accumulated over time and represent an empirical estimate of a correspondence distribution. In the following, I show that this principle can be used to learn correspondence distributions induced by depth or motion, for cameras with quite different optical characteristics, as well as for cameras moving arbitrarily in the world, as long as the *relative* orientation of the cameras is kept fixed. Under certain conditions, TCA may also be used to estimate differential motion parameters (see Sec. 5). Besides learning *geometric* relations between pairs of views I am also interested in learning the *photometric* relationships. In fact, these are intertwined tasks: the learning of correspondences depends on a model of the photometric relationships and the learning of the photometric relationships depends on estimated correspondences.

Obviously, the principle of observing and matching temporal changes within the signal will only be applicable when the scene is not essentially static. We may only learn correspondences for those pixels where we actually observe changes over extended periods of time. However, taking a biological inspired view on vision we may ask what can be learnt from static sensory input?

Before we turn to a formal treatment of TCA, let me present a concluding example for an intuitive understanding of the proposed approach. Regard Fig. 3.6 (top left) which shows 4 views from a multi-camera lab setup. For the blue marked pixel in the upper left view (also denoted as *seed pixel*), its correspondence distribution within the other views is sought. Note that the selected pixel lies within a textureless (=homogeneous) area. Obviously, the pool of potentially corresponding pixels in the other views is very large, when only considering spatial information. As a consequence, any classic spatial feature based method would fail to reliably extract the true correspondence. In TCA, we continuously determine whether the grey value at the selected pixel has changed significantly between two time steps, and if so, an event is detected. A detected event will then be summarised by a descriptor, which in this example is a simple binary variable. Given an event has been detected at the seed pixel, all pixels in the other views showing an event too are determined. Then, events are matched based on the descriptors. Matched pixel locations are then used to update the seed pixel's correspondence distribution per view. For the moment being, this update procedure may be thought of as simply counting the number of coincident events.

In Fig. 3.6 (top right) an event is detected when the truck covers the seed pixel. Then, for each pixel in the other views showing an event too, the probability of being the true correspondence increases, as visualised by the overlaid correspondence distribution. Obviously, the more temporally coincident signal changes are detected, the higher the correspondence probability will be. From Fig. 3.6 (bottom left and bottom right) it can be seen that when the learning is performed over more and more frames, the correspondence distributions converge to the area containing the true correspondence. As we here use a binary descriptor, we do not expect to learn a pixel-to-pixel correspondence distribution. This is due to the fact that an event at the seed pixel will be matched to whole blobs in the other views. By using a different descriptor, e.g. the absolute difference of the grey values, less pixels would be matched. This will be further discussed in the following sections.

Figure 3.7 shows estimates of the correspondence distribution obtained after 100, 300 and 400 processed images, respectively.

Next, I will show that TCA can formally be derived, based on a probabilistic model. In this model, the correspondence distribution is represented as a posterior distribution which is updated over time.

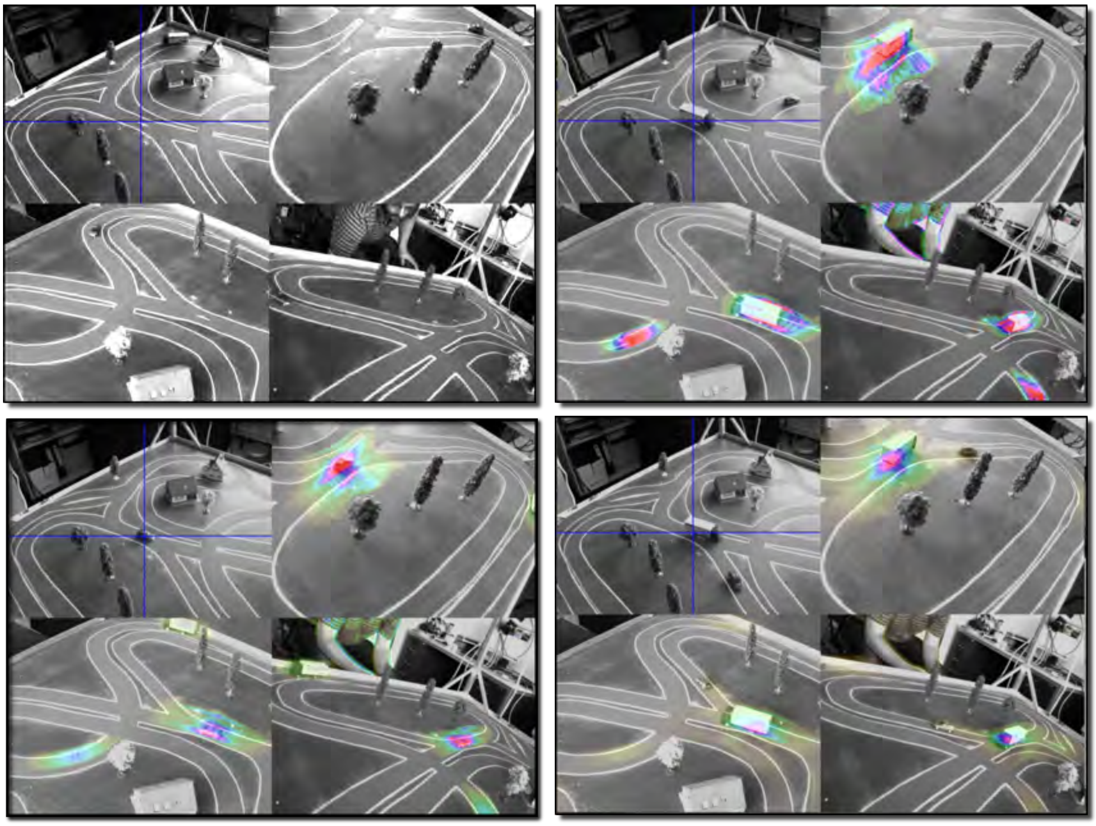


Figure 3.6: **Lab setup experiment:** 4 cameras face the same 3-D world. For the pixel marked with a blue cross (seed pixel) its correspondence distribution in the other views is sought. Correspondence distributions are learnt via TCA and are overlaid within each view. The more events are detected and matched, the better the approximation of the true correspondence distribution. See text for details. Figure only interpretable when viewed in colour.

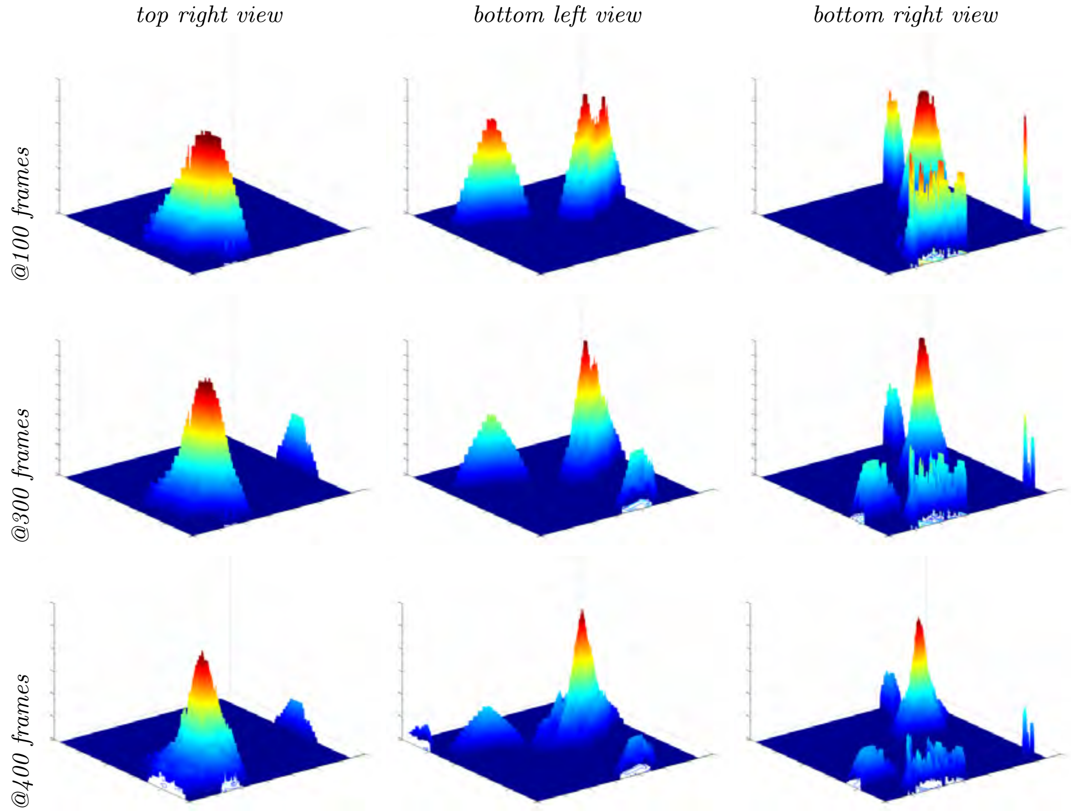


Figure 3.7: **Evolving correspondence distributions for the lab setup:** After 100, 300 and 400 processed images. Peak within each correspondence distribution marks the true correspondence. See text for details.

3.2 Related Work

In Sec. 2.3, principles and methods of correspondence estimation have been introduced. In the following, I will briefly discuss methods which are more closely related to the approach developed in this thesis.

In the approach of (Wexler et al. 2003) the epipolar geometry is learnt in a correspondence free manner. For a small patch centred at a pixel $\boldsymbol{x}_i$ in $\mathcal{C}_i$, its corresponding pixel in $\mathcal{C}_j$ is sought by computing and accumulating the similarity to all patches in $\mathcal{C}_j$ over many frames. The similarity measure is based on the colour difference of the regarded patches. Finally, they obtain an accumulator array which contains clusters of matched points lying on the epipolar line. In this basic version of the algorithm no temporal information is used. While they discuss an extension to penalise temporally non-coherent matches, this is not further analysed. In contrast to this work, I solely regard the temporal information of single pixels and hence make no assumption on the topological ordering of the pixels as is done in (ibid.).

In (Felsberg et al. 2013), an iterative learning scheme for point correspondences is presented. The approach is based on the channel representation of a set of local features in the regarded views. It is assumed that the feature points only move along 2-D surfaces in 3-D space. Their approach differs from the one presented in this thesis, as I never compute local features and make no assumptions on the actual scene geometry/interaction.

The so called Joint-Feature-Distributions (JDF) proposed in (Triggs 2001) share the idea of a correspondence distribution. While I make no assumptions on the parametric form of this distribution, the JDFs are parametric-models of correspondence relations derived for the affine and projective geometry. Once trained, they allow to generate spatial priors for a correspondence given a selected pixel. While JDFs are trained on a set of given correspondences, I never explicitly solve the correspondence problem and I do not assume to have ground truth correspondence available.

Certainly the idea of analysing the temporal change of single pixels is not exclusive to this thesis but appears in several forms. Besides specific (software) algorithms, hardware architectures, so called Address-Event Representation (AER) vision sensors have been developed (Delbrück et al. 2010; Mahowald 1994; Mead and Mahowald 1988). An AER sensor mimics the human retina and hence is often referred to as *silicon retina*. These vision sensors represent a perceived scene by means of intensity changes only and reduce redundant information. In this representation, static parts of the scene are discarded, as their intensity values only vary within the noise range. In contrast to a standard frame based camera, an AER sensor operates at a much higher temporal resolution (several hundred fps). However, these sensors also operate at a much smaller spatial resolution (from 32 × 32 pixels to 304 × 240 pixels) (Delbrück et al. 2010). While an AER type representation can be computed from a standard camera, the temporal resolution will be limited by the camera used and will usually be much smaller than the temporal resolution of an AER sensor.

Based on an AER type representation, in (Humenberger et al. 2012) an algorithm for (person) fall/tumble detection is developed, considering relative light intensity changes only. In (Carneiro et al. 2013), an algorithm for scene reconstruction is proposed. The authors of (Kogler et al. 2009) are interested in using an AER sensor as a cheap replacement for a standard camera in a pre-crash detection system in an automotive environment. They apply standard frame based stereo algorithms to the output of a stereo AER setup. According to the authors, the conversion of the AER to a frame based representation is probably too time consuming to handle the high frame rates of the AER sensor, thus limiting its advantage of a high frame rate.

In (Benosman et al. 2011), an algorithm to learn epipolar geometry, based on an AER sensor is proposed. Similar to the presented approach, they regard so called co-activation sets, and accumulate matched events over time. While I utilise a similar representation of the input data (=events), I work on a standard camera sensor. Compared to (ibid.), I present a probabilistic modelling and believe to be the first in exploiting the temporal coincidence approach for learning the statistics of stereo and image motion for real world camera setups. It is interesting to note that methods in which evidence on the sought entities is accumulated instead of directly being computed are often referred to as *Hough voting schemes* (Gall et al. 2011), in reference to the well-known Hough transform to detect lines, circles (Duda and Hart 1972; Hough 1962) but also arbitrarily shaped objects (D. H. Ballard 1981).

There are also frame based (=standard camera) approaches, which are related to the presented approach. I consider the work by Szlávik et al. in a series of papers (Szlávik et al. 2004, 2007; Szlávik and Szirányi 2006; Szlávik, Szirányi, and Havasi 2007; Szlávik, Szirányi, Havasi, and Benedek 2005) and based on that the work by Ermis et al. (Ermis 2010; Ermis et al. 2008) as the perhaps closest methods to the one presented by me.

Szlávik aims at finding point-to-point correspondences by detecting pixels in two views with similar motion change history. Specifically, Szlávik (Szlávik 2006) determines so called co-motion statistics between pairs of cameras. A co-motion is defined as the observation of concurrent motion at a fixed pixel in one view and all other pixels in the other view. These co-motions are accumulated over time and form so called remote (co-motion across two views) and local (co-motion within a single view) co-motion maps. Within each view, binary motion masks are computed based on background subtraction. Using a heuristic, these motion masks are subject to a post processing step using morphological operators to generate closed blobs for moving objects. Then, co-motions are recorded over the frames of long image sequences. Next, correspondence candidates are extracted from the remote maps as those pixel locations maximising the count of concurrent motion. Based on an engineered pipeline, the correspondence candidates are filtered and fine-tuned. This includes discarding candidates where in a local neighbourhood no, or only few other correspondence candidates can be found (principle of spatial coherence (Zhang et al. 1995)). Furthermore, pixels which show nearly continuous motion are discarded from the co-motion maps (=count is set to zero) as a pixel showing continuous motion would lead to high co-motion counts also for non-corresponding pixels. Obviously, when only considering the binary information of motion/no motion the discriminative power between corresponding and non-corresponding pixels vanishes, given that they show continuous motion. The remaining correspondence candidates are then fitted to specific geometric models, i.e., a homography or the fundamental matrix based on RANSAC (Fischler and Bolles 1981; Szlávik, Szirányi, and Havasi 2007), and a final set of correspondence is extracted and fine-tuned by an iterative process, which aims at minimising the symmetric transfer error. Beyond that, in (Szlávik, Szirányi, Havasi, and Benedek 2005) an entropy-based criterion is incorporated to reduce the number of false correspondences due to flashes or random noise on the background. As Szlávik aims at learning point-to-point correspondences only, their processing pipeline cannot handle scenes which deviate from the ground plane assumption. Therefore, in scenes where the moving objects will not lie on a plane, the algorithm is fed with the output from a shadow detector, as shadows usually lie within the ground plane. Note that the approach by Szlávik is robust to illumination difference among the views, at least to some extent. This is due to the fact that co-motions are detected based on the change mask only, but not on relative intensity differences. In contrast, I explicitly model illumination differences and introduce the concept of the *Grey Value Transfer Function* (cf. Sec. 3.6).

The work by Ermis (Ermis 2010) shares the basic processing steps with the approach by Szlávik. However, Ermis' primary interest lies in efficient information processing and communication within a camera network. He defines geometry independent features, which can be used for different tasks in multi-camera systems. These include correspondence estimation or abnormal behaviour detection. He introduces the *activity feature*, which boils down to the output of a change detector. The activity feature is detected at a specific pixel, given that the pixel location is subject to object motion, similar as the co-motion feature by Szlávik. Ermis attempts to give a formal definition of when the activity feature is scene/geometry independent. This is done under the idealised assumption that the observed scene essentially lies in a plane (=2-D). The definition of the activity feature is based on analysing the temporal duration of an activity feature in 3-D, when it is reprojected to separate camera views. Given that a 3-D point is visible in two views and that the scene depth is constant, the activity feature for corresponding pixels will be observable for the same amount of time. However, it is my understanding that this is actually directly given by epipolar geometry. The true correspondence for a pixel $\boldsymbol{x}_i$ in view $\mathcal{C}_i$ will be a pixel $\boldsymbol{z}_j$ in view $\mathcal{C}_j$, which will not change as long as the scene depth does not vary. Here, it does not matter whether one analyses the raw image data, or the output from a change detector.

Given that the depth profile of the scene is not essentially static (=3-D) the theory for the 2-D case will in general not hold. Ermis then defines varying occupancy durations and so called spurious activities. A spurious activity is generated when a signal change can only be seen from one of the cameras, due to an occluder in the other camera. To handle general 3-D (non-planar) scenes, the motion masks are subject to a so called aspect ratio normalisation. Here, a motion blob will be replaced with the bottom part of its enclosing bounding box, given that the height/width ratio is above some threshold. The aspect ratio normalisation uses higher level information (on the object level) rather then processing single pixels only, as I do. This also induces assumptions on the signal topology and the method could not be applied to, e.g., permuted sequences. In (Clarot et al. 2009), the approach is used in a camera network topology recovery task.

There are several aspects in which the TCA approach differs from the one by Szlávik and the one by Ermis, which will be discussed in the following. Firstly, I am interested in learning general mappings in multi-camera setups (=stereo) but also for single views (=motion). In contrast, Szlávik and Ermis focus on estimating point-to-point correspondences in multi-camera setups only.

In a general 3-D scene, pixels will in general not correspond to a single pixel in a second view, but to a set of pixels, as explained by n-view geometry. I accept this fact and my learning scheme can represent point-to-line correspondences. In contrast, Szlávik and Ermis need to handle general 3-D scenes as a special case and need to change the input data of their algorithms to the output of a shadow detector or the aspect ratio normalised change masks, respectively. In contrast, TCA works on the single pixel level and does not use any high level image information (shadows, interpretation of object masks). Furthermore, I provide a probabilistic model where with each correspondence a confidence measure is associated. This allows to detect false correspondences in a statistically principled way. Besides learning stereo correspondences, I will show that TCA may also be used to learn other mappings such as motion or to infer instantaneous motion by means of yaw rates (cf. Sec. 5).

To summarise, I do not aim at representing the epipolar relation by an algebraic expression, but my primary representation is a *correspondence distribution* estimated from the concurrent activity of single pixels. The method allows for arbitrary shapes of the distribution which may depend strongly on the relative sensor orientation, sensor and lens system geometry, and the type of typical motion. Of course, homographies or F-matrices can be computed from TCA results as well.

3.3 A Theory of Matching

In the following, I will formally derive the Temporal Coincidence Analysis (TCA) approach, based on a probabilistic model. I abstract from the matching of visual data and formulate the matching problem for two sets of unordered signal wires, among which corresponding signal wires are to be determined (see Fig. 3.8). Finally, the model can

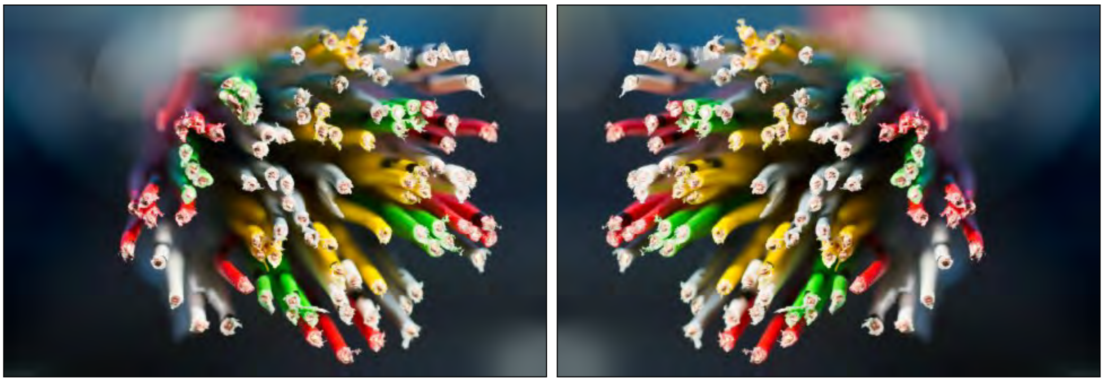


Figure 3.8: **Bunch of signals:** Given two sets of unordered signal wires, I derive a probabilistic model to determine corresponding signal wires. Image source: Deutsche Presse Agentur (dpa).

be instantiated for a specific application, e.g., in order to estimate motion or stereo correspondence distributions.

3.3.1 Definitions and Problem Statement

Let $h[n]$ and $g[n]$ be discrete sequences of random variables, with $n = 0, .., N$, also denoted as channels or signals in the following. I consider $h[n]$ and $g[n]$ to be the noisy observations from source sequences $s_h[n]$ and $s_g[n]$, where the source sequences may be transformed via deterministic functions ϕ_h and ϕ_g, respectively. The deterministic functions ϕ_h and ϕ_g depend on a set of parameters θ_h and θ_g, respectively. Let $v[n]$ and $w[n]$ be sequences modelling the observation noise, then the complete observation model is given as:

$$h[n] = \phi_h(s_h[n], \theta_h) + v[n], \tag{3.3}$$

$$g[n] = \phi_g(s_g[n], \theta_g) + w[n]. \tag{3.4}$$

Let $\mathrm{h}[n]$, $\mathrm{g}[n]$, $\mathrm{s_h}[n]$, $\mathrm{s_g}[n]$, $\mathrm{v}[n]$ and $\mathrm{w}[n]$ denote realisations of the previously defined random sequences. Next, assume that only the realisations of $h[n]$ and $g[n]$ are observable and that the individual source and noise sequences are hidden from us.

The problem addressed here can now be defined as follows: given an observed signal $\mathrm{g}[n]$ and a set of N independent observed signals $\mathcal{H} = \{\mathrm{h}_i[n]\}_N$, select a $\mathrm{h}_i[n] \in \mathcal{H}$, which maximises the probability that $\mathrm{g}[n]$ and $\mathrm{h}_i[n]$ are observations of the same source signal, with $\mathrm{s_{h_i}}[n] = \mathrm{s_g}[n]$. In other words, choose $\mathrm{h}_i[n] \in \mathcal{H}$ which most likely corresponds to $\mathrm{g}[n]$.

To this end, I will now derive a posterior distribution over the correspondence candidates. In order to keep the notation uncluttered, I may denote the observed channel $\mathrm{h}_i[n]$ as a vector $\boldsymbol{h}_i$, and similarly for the other involved variables. Hence, $\boldsymbol{h}_i$ is to be understood as a vector of observed signal values. I will also not index the involved distributions with the associated rv., as this will be clear from the context. For example, instead of $p_{\boldsymbol{h}}(\boldsymbol{h})$ and $p_{\boldsymbol{g}}(\boldsymbol{g})$, we may simply write $p(\boldsymbol{h})$ and $p(\boldsymbol{g})$, but note that the underlying pdfs are not necessarily the same. Note that even in the noise free case, I do not assume that the values of corresponding channels are the same, but that they are related by functions ϕ_h and ϕ_g. This is important, and will later allow us to represent, e.g., affine transformations of the input data to model illumination differences among views.

Let $c_i \in \boldsymbol{c}$ with $i = 1, .., N$ be the symbol denoting that observation signal $\boldsymbol{h}_i$ corresponds to observation signal $\boldsymbol{g}$, such that $\boldsymbol{g} \leftrightarrow \boldsymbol{h}_i$ holds. Let us now derive the posterior probability distribution $Pr[c_i|\{\boldsymbol{h}_j\}, \boldsymbol{g}]$ of c_i, given observation signal $\boldsymbol{g}$ and all candidate signals in $\mathcal{H}$. Here, the notation $\{\boldsymbol{h}_j\}$ is used to enumerate all elements of $\mathcal{H}$.

From the joint density $p(c_i, \{\boldsymbol{h}_j\}, \boldsymbol{g})$ and the basic rules of probability we obtain:

$$p(\{\boldsymbol{h}_j\}, \boldsymbol{g}|c_i) \cdot Pr[c_i] = p(c_i, \{\boldsymbol{h}_j\}, \boldsymbol{g}) = p(\{\boldsymbol{h}_j\}, \boldsymbol{g}) \cdot Pr[c_i|\{\boldsymbol{h}_j\}, \boldsymbol{g}], \tag{3.5}$$

$$\Rightarrow$$

$$Pr[c_i|\{\boldsymbol{h}_j\}, \boldsymbol{g}] = \frac{p(\{\boldsymbol{h}_j\}, \boldsymbol{g}|c_i) \cdot Pr[c_i]}{p(\{\boldsymbol{h}_j\}, \boldsymbol{g})}. \tag{3.6}$$

Note that the sought posterior is a discrete probability distribution, since the set of all c_i is discrete. Next, I will simplify 3.6 in several ways but will not make any specific assumptions about the actual distributions of the signals.

Note that the denominator in Eq. 3.6 is a constant as it does not depend on c_i. Equation 3.6 may thus be rewritten according to:

$$Pr[c_i|\{\boldsymbol{h}_j\}, \boldsymbol{g}] = \frac{p(\{\boldsymbol{h}_j\}, \boldsymbol{g}|c_i) \cdot Pr[c_i]}{\underbrace{p(\{\boldsymbol{h}_j\}, \boldsymbol{g})}_{=Z_1}}, \tag{3.7}$$

$$= \frac{1}{Z_1} \cdot p(\{\boldsymbol{h}_j\}, \boldsymbol{g}|c_i) \cdot Pr[c_i]. \tag{3.8}$$

Based on the assumption that the individual channels in $\mathcal{H}$ are independent of each other, we may rewrite Eq. 3.8 by expanding $\{\boldsymbol{h}_j\}$ and apply the product rule according to:

$$Pr[c_i|\{\boldsymbol{h}_j\}, \boldsymbol{g}] = \frac{1}{Z_1} \cdot p(\{\boldsymbol{h}_j\}, \boldsymbol{g}|c_i) \cdot Pr[c_i], \tag{3.9}$$

$$= \frac{1}{Z_1} \cdot p(\boldsymbol{h}_1, ..., \boldsymbol{h}_n, \boldsymbol{g}|c_i) \cdot Pr[c_i], \tag{3.10}$$

$$= \frac{1}{Z_1} \cdot p(\boldsymbol{h}_1, ..., \boldsymbol{h}_{n-1}, \boldsymbol{g}, |\boldsymbol{h}_n, c_i) \cdot \underbrace{p(\boldsymbol{h}_n|c_i)}_{=p(\boldsymbol{h}_n) \text{ for all } j \neq i} \cdot Pr[c_i]. \tag{3.11}$$

Due to the assumption that the candidate channels are independent of each other, we see that the term $p\left(\boldsymbol{h}_n|c_i\right)$ is independent of c_i for all $j \neq i$ and simplifies to $p\left(\boldsymbol{h}_n\right)$. For the same reason, the term $p\left(\boldsymbol{h}_1, ..., \boldsymbol{h}_{n-1}, \boldsymbol{g}, |\boldsymbol{h}_n, c_i\right)$ in Eq. 3.11 may be simplified to $p\left(\boldsymbol{h}_1, ..., \boldsymbol{h}_{n-1}, \boldsymbol{g}, |c_i\right)$.

Using the same reasoning for all candidate channels, we may rewrite Eq. 3.11 according to:

$$\begin{aligned}
Pr[c_i|\{\boldsymbol{h}_j\}, \boldsymbol{g}] &= \frac{1}{Z_1} \cdot p\left(\boldsymbol{h}_1, ..., \boldsymbol{h}_{n-1}, \boldsymbol{g}, |\boldsymbol{h}_n, c_i\right) \cdot p\left(\boldsymbol{h}_n|c_i\right) \cdot Pr[c_i], &\quad (3.12)\\
&= \frac{1}{Z_1} \cdot p\left(\boldsymbol{h}_1, ..., \boldsymbol{h}_{n-1}, \boldsymbol{g}, |c_i\right) \cdot p\left(\boldsymbol{h}_n\right) \cdot Pr[c_i], &\quad (3.13)\\
&= \frac{1}{Z_1} \cdot p\left(\boldsymbol{h}_i, \boldsymbol{g}|c_i\right) \cdot \prod_{j, j\neq i} p\left(\boldsymbol{h}_j\right) \cdot Pr[c_i]. &\quad (3.14)
\end{aligned}$$

The posterior of c_i in Eq. 3.14 depends on the product term over the candidate channels, which obviously is different for each $c_i \in \boldsymbol{c}$. However, we may relax this dependence into a constant term as follows. We multiply Eq. 3.14 with $\frac{p(\boldsymbol{h}_i)}{p(\boldsymbol{h}_i)}$ and rearrange according to:

$$\begin{aligned}
Pr[c_i|\{\boldsymbol{h}_j\}, \boldsymbol{g}] &= \frac{1}{Z_1} \cdot p\left(\boldsymbol{h}_i, \boldsymbol{g}|c_i\right) \cdot \prod_{j, j\neq i} p\left(\boldsymbol{h}_j\right) \cdot Pr[c_i], &\quad (3.15)\\
&= \frac{1}{Z_1} \cdot \frac{p\left(\boldsymbol{h}_i\right)}{p\left(\boldsymbol{h}_i\right)} \cdot p\left(\boldsymbol{h}_i, \boldsymbol{g}|c_i\right) \cdot \prod_{j, j\neq i} p\left(\boldsymbol{h}_j\right) \cdot Pr[c_i], &\quad (3.16)\\
&= \frac{1}{Z_1} \cdot \frac{1}{p\left(\boldsymbol{h}_i\right)} \cdot p\left(\boldsymbol{h}_i, \boldsymbol{g}|c_i\right) \prod_{j, j\neq i} p\left(\boldsymbol{h}_j\right) \cdot p\left(\boldsymbol{h}_i\right) \cdot Pr[c_i], &\quad (3.17)\\
&= \frac{1}{Z_1} \cdot \frac{1}{p\left(\boldsymbol{h}_i\right)} \cdot p\left(\boldsymbol{h}_i, \boldsymbol{g}|c_i\right) \underbrace{\prod_{j} p\left(\boldsymbol{h}_j\right)}_{\frac{1}{Z_2}} \cdot Pr[c_i], &\quad (3.18)\\
&= \frac{1}{Z_1} \cdot \frac{1}{Z_2} \cdot \frac{p\left(\boldsymbol{h}_i, \boldsymbol{g}, |c_i\right)}{p\left(\boldsymbol{h}_i\right)} \cdot Pr[c_i]. &\quad (3.19)
\end{aligned}$$

Finally, we may rearrange and further simplify Eq. 3.19 to obtain:

$$\begin{aligned}
Pr[c_i|\{\boldsymbol{h}_j\},\boldsymbol{g}] &= \frac{1}{Z_1}\cdot\frac{1}{Z_2}\cdot\frac{p\left(\boldsymbol{h}_i,\boldsymbol{g},|c_i\right)}{p\left(\boldsymbol{h}_i\right)}\cdot Pr[c_i], && (3.20)\\
&= \frac{1}{Z_1}\cdot\frac{1}{Z_2}\cdot\frac{p\left(\boldsymbol{g}|\boldsymbol{h}_i,c_i\right)\cdot p\left(\boldsymbol{h}_i|c_i\right)}{p\left(\boldsymbol{h}_i\right)}\cdot Pr[c_i], && (3.21)\\
&= \frac{1}{Z_1}\cdot\frac{1}{Z_2}\cdot\frac{p\left(\boldsymbol{g}|\boldsymbol{h}_i\right)\cdot p\left(\boldsymbol{h}_i\right)}{p\left(\boldsymbol{h}_i\right)}\cdot Pr[c_i], && (3.22)\\
&= \frac{1}{Z_1}\cdot\frac{1}{Z_2}\cdot p\left(\boldsymbol{g}|\boldsymbol{h}_i\right)\cdot Pr[c_i]. && (3.23)
\end{aligned}$$

Let us now derive the conditional density $p\left(\boldsymbol{g}|\boldsymbol{h}_i\right)$ from Eq. 3.23. To keep the notation uncluttered, I may drop index i of $\boldsymbol{h}_i$ but note that I will assume from now on that channels $\boldsymbol{g}$ and $\boldsymbol{h}$ correspond. From the joint distribution $p\left(\boldsymbol{h},\boldsymbol{g}\right)$ we have:

$$\begin{aligned}
p\left(\boldsymbol{h},\boldsymbol{g}\right) &= p\left(\boldsymbol{g}|\boldsymbol{h}\right)\cdot p\left(\boldsymbol{h}\right), && (3.24)\\
&\Rightarrow\\
p\left(\boldsymbol{g}|\boldsymbol{h}\right) &= \frac{p\left(\boldsymbol{h},\boldsymbol{g}\right)}{p\left(\boldsymbol{h}\right)}. && (3.25)
\end{aligned}$$

Let us now derive the joint distribution $p\left(\boldsymbol{h},\boldsymbol{g}\right)$. According to the observation model in Eq. 3.4 we know that the joint distribution depends on the hidden source signal $\boldsymbol{s}$. Thus $p\left(\boldsymbol{h},\boldsymbol{g}\right)$ is given by marginalising the joint distribution $p\left(\boldsymbol{h},\boldsymbol{g},\boldsymbol{s}\right)$ over $\boldsymbol{s}$ according to:

$$p\left(\boldsymbol{h},\boldsymbol{g}\right) = \int_{\boldsymbol{s}} p\left(\boldsymbol{h},\boldsymbol{g},\boldsymbol{s}\right)\mathrm{d}\boldsymbol{s}. \tag{3.26}$$

Next, we expand Eq. 3.26 as follows:

$$\begin{aligned}
p\left(\boldsymbol{h},\boldsymbol{g}\right) &= \int_{\boldsymbol{s}} p\left(\boldsymbol{h},\boldsymbol{g},\boldsymbol{s}\right)\mathrm{d}\boldsymbol{s}, && (3.27)\\
&= \int_{\boldsymbol{s}} p\left(\boldsymbol{h},\boldsymbol{g}|\boldsymbol{s}\right)\cdot p\left(\boldsymbol{s}\right)\mathrm{d}\boldsymbol{s}, && (3.28)\\
&= \int_{\boldsymbol{s}} p\left(\boldsymbol{h}|\boldsymbol{g},\boldsymbol{s}\right)\cdot p\left(\boldsymbol{g}|\boldsymbol{s}\right)\cdot p\left(\boldsymbol{s}\right)\mathrm{d}\boldsymbol{s}, && (3.29)\\
&= \int_{\boldsymbol{s}} p\left(\boldsymbol{h}|\boldsymbol{s}\right)\cdot p\left(\boldsymbol{g}|\boldsymbol{s}\right)\cdot p\left(\boldsymbol{s}\right)\mathrm{d}\boldsymbol{s}, && (3.30)
\end{aligned}$$

where the transition from Eq. 3.29 to Eq. 3.30 is due to the conditional independence of $\boldsymbol{h}$ and $\boldsymbol{g}$ when $\boldsymbol{s}$ is *known* and is assumed to be the true cause of observing $\boldsymbol{h}$ and $\boldsymbol{g}$.

Furthermore, if we assume that $\boldsymbol{s}$ is *known*, then observations $\boldsymbol{h}$ and $\boldsymbol{g}$ are distributed according to the assumed noise distribution, such that we have:

$$p\left(\boldsymbol{h}|\boldsymbol{s}\right) = p\left(\boldsymbol{v} = \boldsymbol{h} - \boldsymbol{s}\right), \tag{3.31}$$

$$p\left(\boldsymbol{g}|\boldsymbol{s}\right) = p\left(\boldsymbol{w} = \boldsymbol{g} - \boldsymbol{s}\right). \tag{3.32}$$

Finally, we obtain the joint distribution as:

$$p\left(\boldsymbol{h}, \boldsymbol{g}\right) = \int_{\boldsymbol{s}} p\left(\boldsymbol{v} = \boldsymbol{h} - \boldsymbol{s}\right) \cdot p\left(\boldsymbol{w} = \boldsymbol{g} - \boldsymbol{s}\right) \cdot p\left(\boldsymbol{s}\right) \mathrm{d}\boldsymbol{s}. \tag{3.33}$$

Inserting the joint distribution from Eq. 3.33 in the conditional distribution in Eq. 3.25, we obtain:

$$p\left(\boldsymbol{g}|\boldsymbol{h}\right) = \frac{\int_{\boldsymbol{s}} p\left(\boldsymbol{h}|\boldsymbol{s}\right) \cdot p\left(\boldsymbol{g}|\boldsymbol{s}\right) \cdot p\left(\boldsymbol{s}\right) \mathrm{d}\boldsymbol{s}}{p\left(\boldsymbol{h}\right)}. \tag{3.34}$$

If we assume the observation signals to be noise free, the pdfs $p\left(\boldsymbol{h}\right)$ and $p\left(\boldsymbol{g}\right)$ are simply given by the pdf of the source signal according to:

$$p\left(\boldsymbol{h}\right) = p\left(\boldsymbol{s}\right), \tag{3.35}$$

$$p\left(\boldsymbol{g}\right) = p\left(\boldsymbol{s}\right). \tag{3.36}$$

I stress that the densities for noise free observations are defined only for completeness. If for real world data, the observations would be noise free, the matching problem would be trivial and any correspondence candidate could be excluded upon the first observation of differing symbols.

For the case of noise afflicted observations, we obtain the pdfs $p\left(\boldsymbol{h}\right)$ and $p\left(\boldsymbol{g}\right)$ again by marginalisation over the source signal:

$$\begin{aligned} p\left(\boldsymbol{h}\right) &= \int_{\boldsymbol{s}_{\boldsymbol{h}}} p\left(\boldsymbol{h}, \boldsymbol{s}_{\boldsymbol{h}}\right) \mathrm{d}\boldsymbol{s}_{\boldsymbol{h}}, && (3.37)\\ &= \int_{\boldsymbol{s}_{\boldsymbol{h}}} p\left(\boldsymbol{h}|\boldsymbol{s}_{\boldsymbol{h}}\right) \cdot p\left(\boldsymbol{s}_{\boldsymbol{h}}\right) \mathrm{d}\boldsymbol{s}_{\boldsymbol{h}}, && (3.38)\\ &= \int_{\boldsymbol{s}_{\boldsymbol{h}}} p\left(\boldsymbol{v} = \boldsymbol{h} - \boldsymbol{s}_{\boldsymbol{h}}\right) \cdot p\left(\boldsymbol{s}_{\boldsymbol{h}}\right) \mathrm{d}\boldsymbol{s}_{\boldsymbol{h}}, && (3.39) \end{aligned}$$

and

$$\begin{aligned} p\left(\boldsymbol{g}\right) &= \int_{\boldsymbol{s}_{\boldsymbol{g}}} p\left(\boldsymbol{g}, \boldsymbol{s}_{\boldsymbol{g}}\right) \mathrm{d}\boldsymbol{s}_{\boldsymbol{g}}, && (3.40)\\ &= \int_{\boldsymbol{s}_{\boldsymbol{g}}} p\left(\boldsymbol{g}|\boldsymbol{s}_{\boldsymbol{g}}\right) \cdot p\left(\boldsymbol{s}_{\boldsymbol{g}}\right) \mathrm{d}\boldsymbol{s}_{\boldsymbol{g}}, && (3.41)\\ &= \int_{\boldsymbol{s}_{\boldsymbol{g}}} p\left(\boldsymbol{w} = \boldsymbol{g} - \boldsymbol{s}_{\boldsymbol{g}}\right) \cdot p\left(\boldsymbol{s}_{\boldsymbol{g}}\right) \mathrm{d}\boldsymbol{s}_{\boldsymbol{g}}. && (3.42) \end{aligned}$$

For completeness, let us also derive the joint distribution for non-corresponding channels. The joint pdf $p(\boldsymbol{h}, \boldsymbol{g})$ of *independent* observation signals $\boldsymbol{h}$ and $\boldsymbol{g}$ is given according to:

$$
\begin{aligned}
p(\boldsymbol{h}, \boldsymbol{g}) &= p(\boldsymbol{h}) \cdot p(\boldsymbol{g}), && (3.43)\\
&= \int_{\boldsymbol{s_h}} p(\boldsymbol{h}, \boldsymbol{s_h}) \, \mathrm{d}\boldsymbol{s_h} \cdot \int_{\boldsymbol{s_g}} p(\boldsymbol{g}, \boldsymbol{s_g}) \, \mathrm{d}\boldsymbol{s_g}, && (3.44)\\
&= \int_{\boldsymbol{s_h}} p(\boldsymbol{h}|\boldsymbol{s_h}) \cdot p(\boldsymbol{s_h}) \, \mathrm{d}\boldsymbol{s_h} \cdot \int_{\boldsymbol{s_g}} p(\boldsymbol{g}|\boldsymbol{s_g}) \cdot p(\boldsymbol{s_g}) \, \mathrm{d}\boldsymbol{s_g}, && (3.45)\\
&= \int_{\boldsymbol{s_h}} p(\boldsymbol{v} = \boldsymbol{h} - \boldsymbol{s_h}) \cdot p(\boldsymbol{s_h}) \, \mathrm{d}\boldsymbol{s_h} \cdot \int_{\boldsymbol{s_g}} p(\boldsymbol{w} = \boldsymbol{g} - \boldsymbol{s_g}) \cdot p(\boldsymbol{s_g}) \, \mathrm{d}\boldsymbol{s_g}. && (3.46)
\end{aligned}
$$

The derivation of the posterior distribution in Eq. 3.6 is now complete. In order to instantiate the model and to compute the posterior, we now have to specify the individual distributions and evaluate Eq. 3.23 for each candidate channel. The computation of the posterior becomes extremely easy if we assume that the individual signals are represented in their *innovations* representation. This means that we regard each signal as a sequence of independent and identically distributed symbols.

Then, the posterior distribution factors and may be updated sequentially at each time step, as will be shown next.

3.3.2 Temporal Update Scheme

The main result I derive and obtain in the following is a temporal update scheme for the posterior distribution $Pr[c_i|\{\boldsymbol{h}_j\}, \boldsymbol{g}]$. I assume that all channels are given in (or are converted into) their innovations representation and are i.i.d. sequences. This implies that we may update the posterior sequentially.

Let us now assume that we observe one symbol per signal and time step t. The posterior distribution of c_i, given observations $\mathrm{h}_j[n = t]$ and $\mathrm{g}[n = t]$ at time t is then given as:

$$
Pr_{[t]}\left[c_i|\{\mathrm{h}_j[n = t]\}, \mathrm{g}[n = t]\right] = \frac{1}{Z_1} \cdot \frac{1}{Z_2} \cdot p\left(\mathrm{g}[n = t]|\mathrm{h}_i[n = t]\right) \cdot Pr_{[t]}\left[c_i\right], \qquad (3.47)
$$

with

$$
\sum_i Pr_{[t]}\left[c_i|\{\mathrm{h}[n]_{j,t}\}, \mathrm{g}[n]_t\right] = 1. \qquad (3.48)
$$

The term $Pr_{[t]}\left[c_i\right]$ in Eq. 3.47 represents the prior probability or our prior belief that the signal $\boldsymbol{g}$ and the i'th candidate channel correspond. At time t, we want to incorporate our knowledge about the posterior distribution from previous time steps. To this end, the posterior distribution of c_i at time $t-1$ is used as the prior distribution over c_i at time t. Initially, before any signal observations are available, we have no preference for a specific channel. Therefore, we represent the prior at time step $t = 0$ (=init) by means of a flat (=uniform) prior with:

$$Pr_{[init]}\left[c_i\right] = \frac{1}{N}, \tag{3.49}$$

where N denotes the number of correspondence candidates.

We may now iteratively update both the posterior distribution and the prior distribution as follows. At time step $t = 0$ we have:

$$Pr_{[0]}\left[c_i|\{\mathrm{h}_j[n=0]\},\mathrm{g}[n=0]\right] = \frac{1}{Z_1}\cdot\frac{1}{Z_2}\cdot p\left(\mathrm{g}[n=0]|\mathrm{h}_i[n=0]\right)\cdot Pr_{[init]}\left[c_i\right]. \tag{3.50}$$

At time t we have:

$$Pr_{[t]}\left[c_i|\{\mathrm{h}_j[n=t]\},\mathrm{g}[n=t]\right] = \frac{1}{Z_1}\cdot\frac{1}{Z_2}\cdot p\left(\mathrm{g}[n=t]|\mathrm{h}_i[n=t]\right)\cdot Pr_{[t]}\left[c_i\right], \tag{3.51}$$

with

$$Pr_{[t]}\left[c_i\right] = Pr_{[t-1]}\left[c_i|\{\mathrm{h}_j[n=t-1]\},\mathrm{g}[n=t-1]\right]. \tag{3.52}$$

Note that there is no need to explicitly solve for the normalisation constants Z_1 and Z_2, in order to guarantee that the posterior distribution at time t is properly normalised. First, we observe that the prior distribution of $\boldsymbol{c}$ at time $t-1$ is normalised by definition (see initialisation step). The posterior distribution at time t is then normalised by dividing each $Pr_{[t]}\left[c_i|\{\mathrm{h}_j[n=t]\},\mathrm{g}[n=t]\right]$ by the sum of the non-normalised posterior:

$$\sum_i Pr_{[t]}\left[c_i|\{\mathrm{h}_j[n=t]\},\mathrm{g}[n=t]\right] = 1, \tag{3.53}$$

$$\sum_i \frac{1}{Z_1}\cdot\frac{1}{Z_2}\cdot p\left(\mathrm{g}[n=t]|\mathrm{h}_i[n=t]\right)\cdot Pr_{[t]}\left[c_i\right] = 1, \tag{3.54}$$

$$\frac{1}{Z_1\cdot Z_2}\sum_i p\left(\mathrm{g}[n=t]|\mathrm{h}_i[n=t]\right)\cdot Pr_{[t]}\left[c_i\right] = 1, \tag{3.55}$$

$$\sum_i p\left(\mathrm{g}[n=t]|\mathrm{h}_i[n=t]\right)\cdot Pr_{[t]}\left[c_i\right] = Z_1\cdot Z_2. \tag{3.56}$$

To summarise, the posterior distribution at time t is properly normalised by setting the product of $Z_1 \cdot Z_2$ to the sum of the non-normalised posterior.

Due to the multiplication of potentially very small numbers, the update procedure may run into numerical instabilities. This may be circumvented by computing the posterior distribution in the log domain. Taking the log of Eq. 3.51 we obtain:

$$\log\left[Pr_{[t]}\left[c_i|\{\mathrm{h}_j[n=t]\},\mathrm{g}[n=t]\right]\right] = \log\left[\frac{1}{Z_1}\cdot\frac{1}{Z_2}\cdot p\left(\mathrm{g}[n=t]|\mathrm{h}_i[n=t]\right)\cdot Pr_{[t]}\left[c_i\right]\right], \tag{3.57}$$

$$= -\log Z_1 - \log Z_2 + \log\left[p\left(\mathrm{g}[n=t]|\mathrm{h}_i[n=t]\right)\right] + \log\left[Pr_{[t]}\left[c_i\right]\right]. \tag{3.58}$$

However, to compute the posterior in the log domain, we would now have to explicitly solve for the normalisation constants Z_1 and Z_2. Instead, I propose to approximate the posterior as:

$$\log\left[Pr_{[t]}\left[c_i|\{\mathrm{h}_j[n=t]\},\mathrm{g}[n=t]\right]\right] = -\log Z_1 - \log Z_2 + \log\left[p\left(\mathrm{g}[n=t]|\mathrm{h}_i[n=t]\right)\right] + \log\left[Pr_{[t]}\left[c_i\right]\right], \tag{3.59}$$

$$\to \underbrace{\log\left[p\left(\mathrm{g}[n=t]|\mathrm{h}_i[n=t]\right)\right]}_{\text{data term}} + \underbrace{\log\left[Pr_{[t]}\left[c_i\right]\right]}_{\text{accumulator}}. \tag{3.60}$$

The approximate model may be updated sequentially as well. The data term in Eq. 3.60 will be evaluated at each time instant and its result is added to the prior term, i.e., the posterior from the previous time step. The prior term at time t may be interpreted as an accumulator, containing the summed data terms from previous time steps. I implement the data term as a threshold operation based on a similarity measure of pairs of signal values. Given that m candidate signals pass the threshold test at time t, I assign each candidate channel a probability of $\frac{1}{m}$ of being the true correspondence. I then update the accumulator cell indexed by the candidate channel i as:

$$\boldsymbol{A}_t[i] = \begin{cases} \log(\frac{1}{m}) + \boldsymbol{A}_{t-1}[i], & \text{threshold test passed,} \\ \boldsymbol{A}_{t-1}[i], & \text{else.} \end{cases} \tag{3.61}$$

Due to $\log(\frac{1}{m}) = \log(1) - \log(m) = -\log(m)$, we may update the accumulator with positive weights given by $\log(m)$ which only changes the sign.

3.3.3 Posterior Entropy and Conditional Update Scheme

Previously, we have seen that the channel matching problem, as defined in Sec. 3.3.1, may be formulated by means of a probabilistic model which leads to a posterior distribution over the correspondence candidates. The most likely corresponding channel is then given as the i-th channel, for which the posterior distribution attains its maximum value.

Let us now assume that the system which implements the posterior update scheme has to operate under an energy constraint. I will assume that each posterior update is associated with a constant cost term Q_p and the goal is to learn the true correspondence, while minimising the consumed energy. To this end, we first have to answer the question when does the posterior encode a reliable estimate of the true corresponding channel, or in other words, when have we learnt enough? I address this question by means of basic results of information theory. Specifically, I will regard the *entropy* of the posterior distribution, as described in the following.

Let C be a discrete random variable, which may take on values in the domain $[1, .., |\mathcal{H}|]$. Let C be distributed according to the posterior distribution derived previously with:

$$f_C\left[c_i\right] = Pr[C = c_i | \{\boldsymbol{h}_j\}, \boldsymbol{g}]. \tag{3.62}$$

We will now regard the entropy of random variable C, which is a measure of its uncertainty (Cover and Joy Thomas 2009, p. 13). The entropy of C under the posterior distribution is given as (ibid., p. 14):

$$H(C) = \mathbb{E}\left[\log_2 \frac{1}{f_C\left[c_i\right]}\right] = -\sum_{c_i} f_C\left[c_i\right] \cdot \log_2 f_C\left[c_i\right]. \tag{3.63}$$

As the $\log_2$ is taken w.r.t. base 2, the entropy is measured in bits (ibid., p. 14). If rv. C is uniformly distributed, every possible outcome (event) is equally likely, hence, the uncertainty about an unobserved event is maximum. Therefore, the entropy will be maximum for a uniformly distributed rv.. The entropy will be minimum, given that the probability mass concentrates at a single event as the uncertainty about an unobserved event is minimum.

According to Eq. 3.49, the posterior will be initialised as a uniform distribution. As each of the N candidate channels *may* be the true corresponding channel with a probability of $\frac{1}{N}$, the *uncertainty* about the true correspondence is maximum. Hence, the entropy of C is maximum and is given by:

$$\begin{aligned} H(C)_{init} &= -\sum_{c_i} f_C\left[c_i\right] \cdot \log f_C\left[c_i\right], && (3.64) \\ &= -\sum_{c_i} \frac{1}{N} \cdot \log_2 \frac{1}{N}, && (3.65) \\ &= -\sum_{c_i} \frac{1}{N} \cdot (\log_2(1) - \log_2 N), && (3.66) \\ &= -\sum_{c_i} \frac{-\log_2 N}{N}, && (3.67) \\ &= -N \cdot \frac{-\log_2 N}{N}, && (3.68) \\ &= \log_2 N. && (3.69) \end{aligned}$$

We may now address the question, when we have learnt enough. When the posterior is updated over time, we expect to *learn* or gain information about the true correspondence, such that the uncertainty decreases. This means that the entropy of the posterior distribution should decrease. As already mentioned, the entropy will be minimum (=0) when the probability mass concentrates at a single event, i.e., when the probability for one of the candidate channels is 1. Of course, this has to be regarded as a theoretical lower limit, which will rarely be achievable in practice, due to signal noise.

Based on the entropy, we may also asses how *much* we have learnt about the correspondence after each posterior update. This allows to differentiate between 'useful' and 'useless' posterior updates. To this end, I regard the difference of the posterior's entropy between two time steps, defined as:

$$\Delta H(C) = H(C)_t - H(C)_{t+1}. \tag{3.70}$$

The larger this difference is, the more we learn about the correspondence. The smaller the difference, the less we learn about the correspondence and we may completely discard this update and simultaneously save energy.

3.3.4 Instantiating the Model - Gaussian Case

Let us now instantiate the model derived in Sec. 3.3.1. To this end, we have to specify the source signal and noise distribution and derive the functional form of the posterior in Eq. 3.51. For the moment being, I assume that the deterministic mapping functions ϕ_h and ϕ_g represent the identify function. We may also assume that these mappings have been compensated already, and that uncertainties of this process are normally distributed which may then be subsumed into the observation noise term.

Recall from Sec. 3.3.1 that I assume the signals to be sequences of i.i.d. symbols. To keep the notation uncluttered, I may drop the index $[n]$ from all involved signals and write s, h etc. to denote these scalar random variables.

I assume that the source signal is Gaussian distributed according to:

$$p(s) \sim \mathcal{N}(0, \sigma_s^2). \tag{3.71}$$

The conditional distributions of signals h and g, given an observed source signal correspond to the noise distributions, which are assumed to be Gaussian with mean 0 and channel specific variances:

$$p(h|s) \sim \mathcal{N}(s, \sigma_{h|s}^2), \tag{3.72}$$

$$p(g|s) \sim \mathcal{N}(s, \sigma_{g|s}^2). \tag{3.73}$$

Clearly, $\sigma_{h|s}^2$ and $\sigma_{g|s}^2$ are given as the assumed noise variances σ_v^2 and σ_w^2, respectively.

The distributions of h and g are then given by marginalising the joint distributions as given in Eq. 3.39 and Eq. 3.42 as:

$$\begin{aligned} p(h) &= \int_{s_h} p(h|s_h) \cdot p(s_h)\, \mathrm{d}s_h, \\ &= \frac{1}{\sqrt{2 \cdot \pi \cdot (\sigma_{h|s}^2 + \sigma_s^2)}} \cdot \exp\left(-\frac{h^2}{2 \cdot (\sigma_{h|s}^2 + \sigma_s^2)}\right), \\ &\sim \mathcal{N}(0, \sigma_{h|s}^2 + \sigma_s^2), \end{aligned} \tag{3.74}$$

and similarly for $p(g)$:

$$p(g) \sim \mathcal{N}(0, \sigma_{g|s}^2 + \sigma_s^2). \tag{3.75}$$

Next, let us derive the conditional distribution $p(g|h)$ as is needed to evaluate Eq. 3.51. We start with the derivation of $p(g, h)$. As shown in Eq. 3.33, we have to marginalise the joint distribution $p(g, h_i, s)$ w.r.t. s. For the Gaussian model we obtain[1]:

$$p(h_i, g|c_i) = \frac{\exp\left(-\frac{h^2\sigma_{g|s}^2 + \sigma_s^2(g-h)^2 + g^2\sigma_{h|s}^2}{2\sigma_{g|s}^2\left(\sigma_{h|s}^2 + \sigma_s^2\right) + 2\sigma_{h|s}^2\sigma_s^2}\right)}{2\pi\sqrt{\sigma_{g|s}^2\left(\sigma_{h|s}^2 + \sigma_s^2\right) + \sigma_{h|s}^2\sigma_s^2}}. \tag{3.76}$$

---

[1] via pen and paper or with a computer algebra program as is done here

The joint distribution in Eq. 3.76 is again Gaussian. We can determine the mean and covariance of this distribution, e.g., via 'completing the square' (cf. (Bishop 2006, p. 86)). To this end, we regard the term in the exponential of Eq. 3.76 and rearrange to obtain the canonical form of the exponential term of a multivariate Gaussian. In our case, this is given as (ibid., p. 86):

$$-\frac{1}{2} \cdot \begin{bmatrix} h & g \end{bmatrix} \Sigma^{-1} \begin{bmatrix} h \\ g \end{bmatrix} + \begin{bmatrix} h & g \end{bmatrix} \Sigma^{-1} \begin{bmatrix} \mu_h \\ \mu_g \end{bmatrix} + \text{const.} \tag{3.77}$$

As the exponential in Eq. 3.76 contains no linear terms in h or g, we have that $\mu_h = \mu_g = 0$. For the quadratic form, we determine the precision matrix Σ^{-1} and invert it to obtain the covariance matrix:

$$\Sigma = \begin{bmatrix} \Sigma_{hh} & \Sigma_{hg} \\ \Sigma_{gh} & \Sigma_{gg} \end{bmatrix} = \begin{bmatrix} \sigma^2_{h|s} + \sigma^2_s & \sigma^2_s \\ \sigma^2_s & \sigma^2_{g|s} + \sigma^2_s \end{bmatrix}. \tag{3.78}$$

Finally, the joint distribution in 3.76 is given as:

$$p\left(h_i, g|c\right) \sim \mathcal{N}\left(\begin{bmatrix} h \\ g \end{bmatrix}; \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} \sigma^2_{h|s} + \sigma^2_s & \sigma^2_s \\ \sigma^2_s & \sigma^2_{g|s} + \sigma^2_s \end{bmatrix}\right). \tag{3.79}$$

Based on standard results for the multivariate Gaussian distribution, we obtain the conditional distribution $p\left(g|h_i\right)$, which will be Gaussian again. The conditional mean and variance are given as (ibid., p. 87):

$$\mu_{g|h} = \mu_g + \Sigma_{gh}\Sigma^{-1}_{hh}(h - \mu_h), \tag{3.80}$$

$$\Sigma_{g|h} = \Sigma_{gg} - \Sigma_{gh}\Sigma^{-1}_{hh}\Sigma_{hg}, \tag{3.81}$$

and hence:

$$p\left(g|h\right) \sim \mathcal{N}\left(\begin{bmatrix} h \\ g \end{bmatrix}; \frac{\sigma^2_s}{\sigma^2_{h|s} + \sigma^2_s} \cdot h, \sigma^2_{g|s} + \frac{\sigma^2_{h|s} \cdot \sigma^2_s}{\sigma^2_{h|s} + \sigma^2_s}\right). \tag{3.82}$$

$$\tag{3.83}$$

For the following discussion, we will also need the joint distribution of non-corresponding channels. This distribution is given according to Eq. 3.46. Under the Gaussian model, this distribution has a diagonal covariance matrix and is given as:

$$p\left(h, g|h \not\leftrightarrow g\right) \sim \mathcal{N}\left(\begin{bmatrix} h \\ g \end{bmatrix}; \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} \sigma^2_{h|s} + \sigma^2_s & 0 \\ 0 & \sigma^2_{g|s} + \sigma^2_s \end{bmatrix}\right). \tag{3.84}$$

Let us now regard Fig. 3.9, which visualises prototypical joint pdfs for non-corresponding and corresponding channels under the Gaussian model.

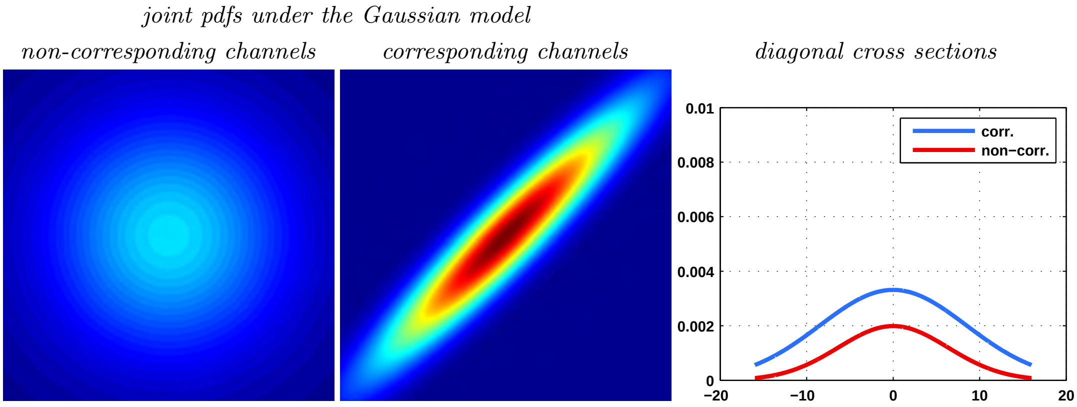


Figure 3.9: **Joint distributions under the Gaussian model:** Prototypical joint distributions of (left) non-corresponding and (middle) corresponding channels. (right) Cross section of the pdfs. See text for details.

Parameters were set to $\sigma_s^2 = 8, \sigma_{h|s}^2 = \sigma_{g|s}^2 = 4$. Note that the plots of the joint pdfs were set to the same scale. As expected, the joint pdf for independent channels is isotropic, while the joint pdf for dependent channels is anisotropic. We see that the relative propensity of observing largely *different* symbols is larger for independent channels than for corresponding channels. Furthermore, we see that the relative propensity of *simultaneously* observing two symbols at the low or high end of the observable intervals is considerably larger for corresponding channels than for independent channels. This can also be seen from the diagonal cross sections of the pdfs in Fig. 3.9 (right). When we update the posterior distribution, the model always assumes that the regarded channels correspond and only the joint pdf for corresponding channels is regarded (or the conditional derived thereof). However, if the channels do not correspond, we expect to observe largely different symbols, and the model will assign only a small probability to them. From this, we may conclude that the probability of observing two *rare* signal values at the same time is larger for corresponding channels than for non-corresponding channels.

Figure 3.10 shows the distributions $p\,(s)$, $p\,(h)$ and $p\,(g)$ for $\sigma_s^2 = 16$ and $\sigma_{h|s}^2 = \sigma_{g|s}^2 = 4$. Now assume that we draw a sample from the hidden source signal with $s = -8$. Figure 3.10 also shows the conditional distribution $p\,(h|s = -8)$. If the channels h and g truly correspond, the conditional pdf $p\,(g|h)$ will be large for values of g which are in the vicinity of h. However, if the channels do not correspond, the true conditional pdf $p\,(g|h)$ becomes $p\,(g)$, which we see is centred around 0. From these plots, we see that the relative propensity of observing extreme values of s is higher for corresponding channels, than for non-corresponding channels.

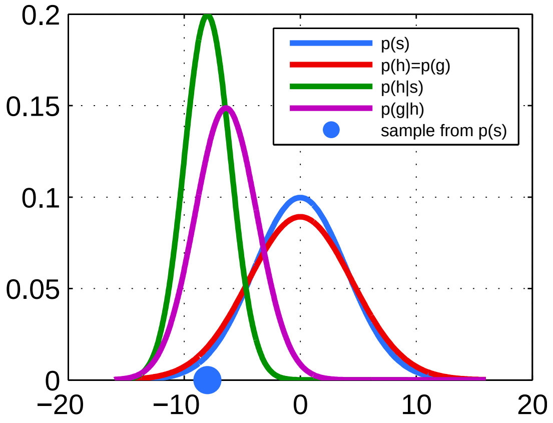


Figure 3.10: **Exemplary distributions under the Gaussian model:** For $\sigma_s^2 = 16$ and $\sigma_{h|s}^2 = \sigma_{g|s}^2 = 4$. A signal value $s = -8$ is sampled. See text for details. Figure only interpretable when viewed in colour.

Let us now return to the discussion on how to decrease the entropy of the posterior distribution (cf. Sec. 3.3.3). In order to decrease the entropy of the posterior distribution, we should regard rare events. The probability of observing rare events will be larger for corresponding channels than for non-corresponding channels, simultaneously we minimise the probability for a false match.

3.3.5 Simulation and Comparison to Correlation

Let us now simulate the model derived in Sec. 3.3.4 as follows. We regard a seed channel $\boldsymbol{g}$ and a set of $|\mathcal{H}| = 100$ candidate channels. All channels are assumed to be sequences of i.i.d. symbols, which are Gaussian distributed according to the derivations in Sec. 3.3.4. Among the set of candidate channels, there is one channel $\boldsymbol{h}_i$ for which $\boldsymbol{g} \leftrightarrow \boldsymbol{h}_i$ holds. Recall that the model is parameterised by $\sigma_s^2, \sigma_{h|s}^2$ and $\sigma_{g|s}^2$. Throughout the following I set $\sigma_{h|s}^2 = \sigma_{g|s}^2$. I set $\sigma_s = 10$ and vary the observation noise $\sigma_{h|s}$ within $[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]$. Note that not the actual values of the parameters is important for the analysis, but only their ratio. The chosen parameters simulate a noise level from 10% to 100% of the source signal standard deviation. For each triplet of parameters, I generate 100 test sets. Each test set consists of an observed seed channel and $\mathcal{H} = \{\boldsymbol{h}_j\}_N$, $N = 100$ candidate channels. Each channel consists of $3,000$ samples (=signal values). A sample for the seed channel and the true corresponding channel is generated as follows: First, I sample a source signal value from the distribution defined in Eq. 3.71, and then sample g and h from Eq. 3.73 and Eq. 3.72, respectively.

I then estimate the posterior distribution, based on the temporal update scheme described in Sec. 3.3.2. Besides the posterior model, I will also regard the approximate model of the posterior as defined in Eq. 3.61. If the samples originate from the same source signal, I allow them to vary within a matching envelope, where the extent of the envelope is given by the standard deviation of the observation noise. The threshold test for the approximate model is then defined as:

$$|g - h| < \sigma_{g|s} + \sigma_{h|s}. \tag{3.85}$$

I will also compare the proposed model to a standard correlation approach. To this end, I perform a full correlation of the seed signal channel $\boldsymbol{g}$ to all other candidate channels $\{\boldsymbol{h}_j\}$. The correlation of the seed channel and the $i'th$ candidate channel is given as (Therrien 1992, p. 140):

$$\rho_{g,\boldsymbol{h}_i} = \frac{\mathrm{Cov}(\boldsymbol{g}, \boldsymbol{h}_i)}{\sigma_{\boldsymbol{g}}\sigma_{\boldsymbol{h}}}, \tag{3.86}$$

and the corresponding sample correlation coefficient as:

$$\hat{r}_{g,\boldsymbol{h}_i} = \frac{1}{N}\frac{\sum_n^N g[n] \cdot h_j[n]}{\sigma_g \sigma_h}. \tag{3.87}$$

For all three methods, I estimate the correspondence over all 3,000 samples. Recall that I would like to minimise the number of update steps and only perform an update, when we may expect to gain information about the true correspondence. In Sec. 3.3.3, I hypothesised that we gain the more information the rarer the regarded signal symbols are. This should result in larger (negative) entropy differences between update steps, hence, the curve of the entropy should decrease faster. In order to analyse the influence of the rareness of the individual symbols on the correspondence estimate I proceed as follows. For the $i'th$ signal sample, I perform the model updates only when $p\,(g) <= p\,(\tau)$ holds. I vary τ within $[0, 1 \cdot \sigma_{g|s}, 2 \cdot \sigma_{g|s}, 3 \cdot \sigma_{g|s}, 4 \cdot \sigma_{g|s}]$. For $\tau = 0$, the model will be updated for each of the 3,000 samples. The larger τ is set the less model updates will be performed.

In Figs. 3.12 to 3.21, I visualise for each method the entropy of the correspondence distribution and extract the estimated correspondence as the coordinates (=index of the candidate channel in $\mathcal{H}$) for which the distribution takes on its maximum value (MAP estimate). Recall that for each noise level 100 independent test sets of seed and candidate channels were generated. Hence, the plots shown in the following are results which were averaged over the 100 test sets. Within the plots visualising if the true correspondence was estimated, the plotted values are the fraction of successfully estimated correspondences over the 100 test sets after the given number of model updates and, hence, the plotted values are in the interval $[0, 1]$. While only the posterior is a

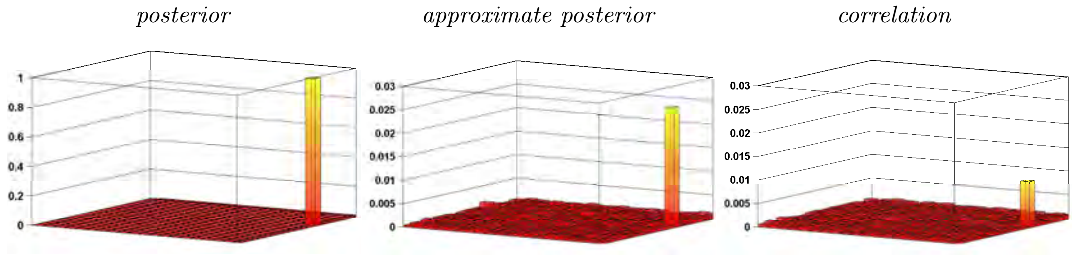


Figure 3.11: **Sample posterior and approximations:** (left) Posterior distribution estimated according to the proposed model, (middle) approximate posterior, (right) correlation result. While the approximate model and correlation are able to determine the true correspondence, the corresponding distributions contain a noise floor, which consumes most of the probability mass. Hence, the confidence in the correspondence is small. See text for details. Best viewed up scaled.

true probability distribution, I will also regard the results for the approximate model and the correlation as a distribution over the correspondence candidates (upon proper normalisation).

For reasons of a clear presentation, throughout the figures, I limit the plot range on the abscissa to 40 (model update steps).

Regard Fig. 3.12. For a noise level of 10%, we see that the proposed model is able to learn the correspondence after 5 model update steps on average. This is the case for the full posterior update and when updates are skipped. Furthermore, the entropy of the posterior distribution continuously decreases until a minimum is reached. However, it is important to note that the number of observed signal values and the number of performed model updates are in general different, only for the full posterior update where the model is updated for every new symbol these numbers are identical.

The performance of the approximate model, w.r.t. the needed number of updates until the true correspondence is learnt, is slightly worse than for the exact model as expected. We see that skipping certain updates, as described above, has a small influence on the number of updates needed (less are needed), until the correspondence is learnt. More importantly, we see that the entropy of the correspondence distribution decreases only marginally. While the correlation model is also able to determine the true correspondence, we see that considerably more update steps are needed. Similar as for the approximate model, the entropy of the correspondence distribution does not decrease substantially. The reason for this is that the approximate correspondence distributions contain a 'noise floor' which consumes most of the probability mass. The noise floor is distributed rather uniformly, hence the entropy will be close to the maximum entropy of the distribution (cf. Fig. 3.11).

Regard Fig. 3.13. For a noise level of 20%, we see that the proposed model needs more update steps until the true correspondence is learnt. Clearly, this is to be expected, as the involved distributions are more flat and within each update step there are more compatible signal values among the channels. Furthermore, we see that more likely signal values may be skipped which allows to learn the correspondence with a fewer number of update steps. This can also be seen from the entropy curves. The entropy curve, when skipping all signal values with $p(g) > p\left(4 \cdot \sigma_{g|s}\right)$ decreases the fastest. For the approximate model and the correlation, we also see that the correspondence may be learnt with fewer update steps by skipping more likely signal values. However, the proposed method clearly outperforms the approximate model and the correlation approach. The entropy curves of the approximate and correlation model show a similar behaviour as before, and do not decrease.

For noise levels of 30% to 50% we now see more clearly that when updates for likely signal values are skipped, the less update steps are needed overall. See how the entropy curve decrease the faster, the more likely update steps are discarded. This is the case for all 3 models, however, the proposed method still outperforms the other two. Also note that the approximate model and the correlation approach are on par for a noise level of 40%, and that the correlation approach slightly outperforms the approximate model for a noise level of 50%. This is to be expected, and is due to the chosen threshold test for the approximate model, as defined in Eq. 3.85. For this threshold test and a noise level of 50%, on average every second candidate channel will be matched.

When we further increase the noise level up to 80%, we see that the correspondence curve starts to decrease for a skip level of $p(g) > p\left(4 \cdot \sigma_{g|s}\right)$. Note that this is *not* a failure case of the method but rather a matter of the script which generated the figures for visualisation. For the given parameter settings, there will be on average $2 \cdot (1 - F(\tau, \mu, \sigma^2)) \cdot N$ samples which will pass the threshold test $p(g) <= p(\tau)$ such that a model update is carried out. Here, F is the cdf of the distribution associated with the seed channel. In a set of 3,000 i.i.d samples and for a noise level of 80% the expected number of samples which pass the threshold test and thus lead to a model update is $2 \cdot (1 - F(4 \cdot \sigma_{g|s}, 0, \sigma_g^2)) \cdot 3000 = 37.89$. The plotted data points are initialised to 0 (=failed to estimate the true correspondence), hence, the fraction of successfully estimated correspondences over the 100 test sets after more than ≈ 38 model updates will decrease to 0 simply because there are no available data points to be plotted.

For noise levels of 90% and 100% we obtain an average number of samples of 22.36 and 14.03, respectively. Obviously, these bounds also exist for the other noise and skip levels. However, these cannot be seen from the plots as the actual numbers are always above the plot range limit which was set to 40 update steps.

Let us now summarise the simulation results. We have seen that the true correspondence can be learnt based on the proposed model, its approximate counterpart and based on standard correlation. The proposed method clearly outperforms the other two. The approximate model outperforms standard correlation w.r.t. the average number of needed update steps, given that the threshold test performs better than chance. We have seen that only for the proposed scheme, the entropy of the correspondence distribution decreases as expected. For the approximate and correlation model, the distributions contain a rather uniformly distributed noise floor, which consumes most of the probability mass and leads to an entropy which is close to the maximum entropy of the correspondence distribution. For all three models, we have seen that we may decrease the number of needed update steps, by discarding those update steps, in which the seed signal value has a high probability to be observed. The more rarer a seed signal value is, the less compatible candidate signal values exist (assuming a reasonable observation noise level). In order to minimise the number of update steps, we should regard signal values with minimum probability of being observed. However, in the extreme case we need to wait forever, until a suitable observation is actually observed.

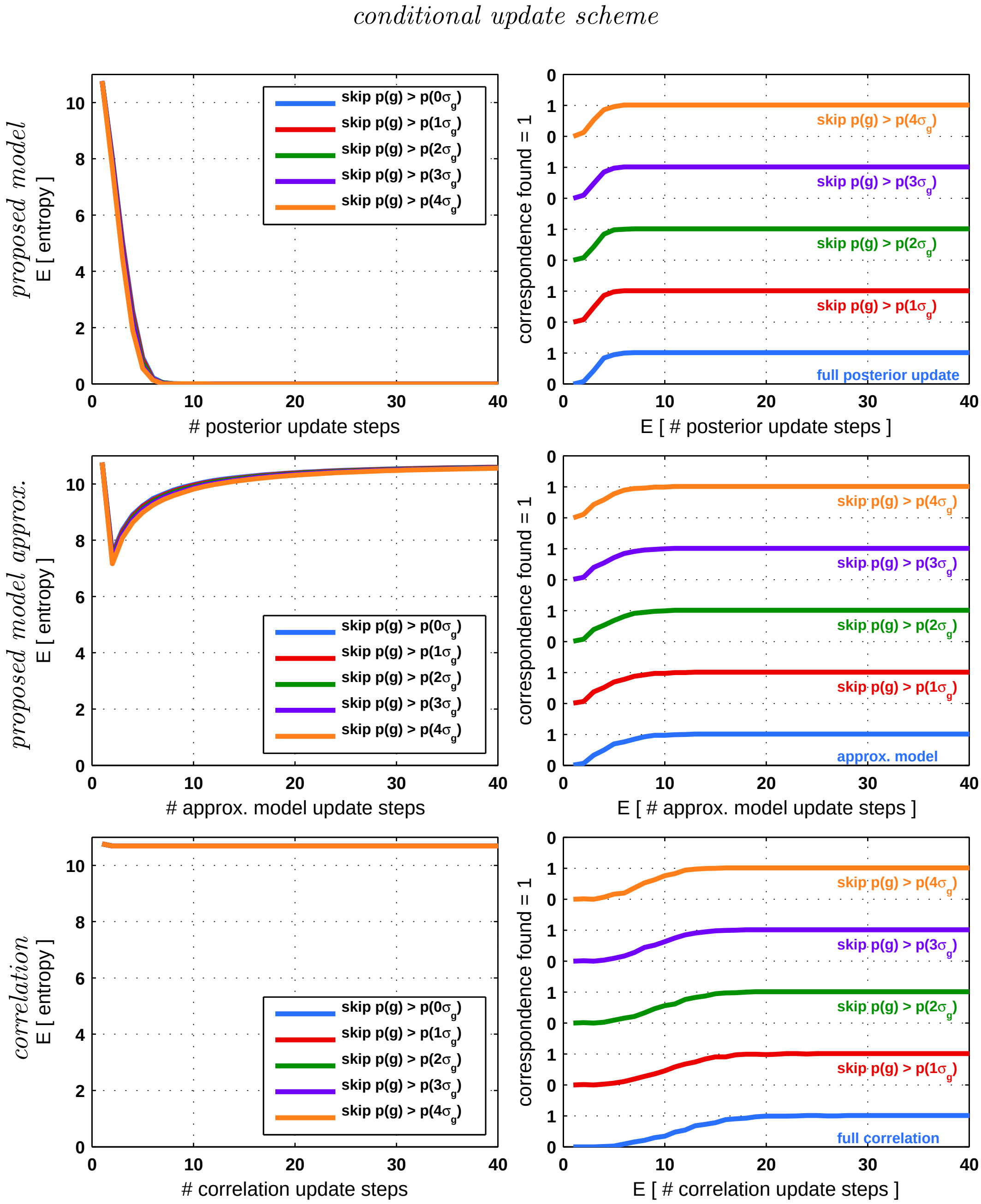


Figure 3.12: **Simulation results:** Simulation results averaged over 100 runs, based on 3,000 sampled signal values per channel and a noise level of 10% of the source signal standard deviation. See text for details. Figure only interpretable when viewed in colour.

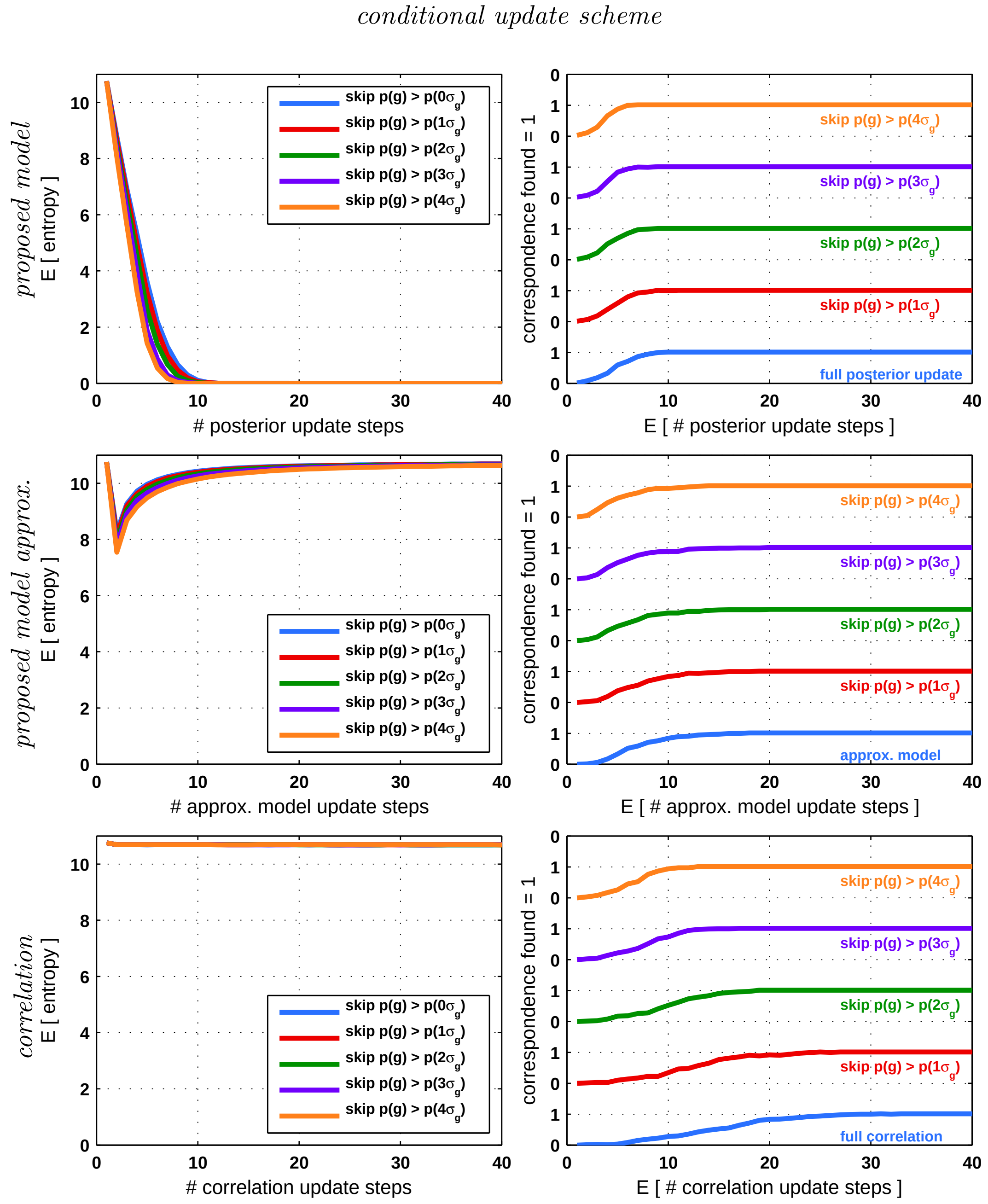


Figure 3.13: **Simulation results:** Simulation results averaged over 100 runs, based on 3,000 sampled signal values per channel and a noise level of 20% of the source signal standard deviation. See text for details. Figure only interpretable when viewed in colour.

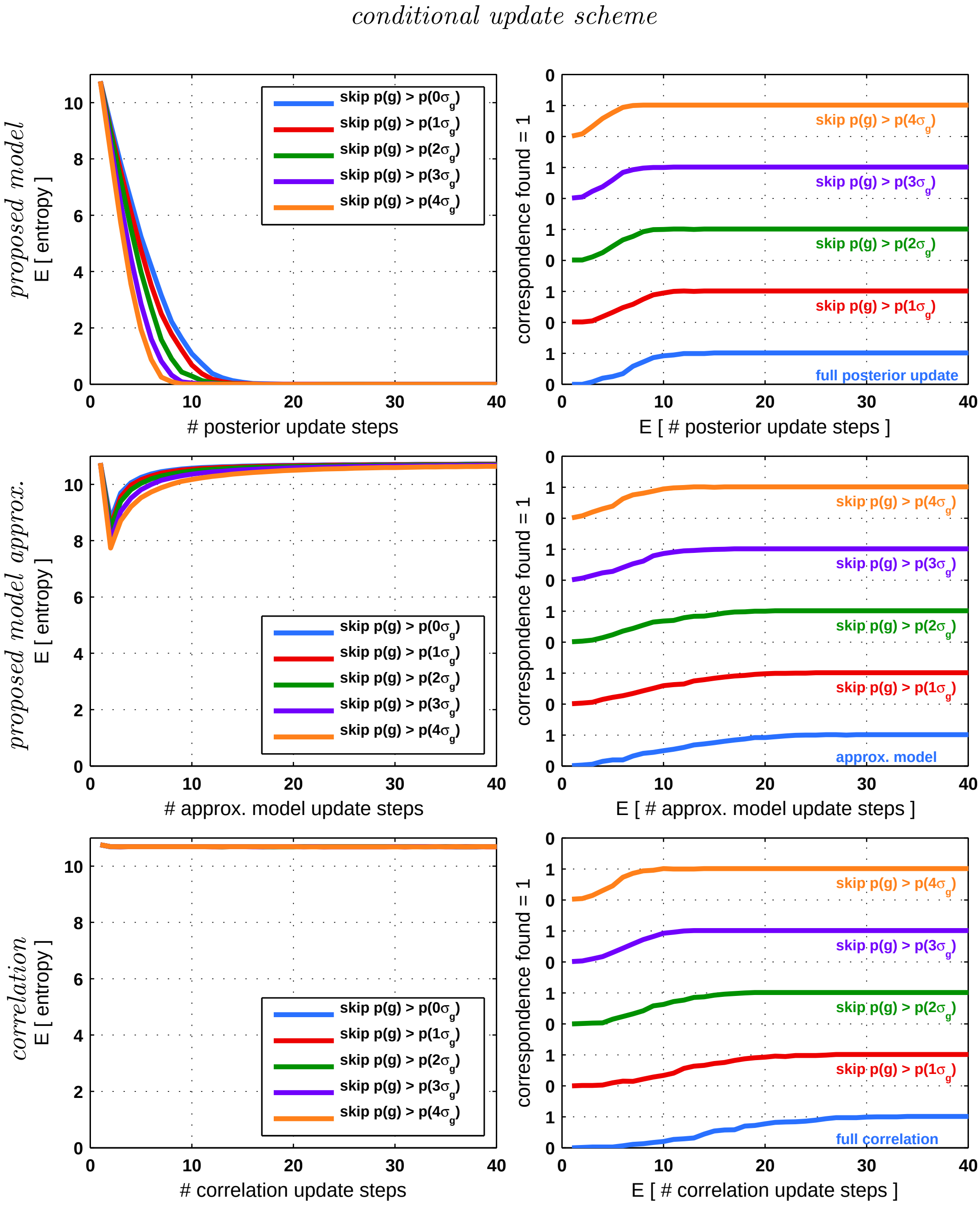


Figure 3.14: **Simulation results:** Simulation results averaged over 100 runs, based on 3,000 sampled signal values per channel and a noise level of 30% of the source signal standard deviation. See text for details. Figure only interpretable when viewed in colour.

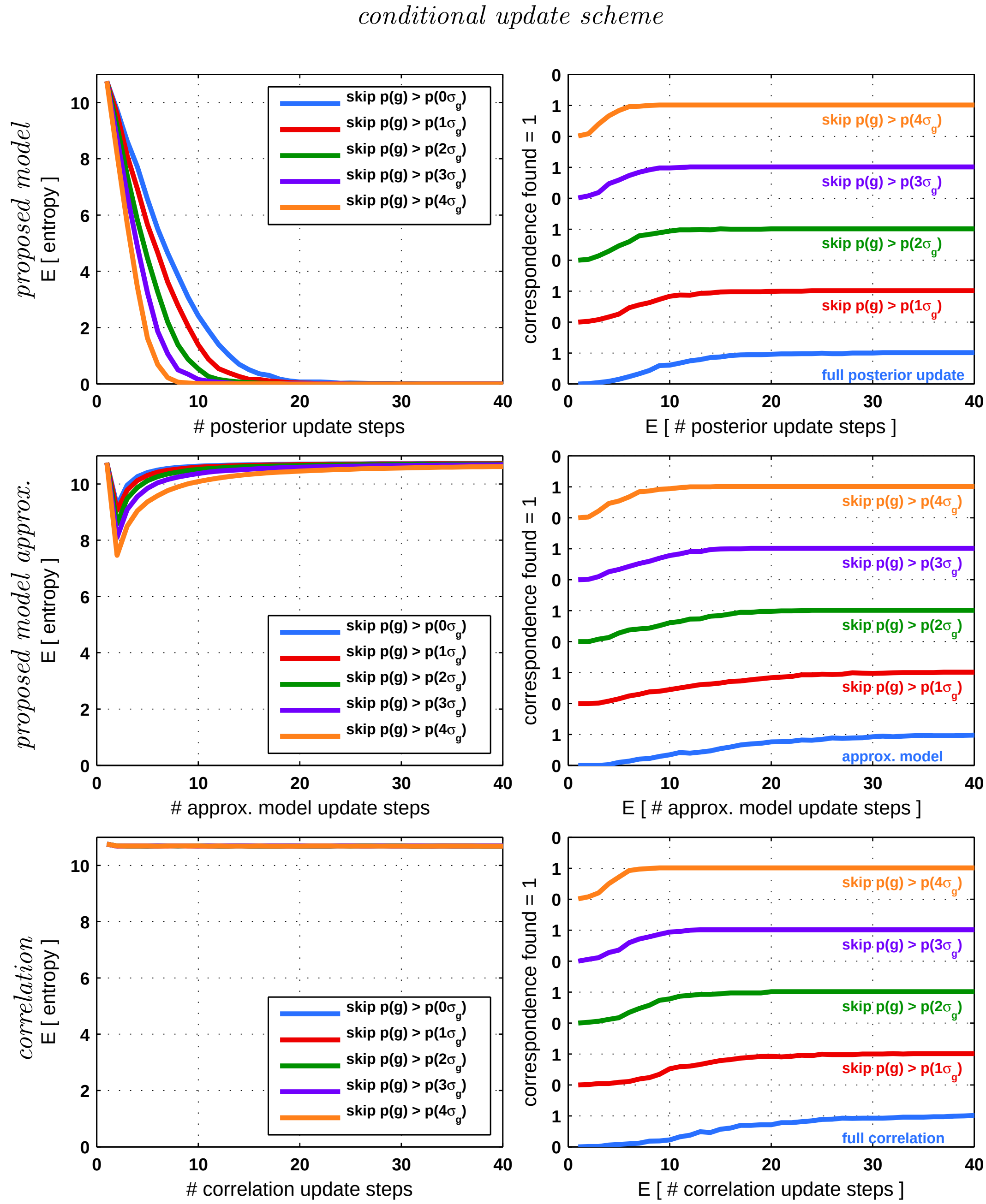


Figure 3.15: **Simulation results:** Simulation results averaged over 100 runs, based on 3,000 sampled signal values per channel and a noise level of 40% of the source signal standard deviation. See text for details. Figure only interpretable when viewed in colour.

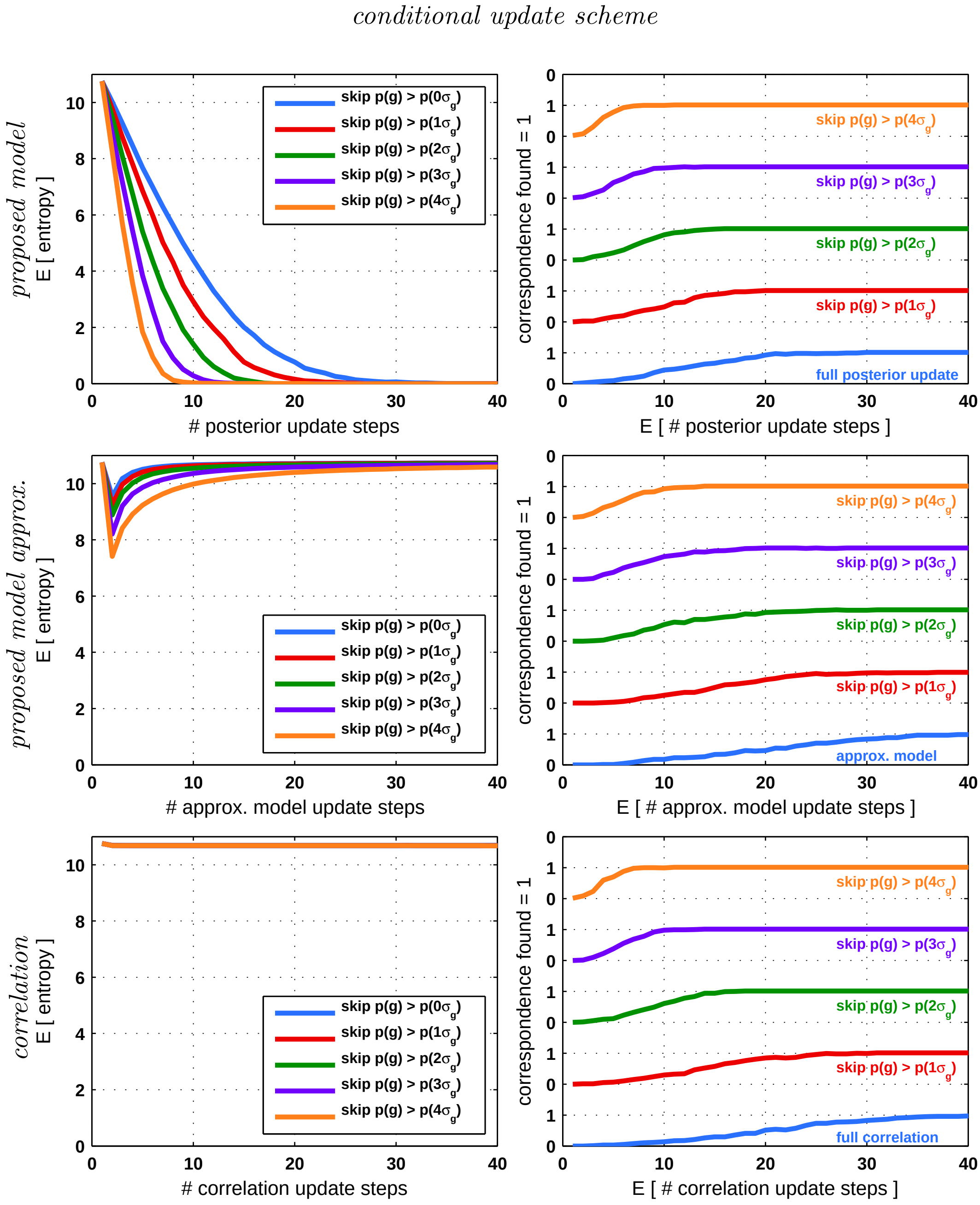


Figure 3.16: **Simulation results:** Simulation results averaged over 100 runs, based on 3,000 sampled signal values per channel and a noise level of 50% of the source signal standard deviation. See text for details. Figure only interpretable when viewed in colour.

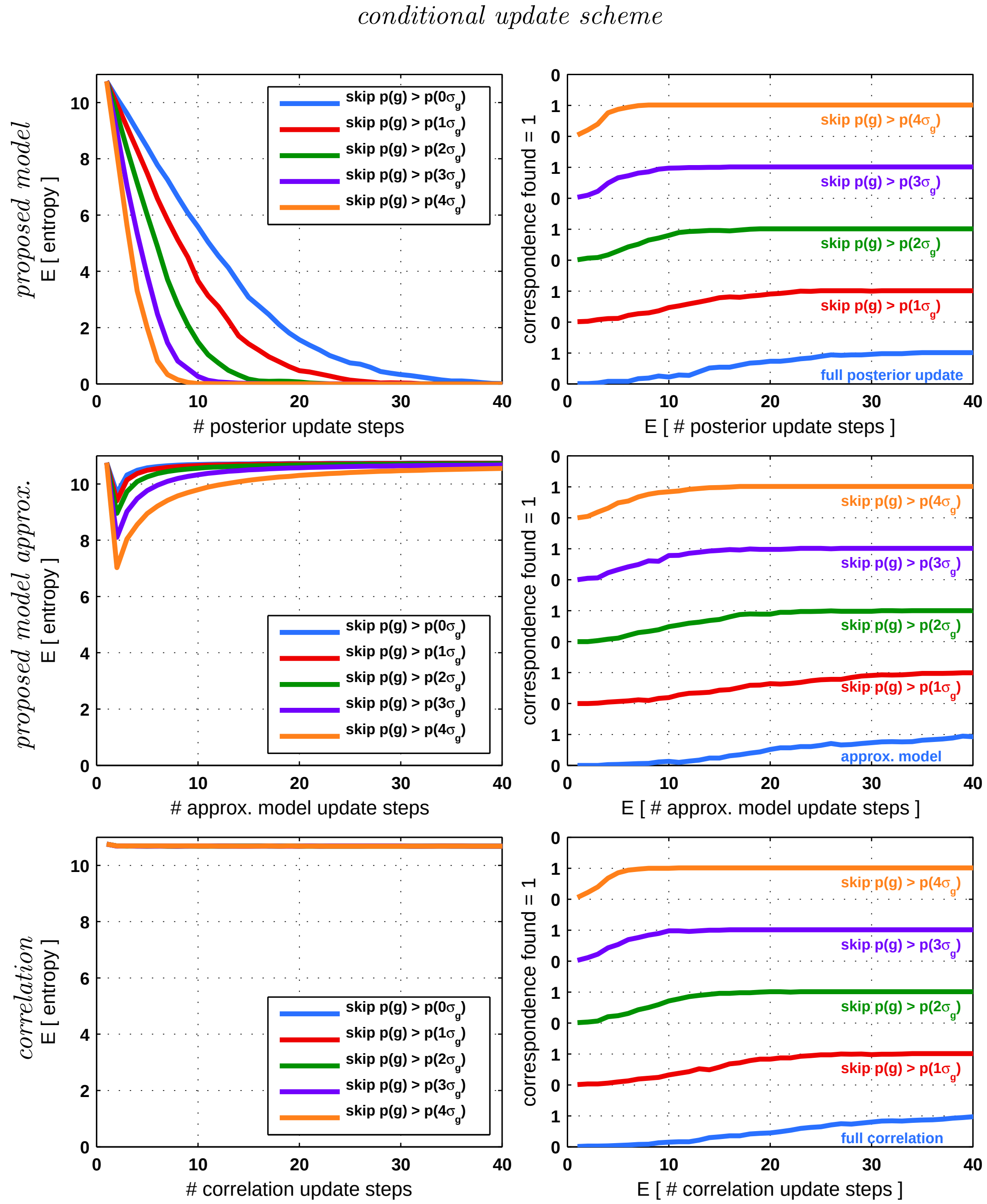


Figure 3.17: **Simulation results:** Simulation results averaged over 100 runs, based on 3,000 sampled signal values per channel and a noise level of 60% of the source signal standard deviation. See text for details. Figure only interpretable when viewed in colour.

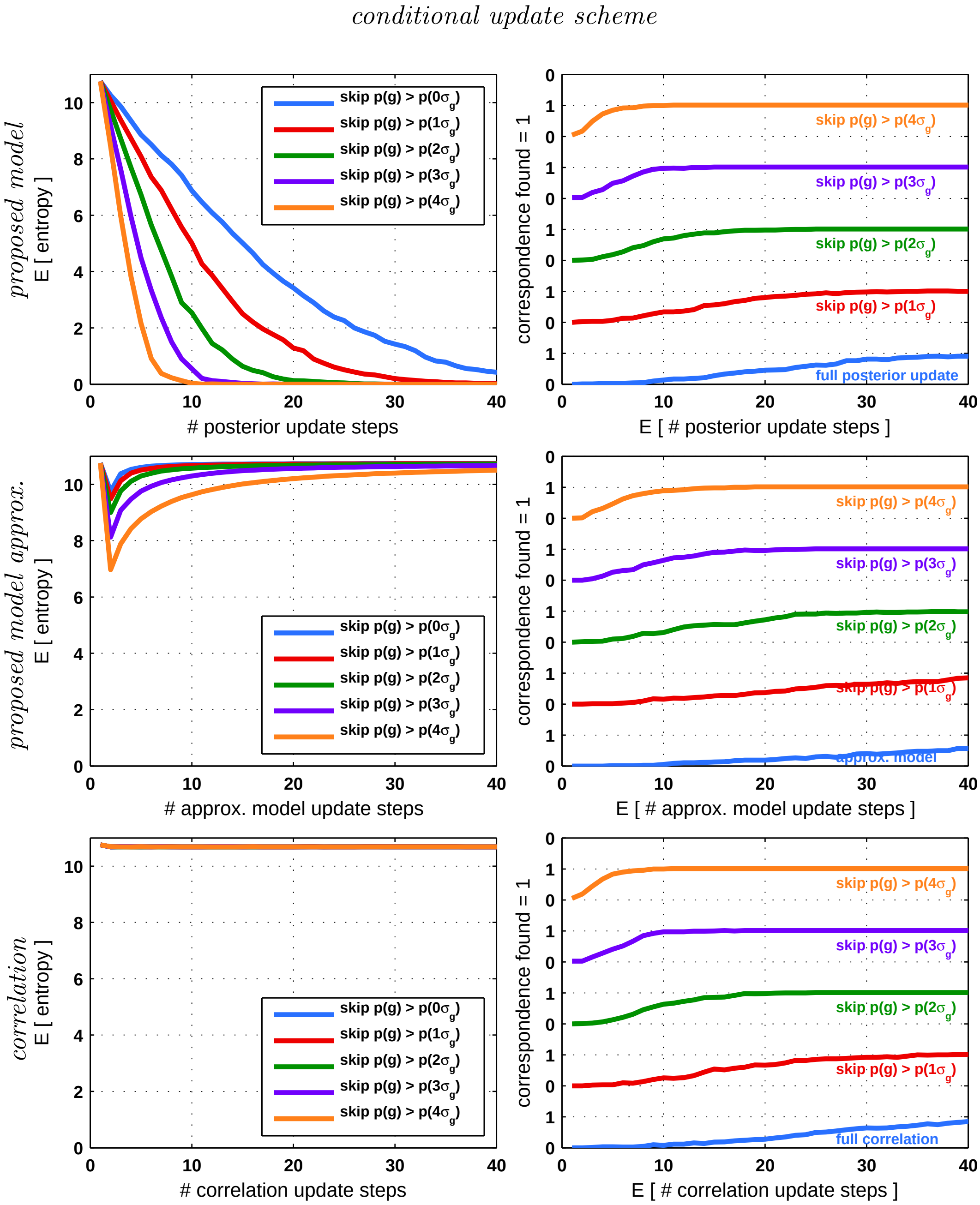


Figure 3.18: **Simulation results:** Simulation results averaged over 100 runs, based on 3,000 sampled signal values per channel and a noise level of 70% of the source signal standard deviation. See text for details. Figure only interpretable when viewed in colour.

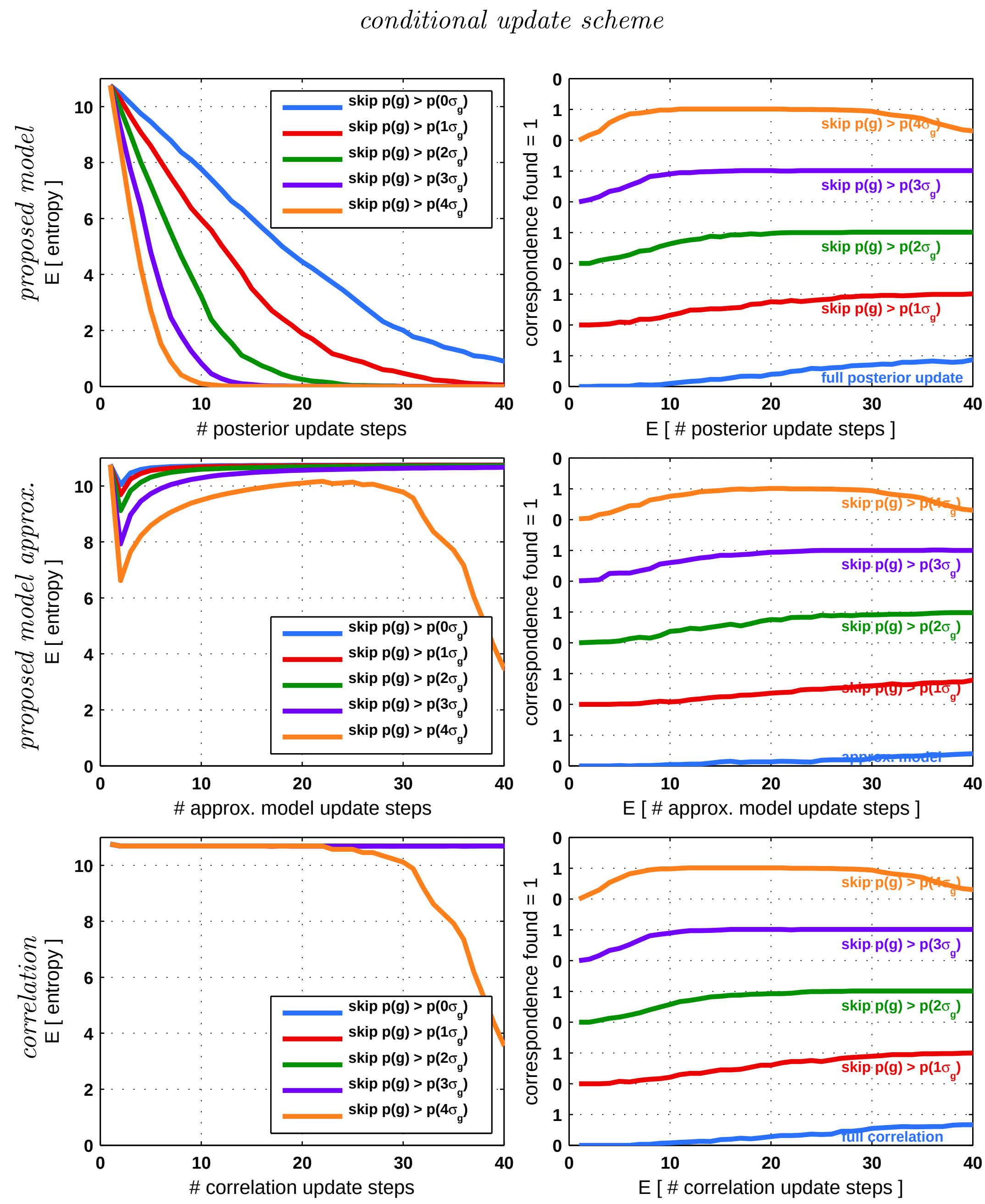


Figure 3.19: **Simulation results:** Simulation results averaged over 100 runs, based on 3,000 sampled signal values per channel and a noise level of 80% of the source signal standard deviation. See text for details. Figure only interpretable when viewed in colour.

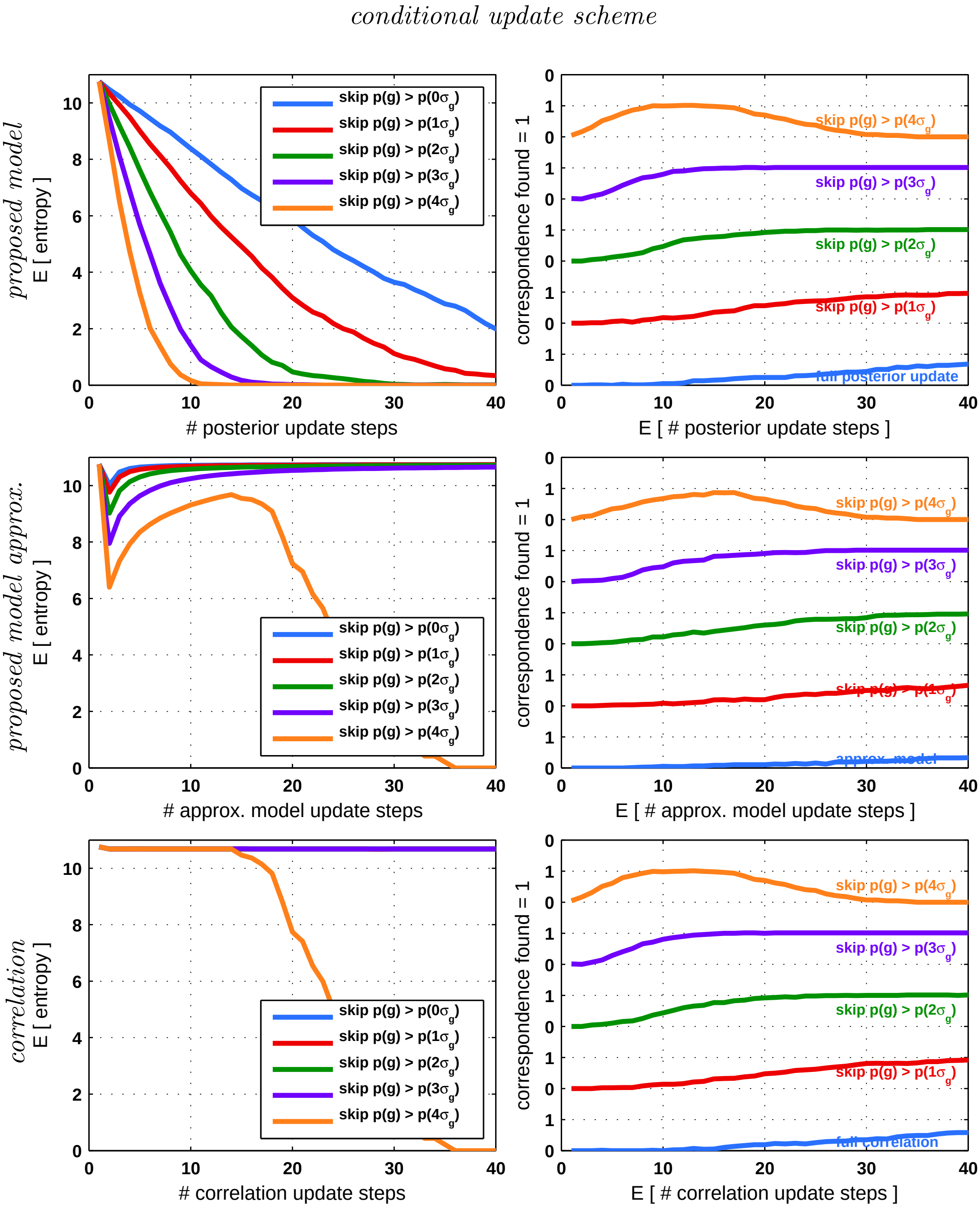


Figure 3.20: **Simulation results:** Simulation results averaged over 100 runs, based on 3,000 sampled signal values per channel and a noise level of 90% of the source signal standard deviation. See text for details. Figure only interpretable when viewed in colour.

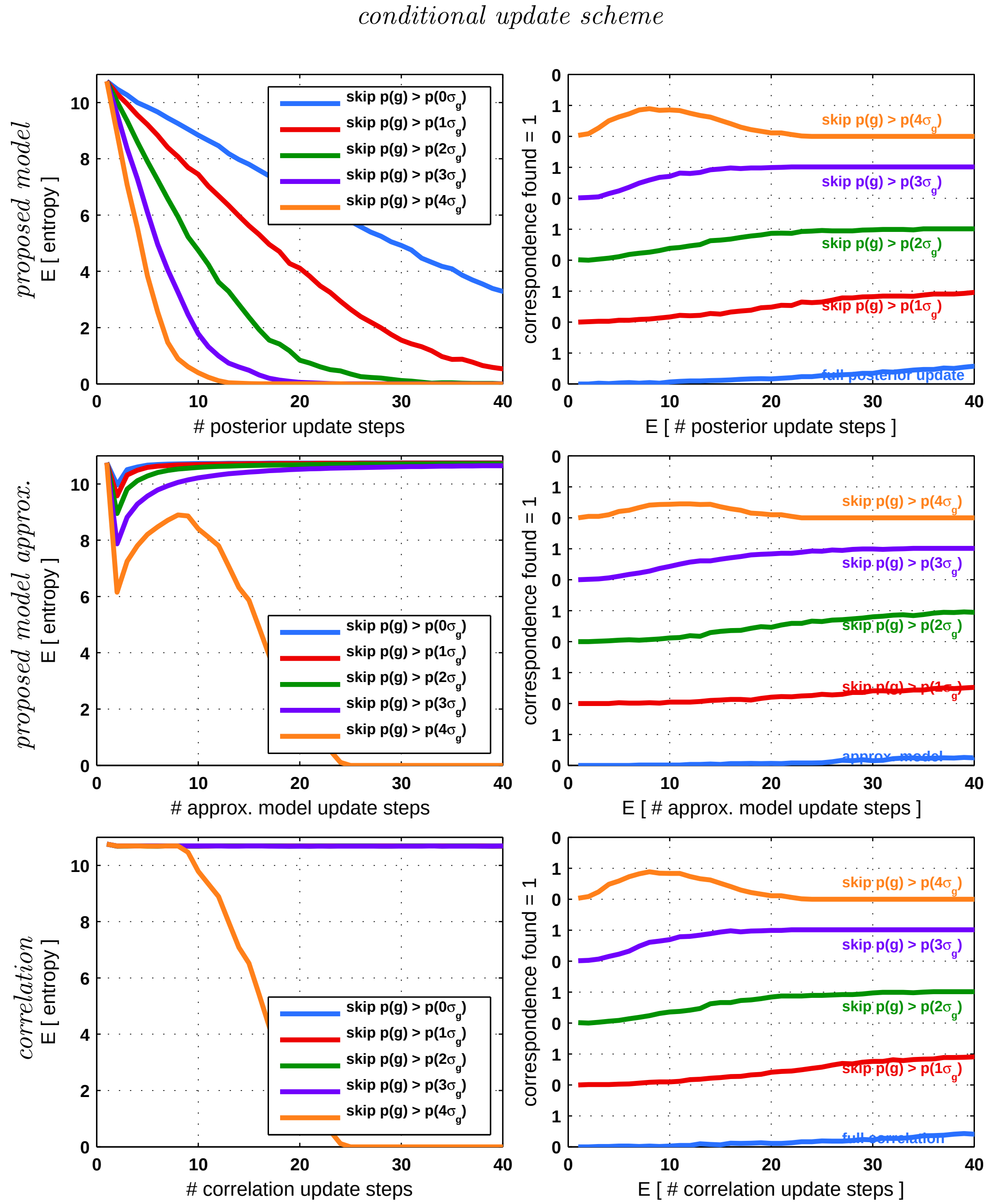


Figure 3.21: **Simulation results:** Simulation results averaged over 100 runs, based on 3,000 sampled signal values per channel and a noise level of 100% of the source signal standard deviation. See text for details. Figure only interpretable when viewed in colour.

3.4 TCA Algorithm

The *Temporal Coincidence Analysis* (TCA) approach, which was informally introduced in Sec. 3.1 directly follows from the probabilistic model derived in Sec. 3.3. TCA is thus a method to learn correspondence distributions among pairs of views in an autonomous way.

In the following, I regard image sequences of a binocular camera setup with cameras $\mathcal{C}_i$ and $\mathcal{C}_j$. For a selected pixel $\boldsymbol{x}_i \in \mathcal{I}_i$ in the spatial domain of camera $\mathcal{C}_i$ (occasionally also called first view or source view), its correspondence distribution over the spatial domain of $\mathcal{C}_j$ (occasionally referred to as second view or target view) is sought. Similar as defined in Sec. 3.3, I regard the temporal signal of each pixel as a signal channel, which is the observation of a hidden source channel. The source channel is considered to emit from the 3-D point, which we observe at a specific pixel location. Within each view, the individual pixels (=channels) are assumed to be independent of each other. This is certainly not the case for natural images and video and means that I disregard the topological order of the image data. However, at the same time this induces invariances to certain geometric transformations of the input data and lets us learn correspondences for randomly permuted data. These invariances include translational and rotational invariance and scaling to a large extend (see Sec. 4.2).

Recall that the signal channels of the probabilistic model are considered to be sequences of i.i.d. symbols. Clearly, the temporal signal of single pixels will contain correlations. Therefore, I will not directly work on the pixel signals but instead determine an event signal which at least approximates an i.i.d. sequence. Each event signal value is a function of temporally consecutive signal value pairs. While an event value may be determined from longer signal value strings, we have to take into account that a correspondence may only exist within short time intervals as discussed in Sec. 3.1.

3.4.1 Temporal Event Signal and Event Detection

Consider the temporal grey value signal of a pixel $\boldsymbol{x}_i$ in camera $\mathcal{C}_i$. I convert the grey value signal into an event signal as follows. The event signal at time t is formed from the original signal values $s_i(\boldsymbol{x}_i, t-1)$ and $s_i(\boldsymbol{x}_i, t)$ of the seed pixel at time instants $t-1$ and t via a monotone function f_e according to:

$$f_{e,t}(s_i(\boldsymbol{x}_i, t-1), s_i(\boldsymbol{x}_i, t)). \tag{3.88}$$

I define the event function as the squared difference of the input values

$$f_{e,t}(a, b) = (a-b)^2. \tag{3.89}$$

This choice of the event function is certainly not the only which could be considered. We could, e.g., also regard the signed difference of the signal values. However, as will

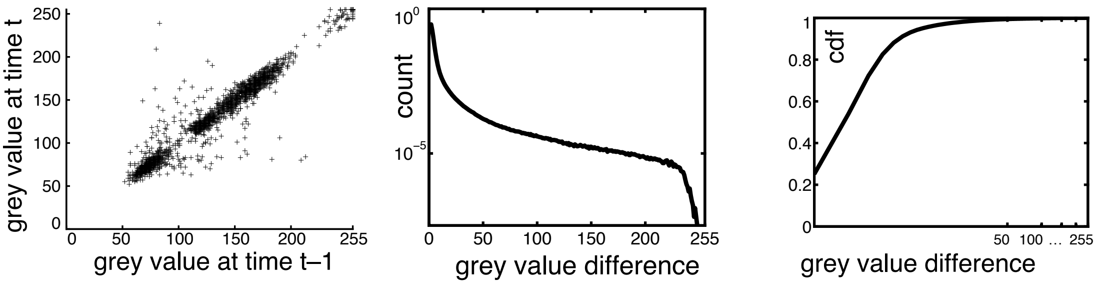


Figure 3.22: **Statistics of temporally subsequent grey values in natural video:** (left) Joint histogram of temporally subsequent grey values $s_i(\boldsymbol{x}_i, t-1)$ and $s_i(\boldsymbol{x}_i, t)$ of a fixed pixel location $\boldsymbol{x}_i$, (middle) normalised histogram of temporal grey value differences, (right) cdf of temporal grey value differences.

be explained shortly, if nothing is known about the photometric differences among the views, matching could fail, e.g., if the grey values for corresponding pixels are inverted. Therefore, at least initially we should retain to the squared difference.

We may now estimate the correspondence distribution by estimating the posterior distribution over the candidate (event) signals, as described in Sec. 3.3. Recall from Sec. 3.3.5 that the number of update steps needed to convey the true association depends on the probability of observing the respective signal values. The rarer the symbols, the more information they convey about the true correspondence. Therefore, I update the posterior only for significant events, or key events in the event signal, which are above an event threshold value T_e according to:

$$f_{e,t}(a,b) > T_e. \tag{3.90}$$

When we regard the joint histogram of temporally subsequent grey values $s_i(\boldsymbol{x}_i, t-1)$ and $s_i(\boldsymbol{x}_i, t)$, we see that these show certain regularities. Regard Fig. 3.22, which shows such a histogram for a fixed pixel location $\boldsymbol{x}_i$ estimated from a natural image sequence. Most of the time, the subsequent signal values are similar, as the histogram clusters along the main diagonal. Furthermore, we see that the histogram counts systematically decrease with increasing distance from the main diagonal. From this, we conclude that a combination of two specific grey values occurs the less often, the larger their difference is, since this is directly equivalent to the distance from the main diagonal in the joint histogram. This can also be seen from Fig. 3.23, which shows joint histograms for sequences `GUBo1616` and `GUCar`. Therefore, key events are detected for rather large threshold values T_e. We can determine a suitable threshold T_e from the cumulative distribution function of pixel grey value differences as an α-percentile. This can be done on a global basis, such that the threshold is held fixed over time, but could equally be

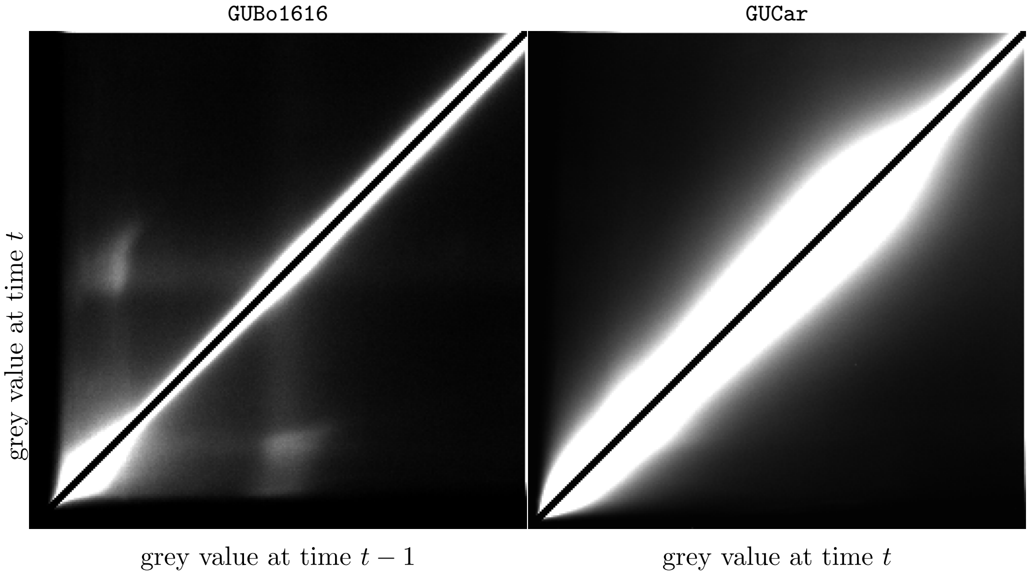


Figure 3.23: **Joint histograms for sequences `GUBo1616` and `GUCar`:** Histograms are averaged over a regular subset of all pixels within the respective sequence. Images are rescaled w.r.t. their grey value range for visualisation purposes. Compare with Fig. 3.22. See text for details.

updated on a per frame basis. Clearly, the larger T_e is chosen, the rarer are the events which fulfil Eq. 3.90.

According to the posterior and approximate model update, we compare grey values from different cameras. We have to take into account that a 3-D point captured by two cameras rarely shows the same grey value. Clearly, this is due differing camera characteristics and signal noise. To this end, I explicitly model photometric differences among the cameras as described next.

3.4.2 Grey Value Transfer Function

To model the different photometric characteristics of two cameras, I define the function that maps grey values acquired by one camera onto the observed grey value in the other camera. This function is denoted as the *Grey Value Transfer Function* (GVTF) ϕ_{ij}; it models the characteristic curve of the mapping of grey values s_i produced by camera $\mathcal{C}_i$ to the corresponding grey values s_j acquired with camera $\mathcal{C}_j$. Thus, if pixel $\boldsymbol{x}_i$ in image $\mathcal{I}_i$ and pixel $\boldsymbol{y}_j$ in image $\mathcal{I}_j$ correspond to the same point in 3-D space at time t we have:

$$s_j(\boldsymbol{y}_j, t) = \phi_{ij}(s_i(\boldsymbol{x}_i, t)). \tag{3.91}$$

If we assume a set of ideal cameras with exactly matching photometric characteristics (including same exposure settings, 3-D points follows Lambertian reflectance model), then the GVTF ϕ_{ij} is simply the identity function. The GVTF will usually be unknown and needs to be estimated in practice along learning the correspondence distribution. In Sec. 3.6 I develop an approach to learn the GVTF.

3.4.3 Event Matching and Estimation of Correspondence Distributions

Once a key event is triggered on the seed pixel $\boldsymbol{x}_i$, I update the posterior distribution as described in Sec. 3.3.2. This update is based on the GVTF transformed event signal values. If the uncertainties in the GVTF estimate are assumed to be Gaussian distributed with mean zero and a variance σ_ϕ^2, this term may be added to the assumed observation noise.

In the approximate model, matching is performed based on a threshold test. Specifically, I determine the set of *possible* correspondences Ω_{pc} which includes all pixels in $\mathcal{C}_j$ according to:

$$\begin{aligned}\Omega_{pc}(\boldsymbol{x}_i, t) = \{\boldsymbol{y}_j \in \mathcal{I}_j :& \\ & f_{e,t}(s_i(\boldsymbol{x}_i, t-1), s_i(\boldsymbol{x}_i, t)) && > T_e \\ \wedge\ & f_m(\phi_{ij}(s_i(\boldsymbol{x}_i, t-1)), s_j(\boldsymbol{y}_j, t-1)) && < T_m \\ \wedge\ & f_m(\phi_{ij}(s_i(\boldsymbol{x}_i, t)), s_j(\boldsymbol{y}_j, t)) && < T_m\}.\end{aligned} \tag{3.92}$$

Here, f_m is a matching or distance function of its input arguments. Observe that in this formulation of the matching function, I implicitly test for an event in the candidate channel. Threshold value T_m models the signal noise and allows to define a matching envelope to account for uncertainties in the estimate of the GVTF. The accumulator is then updated according to Eq. 3.61. For natural scenes, a reasonable choice for the matching function could be the squared or absolute difference. However, this also implies that the matching would fail if the polarity of the grey values among the views is opposite and nothing is known about the true GVTF. However, different matching functions may easily be integrated.

To summarise, I learn correspondence distributions via TCA by the repeated detection and matching of decisive events. In the probabilistic model, I update the posterior distribution as described in Sec. 3.3.2 whenever an event has been detected at the seed pixel $\boldsymbol{x}_i$. The posterior distribution from the previous time step serves as a prior distribution for the current time step. In the approximate model, I build an accumulator array. Within each matching pass, the count of the cells indexed by the set $\Omega_{pc}(\boldsymbol{x}_i, t)$ on the accumulator is incremented based on the number of matched events $M = |\Omega_{pc}(\boldsymbol{x}_i, t)|$.

Algorithm 1 Pixel wise Temporal Coincidence Analysis

```
Input a pixel $x_i$ in camera $\mathcal{C}_i$ and a camera $\mathcal{C}_j$
Approximate model: Set the accumulator $A_{ij}$ to zero
Exact model: Initialise posterior
while images to be processed do
  if an event occurred on pixel $x_i$ then
    Approximate model: raise the count in $A_{ij}$ for those pixels in camera $\mathcal{C}_j$ which are in the set $\Omega_{pc}(x_i, t)$
    Exact model: update posterior distribution
  end if
end while
```

Algorithm 1 summarises the basic algorithm to estimate the correspondence distribution via temporal coincidences.

I am now interested in monitoring the learning process by means of several attributes of the correspondence distribution. These attributes then allow the system to have a measure of confidence about the learnt correspondence and assess when a correspondence has successfully been learnt. These self monitoring capabilities are a very important property of autonomous systems which need to be able to assess their learning performance (Förstner 1991).

3.5 On the Analysis of Correspondence Distributions

Within the previous sections, I derived the TCA approach to learn correspondence distributions. Recall from Sec. 3.1.1 that there are three classes of correspondence distributions I consider: i) point-to-point correspondence, ii) point-to-line correspondence and iii) no correspondence. Which type of correspondence is learnt depends on the scene depth structure and the scene content (cf. Sec. 3.1.1). Let a seed pixel $\boldsymbol{x}_i$ and its true corresponding pixel $\boldsymbol{y}_j$ at time t be given. If only small depth variations occur, $\boldsymbol{x}_i \leftrightarrow \boldsymbol{y}_j$ will be valid for all time steps and the correspondence distribution will be isotropic and point like. If large depth variations occur, the true corresponding pixel will vary along the epipolar ray (which not necessarily needs to be a line). In this case, the true correspondence distribution will be anisotropic with a predominant direction. If no correspondence exists the distribution will be scattered.

I am now interested in monitoring the learning process and assess the confidence in the correspondence estimate at every time step of the learning process. Furthermore, I want to decide to which class the learnt correspondence belongs. All this can be accomplished by means of a set of attributes of the correspondence distributions as follows. Recall the definition of a correspondence distribution from Sec. 3.3: The correspondence distribution is given by means of a pmf $f_{\boldsymbol{Y}}$, which describes the distribution of the random vector $\boldsymbol{Y}_j = (Y_1, Y_2)_j^T$, where $\boldsymbol{Y}_j$ is considered to represent the spatial coordinates in the j-th camera $\mathcal{C}_j$. Specifically, I will regard the following set of attributes of the correspondence distribution:

- mean vector $\boldsymbol{\mu_y}$,
- covariance matrix $\boldsymbol{C_y}$,
- eigenvalues of the covariance matrix λ_1, λ_2,
- scalar coherence measure f_c without the need for an eigenvalue decomposition of $\boldsymbol{C_y}$,
- coordinates of the maximum value $\boldsymbol{m}$,
- entropy $H(\boldsymbol{Y})$.

Mean vector The expected correspondence coordinates under the correspondence distribution are given as the expectation of rv. $\boldsymbol{Y}_j$:

$$\boldsymbol{\mu_y} = \begin{bmatrix} \mu_{y_1} \\ \mu_{y_2} \end{bmatrix}, \tag{3.93}$$

with:

$$\mu_{y_1} = \mathbb{E}\left[Y_1\right] = \sum_{y_1} \sum_{y_2} y_1 \cdot Pr[\boldsymbol{Y} = (y_1, y_2)], \tag{3.94}$$

$$\mu_{y_2} = \mathbb{E}\left[Y_2\right] = \sum_{y_1} \sum_{y_2} y_2 \cdot Pr[\boldsymbol{Y} = (y_1, y_2)]. \tag{3.95}$$

Covariance matrix The second order central moment of the correspondence distribution is given as the covariance matrix according to:

$$\boldsymbol{C_y} = \begin{bmatrix} c_{y_1,y_1} & c_{y_1,y_2} \\ c_{y_2 y_1} & c_{y_2,y_2} \end{bmatrix}, \tag{3.96}$$

with:

$$c_{y_1,y_1} = \sigma_{y_1}^2 = \mathbb{E}\left[(Y_1 - \mathbb{E}\left[Y_1\right])^2\right], \tag{3.97}$$

$$c_{y_2,y_2} = \sigma_{y_2}^2 = \mathbb{E}\left[(Y_2 - \mathbb{E}\left[Y_2\right])^2\right], \tag{3.98}$$

$$c_{y_1,y_2} = \sigma_{y_1,y_2}^2 = \mathbb{E}\left[(Y_1 - \mathbb{E}\left[Y_1\right]) \cdot (Y_2 - \mathbb{E}\left[Y_2\right])\right]. \tag{3.99}$$

Eigenvalues of the covariance matrix From an eigen-decomposition of the covariance matrix, we obtain eigenvectors $\boldsymbol{v}_1, \boldsymbol{v}_2$ and eigenvalues λ_1, λ_2 with:

$$\boldsymbol{C_y}\boldsymbol{v}_1 = \lambda_1 \boldsymbol{v}_1, \tag{3.100}$$

$$\boldsymbol{C_y}\boldsymbol{v}_2 = \lambda_2 \boldsymbol{v}_2, \tag{3.101}$$

with $\lambda_1 \geq \lambda_2$. The eigenvalues can be determined in closed form as the roots of the characteristic polynomial:

$$\det(\lambda \boldsymbol{I} - \boldsymbol{C_y}) = 0, \tag{3.102}$$

according to:

$$\lambda_1 = \frac{1}{2}(c_{y_1,y_1} - c_{y_2,y_2}) + \sqrt{4c_{y_1,y_2}^2 + (c_{y_1,y_1} - c_{y_2,y_2})^2}, \tag{3.103}$$

$$\lambda_2 = \frac{1}{2}(c_{y_1,y_1} - c_{y_2,y_2}) - \sqrt{4c_{y_1,y_2}^2 + (c_{y_1,y_1} - c_{y_2,y_2})^2}. \tag{3.104}$$

The eigenvalues may be regarded as a confidence measure about the learnt correspondence and the eigenvectors encode the direction of highest and lowest confidence scaled by their associated eigenvalue. The eigenvectors define the principal axes of an uncertainty or 'error ellipse', where the extend of the ellipse is governed by the eigenvalues.

Coherence measure I adopt the so called coherence measure from (Förstner 1991; Jähne 2012), which is a monotone and scalar valued function of the eigenvalues of the covariance matrix. The coherence measure indicates whether the correspondence distribution is of isotropic or anisotropic nature, with a dominant direction. It is defined as:

$$f_c = \frac{\lambda_1 - \lambda_2}{\lambda_1 + \lambda_2}. \tag{3.105}$$

While the coherence measure is a ratio of the eigenvalues, the following derivation shows that there is no need to explicitly solve for the eigenvalues. From the squared coherence measure we obtain:

$$f_c^2 = \frac{(\lambda_1 - \lambda_2)^2}{(\lambda_1 + \lambda_2)^2}, \tag{3.106}$$

$$= \frac{\lambda_1^2 - 2\lambda_1\lambda_2 + \lambda_2^2}{(\lambda_1 + \lambda_2)^2}, \tag{3.107}$$

$$= \frac{\lambda_1^2 + 2\lambda_1\lambda_2 + \lambda_2^2 - 4\lambda_1\lambda_2}{(\lambda_1 + \lambda_2)^2}, \tag{3.108}$$

$$= \frac{(\lambda_1 + \lambda_2)^2 - 4\lambda_1\lambda_2}{(\lambda_1 + \lambda_2)^2}. \tag{3.109}$$

For any square matrix its determinant is given by the product of its eigenvalues (Golub and Loan 2012):

$$\det\{\boldsymbol{A}\} = \prod_{i=1}^{N} \lambda_i. \tag{3.110}$$

The trace of a matrix is given by the sum of its eigenvalues (ibid.):

$$\mathrm{Tr}\{\boldsymbol{A}\} = \sum_{i=1}^{N} \lambda_i. \tag{3.111}$$

From these definitions Eq. 3.109 may be rewritten as

$$f_c^2 = \frac{(\mathrm{Tr}\{\boldsymbol{C_y}\})^2 - 4\det\{\boldsymbol{C_y}\}}{(\mathrm{Tr}\{\boldsymbol{C_y}\})^2}, \tag{3.112}$$

$$= 1 - \frac{4\det\{\boldsymbol{C_y}\}}{(\mathrm{Tr}\{\boldsymbol{C_y}\})^2}, \tag{3.113}$$

from which we see that no eigenvalues need to be computed explicitly. However, for 2×2 matrices, closed form solutions for the eigenvalue problem exist as shown in Eq. 3.100 and Eq. 3.101. The squared coherence measure f_c^2 is 0 for $\lambda_1 = \lambda_2$ and indicates an isotropic correspondence distribution. For $\lambda_1 > 0, \lambda_2 = 0$ the squared coherence measure is 1, indicating an anisotropic correspondence distribution and thus a line (=1D) structure. For both point-to-point and no-correspondence we expect an isotropic correspondence distribution. However, for the point-to-point correspondence we additionally require that $\mathrm{Tr}\{\boldsymbol{C_y}\} < T_\lambda$, with a suitable chosen threshold T_λ.

Coordinates of the maximum value The spatial coordinates where the correspondence distribution attains its maximum value are given according to:

$$\boldsymbol{m} = \begin{bmatrix} m_{y_1} \\ m_{y_2} \end{bmatrix} = \arg\max_{y_1,y_2} Pr[\boldsymbol{Y} = (y_1, y_2)]. \tag{3.114}$$

Entropy The entropy of $\boldsymbol{Y}$ under the correspondence distribution is given as (Cover and Joy Thomas 2009):

$$H(\boldsymbol{Y}) = \mathbb{E}\left[\log \frac{1}{Pr[\boldsymbol{Y}]}\right] = -\sum_{y_1}\sum_{y_2} Pr[\boldsymbol{Y} = (y_1, y_2)] \cdot \log Pr[\boldsymbol{Y} = (y_1, y_2)]. \tag{3.115}$$

The attributes defined above will now be used to assign the correspondence distribution to one of the three classes defined in Sec. 3.1.1 and as a confidence measure about the learnt correspondence. Specifically, the covariance error ellipse is used to visualise the confidence about a learnt correspondence, which are shown throughout the figures in Sec. 4 and 5. The axes of the error ellipse are given by the eigenvectors of the covariance matrix which are scaled by their eigenvalues. I usually scale the ellipses, such that the true correspondence lies within the ellipse with a probability of $99,7\%$ ($= 3\sigma$). See (J. Davis 2007) or (Hoover 1984) for details.

Next, let us discuss how these attributes are expected to behave for the three classes of correspondence distributions considered.

Point-to-point correspondence For point-to-point correspondences, the expectation of $\boldsymbol{Y}_j$, i.e., $\boldsymbol{\mu_y}$ marks the true corresponding pixel. This correspondence will hold over the whole image sequence. The posterior distribution will contain a distinct peak and both eigenvalues of the covariance matrix $\boldsymbol{C_y}$ are small. Thus the error ellipse for the location of the corresponding pixel is concentrated around the spatial location of the average correspondence $\boldsymbol{\mu_y}$ and the coherence measure is close to 0. The entropy of the correspondence distribution should be close to 0. We expect the coordinates of the distributions maximum $\boldsymbol{m}$ as an early indicator for the true correspondence. The coordinates of the distributions mean may be biased, i.e., they do not indicate the true correspondence as long as the entropy is not close to zero.

Point-to-line correspondence For point-to-line correspondences, $\boldsymbol{\mu_y}$ only represents the average correspondence location. Depending on the magnitude of the depth variations over time, the eigenvalues λ_1, λ_2 of the covariance matrix $\boldsymbol{C_y}$ satisfy $\lambda_1 \gg \lambda_2$; the eigenvector associated with λ_1 represents the axis along which the truly corresponding pixel is situated. Note that the average correspondence will not be the true correspondence at any specific time step. Instead, the correspondence distribution serves as a prior on the true correspondence. The coherence measure should be close to 1. The entropy should be small but will depend on the actual depth range observed.

No-correspondence In the absence of a correspondence, the accumulator will in general not be empty, but will contain a scattered distribution. In this case, the eigenvalues λ_1, λ_2 of the covariance matrix $\boldsymbol{C_y}$ both are of larger magnitude compared to those encountered in point-to-point or point-to-line correspondences. The entropy of the correspondence distribution should be rather large and the coherence measure close to 0. The mean of the distribution will lie close to the centre. The distribution's maximum $\boldsymbol{m}$ will be uninformative.

3.6 Learning the Grey Value Transfer Function

Recall from Sec. 3.4.2 that the GVTF models the mapping of a grey value $s_i(\boldsymbol{x}_i)$ at pixel $\boldsymbol{x}_i$ in view $\mathcal{C}_i$ to its corresponding grey value $s_j(\boldsymbol{y}_j)$ in view $\mathcal{C}_j$, observed at the true corresponding pixel $\boldsymbol{y}_j$. I am now interested in learning the GVTF in an autonomous fashion based on a parametric model but without the need for manually selected correspondences. The learning of correspondences and the learning of the GVTF parameters are intertwined tasks: the learning of correspondences depends on the GVTF estimate and the learning of the GVTF depends on estimated correspondences. In the beginning of the learning phase, when no correspondences are available, the GVTF has to be properly initialised, as will be discussed in the following. In a general multi camera setup, we also have to assume that the GVTF is not a static function, but that its parameters may vary over time. It is thus necessary to continuously update the estimate of the GVTF. The learning scheme proposed in the following can handle slow and gradual illumination changes (such as daylight change). However, if the regarded cameras operate with auto exposure, the GVTF may change suddenly and abruptly between frames. I will not explicitly model these sudden changes but suggest to detect them and signal that the GVTF transform will be invalid for the afflicted frames (cf. (Dederscheck et al. 2012)).

Note that even if the camera transfer functions are identical and under constant global illumination, the perceived object appearance may be different across the cameras if the object's reflectance does not follow a Lambertian model. In the following I make the assumption that the object's surface reflectance follows at least approximately a Lambertian model.

When nothing is known about the true GVTF I initialise it as the identity function and simultaneously enlarge the assumed observation noise in the posterior update scheme. Clearly, if we assume the GVTF to be the identity function, but that the true GVTF is, e.g., an affine transformation of the grey values, the posterior update scheme may fail to estimate the correspondence distribution. This is simply because the grey values of the corresponding pixels may differ severely, which leads to small likelihoods in the posterior update. This can only be compensated by increasing the assumed observation noise.

3.6.1 Related Work

The sensitivity to illumination differences is an inherent problem to all appearance based algorithms in which grey values are matched across images. As an example, the underlying assumption of template matching approaches is that the brightness between the template patch and the sought correspondence (patch) is identical (brightness constancy constraint equation, BCCE (Jähne 2012, p. 449)). Clearly, even in the absence of illumination differences, the BCCE will almost never be exactly fulfilled in practice, due to signal noise. Therefore, the difference between the template and the correspondence candidates is minimised, based on a suitable loss function. A common approach to reduce the influence of illumination differences on the matching score is to compute the matching score on *normalised* patches. For example, using the *zero normalised sum of squared differences* (NSSD) (Criminisi et al. 2007) or the *zero mean normalised cross correlation*

(ZNCC) (Hirschmüller and Daniel Scharstein 2009), the matching score is invariant to affine illumination differences (multiplicative gain and additive offset model). Robustness to affine illumination differences may also be achieved by regarding a collinearity measure of the (vectorised) image patches (Mester, Aach, et al. 2001; Mester and Conrad 2014). Given the patches only differ by means of a gain and offset factor, they should be collinear (given that the mean has been compensated). Note that normalisation should always be done 'with care', otherwise unrealistic and large differences may be compensated, still resulting in a high matching score. The *homogeneous matching measure* (HMM) proposed in (Mester and Conrad 2014) constrains the gain and offset ratios to realistic intervals. HMM is shown to be able to detect false matches, which are (false positive) matches under the ZNSSD and ZNCC measures due to the unchecked normalisation factors.

Robustness to illumination differences may also be achieved by transforming the input patches into a representation in which only the relative ratios between grey values are regarded. For example, the census transformation (Zabih and Woodfill 1994), represents a patch as a binary valued patch. Specifically, the grey value of the i-th pixel within the patch is set to 0 (1), given that the grey value of the centre pixel is larger (smaller). The binary valued patch may then be regarded as a hash code or signature. In (Stein 2004), the basic census approach was extended to a ternary representation which also codes 'similar' grey values. The method was used to estimate sparse optical flow in an automotive environment. See (Vogel et al. 2013) for a recent benchmark and (Hafner et al. 2013) for a review of the mathematical concepts underlying the census approach.

Let us now turn to approaches which actually try to model illumination differences among views instead of just compensating them. These approaches are closer related to the one presented by me in which illumination differences are modelled by means of an intensity transfer function.

In (Javed, Rasheed, et al. 2003), inter camera illumination differences are represented by means of a Gaussian distribution fitted to expected colour differences. Specifically, expected colour differences among views are computed from ground truth object correspondences by measuring the so called modified Bhattacharyya coefficient between corresponding histogram bins. The mean and variance of the determined distances are then used to parameterise the Gaussian distribution. The training phase of the approach relies on a known set of object correspondences, which could be generated by a single person moving through the scene. Certainly, this is a weak point of the method, especially in large/public camera networks.

In (Javed, Shafique, et al. 2005), the estimation of the transfer function is extended, such that a subspace of possible transfer functions is determined from ground truth data. From given object correspondences, normalised histograms are computed and a transfer function is determined from the inverted cumulative histogram. Then, to all these hypothesised transfer functions PCA is applied to obtain a lower dimensional subspace of possible transfer functions.

In (Cheng and Piccardi 2006), illumination differences are handled based on histogram normalisation. First, the histogram normalisation is applied to the RGB colour channels, subsequently computing the so called *major colour spectrum histogram*, which only contains a subset of all observed colour values of a tracked object. Object correspondences are then identified given that the histograms distances are below a threshold.

In (Porikli 2003), a colour transfer function is computed as the minimum cost path within the correlation matrix of 1-D histograms of given image pairs from different cameras.

In (Dederscheck et al. 2012) the *relative intensity transfer function* (RITF) between consecutive frames of a video stream is estimated. For two images given at time $t-1$ and t, first, the images are subdivided into small patches. Then, all patches showing apparent motion within the regarded time steps are discarded. A patch centred at location (x, y) in the image at time $t-1$ and the patch at the same spatial location in the image at time t are now assumed to be (approx.) identical except for an illumination difference. Next, from the mean grey values of corresponding patches a joint histogram is build. The RITF is then obtained as a constrained least squares fit to the (filtered) joint histogram. Based on a quality measure of the fit, the method is able to detect adverse illumination differences which cannot be compensated. Experimental results suggest that the proposed method allows to improve the results of a dense optical flow algorithm.

Related to the problem of modelling intensity differences in grey scale images is the problem of achieving colour constancy among different colour images. However, as I only regard grey scale images, I refer to (Hordley 2006) for an overview.

There is also interesting work in modelling the intensity mappings between images of the same camera, which allows to estimate the camera response function and which I mention here for completeness. Grossberg and Nayar (Grossberg and Nayar 2003, 2002) estimate the camera response function, i.e., the mapping from scene radiance to image intensity values. They show how the camera response function can be inferred from brightness transfer functions, estimated from joint histograms of grey values in pairs of images. While this finding is not completely new, they derive several ambiguities of the estimation process and show that an unambiguous recovery of the response function is only possible when assumptions on the response function are made. In previous work (e.g., (S. Mann 2000; Mitsunaga and Nayar 1999), these assumptions were made only implicitly, like smoothness of the response function etc.

3.6.2 GVTF Model

Let me now formally introduce my GVTF model. I model the GVTF by an N-th order polynomial:

$$\phi_{ij}(s_i(\boldsymbol{x}_i);\boldsymbol{\theta}) = \sum_{p=0}^{N} \theta_p s_i(\boldsymbol{x}_i)^p, \tag{3.116}$$

where $\boldsymbol{\theta} = \{\theta_0, .., \theta_N\}$ are the parameters to be estimated. While the order of the polynomial is in general unknown as well, typical camera transfer functions are monotone functions and may be modelled by a low order polynomial.

If a set of pixel to pixel correspondences is available, from which we obtain P pairs of colour values $\{s_i(\boldsymbol{x}_i), s_j(\boldsymbol{y}_j)\}_P$, the estimation of the GVTF boils down to a model selection (order of the polynomial) and regression problem (fitting the polynomial). In the literature, the set of grey value pairs used to estimate the photometric mapping is also denoted as *comparagram* (S. Mann and R. Mann 2001).

We have to take into account that the observed grey value pairs are only noisy measurements of the true grey values. The non-trivial task in the GVTF estimation process is thus the extraction of corresponding grey value pairs in the presence of *geometric* and *photometric* errors. This task becomes even more difficult when the correspondence type is not essentially point-to-point like. For the moment being, I assume that grey values are only extracted for point-to-point correspondences.

3.6.3 Observation Model

I regard a pixel correspondence $\boldsymbol{x}_i \leftrightarrow \boldsymbol{y}_j$ and the associated grey values $s_i(\boldsymbol{x}_i, t)$ and $s_j(\boldsymbol{y}_j, t)$ at time t. Let $\hat{\boldsymbol{x}}_i$, $\hat{\boldsymbol{y}}_j$, $\hat{s}_i(\hat{\boldsymbol{x}}_i, t)$ and $\hat{s}_j(\hat{\boldsymbol{y}}_j, t)$ denote the respective observations. In my observation model, the pixel coordinates may contain a *geometric* error. The respective grey value signal may contain a geometric and a *photometric error*. The geometric error is due to inaccuracies in the estimate of the correspondence, while the photometric error is due to camera noise.

The observation model for the seed pixel $\boldsymbol{x}_i$ and its grey value $s_i(\boldsymbol{x}_i, t)$ is given by:

$$\hat{\boldsymbol{x}}_i = \boldsymbol{x}_i, \tag{3.117}$$

$$\hat{s}_i(\hat{\boldsymbol{x}}_i, t) = s_i(\boldsymbol{x}_i, t) + e_{photo}. \tag{3.118}$$

The observation model reflects that the seed pixel does not contain a geometric error (Eq. 3.117), as it is (automatically) selected prior learning any correspondence. However, the observed grey value will contain a photometric error e_{photo} (Eq. 3.118).

The observation model for $\boldsymbol{y}_j$ and $s_j(\boldsymbol{y}_j, t)$ is given by:

$$\hat{\boldsymbol{y}}_j = \boldsymbol{y}_j + \boldsymbol{e}_{spatial}, \tag{3.119}$$

$$\hat{s}_j(\hat{\boldsymbol{y}}_j) = s_j(\hat{\boldsymbol{y}}_j, t) + e_{photo}, \tag{3.120}$$

with $\boldsymbol{e}_{spatial} = (e_u, e_v)^T$ representing the spatial error by a shift in x and y direction. Besides the camera noise, the observed grey value now depends on the spatial error of the correspondence estimate as well. Note that the spatial error is vector valued, while the photometric error is a scalar value.

3.6.4 Filling the Comparagram

The simplest strategy to generate observation pairs $\{\hat{s}_i(\boldsymbol{x}_i), \hat{s}_j(\hat{\boldsymbol{y}}_j)\}_P$ is to read off the signal values at the given spatial locations (seed pixel and its correspondence) for many frames[2]. However, this strategy may induce severe outliers. Even if the spatial error term is rather small (less than a few pixels) the induced photometric error could be orders of magnitude larger. To see this, regard Fig. 3.24 (left) where two views of a synthetic scene are shown. Assume that for the seed pixel in the left view, its corresponding pixel in the right view contains a spatial error of one pixel, the smallest achievable error disregarding sub-pixel accuracy. At the regarded time step, the seed pixel is covered by the white square, thus its signal value will be close to the maximum grey value. However, the signal value at the correspondence estimate lies within the dark background and its grey value will be close to the minimum grey value. Adding this grey value pair to the comparagram would obviously generate a severe outlier. Next, regard Fig. 3.24 (right). While the spatial error is still present, the photometric error is in the signal noise range (assuming we regard GVTF transformed grey values), as the seed pixel, as well as its correspondence estimate lie within a homogeneous area and not at an edge as before.

The example motivates that a signal pair should only be added to the comparagram, given that the spatial error is small and the local area around the regarded pixel is essentially homogeneous, i.e., does not contain high frequency components. From a statistical point of view, the (local) autocorrelation function of the regarded pixel should have low curvature, i.e., the signal around the regarded pixel should vary slowly.

I am now interested in determining those correspondences for which the difference between the GVTF transformed grey values is minimum. Obviously, the best that can be achieved is when the magnitude of the spatial error is zero and the photometric error is within the assumed signal noise level. However, the spatial error will rarely be zero in practice and my goal is to identify those signal pairs for which a spatial error in the correspondence estimate leads to a small photometric error.

To this end, I regard how the image signal in the vicinity of the correspondence estimate varies for small shifts $\boldsymbol{d} = (d_x, d_y)^T$ in x and y direction, respectively. Specifically, I regard the squared difference of the image signal at the correspondence estimate

---

[2] Throughout the following, I may refer to this type of approach as the *naïve* scheme.

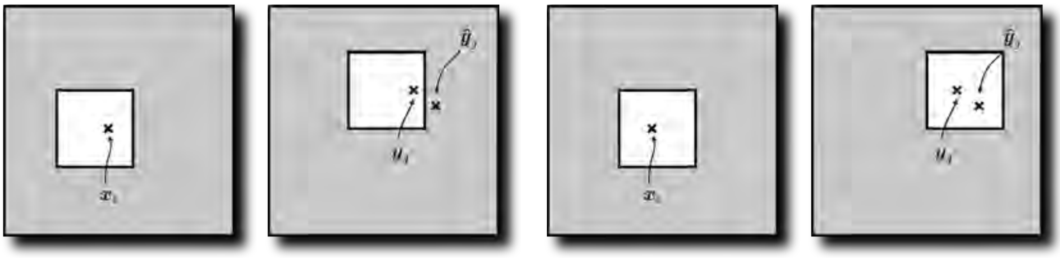


Figure 3.24: **Photometric error due to spatial error:** Visualisation of how a small geometric error may induce a large photometric error, depending on local signal structure (left) in the vicinity of structured/textured regions and (right) in homogenous areas. See text for details.

and its shifted counterpart within a small window around the correspondence estimate according to:

$$Q(\boldsymbol{d}) \overset{def.}{=} \sum_k w(\hat{\boldsymbol{y}}_{j;k})(\hat{s}_j(\hat{\boldsymbol{y}}_{j;k} + \boldsymbol{d}) - \hat{s}_j(\hat{\boldsymbol{y}}_{j;k}))^2. \tag{3.121}$$

Note that the summation $\sum_k$ is over a rectangular subset (=patch) of pixels indexed by $\hat{\boldsymbol{y}}_{j;k}$ where the patch is centred on $\hat{\boldsymbol{y}}_j$. The term $w(\hat{\boldsymbol{y}}_{j;k})$ is an additional weight factor which allows to have pixels close to the centre of the window more importance than those at the border (cf. (Szeliski 2010)).

Intuitively, we should only add those signal pairs to the comparagram, for which Eq. 3.121 is small despite a shift by $\boldsymbol{d}$. In this case, the influence of the spatial error on the photometric error is assumed to be negligible as the correspondence then lies within a homogeneous area, as has been illustrated in Fig. 3.24.

For the following analysis, I assume that the correspondence estimate is sufficiently close to the true correspondence, but that it may contain a spatial error.

I approximate the spatial error dependent term in Eq. 3.121 by a Taylor expansion according to:

$$\hat{s}_j(\hat{\boldsymbol{y}}_j + \boldsymbol{d}) = \hat{s}_j(\hat{\boldsymbol{y}}_j) + \left.\frac{\partial \hat{s}_j}{\partial x}\right|_{\hat{\boldsymbol{y}}_j} \cdot d_x + \left.\frac{\partial \hat{s}_j}{\partial y}\right|_{\hat{\boldsymbol{y}}_j} \cdot d_y + \mathcal{O}(d_x^2, d_y^2), \tag{3.122}$$

$$\approx \hat{s}_j(\hat{\boldsymbol{y}}_j) + \left.\frac{\partial \hat{s}_j}{\partial x}\right|_{\hat{\boldsymbol{y}}_j} \cdot d_x + \left.\frac{\partial \hat{s}_j}{\partial y}\right|_{\hat{\boldsymbol{y}}_j} \cdot d_y, \tag{3.123}$$

where $\mathcal{O}(d_x^2, d_y^2)$ denotes the order of the approximation error.

Plugging Eq. 3.123 in Eq. 3.121 we obtain:

$$
\begin{aligned}
Q(\boldsymbol{d}) &= \sum_k w(\hat{\boldsymbol{y}}_{j;k}) \left[\hat{s}_j(\hat{\boldsymbol{y}}_{j;k} + \boldsymbol{d}) - \hat{s}_j(\hat{\boldsymbol{y}}_{j;k})\right]^2 , && (3.124)\\
&\approx \sum_k w(\hat{\boldsymbol{y}}_{j;k}) \left[\hat{s}_j(\hat{\boldsymbol{y}}_{j;k}) + \left.\frac{\partial \hat{s}_j}{\partial x}\right|_{\hat{\boldsymbol{y}}_{j;k}} \cdot d_x + \left.\frac{\partial \hat{s}_j}{\partial y}\right|_{\hat{\boldsymbol{y}}_{j;k}} \cdot d_y - \hat{s}_j(\hat{\boldsymbol{y}}_{j;k})\right]^2 , && (3.125)\\
&= \sum_k w(\hat{\boldsymbol{y}}_{j;k}) \left[\left.\frac{\partial \hat{s}_j}{\partial x}\right|_{\hat{\boldsymbol{y}}_{j;k}} \cdot d_x + \left.\frac{\partial \hat{s}_j}{\partial y}\right|_{\hat{\boldsymbol{y}}_{j;k}} \cdot d_y\right]^2 , && (3.126)\\
&= \sum_k w(\hat{\boldsymbol{y}}_{j;k}) \left[\left(\left.\frac{\partial \hat{s}_j}{\partial x}\right|_{\hat{\boldsymbol{y}}_{j;k}}\right)^2 \cdot d_x^2 + 2\left.\left(\frac{\partial \hat{s}_j}{\partial x}\frac{\partial \hat{s}_k}{\partial y}\right)\right|_{\hat{\boldsymbol{y}}_{j;k}} \cdot d_x \cdot d_y + \left(\left.\frac{\partial \hat{s}_{j;k}}{\partial y}\right|_{\hat{\boldsymbol{y}}_{j;k}}\right)^2 \cdot d_y^2\right] . && (3.127)
\end{aligned}
$$

Rearranging Eq. 3.127 we obtain:

$$
\begin{aligned}
Q(\boldsymbol{d} = (d_x, d_y)^T) &= \begin{bmatrix} d_x & d_y \end{bmatrix} \sum_k w(\hat{\boldsymbol{y}}_{j;k}) \begin{bmatrix} \left(\left.\frac{\partial \hat{s}_j}{\partial x}\right|_{\hat{\boldsymbol{y}}_{j;k}}\right)^2 & \left.\left(\frac{\partial \hat{s}_j}{\partial x}\frac{\partial \hat{s}_j}{\partial y}\right)\right|_{\hat{\boldsymbol{y}}_{j;k}} \\ \left.\left(\frac{\partial \hat{s}_j}{\partial x}\frac{\partial \hat{s}_j}{\partial y}\right)\right|_{\hat{\boldsymbol{y}}_{j;k}} & \left(\left.\frac{\partial \hat{s}_j}{\partial y}\right|_{\hat{\boldsymbol{y}}_{j;k}}\right)^2 \end{bmatrix} \begin{bmatrix} d_x \\ d_y \end{bmatrix} , && (3.128)\\
&= \boldsymbol{d}^T \boldsymbol{A}[\hat{\boldsymbol{y}}_j]\boldsymbol{d}. && (3.129)
\end{aligned}
$$

We may rewrite $\boldsymbol{A}$ in a more concise form as follows. Let $\boldsymbol{I}_x$ and $\boldsymbol{I}_y$ denote the spatial image derivatives of s_j in x and y direction, respectively. Then we have:

$$
\boldsymbol{A}[\hat{\boldsymbol{y}}_j] = \begin{bmatrix} \sum_k w(\hat{\boldsymbol{y}}_{j;k})\boldsymbol{I}_x^2 & \sum_k w(\hat{\boldsymbol{y}}_{j;k})\boldsymbol{I}_x\boldsymbol{I}_y \\ \sum_k w(\hat{\boldsymbol{y}}_{j;k})\boldsymbol{I}_y\boldsymbol{I}_x & \sum_k w(\hat{\boldsymbol{y}}_{j;k})\boldsymbol{I}_y^2 \end{bmatrix} . \tag{3.130}
$$

The summation and weighting over a local neighbourhood is usually implemented via convolution with a suitable low-pass filter $\mathcal{G}$:

$$
\boldsymbol{A}[\hat{\boldsymbol{y}}_j] = \begin{bmatrix} \mathcal{G} * \boldsymbol{I}_x^2 & \mathcal{G} * \boldsymbol{I}_x\boldsymbol{I}_y \\ \mathcal{G} * \boldsymbol{I}_y\boldsymbol{I}_x & \mathcal{G} * \boldsymbol{I}_y^2 \end{bmatrix} . \tag{3.131}
$$

The quadratic form in Eq. 3.129 is governed by matrix $\boldsymbol{A}$, which is known as the *structure tensor* (Jähne 2012). The structure tensor is well known in computer vision and appears, e.g., in classic approaches to motion estimation, keypoint detection and tracking (Harris and Stephens 1988; Lucas and Kanade 1981; Shi and Tomasi 1994). It can be shown that the structure tensor approximates the Hessian of the signal's auto-correlation function (Förstner 1991). Hence, the structure tensor may be determined without the need to determine spatial image derivatives (Mester 2000).

The structure tensor summarises and describes the local structure around pixel $\hat{\boldsymbol{y}}_j$. The structure tensor is a symmetric matrix and thus its eigenvectors form an orthogonal basis (Golub and Loan 2012). Let λ_1 and λ_2 be the eigenvalues of the structure tensor with $\lambda_1 \geq \lambda_2$. The eigenvalues are a measure of the signal's variance along its major axes (Bigun 2006) and allow to classify the local structure according to:

$$\lambda_1 \approx \lambda_2 \approx 0 \leftrightarrow \text{ no structure, homogeneous region,} \tag{3.132}$$

$$\lambda_1 \gg 0, \lambda_2 \approx 0 \leftrightarrow \text{ 1d structure, line like,} \tag{3.133}$$

$$\lambda_1 \approx \lambda_2 \gg 0 \leftrightarrow \text{ 2d structure, corner like.} \tag{3.134}$$

See Sec. 3.5 for details on the (closed form) computation of the structure tensor's eigenvalues.

In keypoint detection, one is interested in identifying distinct or unique pixels in an image. Intuitively, these are points with a distinct local structure where small shifts of the signal in *any* spatial direction results in large values of Eq. 3.121. Hence, keypoints are those for which *both* eigenvalues of the structure tensor are large. Pixels for which the second eigenvalue of the structure tensor is close to zero may be considered as line points for which only signal changes orthogonal to the 1-D structure can be detected (aperture problem).

Let us return to the task of filling the comparagram. As our goal is to minimise the influence of the spatial error in the correspondence estimate, we should only regard grey values of pixel pairs where the eigenvalues of the associated structure tensors are *both* close to zero, respectively. Then, the pixels lie within a homogenous region.

Now let $\lambda_{1,i}$ and $\lambda_{2,i}$ with $\lambda_{1,i} \geq \lambda_{2,i}$ be the eigenvalues of the structure tensor associated with seed pixel $\boldsymbol{x}_i$, and let $\lambda_{1,j}$ and $\lambda_{2,j}$ with $\lambda_{1,j} \geq \lambda_{2,j}$ be the eigenvalues of the structure tensor associated with the correspondence estimate $\hat{\boldsymbol{y}}_j$. I then define the decision rule when to add a signal pair to the comparagram according to:

$$\begin{cases} \lambda_{1,i} \leq T_\phi \text{ and } \lambda_{1,j} \leq T_\phi, & \text{add } \{\hat{s}_i(\boldsymbol{x}_i), \hat{s}_j(\hat{\boldsymbol{y}}_j)\} \text{ to comparagram,} \\ \text{else,} & \text{reject.} \end{cases} \tag{3.135}$$

Besides this proposed method for filling the comparagram I will compare the approach with the *naïve* scheme in Sec. 3.6.6, in which the comparagram is filled while disregarding the spatial error in the correspondence estimate.

3.6.5 Estimating the GVTF

I am now interested in estimating the GVTF, based on a given comparagram. To this end, I develop a statistical model, in which with each grey value in view $\mathcal{C}_i$, a distribution over the grey values in view $\mathcal{C}_j$ is associated. This allows to represent and quantify the uncertainties involved in the grey value mapping, which are due to signal noise and errors in the correspondence estimate as described earlier.

Let $\mathcal{G} = \{0, .., 255\}$ be the discrete set of observable grey values in view $\mathcal{C}_i$ and view $\mathcal{C}_j$. Let $g_i \in \mathcal{G}$ be a grey value observed in view $\mathcal{C}_i$. The GVTF maps g_i to a grey value $g_j = \phi_{ij}(g_i)$ in view $\mathcal{C}_j$. I regard the GVTF as a distribution over the (2-D) space of grey value pairs $\{(g_i, g_j) \in \mathcal{G} \times \mathcal{G}\}$. For a grey value g_i in view $\mathcal{C}_i$, I regard its first order and second order central moment of the GVTF transformed grey value, which are given as:

$$\mathbb{E}\left[\phi_{ij}(g_i)\right] = \mu_{\phi_{ij}(g_i)}, \tag{3.136}$$

and

$$\mathbb{E}\left[\left(\phi_{ij}(g_i) - \mu_{\phi_{ij}(g_i)}\right)^2\right] = \sigma^2_{\phi_{ij}(g_i)}. \tag{3.137}$$

If we assume the signal noise to be Gaussian distributed, the distribution of grey value g_i over the grey values in view $\mathcal{C}_j$ is given as:

$$\phi_{ij}(g_i) \sim \mathcal{N}(\mu_{\phi_{ij}(g_i)}, \sigma^2_{\phi_{ij}(g_i)}). \tag{3.138}$$

Learning the GVTF then amounts to determine empirical estimates of first and second order moments for all $k = |\mathcal{G}|$ observable grey values.

We may estimate the moments based on the data available in the comparagram. Let $\mathbb{1}\{\mathcal{X} = \mathcal{Y}\}$ denote the indicator function which evaluates to 1, given that the argument is true. Then, for grey value g_i, the empirical mean of its GVTF transformed counterpart is given by:

$$\hat{\mu}_{\phi_{ij}(g_i)} = \frac{1}{N} \sum_{p=1}^{P} \hat{s}_j(\hat{\boldsymbol{y}}_{j;p}) \cdot \mathbb{1}\{\hat{s}_i(\boldsymbol{x}_{i;p}) = g_i\}, \tag{3.139}$$

with $N = \sum_p^P \mathbb{1}\{\hat{s}_i(\boldsymbol{x}_{i;p}) = g_i\}$.

The empirical variance is given by:

$$\hat{\sigma}^2_{\phi_{ij}(g_i)} = \frac{1}{N-1} \sum_{p=1}^{P} (\hat{s}_j(\hat{\boldsymbol{y}}_{j;p}) - \hat{\mu}_{\phi_{ij}(g_i)})^2 \mathbb{1}\{\hat{s}_i(\boldsymbol{x}_{i;p}) = g_i\}. \tag{3.140}$$

If the true signal noise is assumed to be i.i.d., then the empirical variances $\hat{\sigma}^2_{\phi_{ij}(g_i)}$ should be identical and are itself an estimate of the signal noise variance in view $\mathcal{C}_j$.

Observe that the signal values of the seed pixel itself contain noise in the range of the assumed signal noise of view $\mathcal{C}_i$, but that this is not explicitly represented in the equations for the empirical means and variances in Eq. 3.139 and Eq. 3.140, i.e., I here utilise an unsymmetrical error model. Clearly, we may formulate the model as a symmetric error model (as a TLS problem), however, this is considered future work.

3.6.5.1 Online Update Scheme

In order to learn and update the GVTF online, we have to update the set of statistical moments. To this end, we do *not* need to store an ever growing comparagram. Instead, we only have to update a set of *sufficient statistics* per grey value g_i. The formulae for the empirical mean and variance only depend on the following quantities:

$$N_{g_i} = \sum_{p}^{P} \mathbb{1}\{\hat{s}_i(\boldsymbol{x}_{i;p}) = g_i\}, \tag{3.141}$$

$$S_{g_i} = \sum_{p}^{P} \hat{s}_j(\hat{\boldsymbol{y}}_{j;p}) \cdot \mathbb{1}\{\hat{s}_i(\boldsymbol{x}_{i;p}) = g_i\}, \tag{3.142}$$

$$Q_{g_i} = \sum_{p}^{P} \hat{s}_j(\hat{\boldsymbol{y}}_{j;p})^2 \cdot \mathbb{1}\{\hat{s}_i(\boldsymbol{x}_{i;p}) = g_i\}. \tag{3.143}$$

From N_{g_i}, S_{g_i}, Q_{g_i}, the empirical mean and variance are given as:

$$\hat{\mu}_{\phi_{ij}(g_i)} = \frac{S_i}{N_i}, \tag{3.144}$$

$$\hat{\sigma}^2_{\phi_{ij}(g_i)} = \frac{Q_i}{N_i} - \left(\frac{S_i}{N_i}\right)^2. \tag{3.145}$$

For the Gaussian noise model, we may then directly plug in these values of mean and variance.

3.6.5.2 Deriving a Functional Form

Let us now return to the problem of deriving a functional representation of the GVTF, by means of a polynomial of order N. Given the order of the polynomial, the task is to estimate the parameter vector $\boldsymbol{\theta} = \{\theta_0, .., \theta_N\}$. To this end, I may make use of the previously introduced probabilistic model of the GVTF as follows. I regard a grey value g_i and the empirical mean of its GVTF transformed grey value $\hat{\mu}_{\phi_{ij}(g_i)}$ as a data point, to which I fit the parameter vector of the functional GVTF model in a least squares sense. Additionally, I may use the (inverse) empirical variances $\hat{\sigma}^2_{\phi_{ij}(g_i)}$ as weight factors, giving a data point with small variance a larger influence on the parameters to be estimated. Under this model one may then enforce a monotone GVTF with additional constraints if desired.

Let $\boldsymbol{z}$ be the measurement vector with:

$$\boldsymbol{z} = \begin{pmatrix} \hat{\mu}_{\phi_{ij}(g_i=0)} \\ \vdots \\ \hat{\mu}_{\phi_{ij}(g_i=255)} \end{pmatrix}, \tag{3.146}$$

and let $\boldsymbol{H}$ be the coefficient or observation matrix:

$$\boldsymbol{H} = \begin{pmatrix} (g_i = 0)^0 & (g_i = 0)^1 & \dots & (g_i = 0)^N \\ \vdots & \vdots & \vdots & \vdots \\ (g_i = 255)^0 & (g_i = 255)^1 & \dots & (g_i = 255)^N \end{pmatrix}. \tag{3.147}$$

Furthermore, let $\boldsymbol{v}$ be the observation noise vector. The linear observation model is then defined as:

$$\boldsymbol{z} = \boldsymbol{H}\boldsymbol{\theta} + \boldsymbol{v}, \tag{3.148}$$

where $\boldsymbol{\theta}$ is the sought parameter vector defined as:

$$\boldsymbol{\theta} = \begin{pmatrix} \theta_0 \\ \vdots \\ \theta_N \end{pmatrix}. \tag{3.149}$$

Equation 3.148 represents the standard linear model. Minimising the squared difference between the observations and their approximation leads to the normal equation (Kay 1993):

$$\hat{\boldsymbol{\theta}} = (\boldsymbol{H}^T\boldsymbol{H})^{-1}\boldsymbol{H}^T\boldsymbol{z}. \tag{3.150}$$

Additionally, we may make use of the empirical variances as follows. Let $\boldsymbol{W}$ be the weight matrix defined as:

$$\boldsymbol{W} = \mathrm{diag}(\hat{\sigma}^2_{\phi_{ij}(g_i=0)}, \hat{\sigma}^2_{\phi_{ij}(g_i=1)}, \dots \hat{\sigma}^2_{\phi_{ij}(g_i=255)})^{-1}. \tag{3.151}$$

Then, the weighted least squares estimator of the sought parameter vector (respecting the uncertainties in the data points) is given by (ibid.):

$$\hat{\boldsymbol{\theta}} = (\boldsymbol{H}^T\boldsymbol{W}\boldsymbol{H})^{-1}\boldsymbol{H}^T\boldsymbol{W}\boldsymbol{z}. \tag{3.152}$$

Note that the sought parameter vector may be estimated via a total least squares fit, as the regarded grey values in $\mathcal{C}_i$ and $\mathcal{C}_j$ are afflicted with noise. However, this remains future work.

3.6.5.3 Incorporating the GVTF

Recall from Sec. 3.4.3 that the temporal update of the posterior distribution (and its approximate counterpart) is performed on GVTF transformed signal values. Therefore, I transform the seed pixel's grey value via ϕ_{ij} prior the update step. Recall that the GVTF will induce additional uncertainties in the matching process which are at best on par with the signal noise variance, as the GVTF estimate itself is noisy. Recall that the noise variance for the seed pixel channel (cf. 3.3.4) is given by $\sigma^2_{g|s}$. Assuming that the signal and GVTF noise are independent of each other, I then replace this term by $\sigma^2_{g|s} + \sigma^2_{\phi_{ij}}$.

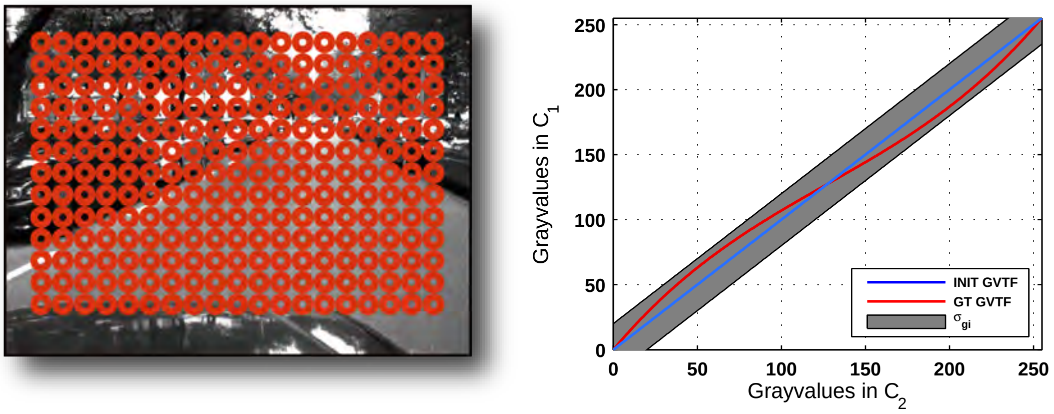


Figure 3.25: **GVTF measuring points and initialisation:** (left) Sample frame of view $\mathcal{C}_i$ of sequence `GUCar` where measurement points are marked by a red circle, (right) Initialised GVTF in blue and its initial uncertainty area in grey. Ground truth GVTF plotted in red. See text for details. Figure only interpretable when viewed in colour.

3.6.6 Simulation

Let me now simulate the proposed GVTF learning scheme on a pseudo stereo setup. To this end, I regard the left view of sequence `GUCar` as camera $\mathcal{C}_1$ and generate a second view $\mathcal{C}_2$ as the *same* left view from `GUCar`. I then select 240 correspondences $\{\boldsymbol{x}_i \leftrightarrow \boldsymbol{y}_j\}_{240}$ laid out in a regular grid.

Obviously, the true correspondences in the second view are identical to the seed pixel coordinates. I now induce a geometric error on the correspondences. Specifically, I independently add zero mean Gaussian noise to each of the correspondence coordinates and to each of its x and y coordinates according to:

$$\hat{\boldsymbol{y}}_j = \boldsymbol{y}_j + \boldsymbol{e}_{spatial}, \tag{3.153}$$

with

$$\boldsymbol{e}_{spatial} \sim \mathcal{N}(\boldsymbol{0}, \sigma^2_{\boldsymbol{e}_{spatial}} \cdot \mathcal{I}), \tag{3.154}$$

where $\mathcal{I}$ denotes the 2×2 identity matrix. In the following, I will only regard the noise afflicted set of correspondences $\{\boldsymbol{x}_i \leftrightarrow \hat{\boldsymbol{y}}_j\}_{240}$.

I vary the spatial noise standard deviation within $\sigma_{\boldsymbol{e}_{spatial}} = [1, 2, 3, 4, 5]$ and vary the comparagram threshold from Eq. 3.135 within $T_\phi = [1, 2, 4, 6, 8, 10]$. Hence, for a given noise std., the average spatial endpoint error (in pixels) is given by $\mathbb{E}\left[EP\right] = \sqrt{\sigma^2_{\boldsymbol{e}_{spatial}} + \sigma^2_{\boldsymbol{e}_{spatial}}}$. I define a GVTF as a 3rd order polynomial with parameter vector $\boldsymbol{\theta} = \{\theta_0 = 0, \theta_1 = 1.53, \theta_2 = -6.2e-3, \theta_2 = -1.64e-5\}$. Note that the actual choice of the GVTF is not relevant in this simulation, as the correspondences are already given. For the moment being, I will not add additional noise to the images, as I focus on the influence of the geometric error on the GVTF estimate.

Figure 3.25 visualises the chosen seed pixels for a sample frame of sequence `GUCar`. Furthermore, the figure shows the ground truth GVTF according to the given parameters as well as its initialisation. I initialise the GVTF estimate as the identity function. The initial variances $\hat{\sigma}^2_{\phi_{ij}(g_i)}$ have to be large enough, such that the true but unknown GVTF is encapsulated. However, this is only needed in practice, when the GVTF is used during correspondence learning and when nothing is known about the true GVTF. Then we have have to accept large (GVTF transformed) grey value differences. Only over time, these variances will decrease the more we learn about the true GVTF. In the simulation I set the initial variances to a fixed value of $\sqrt{20^2}$ which results in an uncertainty area of width ± 20 grey values as shown in Fig. 3.25 (right). In order to initialise the GVTF to the identity I set the parameters according to $N_{(g_i=k)} = 1$, $S_{(g_i=k)} = k$, $Q_{(g_i=k)} = \hat{\sigma}^2_{\phi_{ij}(g_i)} + S^2_{(g_i=k)}$.

Next, for each parameter setting, I build a comparagram over the first 2,500 frames of sequence `GUCar`, by means of updating variables $N_{g_i}, S_{g_i}, Q_{g_i}$. This is done for the naïve update scheme and the proposed update scheme, based on the eigenvalues of the structure tensors (cf. Eq. 3.135).

Figure 3.26 visualises estimated parameters $\hat{\mu}_{\phi_{ij}(g_i)}$ and $\hat{\sigma}^2_{\phi_{ij}(g_i)}$ by means of a grey error bar centred on the mean with a width of one standard deviation. The figure shows plots for $\sigma_{\boldsymbol{e}_{spatial}} = 2$, $\sigma_{\boldsymbol{e}_{spatial}} = 4$ and thresholds $T_\phi = [2, 4, 8]$. Recall that the mean and variance are given according to Eq. 3.144 and Eq. 3.145.

For $\sigma_{\boldsymbol{e}_{spatial}} = 2$ and $T_\phi = 2$, it can be seen that the initial uncertainty for grey values up to 150 considerably decreased and encapsulates the true GVTF. For grey values above 150, the uncertainty is still large but still encapsulates the true GVTF. For $T_\phi = 4$ and $T_\phi = 8$ it can be seen that the uncertainty decreases over the whole grey value range. Clearly, the larger T_ϕ is set, the more grey value pairs will be added to the comparagram. However, the larger the threshold is set, the more will the local area around the seed pixel and its correspondence deviate from a homogenous area. Depending on the geometric error, outliers may be added to the comparagram. This can be seen in Figure 3.26 for $\sigma_{\boldsymbol{e}_{spatial}} = 4$. While the mean values $\hat{\mu}_{\phi_{ij}(g_i)}$ seem to be close to the true GVTF, we see that the associated uncertainties are for some of the grey values larger than the initial value. This effect reduces for $T_\phi = 8$, as more measurements are added to the comparagram.

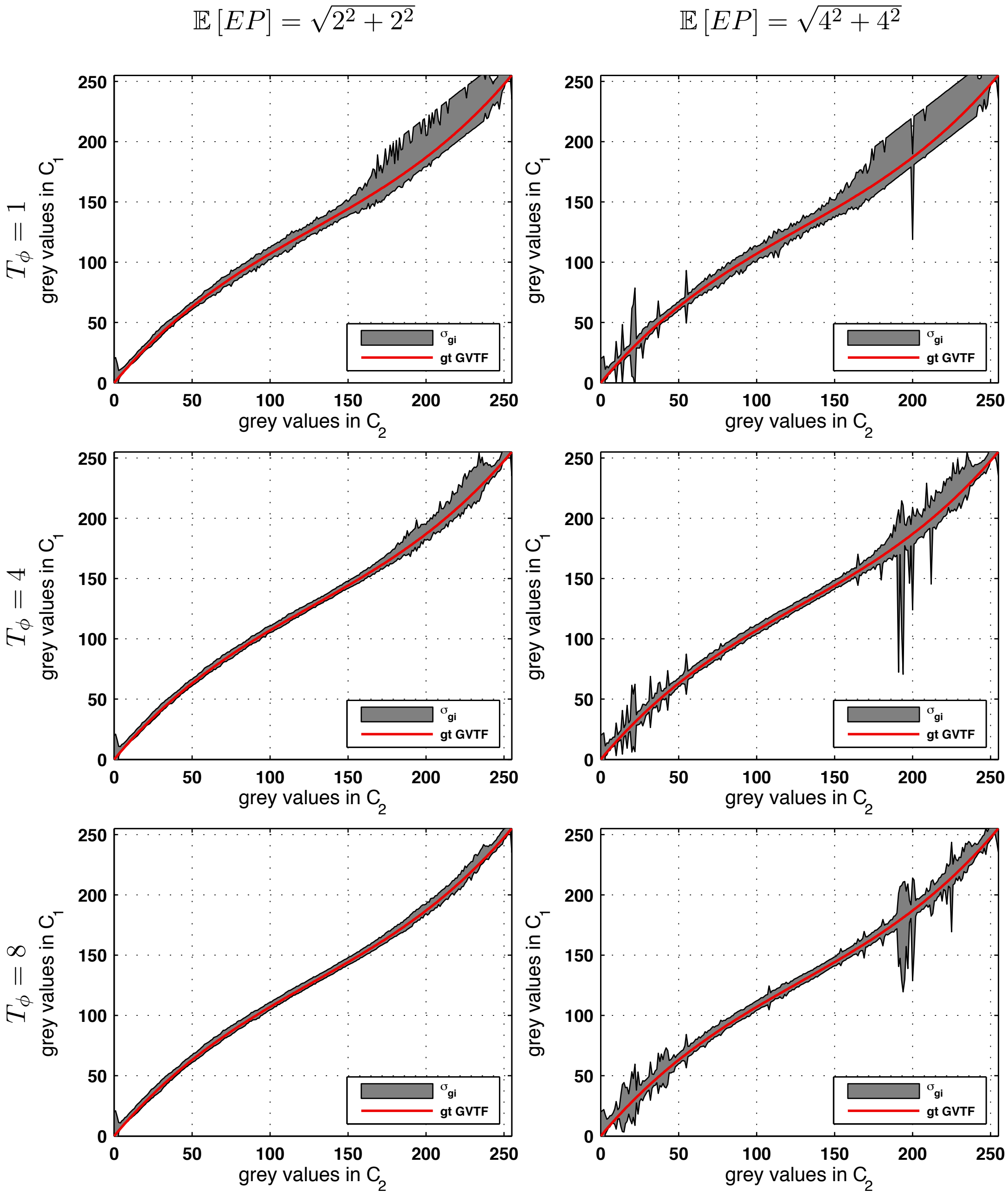


Figure 3.26: **Learning the GVTF via the proposed method:** For different average spatial errors, and different threshold levels. Estimates of $\hat{\mu}_{\phi_{ij}(g_i)}$ and $\hat{\sigma}^2_{\phi_{ij}(g_i)}$ define the grey uncertainty area. Ground truth GVTF in red. See text for details. Figure only interpretable when viewed in colour.

Figure 3.27 compares the proposed and the naïve update scheme. We see that an average spatial error of only 1 pixel induces severe errors in the naïve GVTF estimate. As expected, this becomes even worse when the average spatial error increases.

Next, I determine a functional form of the GVTF, both via a standard (non-weighted) and a weighted least squares fit to $\hat{\mu}_{\phi_{ij}(g_i)}$ (with weights $(\hat{\sigma}^2_{\phi_{ij}(g_i)})^{-1}$) as described in Sec. 3.6.5.2. I assume that the order of the GVTF is known (=3). Figure 3.28 visualises the estimated GVTFs, both for the proposed and the naïve learning scheme after 500, $1,500$ and $2,500$ frames have been processed. The spatial error was set to $\sigma_{\boldsymbol{e}_{spatial}} = 4$ and the threshold to $T_\phi = 10$. In Fig. 3.28 (bottom) I show the average squared residuals between the true GVTF and its estimate. It can be seen that the proposed approach outperforms the naïve method. Overall, the GVTF estimates for the proposed method are less sensitive to the induced spatial error. When comparing the average squared residuals of the weighted and non-weighted fit, we see that the weighted fit improves the GVTF estimate, especially in the early learning phase. Furthermore, using the variance estimates as weighting factors further improves the GVTF estimate. Compare the residuals of the proposed and the naïve scheme; the residuals for the naïve scheme are above 150, while they are close to zero for the proposed scheme and 2,500 processed frames.

To summarise, the simulation has shown the principle applicability of the proposed GVTF learning scheme. In practice, the GVTF estimation will of course be more challenging, as reliable correspondences have to be learnt first.

3.7 TCA on the Pyramid / In Scale Space

So far, I made the implicit assumption that TCA is applied to the full scale images as given by a binocular camera setup. Let the spatial domain of $\mathcal{C}_i$ be of size $M \times N$. Without loss of generality, assume that the spatial domain of $\mathcal{C}_j$ is of the same size. For a given seed pixel $\boldsymbol{x}_i$ in $\mathcal{C}_i$, TCA estimates its correspondence distribution over all $M \cdot N$ candidate pixel locations in $\mathcal{C}_j$. While we may summarise the correspondence distribution with a few attributes (cf. Sec. 3.5), this can only be done once a reliable estimate of the true correspondence distribution has been learnt. The amount of memory needed by TCA thus linearly scales with the number of active seed pixels.

Clearly, we may decrease the amount of memory needed, if we could limit the search space for the sought correspondence in $\mathcal{C}_j$. To this end, we may apply TCA in a coarse to fine manner based on a spatial pyramid of the input images. The coarse to fine approach also allows to implement an energy/memory constraint which could be given by the overall system limits in which TCA is applied. This is also in line with a biological view on vision, where learning is believed to operate in a coarse to fine manner (Marr 1982).

I build on the common *Gaussian pyramid* (Burt and Adelson 1983). For a given image $I_{t,i}$ from $\mathcal{C}_i$ at time t, its pyramidal representation is given as follows. The input image represents the base level of the pyramid:

$$\mathcal{G}^0 = I_{t,i}. \tag{3.155}$$

The $(i+1)$ pyramid level $\mathcal{G}^{i+1}$ is a low pass filtered and subsequently down sampled version of level $\mathcal{G}^i$. Typically, subsampling is performed by a factor of 2 in each dimension (see for example (Jähne 2012, p. 483) or (Szeliski 2010, p. 144) for details).

Returning to the application of TCA, the number of correspondence candidates $|\mathcal{H}|$ on the i-th pyramid level is given by:

$$|\mathcal{H}| = N \cdot M \cdot \left(\frac{1}{4}\right)^i. \tag{3.156}$$

The idea is now to learn a correspondence distribution via TCA on a coarse scale and using the learnt correspondence distribution to constrain the search space (=correspondence candidates) on a finer level (not necessarily the base level).

Recall that the correspondence distribution's covariance matrix encodes the confidence about the learnt correspondence. The level of confidence about the correspondence distribution is encoded by the eigenvalues λ_1 and λ_2 of the covariance matrix and the direction by the respective eigenvectors (cf. Sec. 3.5). For reasons of computational ease, I may constrain the search space on level $i-1$ to a square and axis aligned window of dimensions $\lambda_1 \cdot 2^i \times \lambda_1 \cdot 2^i$, instead of using an elliptical representation as is given by the eigenvalues and eigenvectors. The window is then centred on the coordinates of the empirical mean of the correspondence distribution. Further details and experimental results will be given in Sec. 4.

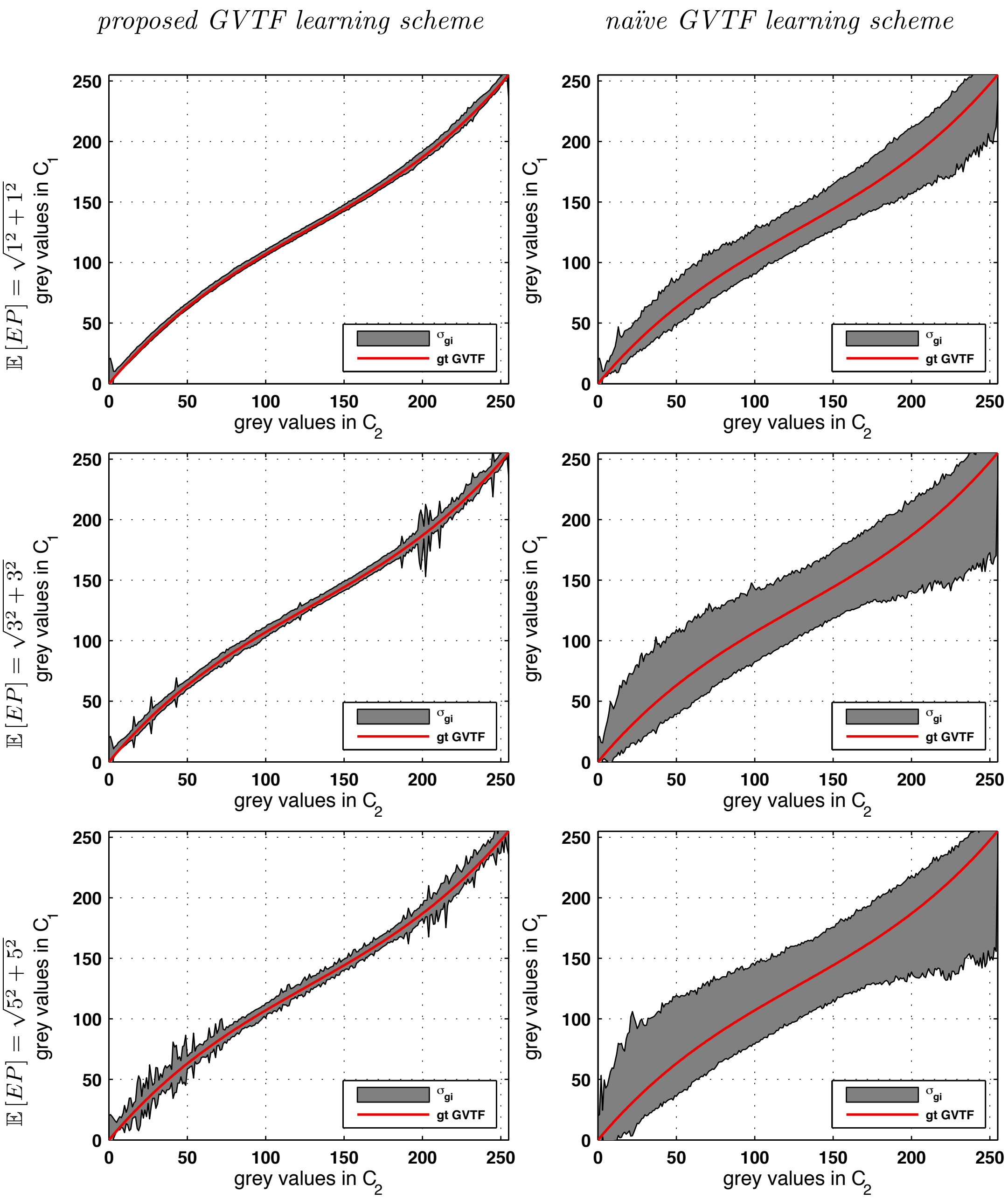


Figure 3.27: **Proposed vs naïve learning scheme:** For different average spatial errors, and different threshold levels. Estimates of $\hat{\mu}_{\phi_{ij}(g_i)}$ and $\hat{\sigma}^2_{\phi_{ij}(g_i)}$ define the grey uncertainty area. Ground truth GVTF in red. See text for details. Figure only interpretable when viewed in colour.

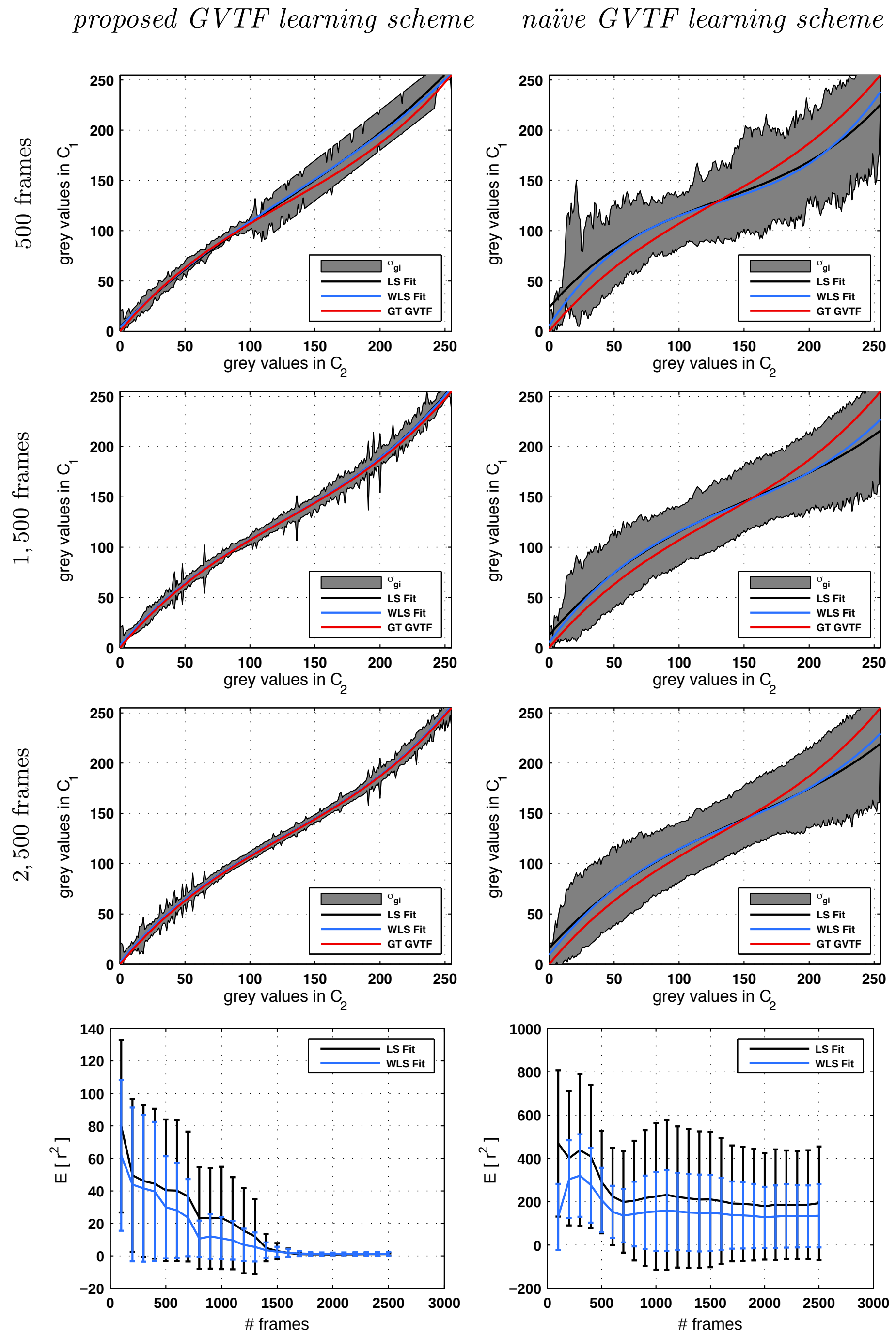


Figure 3.28: **Functional GVTF fitting:** for $\sigma_{\mathbf{e}_{spatial}} = 4$ and $T_\phi = 10$. Weighted and non weighted least squares fit to $\hat{\mu}_{\phi_{ij}(g_i)}$ with weights $(\hat{\sigma}^2_{\phi_{ij}(g_i)})^{-1}$ after 500, 1,500 and 2,500 processed frames are shown. (bottom) Average residuals of the weighted and non-weighted least squares fit. Error bars span 2 times the standard deviation of the squared residuals. See text for details. Figure only interpretable when viewed in colour.

3.8 Summary and Conclusion

I introduced the *Temporal Coincidence Analysis* (TCA) approach to learn pixel correspondences between pairs of views. I derived TCA as a statistical model and showed that the basic principle of detecting and matching temporal events in the grey value signal can be cast as a temporal update scheme of an associated posterior distribution. Correspondences are then represented by a correspondence distribution over the spatial coordinates of the view in which the correspondence is to be learnt. The progress of the learning scheme can be monitored by means of attributes of the correspondence distribution. An important property of TCA is that the uncertainty about a correspondence estimate is explicitly given by means of the second order statistics of the correspondence distribution. This allows to propagate uncertainties to higher level processes which operate on the learnt correspondences, e.g., in the estimation of the fundamental matrix (Brooks et al. 2001; Haralick 2000).

I introduced the *Grey Value Transfer Function* (GVTF), which models the transfer function of grey values in different cameras. The GVTF can be learnt from a set of (learnt) correspondences by fitting a low order polynomial to a comparagram, i.e., a 2-D histogram of corresponding grey values recorded from corresponding pixels. The learning scheme is robust to spatial uncertainties in the correspondence estimates as the local spatial image signal is explicitly regarded. Grey value pairs are only added to the comparagram given that the local signal structure around the regarded pixels is homogeneous.

Simulations of TCA and the GVTF learning scheme have shown that correspondences and transfer functions between pairs of cameras can be learnt by only regarding the temporal signal of single pixels without the need for solving the correspondence problem by means of computing correspondences explicitly.

4 Learning Stereo Correspondences

4.1 Introduction

In the following, I present an evaluation of TCA when applied to binocular (stereo) camera setups. In the first part, the method is analysed based on an artificial binocular setup, where the spatial and photometric transformations between the views can be controlled and are thus known. While the type of correspondence being learnt is restricted to the point-to-point case as explained in the following, the setup allows an evaluation based on ground truth data.

In the second part of the evaluation, results for real world multi-camera setups are presented. This includes real world setups in which the camera views are rotated or perspectively distorted with respect to each other or in which cameras are equipped with different optics and in which cameras are stationary or moving.

4.2 Evaluation

Throughout the evaluation, I assume the cameras to be synchronised and that their relative orientation is fixed over the learning phase. Clearly, the cameras need to have overlapping fields of view, otherwise no spatial correspondences would exist which could be learnt.

4.2.1 Evaluation on Synthetic Data

In the following, I present results obtained on synthetic multi-view data, for which ground truth correspondences are available. Similar to the setup for the GVTF simulation (cf. Sec. 3.6.6), I simulate a binocular camera setup based on a single view. To this end, I regard the monocular sequence `Forrest` as given by camera $\mathcal{C}_1$. The second view $\mathcal{C}_2$ is then given as the first view, which may additionally be transformed by an affine spatial transformation $\boldsymbol{A}$. Within this setup, for each seed pixel $\boldsymbol{x}_i$ in the first view the true corresponding pixel $\boldsymbol{y}_j$ in the second view is given by:

$$\begin{bmatrix} \boldsymbol{y}_j \\ 1 \end{bmatrix} = \boldsymbol{A} \begin{bmatrix} \boldsymbol{x}_i \\ 1 \end{bmatrix}. \tag{4.1}$$

For the results presented in the following, I set $\boldsymbol{A}$ to the identity transformation, merely for visualisation purposes and to ease the understanding of the figures. However, note

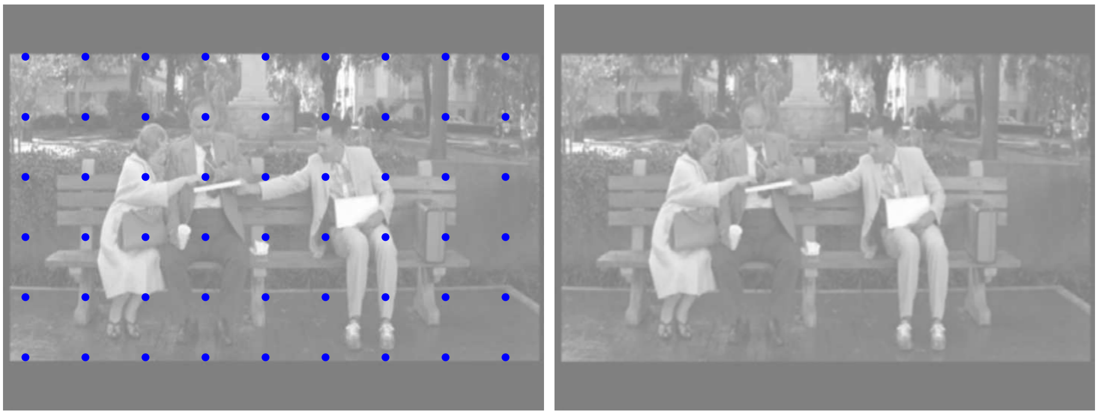


Figure 4.1: **Test setup:** Left and right view of the synthetic binocular camera setup. (left) Sample frame of the left view, here given by sequence `Forrest`. Seed pixels are laid out in a regular grid and are marked by blue dots. (right) Corresponding right view in which correspondences are to be learnt. Initially, the uncertainty about the true correspondence is maximum and will decrease during learning. See text for details. Best viewed in colour.

that TCA is invariant to (in-plane) rotations and translations of the input views as only single pixels are regarded. Therefore, the presented results are transferable for scenarios in which the camera views are rotated and translated w.r.t. each other. Within the evaluation based on real world data, we will see that TCA is also robust to more general perspective transformations of the input data.

Throughout the following experiments, for each frame pair $I_{1,t}$ and $I_{2,t}$ at time t, I add i.i.d. zero mean Gaussian noise independently to every pixel. Note that in this setup, only point-to-point correspondences are to be expected, as the scene depth is constant.

Results presented in the following basically resemble those of the model simulations performed in Sec. 3.3.5 and Sec. 3.6.6. Therefore, I only present several prototypical results for the synthetic setup and introduce the visualisation scheme used throughout the experiments on real world data.

The setup of the evaluation is as follows. I select 54 seed pixels in $\mathcal{C}_1$, laid out in a regular grid as shown in Fig. 4.1. I learn correspondences by means of learning correspondence distributions as described in Sec. 3.4. Recall that the model depends on the parameters:

- σ_s, modelling the standard deviation of the source signal,
- $\sigma_{h|s}$ and $\sigma_{g|s}$, modelling the model uncertainties, i.e., signal noise, uncertainties in the GVTF estimate etc..

Additionally, we may set parameter T_e (=event threshold) to skip model updates which are likely to provide only little information on the true correspondence. As has been discussed previously, events which are very likely to occur will tell us little about the true correspondence, simply because it is very likely that there are too many correspondence candidates (cf. Sec 3.3.4).

4.2.2 Selecting Model Parameters

I determine a suitable value for σ_s by measuring the standard deviation of the event signal value over a representative subset of the frames to be processed. Clearly, in a dynamic scene the std. of the event signal value is likely to change over time and might have to be updated continuously. However, this is considered future work and the presented experiments are based on a fixed value of σ_s.

Parameter $\sigma_{h|s}$ depends on the signal noise and illumination differences among the views. In principle, this parameter allows to define a *matching envelope* which is narrow when the noise level is small and which is wide otherwise. Let the std. of the grey value signal noise be $\sigma_{\mathcal{C}}$ and let κ be a constant value. I then set:

$$\sigma_{h|s} = (2 \cdot \kappa \cdot \sigma_{\mathcal{C}})^2 \tag{4.2}$$

where I typically set $\kappa = 4$. The idea of choosing $\sigma_{h|s}$ in this way is as follows. Under the Gaussian signal noise model, for a given noise standard deviation $\sigma_{\mathcal{C}}$, we have that $99,99\%$ of the values will lie within the interval $\pm 4 \cdot \sigma_{\mathcal{C}}$ (assuming zero mean). According to the definition of the event function (cf. Eq. 3.89), this may result in an event value of $f_e(\kappa \cdot \sigma_{\mathcal{C}}, -\kappa \cdot \sigma_{\mathcal{C}}) = (2 \cdot \kappa \cdot \sigma_{\mathcal{C}})^2$. Similarly, uncertainties due to an unknown GVTF enter $\sigma_{h|s}$ as an additive term, assuming that signal noise and uncertainties of the GVTF estimate are uncorrelated (cf. Sec. 3.6.5.3).

The influence of parameter $\sigma_{h|s}$ on the learning performance is as follows. The larger we set $\sigma_{h|s}$, the more events will be matched as the matching envelope becomes wider. Therefore, we expect that we have to process more frames/events until a specified level of confidence about the learnt correspondence is attained. On the other hand, if $\sigma_{h|s}$ underestimates the true model uncertainty it is likely to match more non-corresponding channels. Clearly, matching of non-corresponding channels may result in a wrong estimate of the true correspondence distribution. I have found that parameter $\sigma_{h|s}$ is rather uncritical as long as it is not underestimating its true (but hidden) value.

In the following experiments, I chose a signal noise level of $\sigma_{\mathcal{C}} = 2$ and an identity GVTF. I estimated the source event signal std. with $\sigma_s = 1,400$ and set $\sigma_{h|s} = \sigma_{g|s} = (2 \cdot \kappa \cdot \sigma_{\mathcal{C}})^2 = 256$. I perform three runs of TCA on 600 frames of sequence `Forrest` where I vary the event threshold in $T_e \in [64, 128, 256]$. After each model update, I update the correspondence distribution's attribute set described in Sec. 3.5.

Figure 4.2 visualises the learnt correspondence distributions after 30, 450 and 590 frames have been processed. Each correspondence distribution is visualised by a red dot, indicating the average correspondence location $\boldsymbol{\mu}_{\boldsymbol{y}_i}$. Additionally, the uncertainty about the learnt correspondence is visualised by means of the distribution's covariance error ellipse, here shown in green.

From the figures corresponding to 30 processed frames, it can be seen that the uncertainty about the learnt correspondence is high as indicated by the large error ellipses. The correspondence estimates seem to cluster in the middle of the right view. This is to be expected, as the correspondence distributions are initialised as a uniform distribution. Hence, the expected correspondence location initially lies in the middle of the image. The more frames are processed, we see that the uncertainty about the correspondence decreases. After 450 processed frames, the uncertainty significantly decreased for most of the seed pixels. After 590 processed frames, the true correspondence has successfully been learnt for each of the seed pixels.

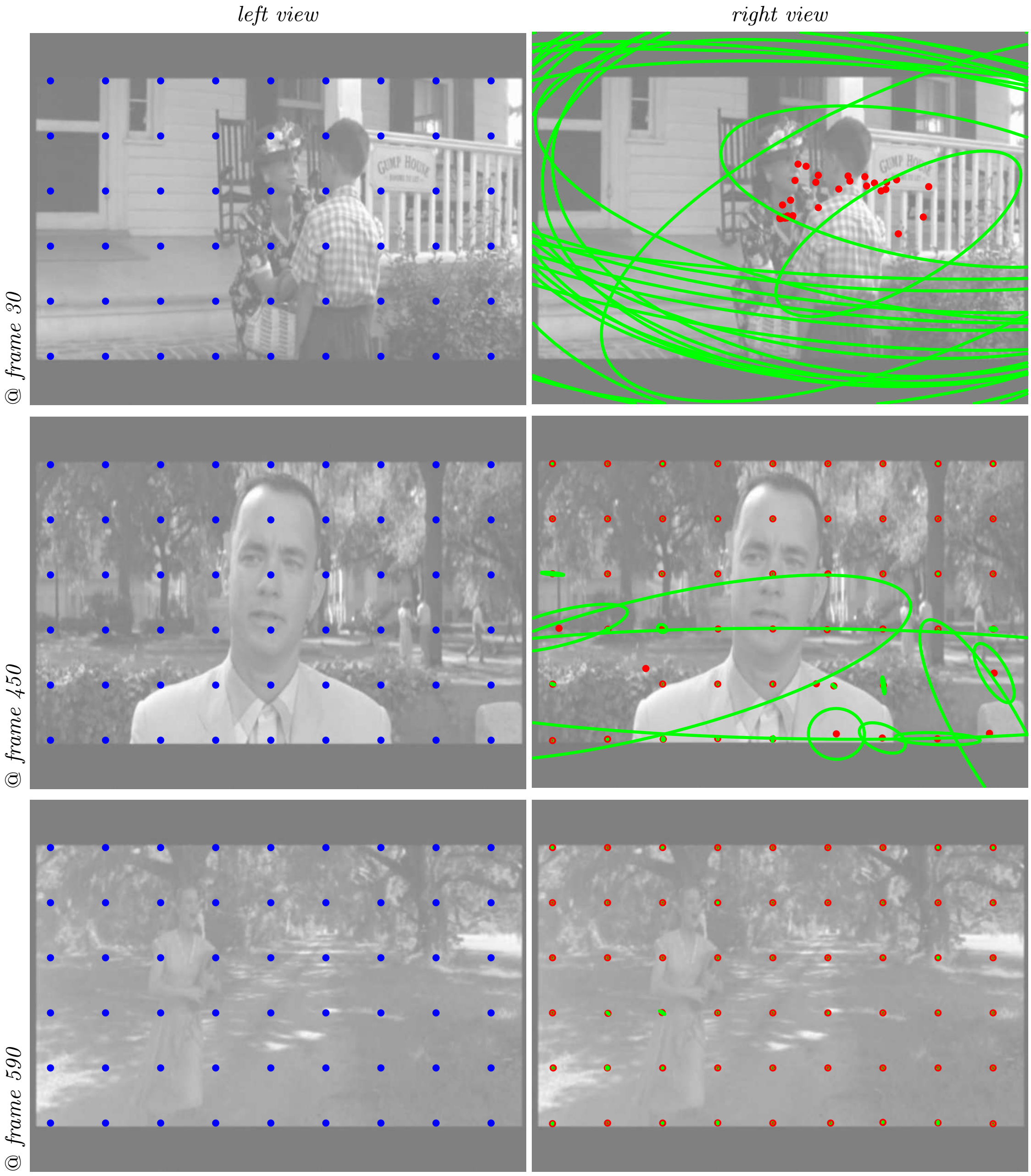


Figure 4.2: **Exemplary results for sequence `Forrest`:** Learning correspondences from a one view setup at frame 30, 450 and 590. For a set of seed pixels in the left view (blue marks), correspondences learnt in the right view are shown (red marks). Green covariance error ellipses per seed pixel visualise the certainty about the estimated correspondence. Best viewed in colour and up scaled. See text for details.

Let us now regard other attributes of the correspondence distributions. Figure 4.3 visualises plots of the entropy, the endpoint error and the coherence measure averaged over the 54 correspondence distributions. Additionally, I determine the energy consumption $\xi = \frac{M}{|\mathcal{H}|}$ per processed frame, which is defined as the ratio of processed (detected) events M to the total number of seed pixels. It follows that $0 \leq \xi \leq 1$. We may influence the average energy consumption by means of the event threshold T_e. Clearly, the larger the event threshold is chosen, the less events will be processed. The aforementioned measures are shown in Fig. 4.3 for the three different event thresholds $T_e \in [64, 128, 256]$. Let us now regard the plots of the average entropy $\mathbb{E}\left[H(\boldsymbol{Y}_j)\right]$ (top row in Fig. 4.3). Recall that the correspondence distribution's entropy upon initialisation is given by Eq. 3.69. As the images have a resolution of 720 × 540 pixels, we have $\mathbb{E}\left[H(\boldsymbol{Y}_j)\right]_{t=0} = \log_2(720 \cdot 540) = 18.56$, at time $t = 0$. It can be seen, that the correspondence distribution's entropy decreases the more frames are processed. Recall that the highest confidence about the learnt correspondence is attained when the entropy is zero. From the plots it can be seen that the average entropy approaches a value of 0, indicating that the correspondences have be learnt successfully.

Let us now turn to the plots of the endpoint error. The endpoint error is defined as the length of the difference vector between the ground truth correspondence $\boldsymbol{y}_j$ and the estimated correspondence $\hat{\boldsymbol{y}}_j$:

$$e_{EP} = ||\boldsymbol{y}_j - \hat{\boldsymbol{y}}_j||_2. \tag{4.3}$$

Here, I regard two different correspondence estimates; i) $\boldsymbol{\mu}_{\boldsymbol{y}_i}$, i.e., the expected value of $\boldsymbol{Y}_j$ under the correspondence distribution and ii) $\boldsymbol{m}_{\boldsymbol{y}_i}$, i.e., the location where the maximum a posteriori (MAP) value of the correspondence distribution is attained. From the plots of the endpoint error in Fig. 4.3, it can be seen that during the early learning stage (in the first 30-40 frames), the error for the mean correspondence is smaller than the error for the map correspondence. This is to be expected as the map correspondence will initially be located in the upper left corner, while the mean correspondence will be located in the middle of the image. This is due to the fact that the correspondence distribution is initialised uniformly and that the MAP correspondence is extracted as the first occurrence of the distribution's maximum value, while the mean value will obviously lie at the centre position.

As learning proceeds, it can be seen that the map estimate slightly outperforms or is on par with the mean correspondence. In the late learning stage we see that the map and mean correspondence approach an average endpoint error of 0. I have found that the map correspondence may serve as an early indicator for the true correspondence, especially when the event threshold is set rather small and the early estimates of the correspondence distribution are rather scattered. However, at the same time, the map correspondence also tends to change or 'jump' quite rapidly until it is 'locked' on the true correspondence.

Obviously, when no ground truth correspondences are available, the endpoint error cannot be determined and cannot serve as a confidence measure for the learnt correspondence. Instead, we then rely on the entropy of the correspondence distribution and/or the eigenvalues of the associated covariance matrix. We therefore require the entropy of the correspondence distribution to be close to 0, before we are confident that the correspondence estimate is close to the true correspondence. Likewise, we may regard the sum of the eigenvalues as a measure of confidence, which should be small when the correspondence estimate is close to the true correspondence. From the plots of the average coherence measure, it can be seen that the learnt correspondence distributions are of isotropic shape, indicating a point-to-point correspondence as expected.

Observe that the plots of the average entropy, the average endpoint error and the average coherence measure are similar when varying the event threshold. However, this is not the case for the plots of the average energy consumption shown in the last row of Fig. 4.3. Notice that the peaks in these plots mark cuts within the Forrest Gump trailer resulting in a large number of events. This behaviour will usually not be seen for natural image sequences. As expected, the average energy consumption reduces for larger event thresholds. Figure 4.4 visualises the average energy consumption by means of processed events versus the attained level of confidence about the true correspondence, here measured by means of the correspondence distribution's entropy. To this end, I accumulate the average energy consumption over the number of processed frames until the specified entropy of the correspondence distribution is attained. The plot in Fig. 4.4 visualises the energy consumptions for $\mathbb{E}\left[H(\boldsymbol{Y}_j)\right] \in [18, 17.5, .., .5]$. From the figure it can be seen that most energy is spent for an event threshold of $T_e = 64$ and least energy is spent for an event threshold of $T_e = 256$. This is to be expected, as for the larger event threshold the model skips considerably more update steps. While more update steps are skipped, this has no negative effect on the learning performance. Only those updates are skipped which are assumed to convey only little information about the true correspondence. Clearly, if the event threshold is set too large we may not learn anything about the true correspondence as there might be no events to be detected and matched.

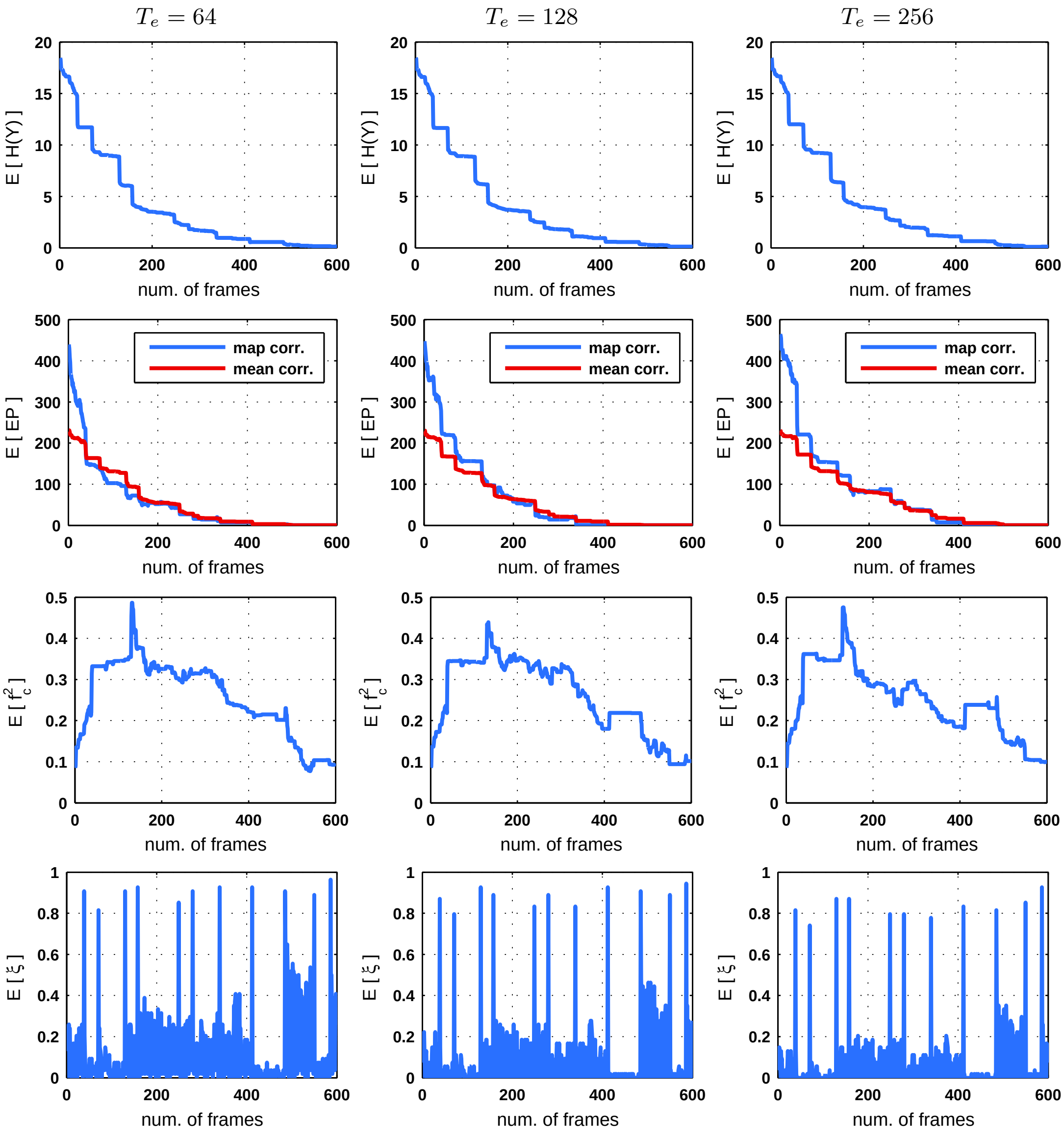


Figure 4.3: **Correspondence distribution's attributes:** (top to bottom) Plots of the average entropy, the average endpoint error, the average coherence measure and the average energy consumption (left to right) for varying event thresholds. Best viewed in colour. See text for details.

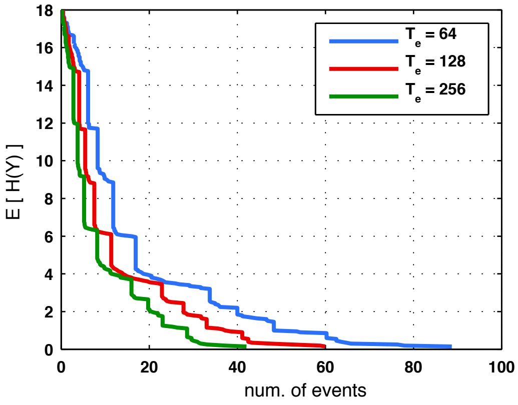


Figure 4.4: **Average energy consumption for sequence** `Forrest`**:** Visualisation of the average energy consumption while varying the event threshold. See text for details.

Learning correspondences on the pyramid Within the previous experiment, TCA was applied to the full scale images of the regarded image sequence. As has been discussed previously in Sec. 3.4, each correspondence distribution is of the same dimension as the view in which the correspondence is to be learnt. Let N and M denote the image width and height, respectively, let P be the number of seed pixels and let b be the number of *bytes* (B) of the datatype used to store the correspondence distribution. The memory footprint $\mathcal{M}$ in *Mebibyte* (MiB) is thus given by:

$$\mathcal{M} = \frac{N \cdot M \cdot P \cdot b}{2^{20}}. \tag{4.4}$$

For a 4 byte floating point data type, the memory consumption per seed pixel is then given by $720 \cdot 540 \cdot 4\text{B} \cdot 2^{-20} \approx 1.5\,\text{Mib}$, resulting in an overall memory footprint of $1.5\ \text{MiB} \cdot 54 \approx 80.1\ \text{MiB}$. The memory footprint may be reduced by either limiting the number of simultaneously active seed pixels or by applying TCA in a coarse-to-fine approach as described in Sec. 3.7.

Figure 4.5 visualises correspondences learnt on a coarse scale of sequence `Forrest`. Specifically, I apply TCA on the 3rd pyramid level with a spatial resolution of 180×135 pixels. The memory consumption per seed pixel is then 0.09MiB with an overall memory footprint of $0.09\ \text{MiB} \cdot 54 \approx 5.00\ \text{MiB}$. Correspondences learnt on a coarse scale may then be back-projected on the original scale as described in Sec. 3.7. Subsequently the back-projected correspondences may serve as a prior and constrain the search space on the fine scale. From Fig. 4.5 it can be seen that the spatial extend of the back projected correspondences (given by the blue squares) is large when the uncertainty about the true correspondence is large. This is due to the fact that the extent of the regions is coupled with the eigenvalues of the distribution's covariance matrix. As learning proceeds, this uncertainty becomes smaller, resulting in a *stronger* prior on the true correspondence on the fine scale.

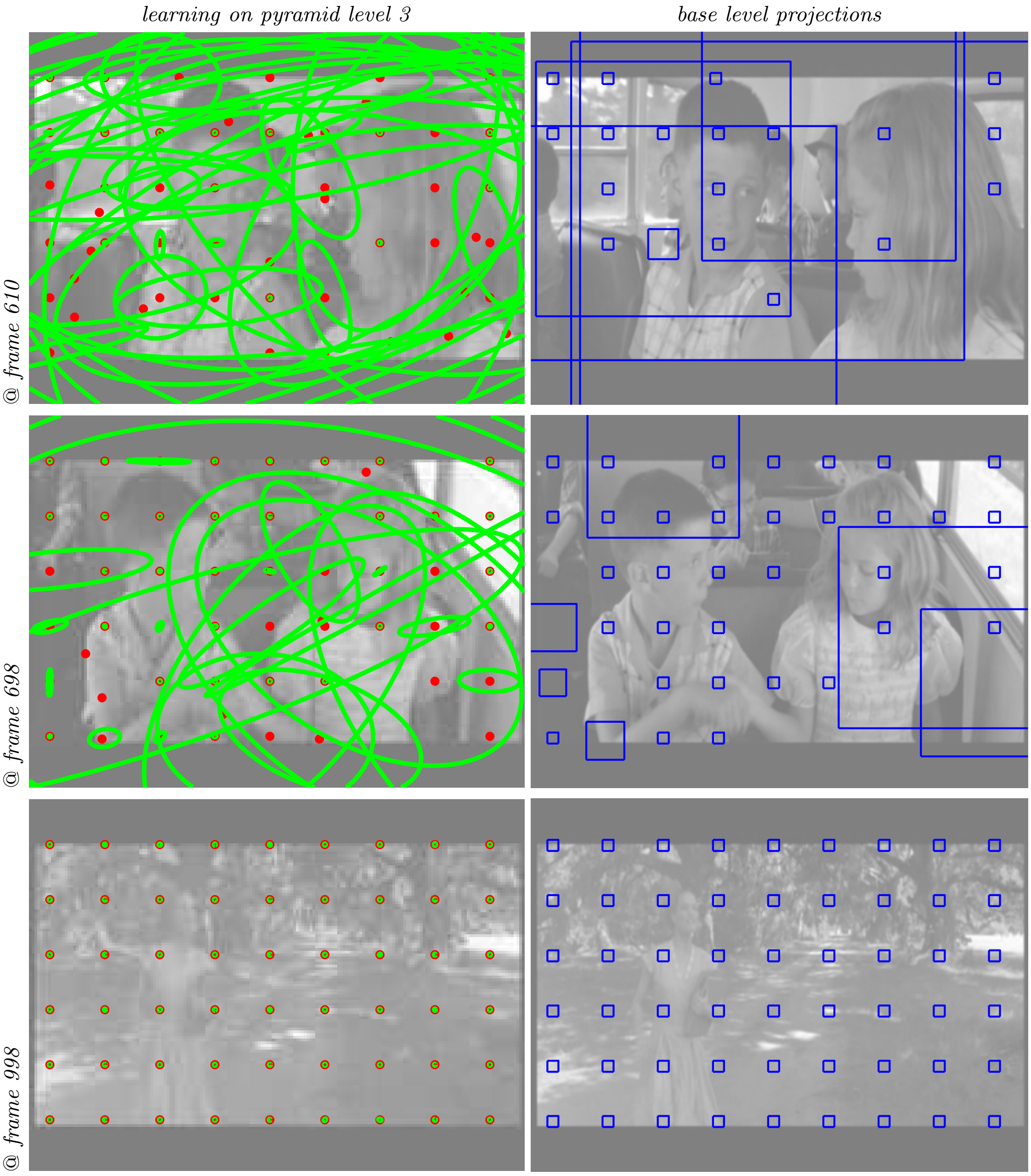


Figure 4.5: **TCA on the pyramid of sequence `Forrest`:** Correspondences are learnt on a coarse scale and constitute a prior on the true correspondence on the base level. In this figure, learning is performed on pyramid level 3. Best viewed in colour. See text for details.

4.2.3 Evaluation on Real World Data

Let us now turn to an evaluation of TCA on real world multi-camera sequences. Specifically, I will present results for sequences `GUBo1616`, `GUBo1606` and `GUCar`. See Sec. A for sample images and details on the camera setup.

Note that TCA requires rather long image sequences, containing a considerable amount of apparent motion. While there are many well-known vision benchmark data sets in the area of stereo and motion estimation, e.g., the Middlebury benchmark (Baker et al. 2011), the problem is that these contain too few images for the presented approach to be applicable.

4.2.4 Experiments on `GUBo1616`

As for the experiments on synthetic data in Sec. 4.2.1, prior learning we have to set the model parameters σ_s and $\sigma_{h|s} = \sigma_{g|s}$. The actual values of these parameters depend on the signal noise level and the GVTF which are initially unknown. Recall from the discussion in Sec. 3.6 that we may approach this problem by either estimating the signal noise and the GVTF prior learning, or by selecting the model parameters in a conservative manner which means to overestimate the true parameters. This means that we enlarge the matching envelope. Therefore, more pixels will be matched in each time step than would be matched when the parameters are precisely known. In the following experiments I chose the second approach and selected the model parameters rather conservatively according to $\sigma_s = 6,000$ and $\sigma_{h|s} = \sigma_{g|s} = 3,000$. As will become clear in the following, the actual choice of these values is not critical as long as the true parameters are not underestimated.

Figure 4.6 visualises the outline of the seed pixels in the left view as well as the learning of correspondences after 100, 2,000 and 4,000 frames have been processed. It can be seen that the uncertainty about the learnt correspondences is large in the early learning stages (after 100 frames). As more frames are processed, i.e., more events are detected and matched, the uncertainty decreases as expected. The last row of Fig. 4.6 shows the *filtered* learning results, where only those correspondences are shown for which the entropy of the associated correspondence distribution is smaller than 1. In this experiment, this is the case for 39 out of the 45 seed pixels. Taking a closer look onto the outline of the seed pixels, it can be seen that some of them lie on bushes or on the pavement, where we expect to detect only few events. This results in correspondence distributions with high entropy, i.e., high uncertainty about the true correspondence. Clearly, these are the seed pixels which are filtered out in Fig. 4.6 (bottom row). By visual inspection, the remaining correspondences have been learnt accurately.

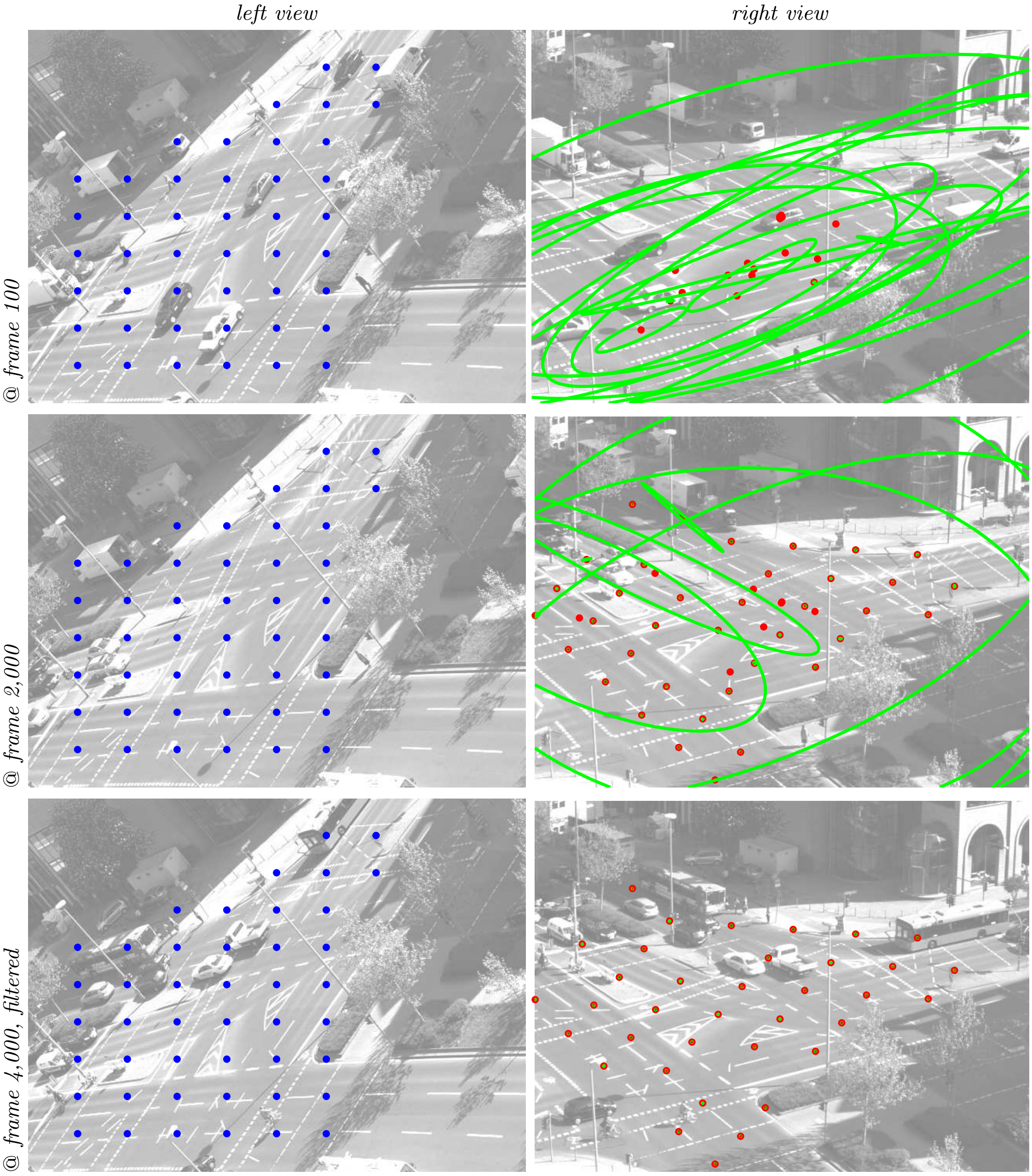


Figure 4.6: **Exemplary results for** `GUBo1616`**:** Learning correspondences from a static binocular camera setup. For a set of seed pixels in the left view (blue marks), correspondences learnt in the right view are shown (red circles) after 100, $2,000$ and $4,000$ frames have been processed, respectively. Green covariance error ellipses per seed pixel visualise the certainty about the estimated correspondence. Best viewed in colour and up scaled. See Sec. 4.2.4 for details.

Let us now regard the attributes of the correspondence distributions. As for the ground truth setup in Sec. 4.2.1, I regard the entropy, the coherence measure, the eigenvalues of the covariance matrix and the spent energy averaged over all seed pixels, respectively. I provide these measures based on the complete set of seed pixels as well as for a filtered subset of seed pixels, as described before. Figure 4.7 shows plots of the correspondence distribution's attributes over $4,000$ frames. It can be seen that the average entropy decreases over time and is close to 0 after $4,000$ frames. This is also the case for the coherence measure and the eigenvalues. When comparing the plots of the filtered and the complete set of seed pixels, it can be seen that the measures are considerably larger for the complete set (= higher uncertainty), as expected. It can be seen that the correspondence distribution's entropy and the associated eigenvalues capture the certainty about the learnt correspondence. Note that the coherence measure alone is only a sufficient condition for a learnt correspondence, as only the ratio of the eigenvalues of the associated covariance matrix is regarded. Observe that the number of processed frames and the number of processed events differ largely. For example, the average entropy drops below 2 after $1,000$ processed frames (cf. Fig. 4.7 (top left)) compared to 20 processed events (cf. Fig. 4.7 (bottom)). Hence the number of processed frames is no indicator for the learning performance.

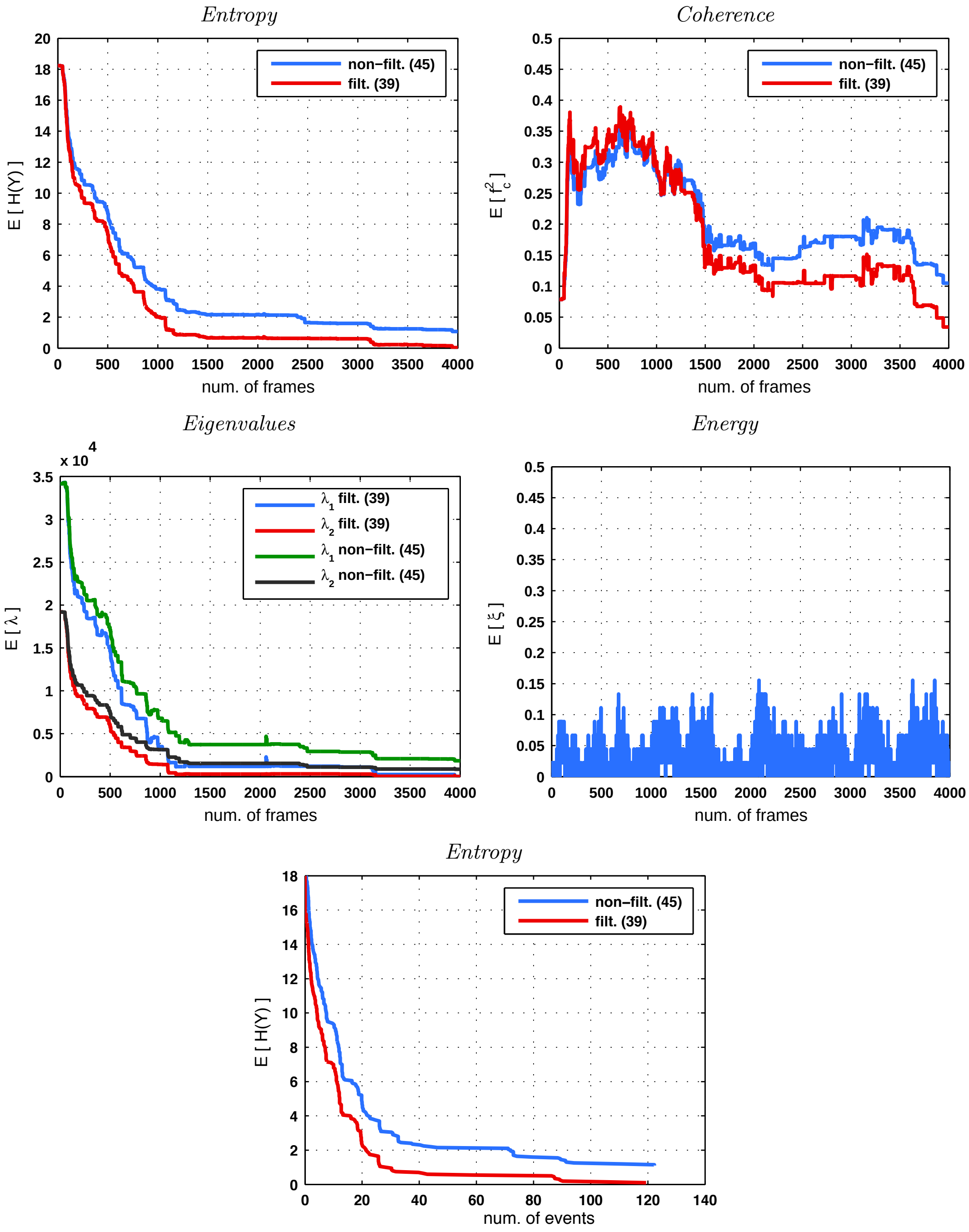


Figure 4.7: **Correspondence distribution's attributes for** `GUBo1616`**:** Plots of the correspondence distribution's entropy, coherence measure, eigenvalues of the covariance matrix, spent energy and entropy vs. spent energy averaged over all seed pixels, respectively. Best viewed in colour. See Sec. 4.2.4 for details.

Next, I learn correspondences for the same set of seed pixels, but on a coarse scale. Specifically, correspondences are learnt on the 3rd level of the image pyramid. The resolution of the input images reduces from 640×480 pixels to 160×120 pixels. Hence the memory footprint reduces from $640 \cdot 480 \cdot 4\mathrm{B} \cdot 2^{-20} \approx 1.18\,\mathrm{Mib}$, to $160 \cdot 120 \cdot 4\mathrm{B} \cdot 2^{-20} \approx 0.07\,\mathrm{Mib}$ per seed pixel (cf. Eq. 4.4). Figure 4.8 (left column) visualises the learnt correspondences after 600, $1,500$ and $4,000$ processed frames. Additionally, the figure visualises (right column) the learnt correspondences when back-projected onto the fine (=original) scale. Recall from Sec. 3.7 that by back-projected I refer to a rectangular region on the original scale, where the true correspondence is expected to lie. Every single pixel position in this region could be the true correspondence with equal probability. Observe that the spatial extend of this region is coupled with the uncertainty about the location of the correspondence on the coarse scale. The larger the uncertainty on the coarse scale, the larger the spatial extend on the fine scale where the true correspondence may lie. This can be seen throughout the plots in Fig. 4.8. Figure 4.9 shows plots of the correspondence distribution's attributes over $7,000$ frames. The plots are similar to the ones obtained when learning on the fine scale, i.e., the average entropy, the coherence measure and the eigenvalues of the covariance matrix decrease over time. As has been explained in Sec. 3.7, we may now refine the learnt correspondences on a finer scale and restrict the search range based on the back-projected correspondences from the coarse scale.

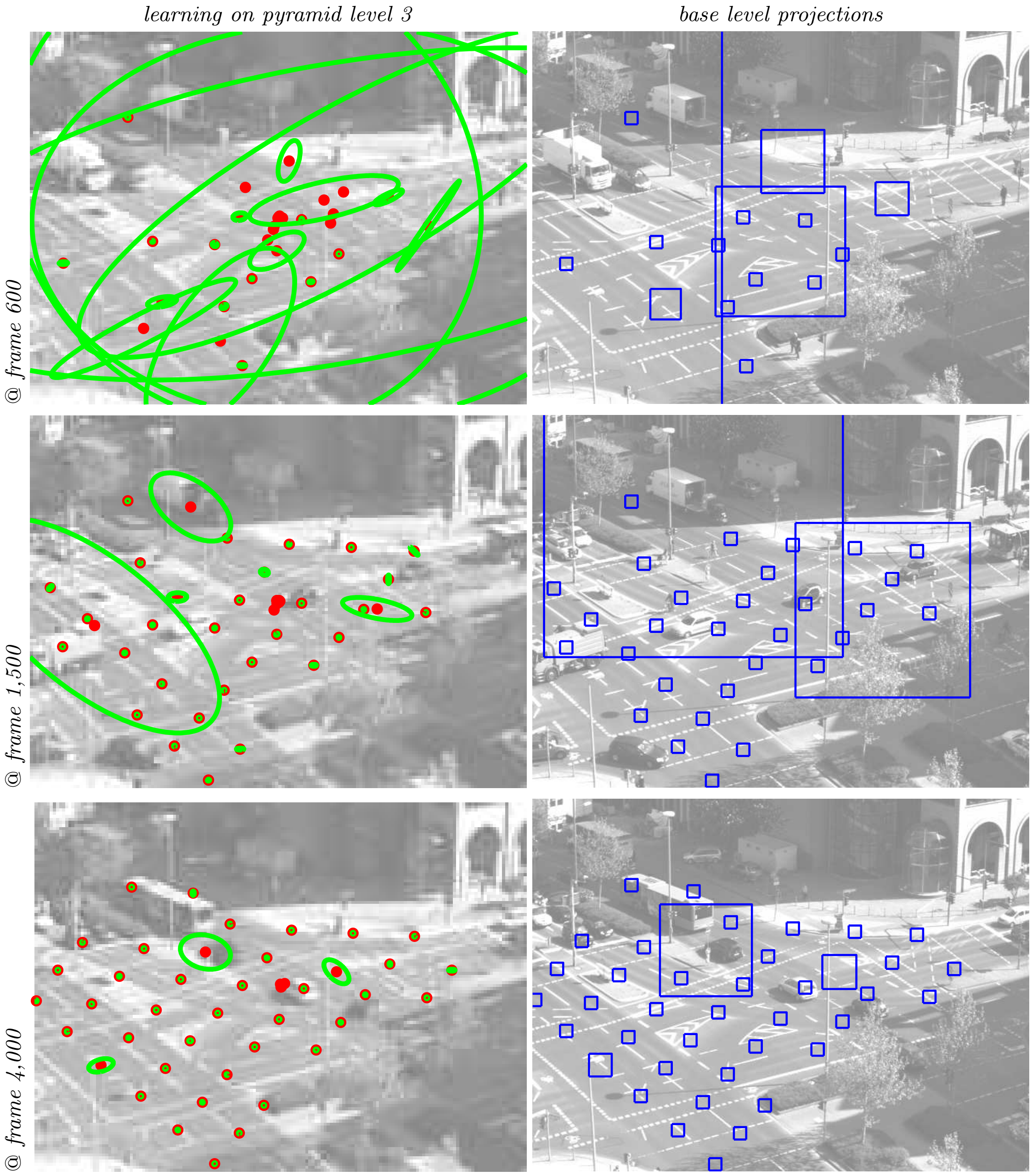


Figure 4.8: **Exemplary results for** `GUBo1616` **(coarse scale):** Learning correspondences from a static binocular camera setup on the third level of the spatial pyramid. (first column) Correspondences learnt on the coarse scale after 600, 1,500 and 4,000 frames have been processed. (second column) Correspondences learnt on the coarse scale are back-projected onto the fine scale and constitute a rectangular region where the true correspondence is expected. The spatial extend of these regions is coupled with the eigenvalues of the correspondence distribution's covariance matrix. See Sec. 4.2.4 for details.

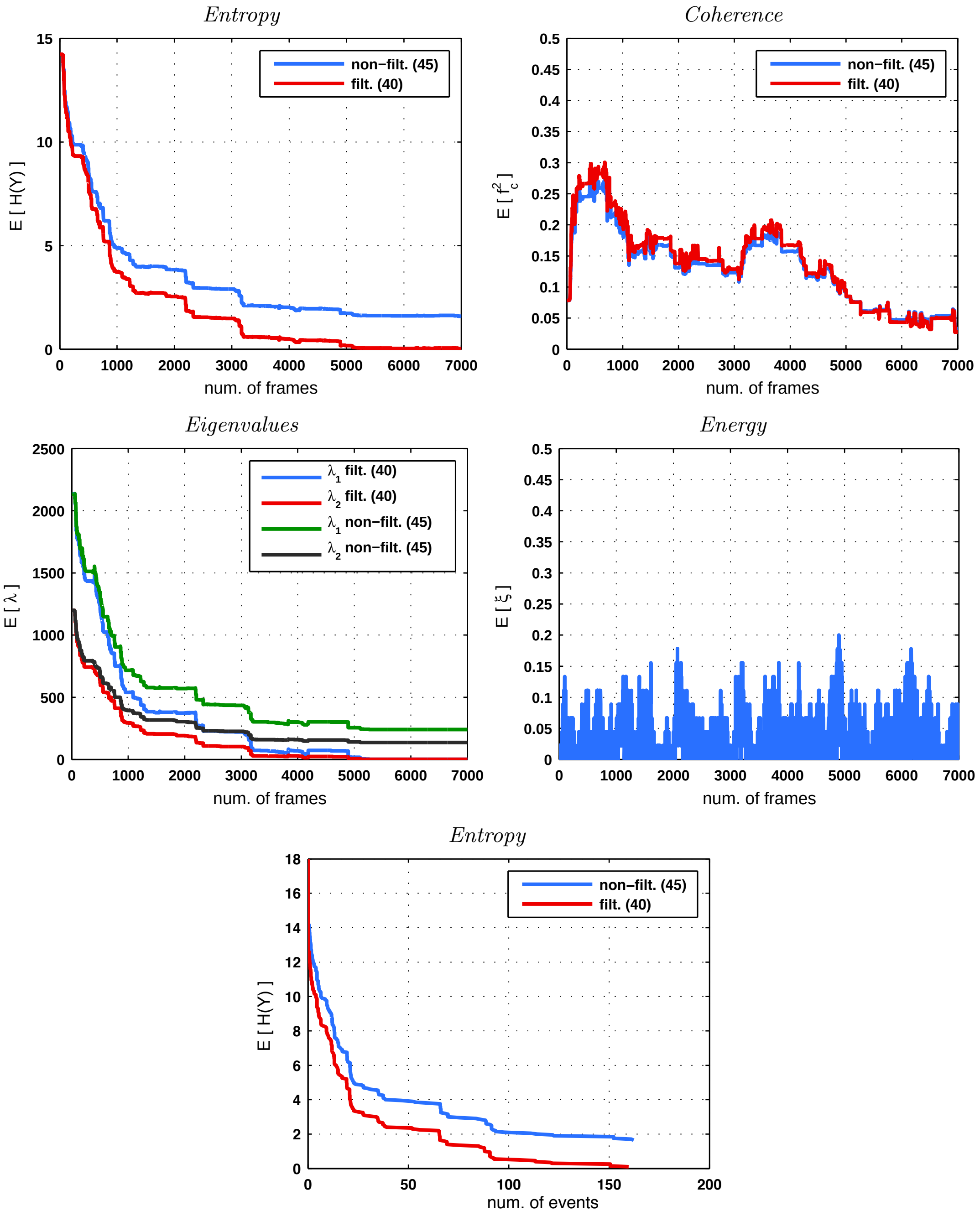


Figure 4.9: **Correspondence distribution's attributes for `GUBo1616` (coarse scale):** Plots of the correspondence distribution's attributes obtained for the 3rd level of the spatial pyramid. Entropy, coherence measure, eigenvalues of the covariance matrix, spent energy and entropy vs. spent energy averaged over all seed pixels, respectively. Best viewed in colour. See Sec. 4.2.4 for details.

Learning the GVTF Once as set of pixel correspondences has been learnt, we may proceed and estimate the GVTF as described in Sec. 3.6. This analysis may not only be done on the fine scale but could as well be performed on a coarse scale as shown in the following.

In a first experiment, I estimate the GVTF on the original resolution of the input images (=fine scale). Recall from Sec. 3.4.2 that I learn a GVTF by means of fitting a low order polynomial to a grey value comparagram. In order to build the comparagram, I select all correspondences for which the entropy of the associated correspondence distribution is below a threshold $T_H = 1$. Then, I fill the comparagram with the pairs of grey values selected at the seed pixel and its correspondence estimate. Only those grey value pairs will be added, for which the local structure of the seed pixel and its correspondence indicates a rather homogeneous area. To this end, I determine the eigenvalues of the structure tensors and only add a grey value pair when Eq. 3.135 is fulfilled with $T_\phi = 10$. (cf. Sec. 3.4.2 for details).

Figure 4.10 visualises the learnt GVTF based on the proposed and the naïve scheme, after having processed 1,000, 5,000 and 10,000 frames. Recall from Sec. 3.4.2 that in the naïve scheme the comparagram is build without regarding the local structure tensor. Let us now regard the learnt GVTFs after 1,000 frames have been processed. In the grey value interval $[100, 150]$ the uncertainty considerably decreased for both the proposed and naïve learning scheme. A noticeable difference between the two learning schemes can be seen in the grey value interval $[160, 250]$. While for the proposed learning scheme, the uncertainty about the grey value mapping is close to the initial uncertainty, the uncertainty considerably decreased for the naïve scheme. This can also be seen for the grey value interval $[30, 100]$. Clearly, this is explained by the filtering procedure employed by the proposed learning scheme based on the analysis of the local structure tensor (cf. Sec. 3.6). In the proposed learning scheme, less grey value pairs are added to the comparagram than for the naïve scheme, hence the remaining uncertainty is larger. Next, let us regard the learnt GVTFs after $5,000$ and $10,000$ frames have been processed. It can be seen that now the uncertainty is larger for the naïve learning scheme, especially in the grey value ranges $[50, 100]$ and $[160, 250]$. From this we conclude that the proposed learning scheme induces less outlier grey value pairs in the comparagram, i.e., the variance of the grey value pairs added to the comparagram is smaller. However, we may also conclude that the geometric error in the learnt correspondences is sufficiently small such that a reasonable GVTF may also be learnt via the naïve scheme.

Figure 4.13 visualises the GVTF learnt on a coarse scale, here for level 3 of the spatial pyramid. It can be seen that for the proposed learning scheme, the results are similar to the ones obtained on the original scale. While the uncertainty for the naïve scheme considerably decreased for the larger grey values, a clear outlier has been introduced for the grey values in the vicinity of grey value 75.

In order to verify that the learnt GVTF indeed captures the illumination differences among the views, I proceed as follows. First, I estimate a spatial transformation relating the left and right view of the regarded image sequences. This is done by selecting the set of correspondences for which the entropy of the associated correspondence distribution is below a threshold $T_H = 1$. From the selected correspondences a homography is estimated (robustly via RANSAC). I then spatially register the left and right view and compute the difference image. Figure 4.11 visualises the difference image for a sample frame pair of sequence `GUBo1616`. Let d be some difference value lying in the interval $[-255, 255]$. I rescale d according to $\hat{d} = \text{uint8}(d \cdot 4 + 128)$ to the interval [0,255], with a value of 128 corresponding to a difference of 0. This is done for visualisation purposes only. By visual inspection, it can be seen that the difference image of the non-compensated views is considerably darker than for the GVTF compensated views, which indicates larger (negative) grey value differences. This can also be seen from the histogram of the (non-rescaled) difference image in Fig. 4.11, for both the GVTF compensated and non-compensated registered views. While the differences of the compensated views cluster around a value of 0, the differences of the non-compensated views shows a bias. Figure 4.12 shows the difference images and histograms for a different pair of frames. From these, we may draw the same conclusions as previously.

Note that I do not expect to obtain perfectly registered images, i.e., differences which are close to 0 for every pixel location. This is due to several reasons, among them that the regarded scene only approximately lies in a single plane. Registering the views by a homography will lead to larger errors where the scene deviates from this planar assumption. This can be seen from large difference values at the light poles. Furthermore, I did not compensate for camera lens distortions.

To summarise, we have seen that the proposed GVTF learning scheme successfully captures the illumination differences for this real world camera setup. In a real world application, the learning of correspondences and the GVTF may now be iterated over time.

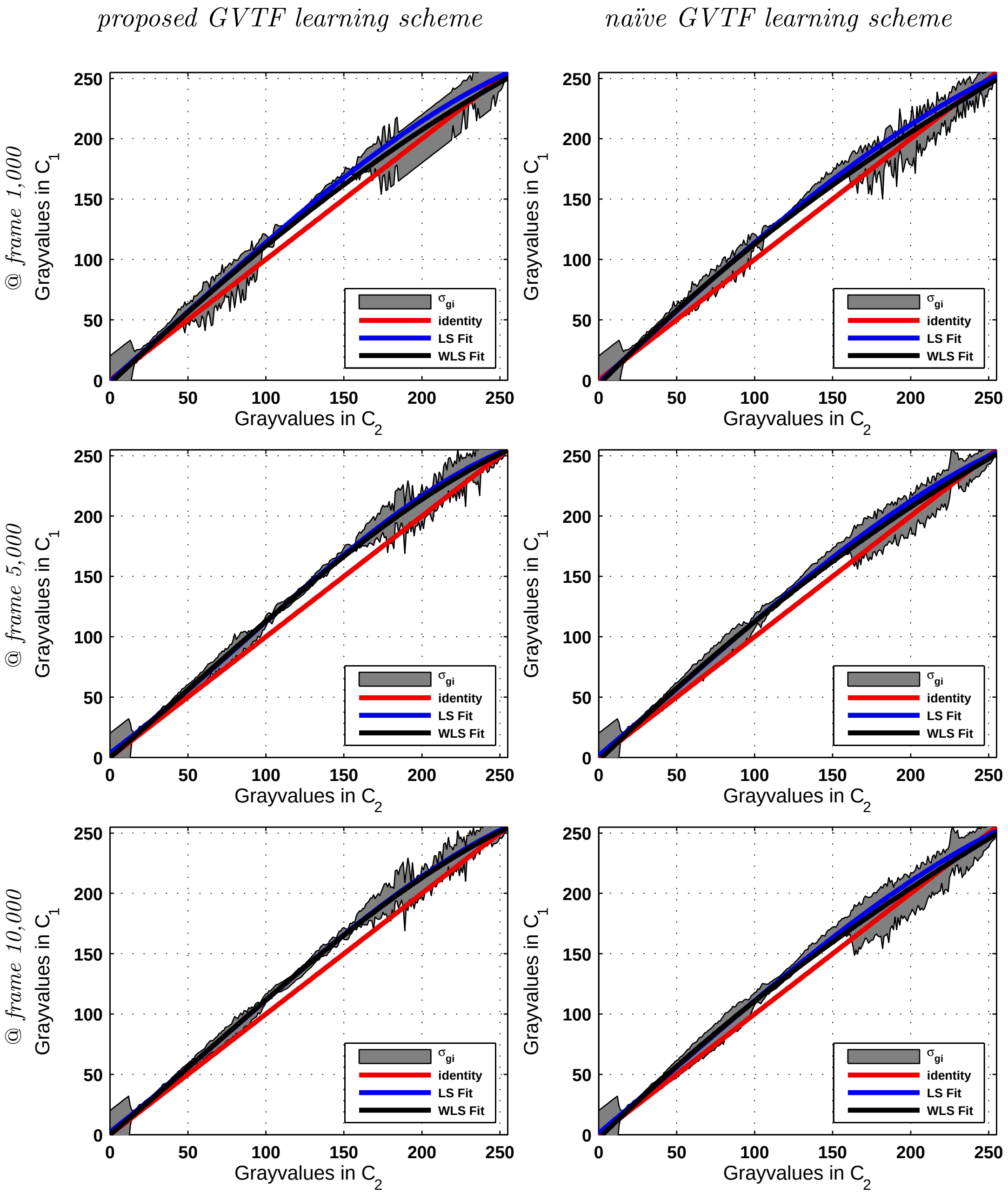


Figure 4.10: **Learning the GVTF for `GUBo1616`:** Given a set of learnt correspondences, the GVTF is estimated as described in Sec. 3.4.2. (first column) Proposed learning scheme, after 1,000, 5,000 and 10,000 frames have been processed. The identity GVTF is shown in red, the fit to a 3rd order polynomial is shown in blue (LS fit) and black (WLS fit). Estimates of $\hat{\mu}_{\phi_{ij}(g_i)}$ and $\hat{\sigma}^2_{\phi_{ij}(g_i)}$ define the grey uncertainty area. (second column) Naïve learning scheme, leading to a GVTF estimate with considerably higher uncertainty. See Sec. 4.2.4 for details. Figure only interpretable when viewed in colour.

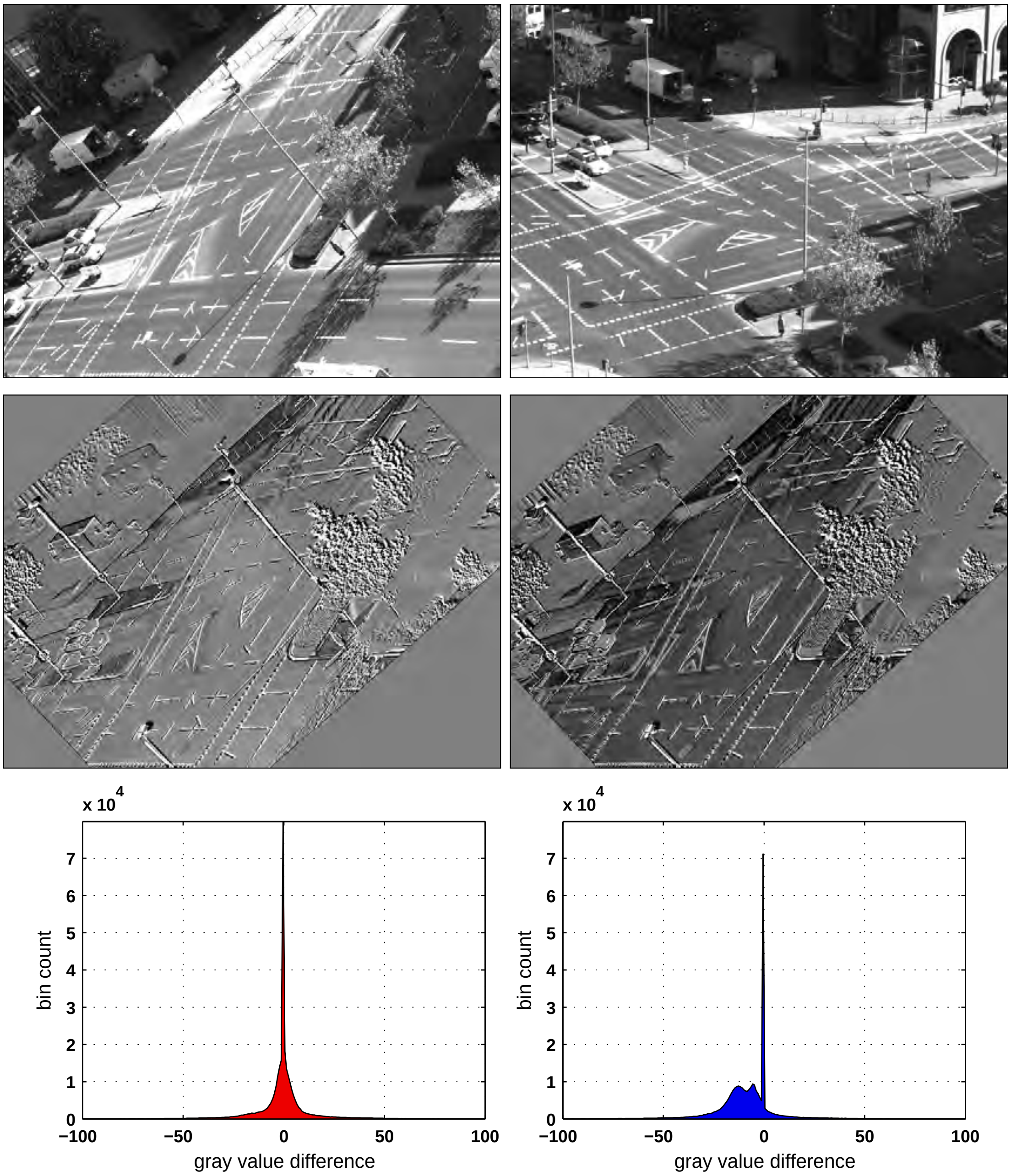


Figure 4.11: **Difference images and error histogram for `GUBo1616`:** Difference image of a sample frame pair, where the left and right view are spatially registered via a homography estimated from learnt correspondences. Grey value differences are mapped to the interval [0,255], where a grey value of 128 indicates a grey value difference of 0. (first column, top to bottom) Sample left view of `GUBo1616`, difference image and histogram of the difference image when the right view is GVTF transformed. (second column, top to bottom) Sample right view of `GUBo1616`, difference image and corresponding histogram when the illumination differences are not compensated. See Sec. 4.2.4 for details.

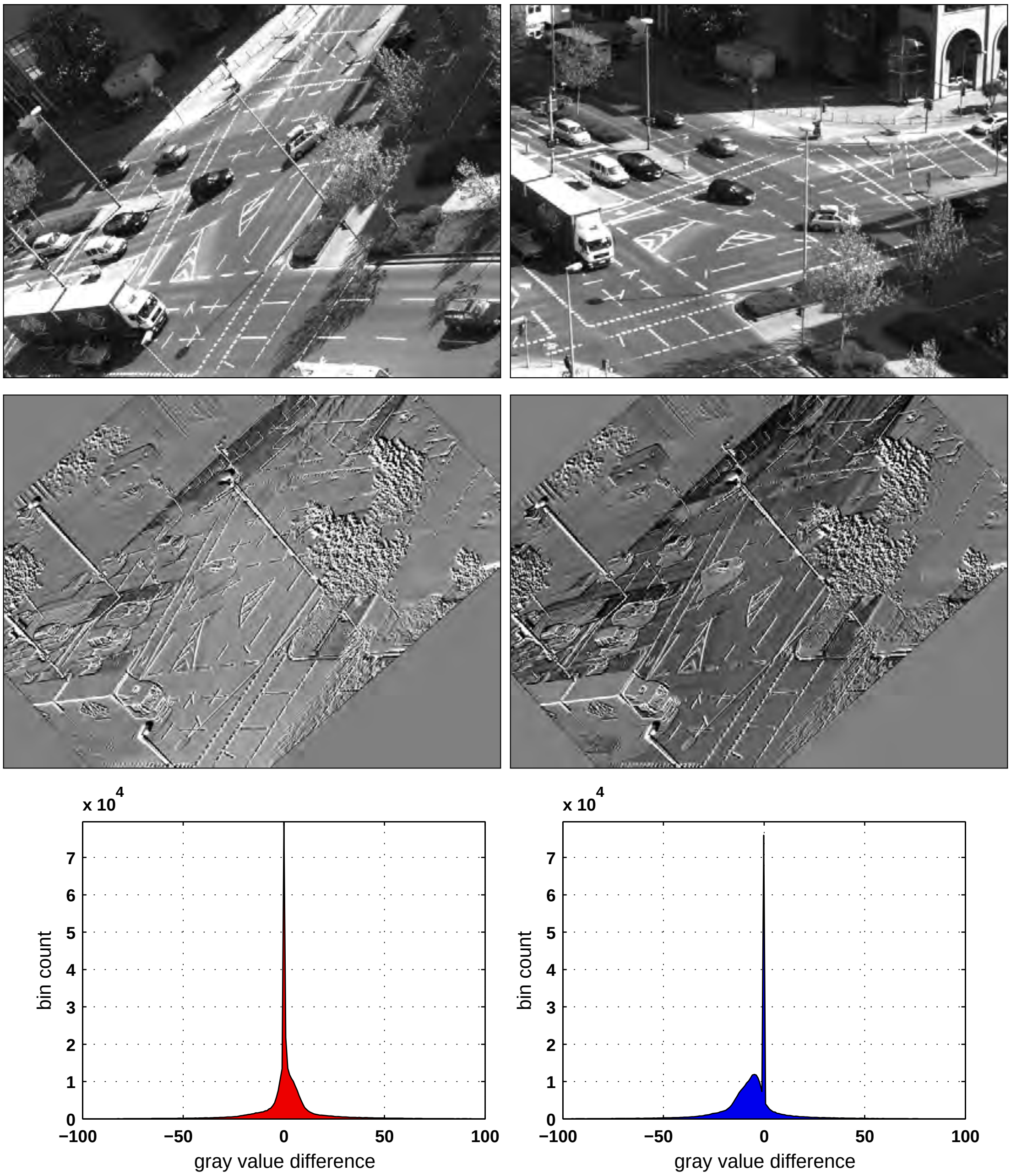


Figure 4.12: **Difference images and error histogram for `GUBo1616` (2):** Difference image of a sample frame pair, where the left and right view are spatially registered via a homography estimated from learnt correspondences. Grey value differences are mapped to the interval [0,255], where a grey value of 128 indicates a grey value difference of 0. (first column, top to bottom) Sample left view of `GUBo1616`, difference image and histogram of the difference image when the right view is GVTF transformed. (second column, top to bottom) Sample right view of `GUBo1616`, difference image and corresponding histogram when the illumination differences are not compensated. See Sec. 4.2.4 for details.

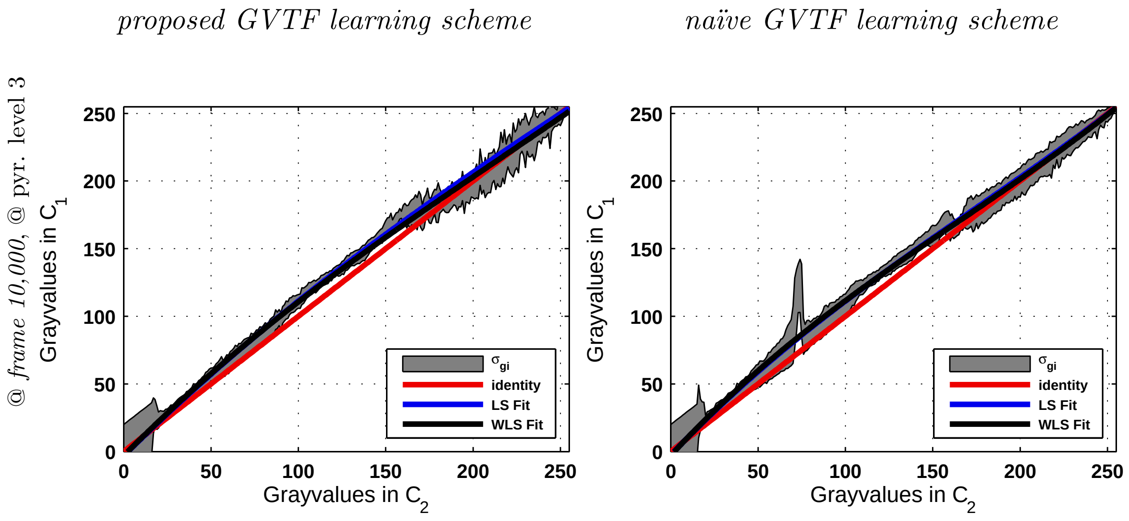


Figure 4.13: **Learning the GVTF for `GUBo1616` (coarse scale):** Given a set of learnt correspondences, the GVTF is estimated as described in Sec. 3.4.2. (first column) Proposed learning scheme, after 10,000 frames have been processed. The identity GVTF is shown in red, the fit to a 3rd order polynomial is shown in blue (LS fit) and black (WLS fit). Estimates of $\hat{\mu}_{\phi_{ij}(g_i)}$ and $\hat{\sigma}^2_{\phi_{ij}(g_i)}$ define the grey uncertainty area. (second column) Naïve learning scheme, leading to a GVTF estimate with considerably higher uncertainty and a severe outlier for grey values in the vicinity of grey value ≈ 75. See Sec. 4.2.4 for details. Figure only interpretable when viewed in colour.

4.2.5 Experiments on `GUBo1606`

Sequence `GUBo1606` comprises a similar setup as `GUBo1616`, except that the cameras are now equipped with two different types of lenses: the focal length of the left camera is 16 mm while the focal length of the right camera is 6 mm.

I select the same model parameters as in Sec. 4.2.5 and again select a regular grid of seed pixels in the left view. Figure 4.14 visualises the seed pixels in the left view as well as the learning of correspondences after 500, 2,000 and 5,100 frames have been processed. Similar to the results presented for sequence `GUBo1616`, the uncertainty about the learnt correspondences decreases over time as more events are detected and matched. When we filter out those correspondences for which the entropy of the associated correspondence distribution is above 1, we see from Fig. 4.14 (bottom row) that for 35 out of the 45 seed pixels, the correspondence has been learnt.

Figure 4.15 shows plots of the correspondence distribution's attributes over 5000 frames. It can be seen that the average entropy decreases over time and is close to 0 after 2,500 frames (for the filtered subset). This is also the case for the coherence measure and the eigenvalues of the covariance matrix. The plot of the average energy spent is similar to the one for sequence `GUBo1616` and 15% of the seed pixels show an event simultaneously. This is to be expected as the scene statistics in terms of observed motion patterns are obviously similar.

As for `GUBo1616`, we again see that the number of processed events until a specified level of confidence is reached is considerably smaller than the number of processed frames. For sequence `GUBo1606` the average entropy becomes smaller than 1 after 40 events, compared to over 1,000 processed frames.

Overall, we see that for most of the seed pixels TCA was able to estimate the correct correspondence, though the sequence is far more challenging than `GUBo1616`, as both the orientation and the scaling between the views is largely different. Taking a closer look on the layout of the seed pixels (cf. Fig. 4.14) it can be seen that only those correspondences are missed which lie on bushes, trees or buildings (top left) where no or only few events are detected and matched. However, based on the correspondence distribution's entropy or associated eigenvalues, these false matches are readily detected.

Learning the GVTF Figure 4.16 visualises the learnt GVTF based on the proposed and the naïve scheme, after having processed 1,000, 5,000 and 10,000 frames. Let us now regard the learnt GVTFs after 1,000 frames have been processed.

For the proposed learning scheme, the uncertainty decreased over the grey value interval $[25, 150]$ but the uncertainty intervals appear more noisy than those obtained for `GUBo1616`. For the naïve update scheme, we see that the comparagram is noisy and contains gross outliers. As more frames are processed, the uncertainty about the GVTF decreases further for the proposed learning scheme. However, only few grey value pairs in the interval $[160, 250]$ are observed. For the naïve scheme, more outliers are added to the comparagram and we may not expect to approximate the true GVTF sufficiently well.

Let us inspect the difference between the results for `GUBo1616` and `GUBo1606` in more detail. Regarding the proposed scheme, we see that the overall uncertainty about the learnt GVTF is smaller for `GUBo1616`. The naïve scheme failed to estimate the GVTF for `GUBo1606`, due to gross outliers in the comparagram. The reason for this is the large scale difference among the views where even a small spatial error in the correspondence estimate may lead to gross outliers in the comparagram.

I estimate a spatial transformation relating the left and right view as described previously. Figure 4.11 and Fig. 4.12 visualise the difference images and difference histograms for two sample frames of `GUBo1606`. It can be seen that the difference image of the non compensated registered views is biased, while the differences for the GVTF compensated views cluster around 0. Overall, the histograms shown have a higher variance than those for `GUBo1616`. One reason for this is that the second view is considerably enlarged during registration, hence, differences due to image interpolation become more apparent.

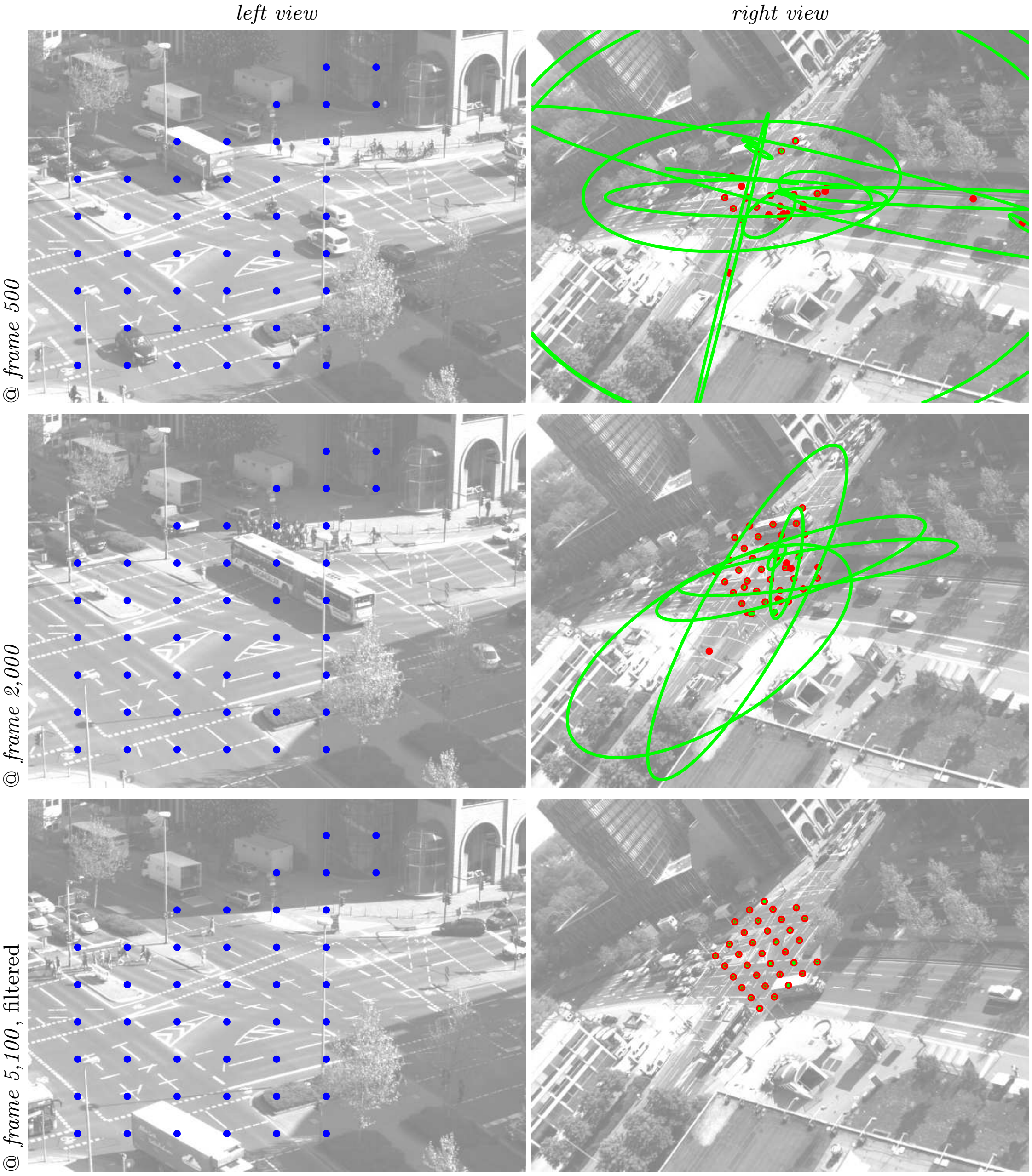


Figure 4.14: **Exemplary results for `GUBo1606`:** Learning correspondences from a static binocular camera setup. For a set of seed pixels in the left view (blue marks), correspondences learnt in the right view are shown (red circles) after 500, 2,000 and 5,100 frames have been processed, respectively. Green covariance error ellipses per seed pixel visualise the certainty about the estimated correspondence. Best viewed in colour and up scaled. See Sec. 4.2.5 for details.

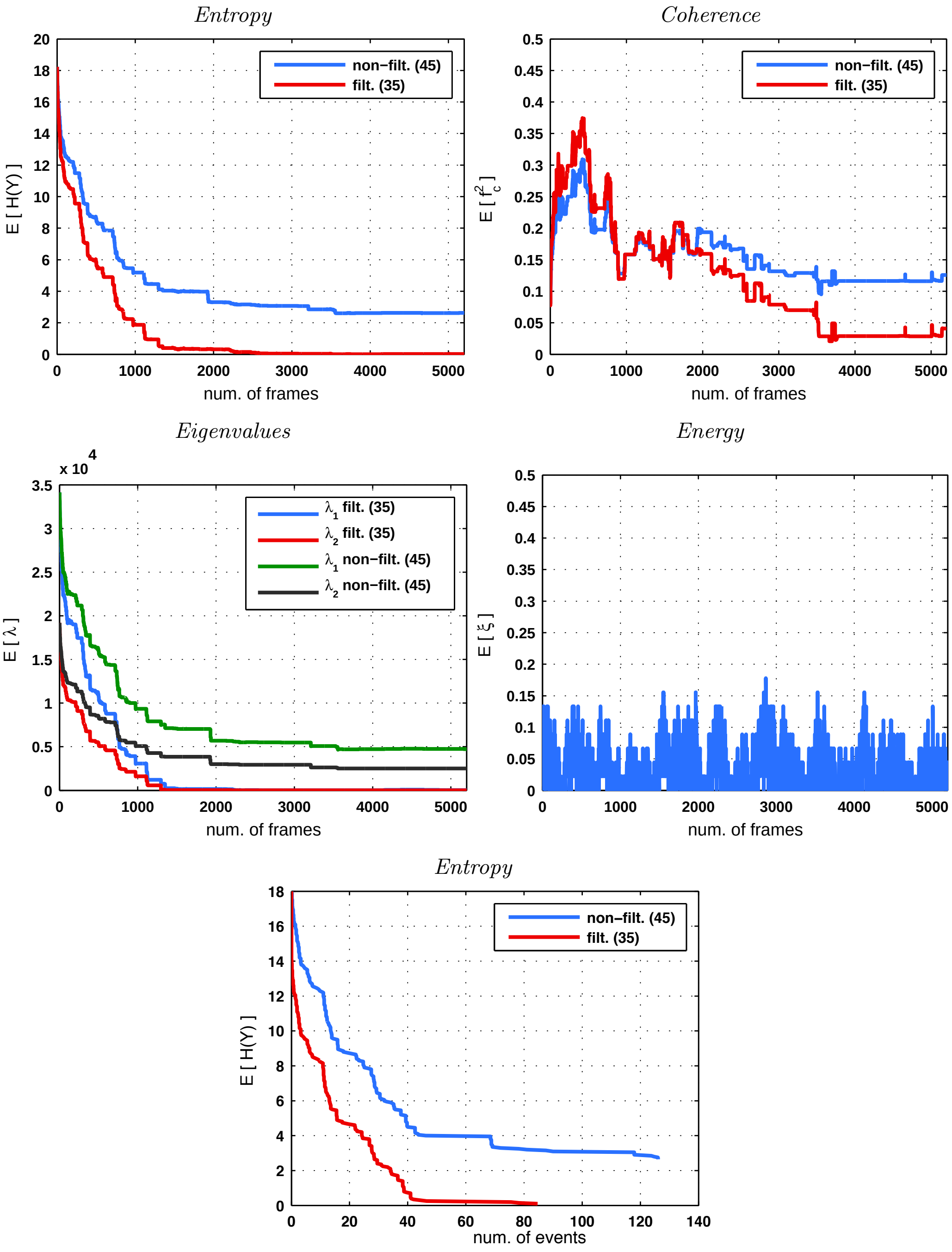


Figure 4.15: **Correspondence distribution's attributes for** `GUBo1606`**:** Plots of the correspondence distribution's entropy, coherence measure, eigenvalues of the covariance matrix, spent energy and entropy vs. spent energy averaged over all seed pixels, respectively. Best viewed in colour. See Sec. 4.2.5 for details.

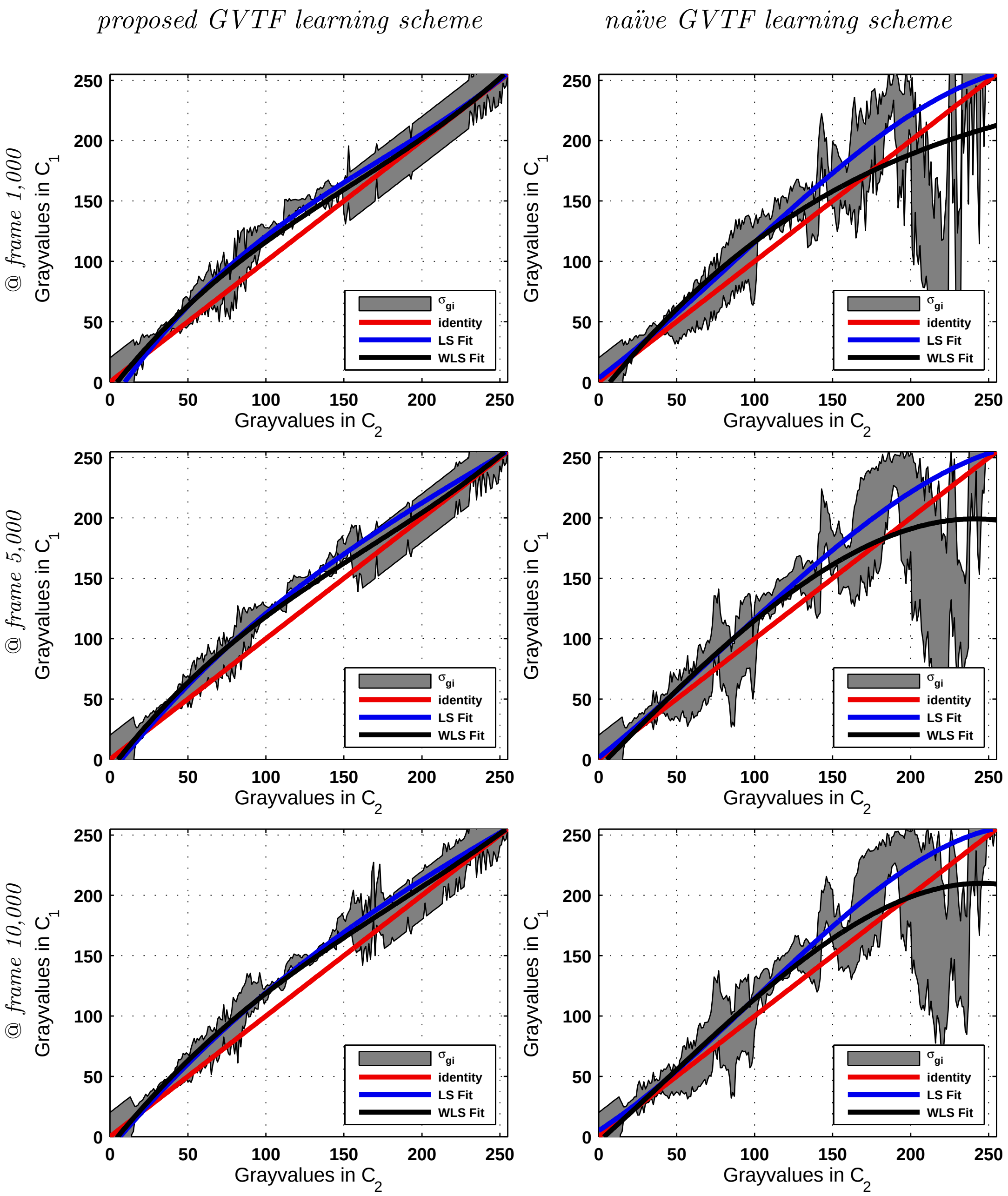


Figure 4.16: **Learning the GVTF for** `GUBo1606`**:** Given a set of learnt correspondences, the GVTF is estimated as described in Sec. 3.4.2. (first column) Proposed learning scheme, after 1,000, 5,000 and 10,000 frames have been processed. The identity GVTF is shown in red, the fit to a 3rd order polynomial is shown in blue (LS fit) and black (WLS fit). Estimates of $\hat{\mu}_{\phi_{ij}(g_i)}$ and $\hat{\sigma}^2_{\phi_{ij}(g_i)}$ define the grey uncertainty area. (second column) Naïve learning scheme, leading to a GVTF estimate with considerably higher uncertainty. See Sec. 4.2.5 for details. Figure only interpretable when viewed in colour.

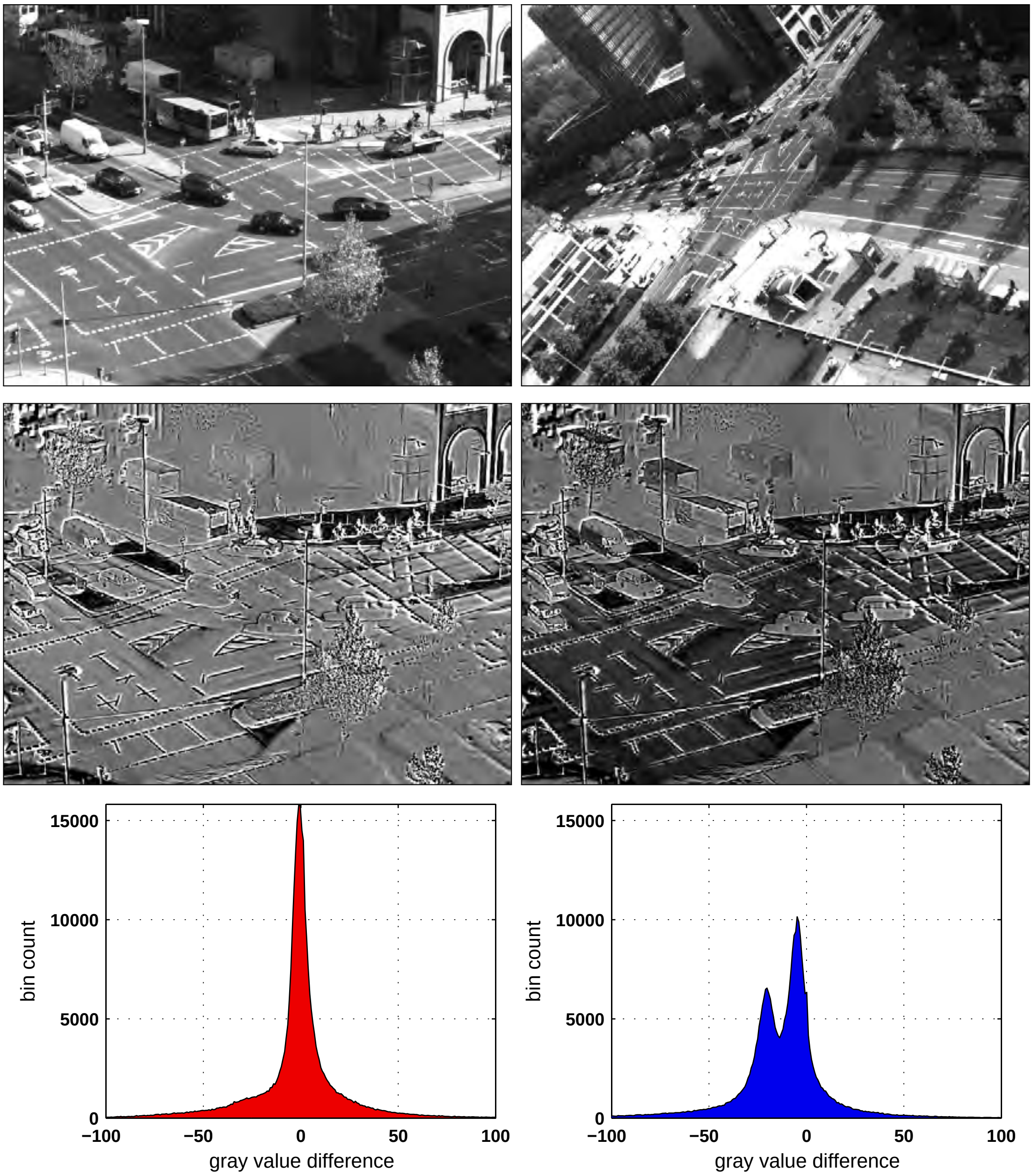


Figure 4.17: **Difference images and error histogram for** `GUBo1606`**:** Difference image of a sample frame pair, where the left and right view are spatially registered via a homography estimated from learnt correspondences. Grey value differences are mapped to the interval [0,255], where a grey value of 128 indicates a grey value difference of 0. (first column, top to bottom) Sample left view of `GUBo1606`, difference image and histogram of the difference image when the right view is GVTF transformed. (second column, top to bottom) Sample right view of `GUBo1606`, difference image and corresponding histogram when the illumination differences are not compensated. See Sec. 4.2.5 for details.

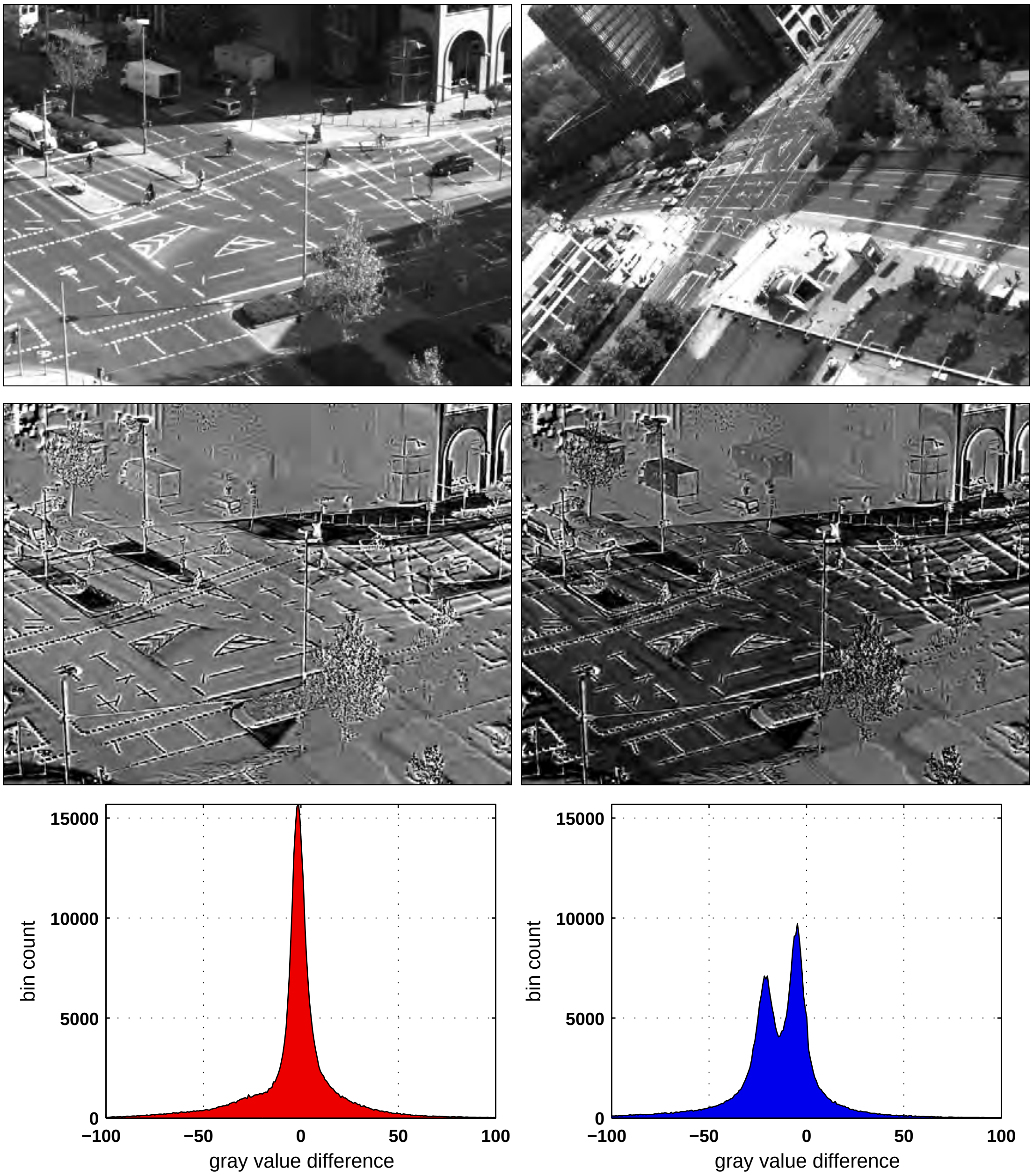


Figure 4.18: **Difference images and error histogram for `GUBo1606` (2):** Difference image of a sample frame pair, where the left and right view are spatially registered via a homography estimated from learnt correspondences. Grey value differences are mapped to the interval [0,255], where a grey value of 128 indicates a grey value difference of 0. (first column, top to bottom) Sample left view of `GUBo1606`, difference image and histogram of the difference image when the right view is GVTF transformed. (second column, top to bottom) Sample right view of `GUBo1606`, difference image and corresponding histogram when the illumination differences are not compensated. See Sec. 4.2.5 for details.

4.2.6 Experiments on `GUCar`

The experiments presented so far consisted of static views of an environment with visual events, where the scene depth was virtually not changing, leading to point-to-point correspondences only. In the following I show that TCA can also be applied to a *moving* stereo camera setup. Sequence `GUCar` was recorded while driving in urban traffic and on a highway. As for the other sequences, the cameras where not photometrically calibrated but operate in auto exposure mode. It can also be seen that the left view is considerably more blurred than the right view. Therefore, I selected very conservative model parameters according to $\sigma_s = 10,000$ and $\sigma_{h|s} = \sigma_{g|s} = 5,000$ with an event threshold of $T_e = 500$.

In this sequence, I expect to learn point-to-line correspondences as the scene depth will change for most of the pixels over time. Nevertheless, I expect the correspondence distributions to be of a point-like structure, as the learning scheme is basically memory less which results in correspondence distributions which only represent short-time relationships. For example, imagine that the scene depth is constant over a period of time and that a correspondence is learnt and represented by the correspondence distribution. If the scene depth now changes, the true correspondence will move along the epipolar ray. Consequently, the correspondence estimate will adapt to the changed scene depth. However, the learning scheme will now 'forget' the previously learnt correspondence as there is no longer evidence for this correspondence among the detected and matched events. This problem may be approached by building a histogram over all correspondence estimates where the associated entropy is below a given threshold. However, this is considered future work.

In order to evaluate the accuracy of the learnt correspondences, I estimate a fundamental matrix (cf. Sec. 2.2) relating the left and right view based on 12 hand selected correspondences. Note that I do not compensate for lens distortions and that a more robust estimate of the fundamental matrix could be obtained when more correspondence would be used (cf. (Hartley and Zisserman 2004)). However, my objective merely is to demonstrate that the learnt correspondences lie on/close to the epipolar line but I will expect mild deviations from it.

Figure 4.19 visualises the outline of the seed pixels in the left view as well as the learning of correspondences after 100, 1,000 and 5,000 frames have been processed. Additionally, for each of the seed pixels I plot the corresponding epipolar line in the right view. Similar to the previous results, the uncertainty about the true correspondence reduces as more events are detected and matched. After 5,000 processed frames (cf. Fig. 4.19 (bottom)) it can be seen that all of the learnt correspondences lie on or close to the epipolar lines. This is also the case for the single correspondence with a large uncertainty.

Let us regard the correspondence with the remaining large uncertainty in more detail (cf. Fig. 4.19 (bottom)). One could ask, why the error ellipse shows an elongated (or

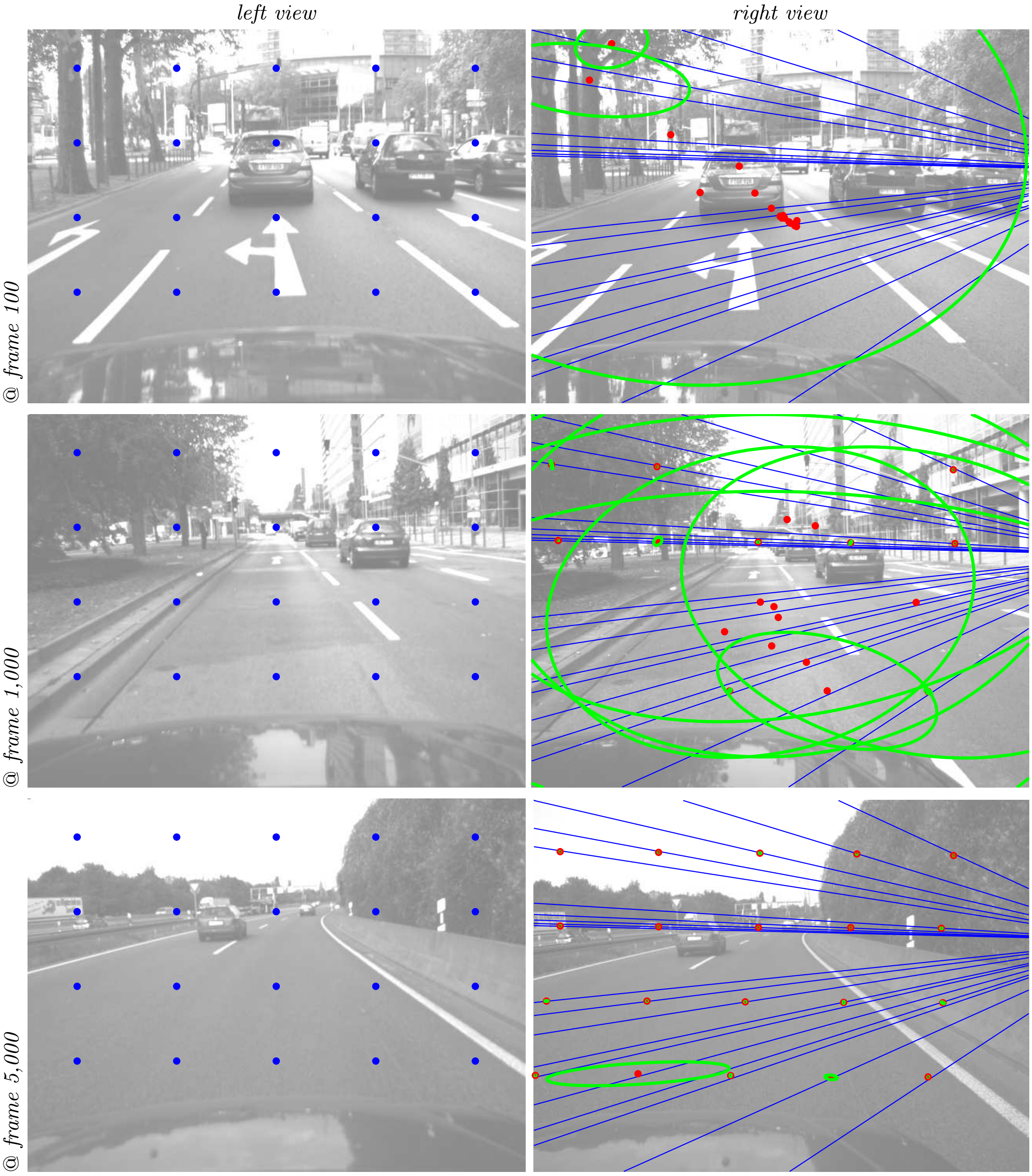


Figure 4.19: **Exemplary results for** `GUCar`**:** Learning correspondences from a moving binocular camera setup. For a set of seed pixels in the left view (blue marks), correspondences learnt in the right view are shown (red circles) after 100, 1,000 and 5,000 frames have been processed, respectively. Green covariance error ellipses per seed pixel visualise the certainty about the estimated correspondence. Best viewed in colour and up scaled. See Sec. 4.2.6 for details.

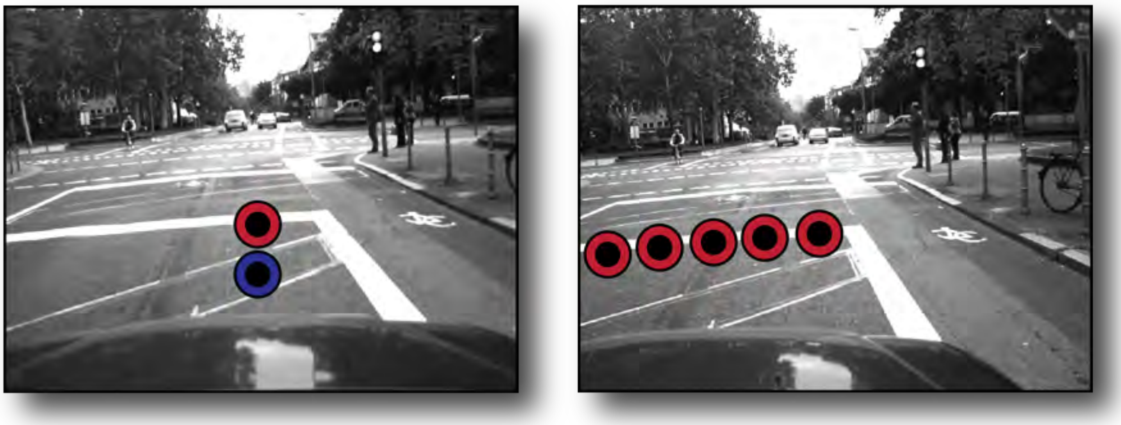


Figure 4.20: **Schematic description of a structure match:** Depending on the scene content, the set of pixels in the right view which show a similar event as the seed pixel may resemble 1d, i.e., line like structures. This may result in line like structures within the correspondence distribution which do not coincide with the true epipolar line.

line like) structure which does not align with epipolar line. The reason for this is most easily explained based on the following example. Regard Fig. 4.20, which shows two sample frames of `GUCar` and a selected seed pixel between two time steps $t-1$ and t. The grey value at the seed pixel at time $t-1$ (blue mark) is small compared to the grey value at time t (close to white) where the seed pixels lies on the white lane mark. Assume that an event is detected at the seed pixel at time t. Let us now regard the set of pixel locations which obey a compatible event in the second view. Clearly, these are now all pixel locations along the white lane mark (where only a subset of them is shown in Fig. 4.20). All of these pixel locations are equally likely to be the true correspondence as all of them obeyed a similar event. If this occurs sufficiently often, a line like structure will evolve within the correspondence distribution. Within sequence `GUCar`, this will often be the case due to lane marks in different orientations.

Figure 4.21 shows plots of the correspondence distribution's attributes over 5,200 frames. From these, we may draw similar conclusions as for the previously shown results. Only the plot of the average energy is considerably different compared to the one obtained for sequences `GUBo1616` and `GUBo1606`. As the cameras are moving in `GUCar`, virtually all pixels will be subject to scene/object motion, resulting in a larger number of pixels showing an event. We may reduce the average energy consumption by choosing a larger event threshold as shown in Fig. 4.22. These plots were obtained for an event threshold of $T_e = 1,200$ (compared to 500) leaving all other parameters as before. It can be seen that the average energy consumption became smaller and that less update steps were needed (on average) to learn the correspondence distribution up to a specified level of uncertainty. For example, for an event threshold of $T_e = 500$, roughly 150 update steps (=detected events) were performed until the correspondence distribution's entropy is smaller than 2, compared to 100 update steps for the larger event threshold.

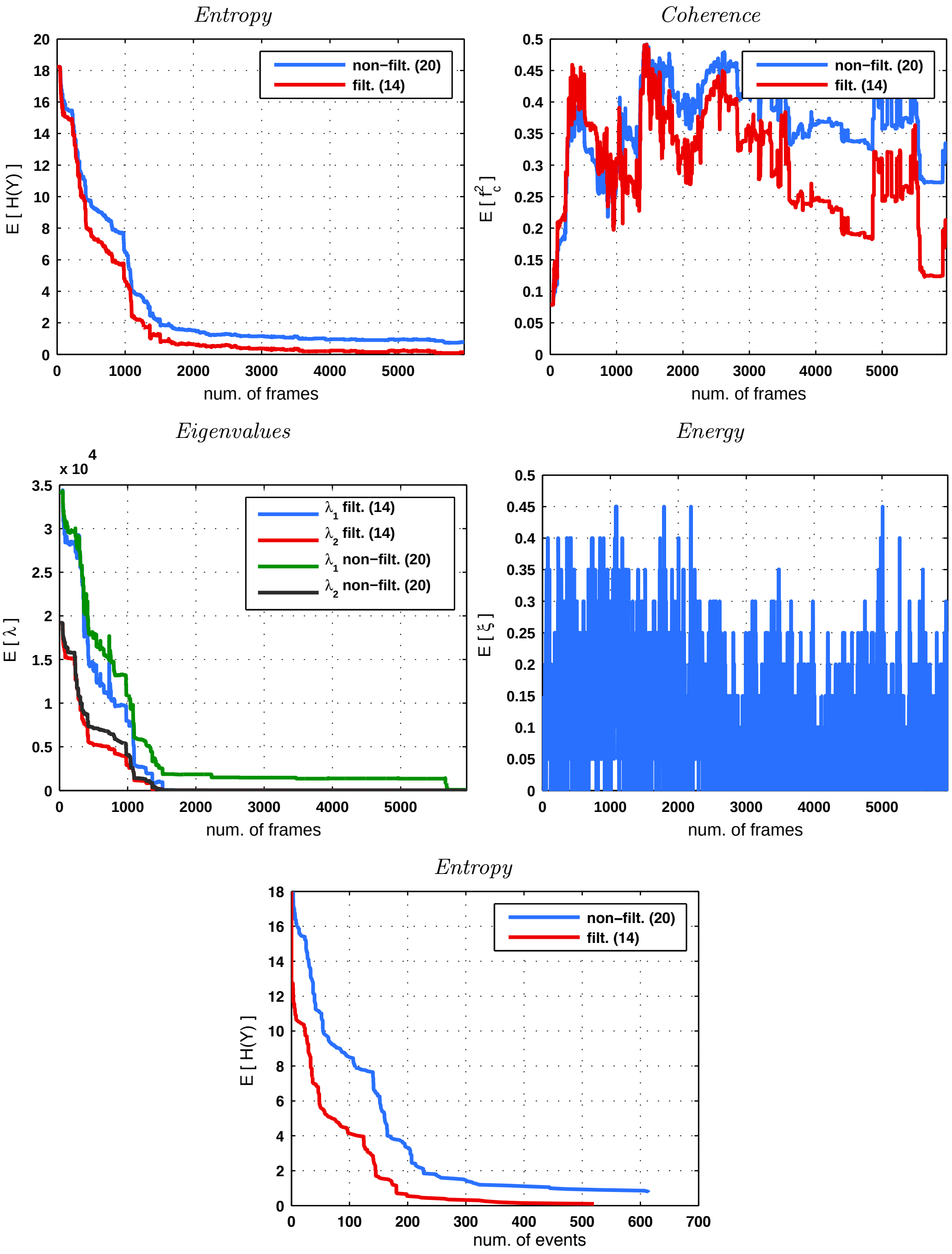


Figure 4.21: **Correspondence distribution's attributes for `GUCar`:** Parameter setting $(10,000, 6,000, 600)$. Plots of the correspondence distribution's entropy, coherence measure, eigenvalues of the covariance matrix, spent energy and entropy vs. spent energy averaged over all seed pixels, respectively. Best viewed in colour. See Sec. 4.2.6 for details.

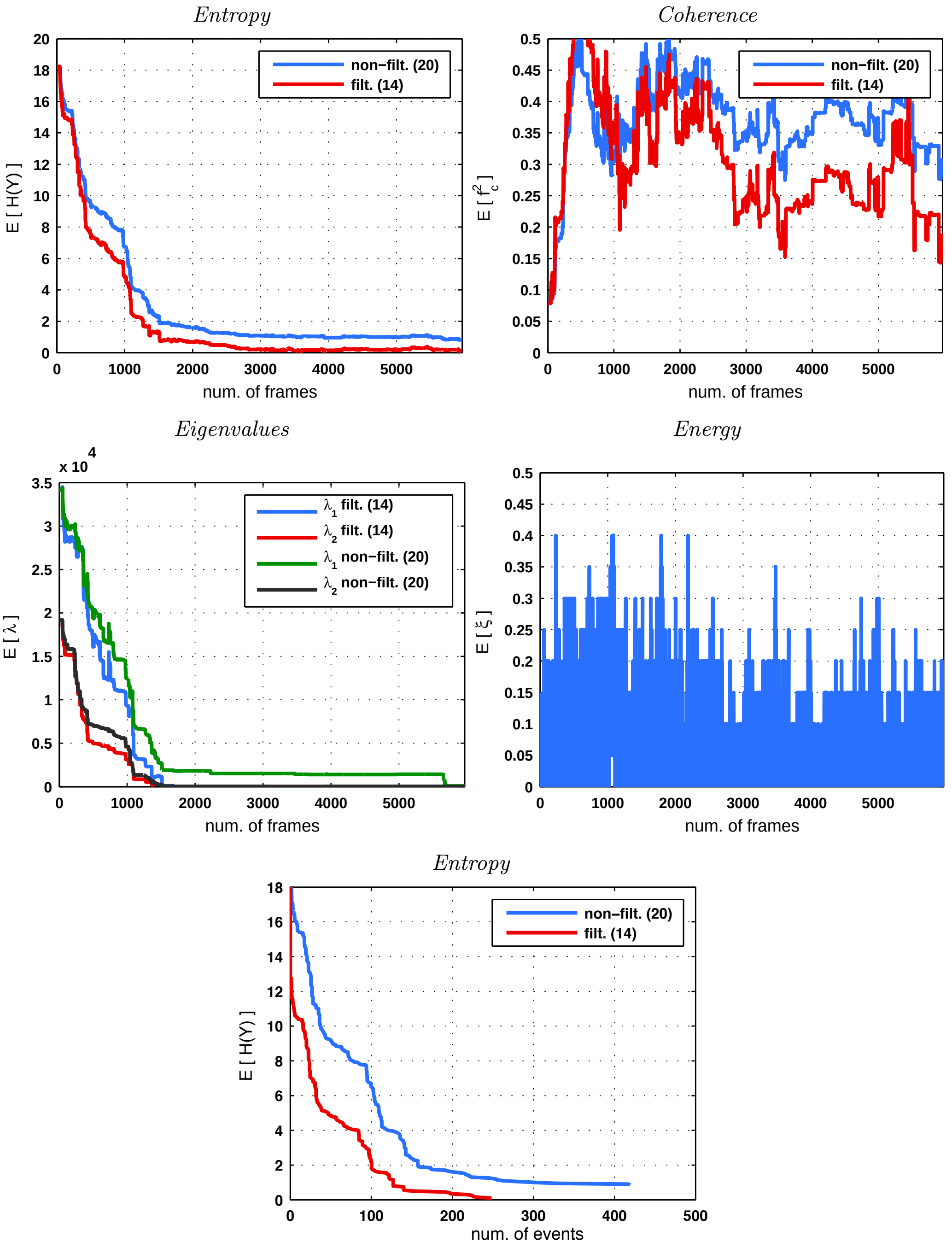


Figure 4.22: **Correspondence distribution's attributes for `GUCar` (doubled event threshold):** Parameter setting $(10,000, 6,000, 1, 200)$. Plots of the correspondence distribution's entropy, coherence measure, eigenvalues of the covariance matrix, spent energy and entropy vs. spent energy averaged over all seed pixels, respectively. Best viewed in colour. See Sec. 4.2.6 for details.

Next, I apply the learning scheme on a coarse scale of `GUCar`, specifically on the 3rd pyramid level. The resolution of the input images reduces from 640×480 pixels to 160×120 pixels. This results in the same memory footprint per seed pixel as in `GUBo1616` (cf. Sec. 4.2.4), given by $160 \cdot 120 \cdot 4\mathrm{B} \cdot 2^{-20} \approx 0.07\,\mathrm{Mib}$ (cf. Eq. 4.4). Figure 4.23 (left column) visualises the learnt correspondences after 800, 4,200 and 14,200 processed frames. In Fig. 4.23 (right column) the learnt correspondences are visualised when back-projected onto the fine (=original) scale. Additionally, the corresponding epipolar lines are shown. It can be seen that the back-projected correspondences encapsulate parts of the epipolar ray. As has been explained in Sec. 3.7, we may now refine the learnt correspondences on a finer scale and restrict the search range based on the back-projected correspondences from the coarse scale.

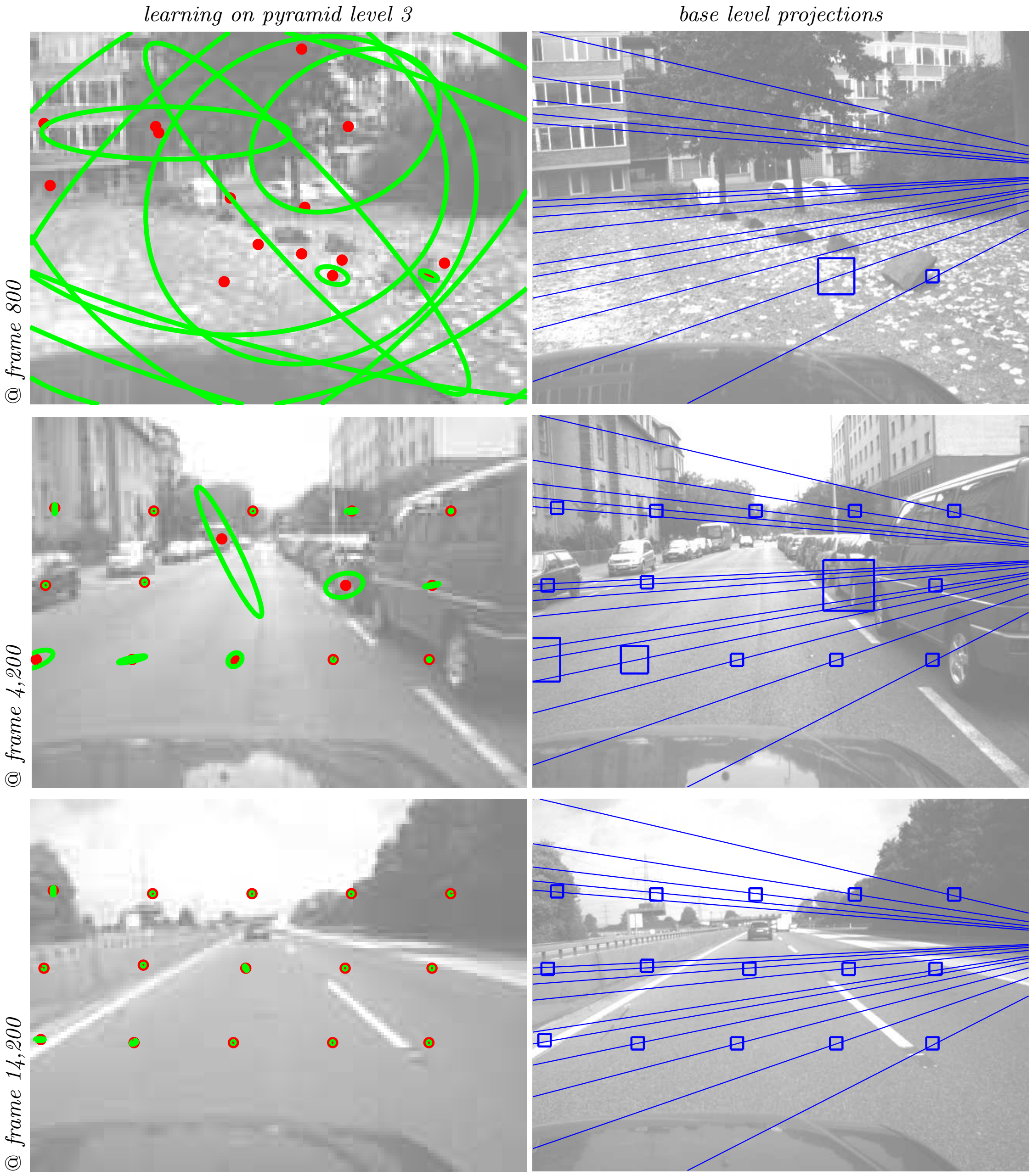


Figure 4.23: **Exemplary results for `GUCar` (coarse scale):** Learning correspondences from a moving binocular camera setup on the third level of the spatial pyramid. (first column) Correspondences learnt on the coarse scale after 800, 4,200 and 14,200 frames have been processed. (second column) Correspondences learnt on the coarse scale are back-projected onto the fine scale and constitute a rectangular region where the true correspondence is expected. The spatial extend of these regions is coupled with the eigenvalues of the correspondence distribution's covariance matrix. See Sec. 4.2.6 for details.

4.3 Summary and Conclusion

I presented the applicability of TCA to learn correspondences in real world binocular camera setups. Based on the proposed scheme, correspondences and grey value transfer functions can be learnt for uncalibrated cameras, where the camera views are translated and/or rotated w.r.t. to each other or show a large scale difference. The cameras may be static or moving. The only assumption that is made is that the relative orientation of the cameras is fixed. I stress that TCA learns correspondence distributions, which not necessarily represent the true correspondence at a specific point in time.

5 Learning Motion Correspondences

5.1 Introduction

Previously, we have seen that stereo correspondences can be learnt unsupervised via TCA. I will now turn to the problem of learning motion *within* a monocular stream of images taken from a (not necessarily) moving camera. As for the stereo case, I do not compute individual motion vector fields, which connect two particular images of a sequence. Instead, I learn the *statistics* of such motion vector fields without explicitly computing motion vectors. Specifically, I learn an average motion map of the regarded scene. Knowledge about the average motion may then serve as a prior in estimating instantaneous motion, i.e., optical flow, or to detect abnormal motion.

Besides learning average motion maps over long image sequences, I am also interested in learning basis flow fields which can be associated with the camera's ego-motion. To this end, I introduce a simple latent variable model in which the ego-motion is the latent variable which may take on the states $\{\textit{forward}, \textit{left}, \textit{right}\}$. As will be shown in the following, with each of these states, a characteristic average motion map is associated which can be learnt via TCA. During training, I extract the current motion state based on a method known as *phase correlation* (Kuglin and Hines 1975b). However, I also show that the state of the latent variable may as well be inferred by applying TCA in a specific manner.

5.1.1 Related Work

Up to now, many different algorithms to estimate the optical flow have been proposed, ranging from local to global methods such as the classical work by Lucas and Kanade (Lucas and Kanade 1981) and Horn and Schunck (Horn and Schunck 1981). Since then, the basic ideas of brightness constancy and/or locally constant motion have been used and extended many times (see (Baker et al. 2011; Sun, Roth, and Black 2010) for a recent overview). However, these methods do not explain how motion perception may evolve and may be learnt over time.

Learning of motion patterns (persistent motion) has found attention in the computer vision community, especially for video surveillance. Here, the goal is to model and extract persistent or normal motion patterns, which may be used to detect abnormal behaviour or may be used as a prior in a tracking application. In (Wright and Pless 2005), motion patterns are extracted based on the 3-D structure tensor computed at each pixel. In (Grimson et al. 1998) motion patterns are extracted from object tracks. In (Hu et al. 2008) motion patterns are determined based on sparse or dense optical flow fields. Motion fields are clustered to form different motion patterns. Compared to these approaches, I never build object tracks and I never explicitly compute the optical flow. Instead, I apply TCA and solely rely on the temporal difference of single pixels and aggregate motion/correspondence candidates over time. I may use the term motion and correspondence candidate interchangeably in the following; a motion vector simply encodes a pixel correspondence between temporally consecutive images. The presented approach is able to represent multiple (different) dominant motions observed at a single pixel as well.

In the vision community, learning optical flow is mainly addressed by estimating (=learning) the parameters of a specific motion model from ground truth data. The focus clearly lies on designing methods that advance the state-of-the-art by means of accuracy and/or running time on, e.g., the Middlebury benchmark (Baker et al. 2011). Sun et al. (Sun, Roth, Lewis, et al. 2008) learn a statistical model of the spatial properties of optical flow by learning the parameters of the model from ground truth data. They show that the model captures the statistics of optical flow and outperforms several standard methods. In contrast to this, I am interested in exploring how the statistics of optical flow may be learnt from scratch without any supervision or ground truth.

Roberts et al. (Roberts et al. 2009) learn a subspace of dense optical flow. Based on the learnt optical flow subspace a dense motion map may be inferred from sparse measurements. Furthermore, the learnt subspace may be used to estimate the ego-motion of a moving platform equipped with cameras. Sparse flow measurements are generated from Harris corners and tracked using the KLT method. A dense map is then generated by identifying the subspace coordinates of the sparse flow. Subsequently, a linear mapping from ground truth platform motion to subspace coordinates is estimated and used for ego-motion estimation. Based on the model described in (ibid.), Herdtweck and Curio (Herdtweck and Curio 2012) estimate the platform heading from monocular visual cues. They infer the *Focus of Expansion* (FoE), and require sparse flow and incremental platform motion for training. The FoE is the spatial location from which all optical flow vectors seem to emanate. The FoE may be computed based on the divergence of the optical flow field. In contrast to this, I present experimental results for inferring the hidden variables proportional to the yaw rate of a moving camera without the need for platform motion data.

In the deep learning community, there is an interest in learning features that relate pairs of images by means of spatial or spatiotemporal mappings. These features are typically given in form of learnt filters (=parameters) of energy-models and probabilistic generative models (Coates et al. 2011; Le et al. 2011; Memisevic 2013). While these methods were shown to extract meaningful mappings, usually the learning step of these models is rather involved. In contrast to this, the presented method can adapt to the computational power available while being easily parallelised if desired.

5.2 TCA for Motion (TCAM)

5.2.1 Approach

As for the stereo case, I learn average motion by applying TCA to two streams of images. However, the image streams are now not originating from two cameras, but are temporally shifted versions of the same monocular image stream.

Specifically, I regard a camera $\mathcal{C}_i$ and its (monocular) image sequence $I_t \in \mathbb{R}^{M \times N}, t = 0, 1, ..., T$ (cf. Sec. 2.1). From image sequence I_t, I generate a temporally shifted image sequence $I_{t;\tau}, t = 0 + \tau, 1 + \tau, ..., T + \tau$. Throughout the following I set $\tau = 1$. Then, between two consecutive time steps t and $t + 1$, a correspondence between two pixel locations is encoded by means of the optical flow vector.

The event signal generation is carried out as for the stereo case. As has been described previously, the correspondence distributions estimated via the posterior update scheme tend to develop a point like structure. This is due to the fact that the model forgets a learnt correspondence as soon as there is not enough evidence for it within the regarded data. A correspondence will only be persistent if evidence in the form of matched events is detected regularly (see Sec. 4.2.6 for a detailed discussion). Within the regarded sequences, the optical flow which encodes the pixel correspondences between consecutive time steps varies considerably. This is different from the stereo case where the observed depth profiles are far more regular. From my experiments, I found it essential to perform the event matching via the approximate update scheme (cf. Eq. 3.61). Specifically, for each event detected at a seed pixel, I identify those pixel locations which show a similar grey value change within time t and $t+1$, as they are likely to be the true corresponding pixel. Typically, there will be a set of pixels showing a similar event, and without taking further information into account it is in general not possible to identify the single true correspondence between two time steps. Instead, we will always have a set of possible correspondences which are given by the set:

$$
\begin{aligned}
\Omega_{pc}(\boldsymbol{x}_i, t) = \{\boldsymbol{y}_j \in \mathcal{I}_j :& && \qquad (5.1)\\
& f_{e,t}(s_i[\boldsymbol{x}_i, t-1], s_i[\boldsymbol{x}_i, t]) && > T_e \\
\wedge \quad & f_m(s_i(\boldsymbol{x}_i, t-1), s_j(\boldsymbol{y}_j, t-1)) && < T_m \qquad (5.2)\\
\wedge \quad & f_m(s_i(\boldsymbol{x}_i, t)), s_j(\boldsymbol{y}_j, t)) && < T_m\}
\end{aligned}
$$

with $f_m(a, b) = |a - b|$ and where $T_m = 2\sigma$ accounts for the noise standard deviation.

The set of matched events $\Omega_{\boldsymbol{x}_i}$ at time t may be seen as a weak hypothesis for the true correspondence; the set of all pixel coordinates is reduced by discarding pixels showing no or a different event. A strong hypothesis for the true correspondence is then formed by adding up the possible correspondences over time in one accumulator array $\boldsymbol{A}_{\boldsymbol{x}_i}$ per seed pixel $\boldsymbol{x}_i$ (cf. Eq. 3.61).

Upon normalisation (such that the accumulator counts sum to one), the accumulator array is processed as the correspondence distribution. However, the accumulator will also contain a considerable amount of noise which makes it necessary to perform a preprocessing prior extracting first and second order central moments. To this end, the accumulator is subject to a threshold operation where each cell count smaller than 50% of the accumulator's maximum value is set to zero. In other words, I only keep those accumulator cells for which the probability of being the true correspondence is above chance. Throughout the following experiments, the event matching is performed via the approximate update scheme.

In contrast to the stereo case, I do not learn and apply a GVTF as I assume the camera transfer function to be static (at least over several frames). Furthermore, I assume that illumination differences between triplets of images are negligible. The method will not suffer from slow and gradual illumination changes, as the events need signal changes above a certain threshold value. Sudden strong illumination changes (flashes etc.) will certainly cause event coincidences at many or all candidate channels. However, these effects will average out over time. Similarly, if occlusions only appear from time to time, this again has no influence on learning the average flow fields as this effect should average out too. In a textureless region, we may not learn anything as no motion can be detected and hence no events occur.

Recall that for the stereo case, I assumed that the true correspondence may be located at any valid pixel location in the second view. Therefore, the event matching was performed over all candidate channels. For the motion case, the range of typical motion can usually be approximated and it is almost always much smaller than the overall image size. The search area, i.e., the area in which events are matched may therefore be restricted to a window of size $w_x \times w_y$ centred on the regarded seed pixel $\boldsymbol{x}_i$ with accumulator $\boldsymbol{A}_{\boldsymbol{x}_i} \in \mathbb{R}^{w_x \times w_y}$. Of course, we now have to assume a topological order on the pixels. Parameters w_x and w_y represent the maximum average flow that can be learnt. If w_x and w_y are too small, all possible correspondences will be random, resulting in a pure noise distribution. If w_x and w_y are larger than the true average flow, we will learn the average flow, but will obviously waste computational resources.

As for the stereo case, in general we may only learn the average correspondence relation and thus the average optical flow at pixel location $\boldsymbol{x}_i$. Recall from Sec. 3.4 that it depends on the actual scene structure whether this average correspondence relation, estimated over many frames, coincides with the true correspondence at any single time step. I stress again that TCA is not meant as a competitor to classic stereo or flow algorithms tailored to return instantaneous (per frame) correspondences, but may be seen as a prior generator. There are good reasons to make use of well-engineered algorithms to estimate highly accurate optical flow, but my objective is to demonstrate that correspondence relations may be learnt and updated over time without any supervision and minimal assumptions about the given data; all that is needed are intensity changes above the sensor noise level. Observe that the average correspondence relations still contain valuable information, e.g. in order to guide a higher level process, restrict search areas, or generate confidence information.

The analysis of learnt correspondence accumulators is done as for the stereo case. After a sufficient number of frames have been processed, the accumulator will encode the average correspondence and thus the average optical flow observed within the sequence at location $\boldsymbol{x}_i$. After pre-processing and normalising the accumulator as described previously, I extract the most likely correspondence as the location of the accumulator's mean and compute a confidence measure from the accumulator's covariance matrix $\boldsymbol{C}$ (cf. Sec. 3.5).

In order to visualise the confidence for a dense flow map, I adopt a colour encoding, similar to the colour coding of flow vectors (see next section for details).

5.2.2 Parametric Flow Model

Typically, a sparse set of average flow vectors is learnt. From this set of sparse flow vectors, we may estimate a parametric model for the optical flow which takes the available uncertainty information explicitly into account. Based on the parametric model, we may then interpolate a dense flow field or initialise the learning phase for a newly selected seed pixel.

To this end, I introduce a bi-quadratic model for the optical flow similar to the one presented by us in (Guevara et al. 2012). For a given set of seed pixels $\{(x_1, x_2)^T_{i,n} = \boldsymbol{x}_{i,n}\}_N$ we learn their corresponding average flow vectors $\{(u, v)^T_{i,n} = \boldsymbol{u}_{i,n}\}_N$ via TCA as described previously. Additionally, we obtain covariance matrices $\{\boldsymbol{C}_n\}_N$ from the accumulator arrays, encoding the uncertainty about the flow estimate. We define the bi-quadratic polynomial:

$$f(\boldsymbol{x}, \boldsymbol{\theta}) = \boldsymbol{A}(\boldsymbol{x}) \cdot \boldsymbol{\theta}, \tag{5.3}$$

where $\boldsymbol{A}(\boldsymbol{x})$ is a 2×12 matrix given as:

$$\boldsymbol{A}(\boldsymbol{x}) = \begin{pmatrix} 1 & x_1 & x_2 & x_1 x_2 & x_1^2 & x_2^2 & \mathbf{0}_6^T \\ \mathbf{0}_6^T & 1 & x_1 & x_2 & x_1 x_2 & x_1^2 & x_2^2 \end{pmatrix}, \tag{5.4}$$

and where $\boldsymbol{\theta}$ is the sought 12×1 parameter vector. Our goal is to estimate the parameter vector $\boldsymbol{\theta}$ by minimising the squared difference between estimated flow vectors and their approximation by the parametric model. Specifically, the loss function Q is given by:

$$Q = \sum_{n=1}^{N} \left[\boldsymbol{u}_{i,n} - f(\boldsymbol{x}_{i,n}, \boldsymbol{\theta})\right]^T \cdot \boldsymbol{C}_n^{-1} \cdot \left[\boldsymbol{u}_{i,n} - f(\boldsymbol{x}_{i,n}, \boldsymbol{\theta})\right]. \tag{5.5}$$

We may rewrite Eq. 5.5 in matrix form as follows. Let $\boldsymbol{H}$ be the coefficient or observation matrix defined as:

$$\boldsymbol{H} = \begin{pmatrix} 1 & (x_1)_1 & (x_2)_1 & (x_1)_1(x_2)_1 & (x_1)_1^2 & (x_2)_1^2 & \mathbf{0}_6^T \\ \mathbf{0}_6^T & 1 & (x_1)_1 & (x_2)_1 & (x_1)_1(x_2)_1 & (x_1)_1^2 & (x_2)_1^2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & (x_1)_N & (x_2)_N & (x_1)_N(x_2)_N & (x_1)_N^2 & (x_2)_N^2 & \mathbf{0}_6^T \\ \mathbf{0}_6^T & 1 & (x_1)_N & (x_2)_N & (x_1)_N(x_2)_N & (x_1)_N^2 & (x_2)_N^2 \end{pmatrix}, \tag{5.6}$$

and let $\boldsymbol{z}$ be the measurement vector defined as:

$$\boldsymbol{z} = \begin{pmatrix} u_1 \\ v_1 \\ \vdots \\ u_N \\ v_N \end{pmatrix}. \tag{5.7}$$

Furthermore, let $\boldsymbol{W}$ be the weight matrix with the given 2×2 inverse covariance matrices along its main diagonal:

$$\boldsymbol{W} = \mathrm{diag}(\boldsymbol{C}_1^{-1}, .., \boldsymbol{C}_N^{-1}). \tag{5.8}$$

The loss function Q may now be rewritten according to:

$$Q = (\boldsymbol{z} - \boldsymbol{H\theta})^T W (\boldsymbol{z} - \boldsymbol{H\theta}). \tag{5.9}$$

Taking the derivative w.r.t. $\boldsymbol{\theta}$, we obtain the sought parameter vector from the normal equation according to (Mendel 1995):

$$\hat{\boldsymbol{\theta}} = (\boldsymbol{H}^T \boldsymbol{W} \boldsymbol{H})^{-1} \boldsymbol{H}^T \boldsymbol{W} \boldsymbol{z}. \tag{5.10}$$

5.2.2.1 Computational Complexity

Regarding the computational complexity, TCAM has $\mathcal{O}(n)$ complexity in the number of seed pixels and $\mathcal{O}(n^2)$ complexity in the size of the accumulator, where the same holds for the memory consumption. Depending on the computational power available, the number of seed pixels may be chosen adaptively; put on hold etc. The scheme is easily parallelised if desired.

5.3 Experiments

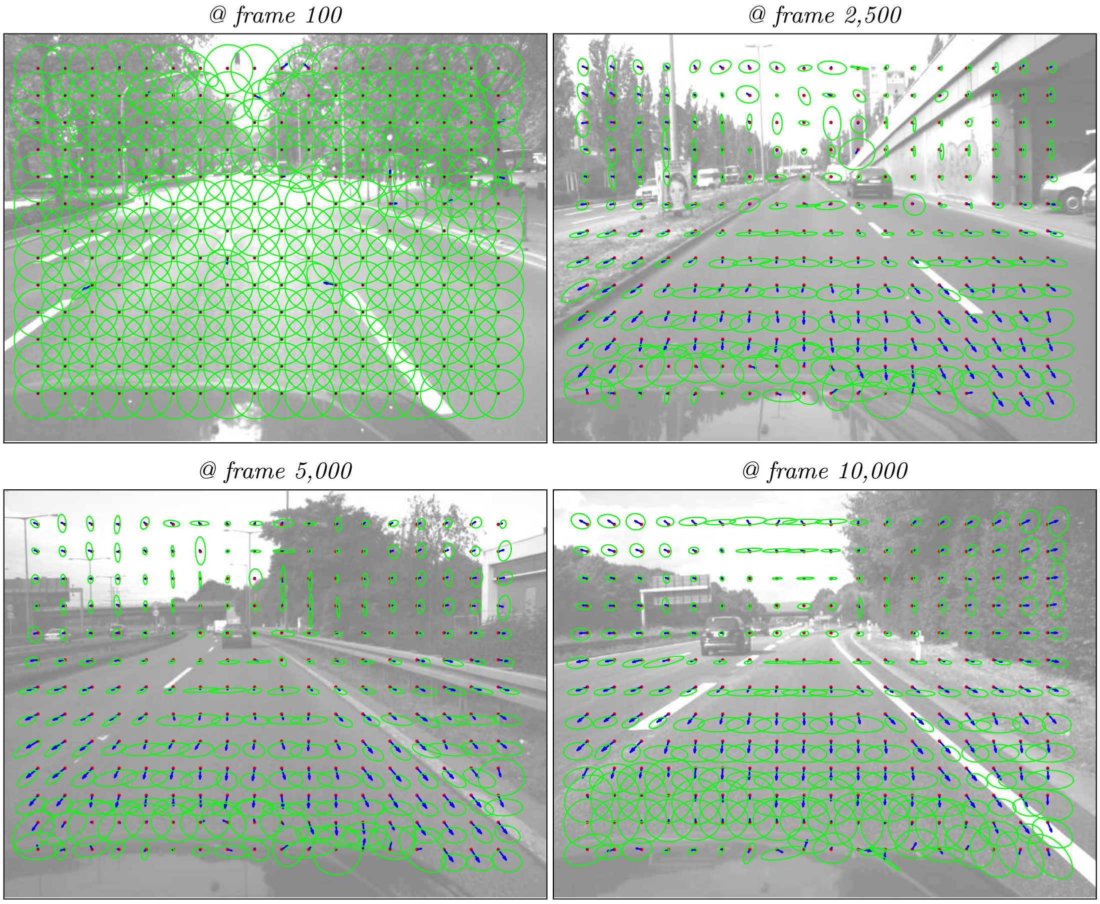
In this section, I present experimental results of the proposed method and show its applicability for several real world sequences. In all experiments, no ground truth was used; all information has been learnt from scratch. Depending on the computational resources available and the number of seed pixels chosen, the method can operate in real time. In order to recover dense flow fields, the scheme should be applied within a multi scale framework; an initial coarse map is then refined, making use of an optimal accumulator size given by the estimated flow from a previous level. Furthermore, the scheme may easily be parallelised and thus harness the computational power of massively parallel architectures such as the GPU. However, for the results presented, I make use of a baseline implementation without coarse to fine and/or parallel computations.

5.3.1 Learning Average Motion Maps

In the first experiment, I learn sparse and dense average motion maps for sequences `GUCar`, `Kitti-Odo`, `GUOmni` and `Virat`. All sequences are converted to greyscale if needed. Given the event threshold T_e, the matching threshold T_m and the window size $w_x \times w_y$, the process of event detection and event matching is performed for many frames (cf. Sec. 3.4). For each seed pixel, I extract the average flow vector observed as the coordinates of the mean of its associated accumulator. For each learnt flow vector, a measure of confidence is provided by the accumulator's covariance matrix (cf. 3.5 for details). For a sparse set of seed pixels, the confidence measure is visualised as a covariance error ellipse (cf. 3.5). For a dense visualisation, I adopt the usual optical flow colour coding scheme and apply it to the first and second eigenvector of the accumulator's covariance matrix, scaled by the associated eigenvalue. Throughout all figures, the colour coded flow vectors are normalised to the maximal possible flow (as given by the accumulator size). Similarly, the confidence maps are normalised as well. See Sec. 2 and Fig. 2.4 for details regarding the colour coding scheme.

Experiments on `GUCar`: I set the event threshold to $T_e = 30$ and the matching threshold to $T_m = 10$. The accumulator size is set to $w_x \times w_y = 40 \times 40$ pixels. I select a regular grid of seed pixels, for which the average optical flow is learnt over 10,000 frames. Figure 5.1 visualises the obtained sparse flow after 100, 2,500, 5,000 and 10,000 frames have been processed. Initially, the uncertainty as reflected by the error ellipses is large and becomes smaller as more frames are processed. The average flow vectors as well as their associated error ellipses develop a characteristic orientation and size, depending on the image region. For example, for pixels in front of the engine hood, where the corresponding 3-D points are close to the camera, the average flow is the largest. For pixels where the corresponding 3-D point is further away, the corresponding average flow is smaller. The focus of expansion is clearly visible, approximately in the middle of the image, in the vicinity of the pixels with minimum average flow vectors.

In order to validate the results obtained for the sparse subset of seed pixels, I also applied the learning scheme to all pixels. Figure 5.3 shows the dense flow and confidence maps after having processed $10,000$ frames. The direction and magnitude of each flow



@ frame 100

@ frame 2,500

@ frame 5,000

@ frame 10,000

Figure 5.1: **Sparse flow for `GUCar`:** (within each image) For each seed pixel (red dot) its learnt average flow (blue vector) and corresponding confidence encoded by means of a (green) covariance error ellipse is shown. Best viewed in colour and up scaled. See text for details.

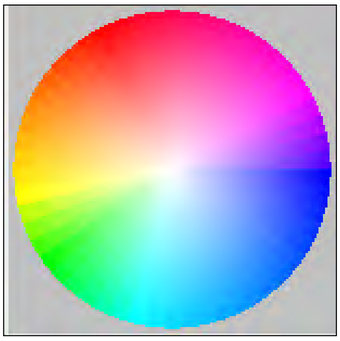
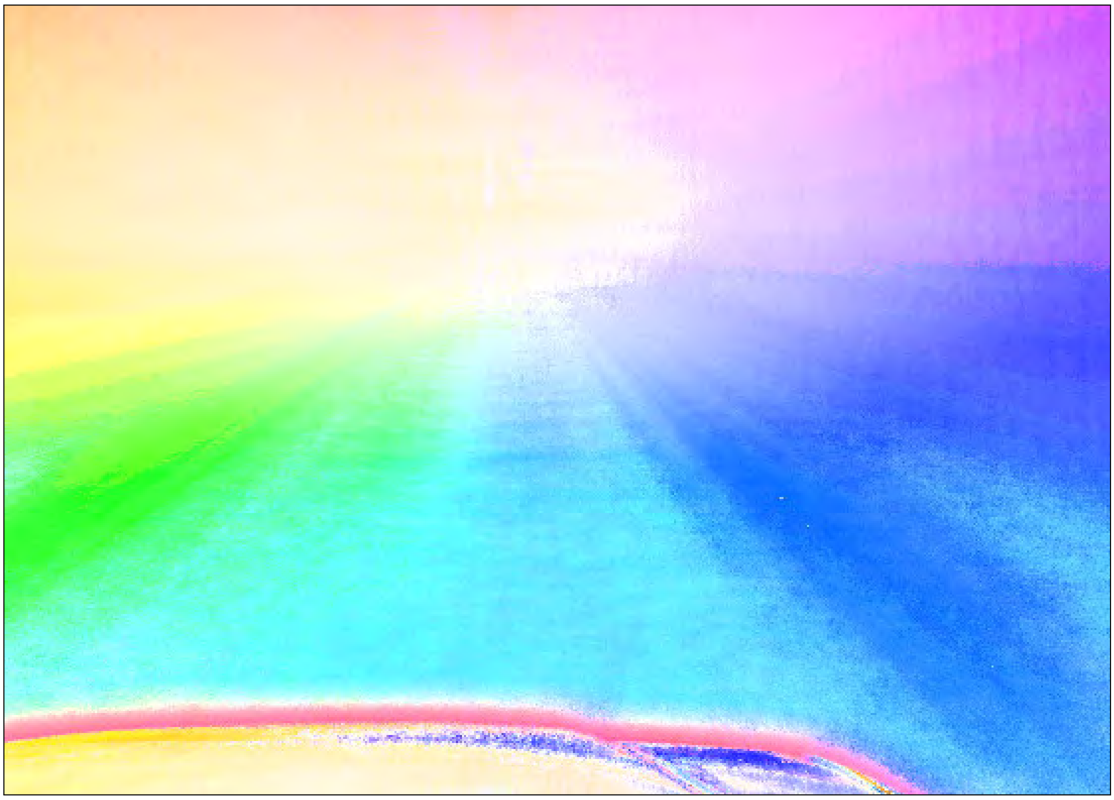


Figure 5.2: **Color wheel:** Throughout Sec. 5.3.1, this color wheel is used to visualise learnt motion maps. Best viewed in colour. See text for details.

vector is converted into a hue and saturation value in the HSV colour space, according to the Middlebury (Baker et al. 2011) standard. The color wheel is shown in Fig. 5.2. It can be seen that the learnt average flow corresponds to the flow expected for a natural street scene, where the camera follows a straight path most of the time. The orientation of the flow vectors in the left half of the map lie within $[90°, 270°]$, while in the right half the orientation lies in $[90°, -90°]$. Note the area where the flow vectors have a low magnitude (white area). This area corresponds to the average position of the focus of expansion (D. Ballard and Brown 1982), i.e. where all points seem to emanate. In Fig. 5.3 I also show a colour coded representation of the eigenvectors of the covariance error ellipses. The orientation of the eigenvectors determines the hue and the corresponding eigenvalues the saturation in HSV colour space. From these confidence maps, it can be seen that within the area directly in front of the car, the confidence about the learnt flow vector is rather low in all directions. This is to be expected, as this area corresponds to the safety clearance, where most of the time only the homogenous road surface is visible and only few events are generated. Highest confidences is attained in the upper half of the image, were most of the events are generated.

Next, I estimate the parameters of the bi-quadratic model introduced in Sec. 5.5. Figure 5.4 visualises a sparse subset of learnt and interpolated flow vectors, respectively. The interpolated flow is evaluated at non-seed pixel locations. Comparing the learnt and interpolated flow vectors, it can be seen that the interpolated flow is more smooth. While I take covariance information into account when estimating the model parameters, the approximation is certainly not perfect. For example, it can be seen that the flow vectors in the upper left and right part are slightly too steep. However, the gist of the average flow observed by a forward moving platform (and static environment) is clearly visible. More accurate results should be obtainable by clustering the learnt sparse flow and learning individual parametric models. However, this remains future work.

dense average flow map



eigenvector for smaller eigenvalue *eigenvector for larger eigenvalue*

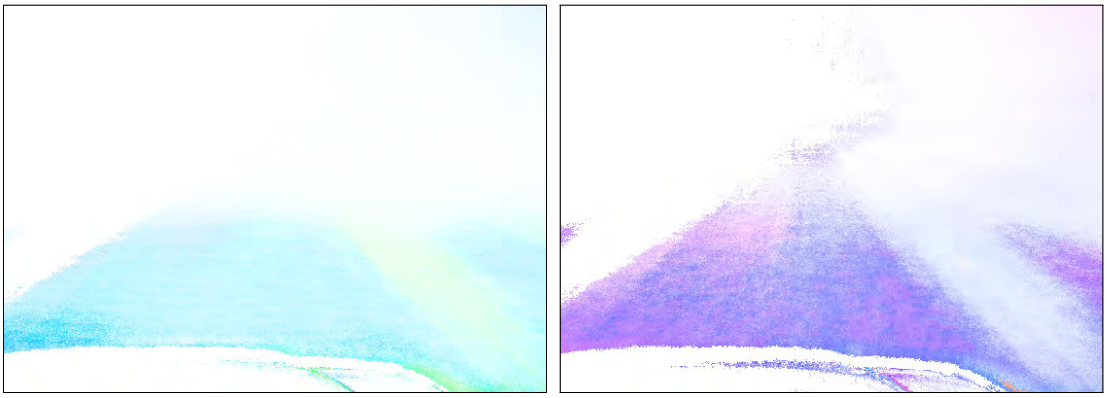


Figure 5.3: **Dense flow and confidence maps for `GUCar`:** (top) Colour coded dense flow map after 10,000 processed frames, (bottom) colour coded eigenvector corresponding to the (left) smaller and (right) larger eigenvalue of the accumulator's covariance matrix, respectively. Colour coding according to the Middlebury (Baker et al. 2011) standard. See text for details. Figure only interpretable when viewed in colour.

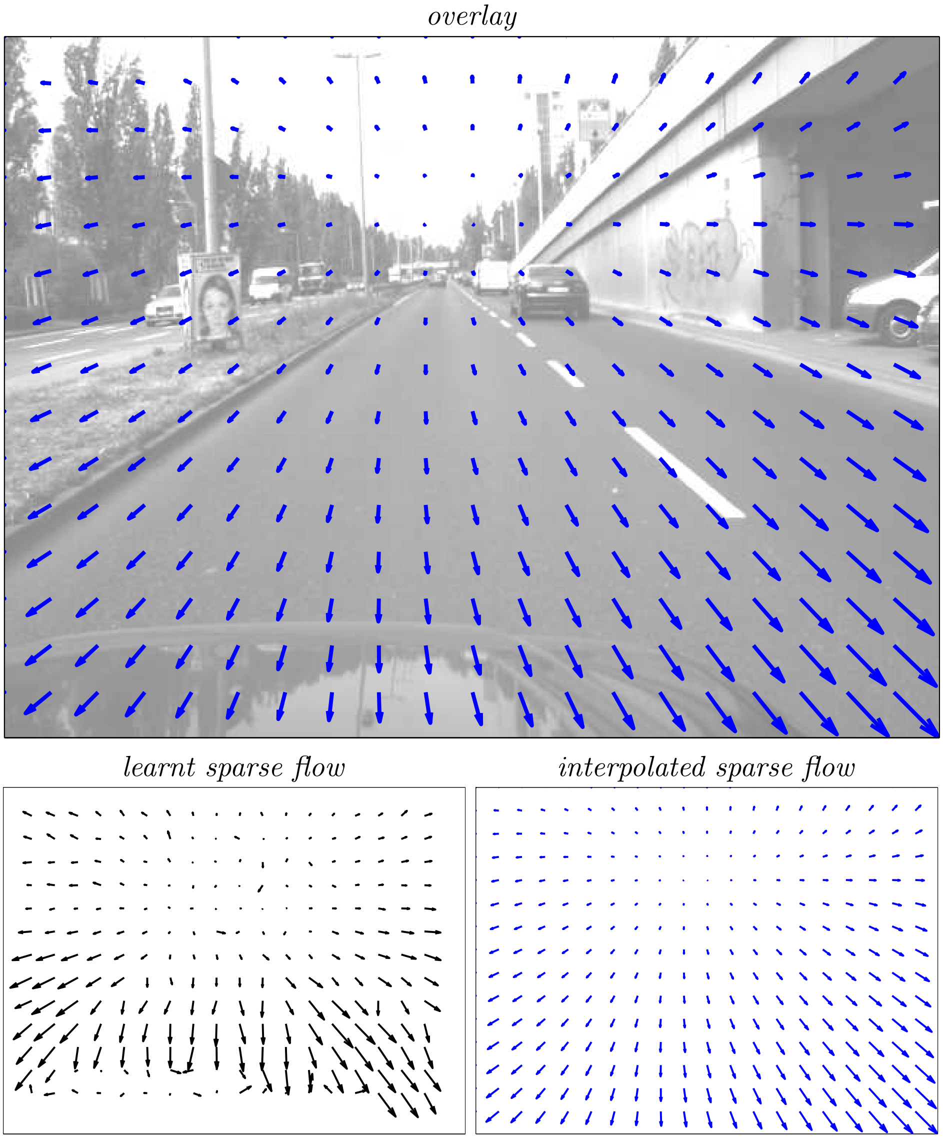


Figure 5.4: **Interpolated sparse flow for `GUCar`:** Based on a sparse set of flow vectors learnt via TCA (bottom left) a bi-quadratic parametric model is fitted. The fitted model may then be used to interpolate the optical flow at non-seed pixel locations as shown in (top) the overlay of interpolated sparse flow vectors in blue and the same set of interpolated flow vectors w/o background (bottom right). Note: Flow vectors are scaled by a factor of 2 for visualisation purposes. See text for details.

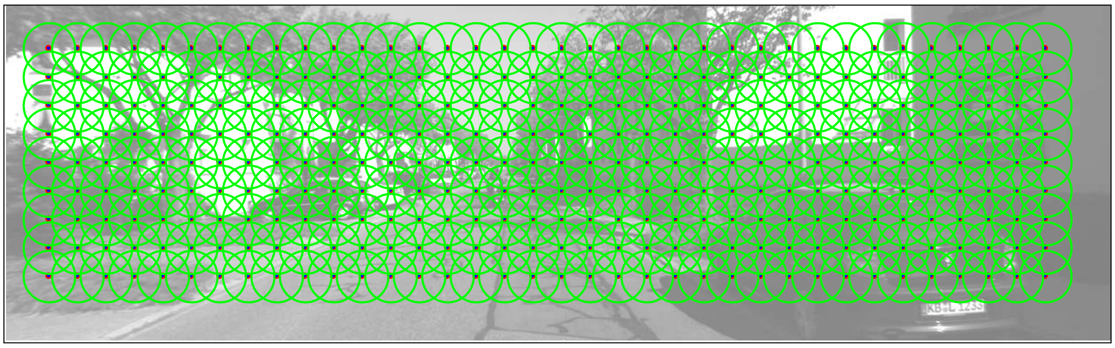
Experiments on `Kitti-Odo`: I select the same parameter setting as for sequence `GUCar` ($T_e = 30$, $T_m = 10$, accumulator size 40×40 pixels). Again I select a set of seed pixels laid out in a regular grid.

Figure 5.5 visualises the learnt average flow vectors at initialisation and after 500 and 4,500 frames have been processed. As it is characteristic for TCA, the uncertainty about the learnt correspondence is large in the early learning stages and becomes smaller the more events are detected and matched. Similar as for sequence `GUCar`, the orientation and magnitude of the learnt flow vectors are characteristic for a camera moving mainly forward through a (most of the time) static urban environment. The focus of expansion lies approximately in the middle of the image, as can be seen from average flow vectors with very small magnitude. As expected, the closer the seed pixels, or rather their corresponding 3-D locations, are to the camera, the larger the magnitude of the associated flow vectors. It is interesting to note that the uncertainty of the flow vectors in the lower left and lower right part of the image have a considerably larger uncertainty than the ones in the lower middle and upper left and right part. This is due to rather large depth and hence motion changes caused by parked and absent cars.

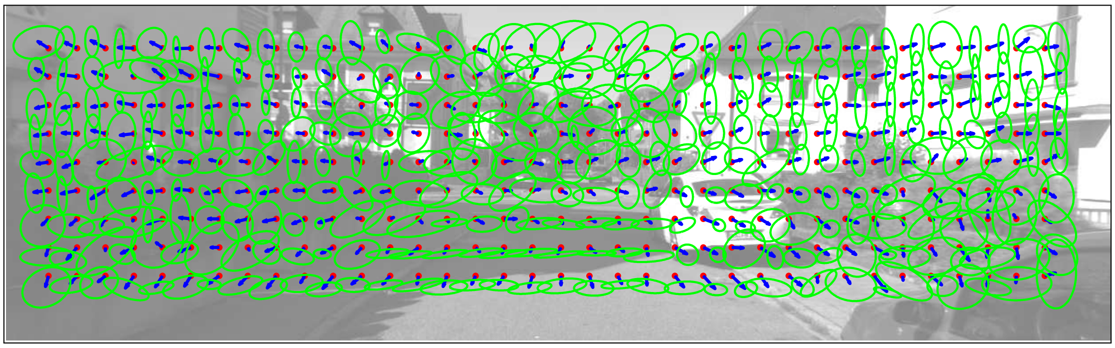
In Fig. 5.6 I visualise a dense flow map and its associated confidence maps after 4,500 frames have been processed. I obtain a similar flow field as for sequence `GUCar`. This is to be expected, as in both sequences the camera is moved through an urban street scene. However, note that the flow map for `Kitti-Odo` is considerably more noisier which is due to the rather small number of frames available.

Next, I estimate the parameters of the parametric flow model and show a sparse set of interpolated flow vectors in Fig. 5.7. We again see that the parametric flow model is able to capture the gist of the expected flow field, i.e., flow vectors with large magnitude for those pixel locations where the corresponding (average) 3-D location is close to the camera. The magnitude decreases towards the focus of expansion, which is approximately located in the centre of the image.

@initialization



@frame 500



@frame 4,500

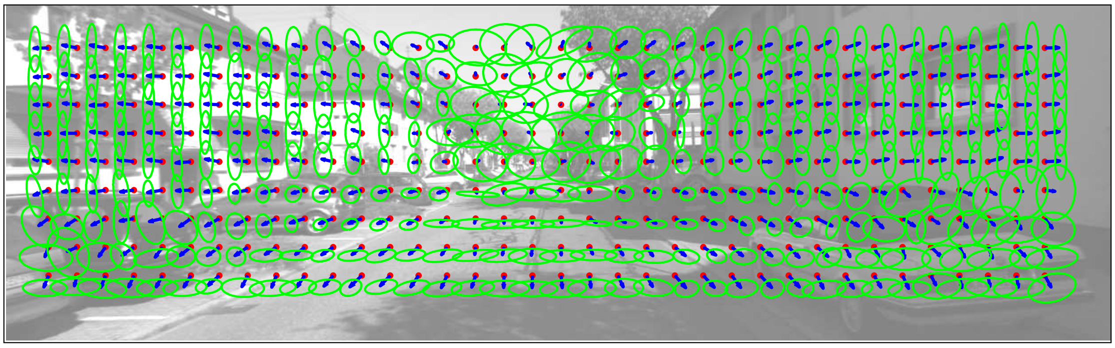
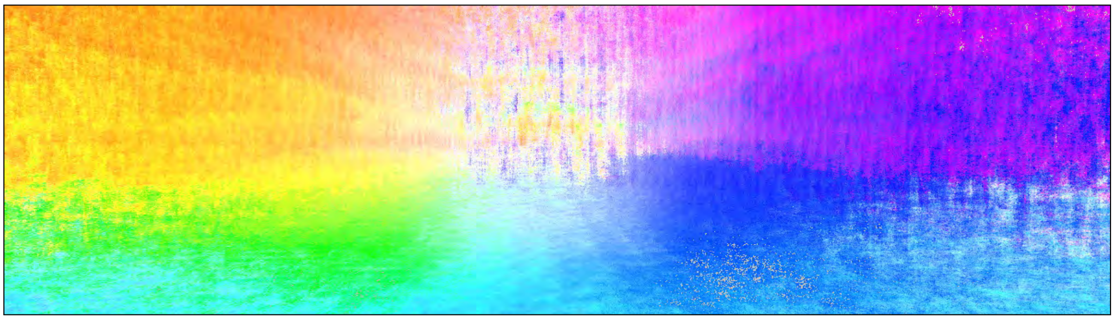
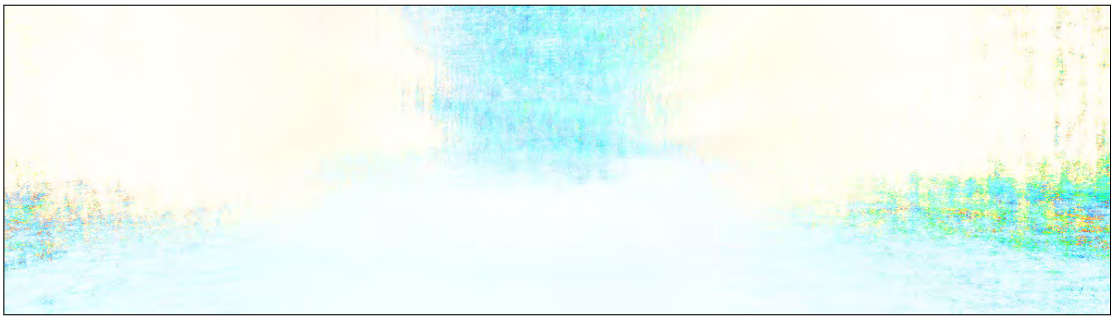


Figure 5.5: **Sparse flow for `Kitti-Odo`:** (within each image) For each seed pixel (red dot) its learnt average flow (blue vector) and corresponding confidence encoded by means of a (green) covariance error ellipse is shown. Best viewed in colour and up scaled. See text for details.

dense average flow map



eigenvector for smaller eigenvalue



eigenvector for larger eigenvalue

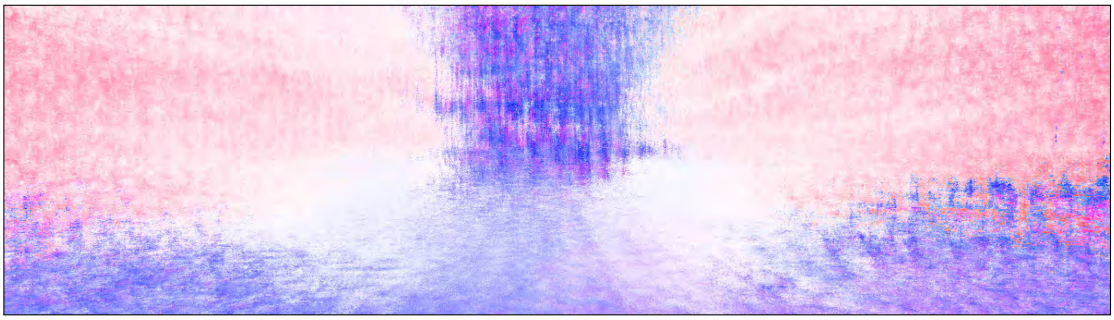


Figure 5.6: **Dense flow and confidence maps for `Kitti-Odo`:** (top) Colour coded dense flow map after 4,500 processed frames, colour coded eigenvector corresponding to the (middle) smaller and (bottom) larger eigenvalue of the accumulator's covariance matrix, respectively. Colour coding according to the Middlebury (Baker et al. 2011) standard. Figure only interpretable when viewed in colour.

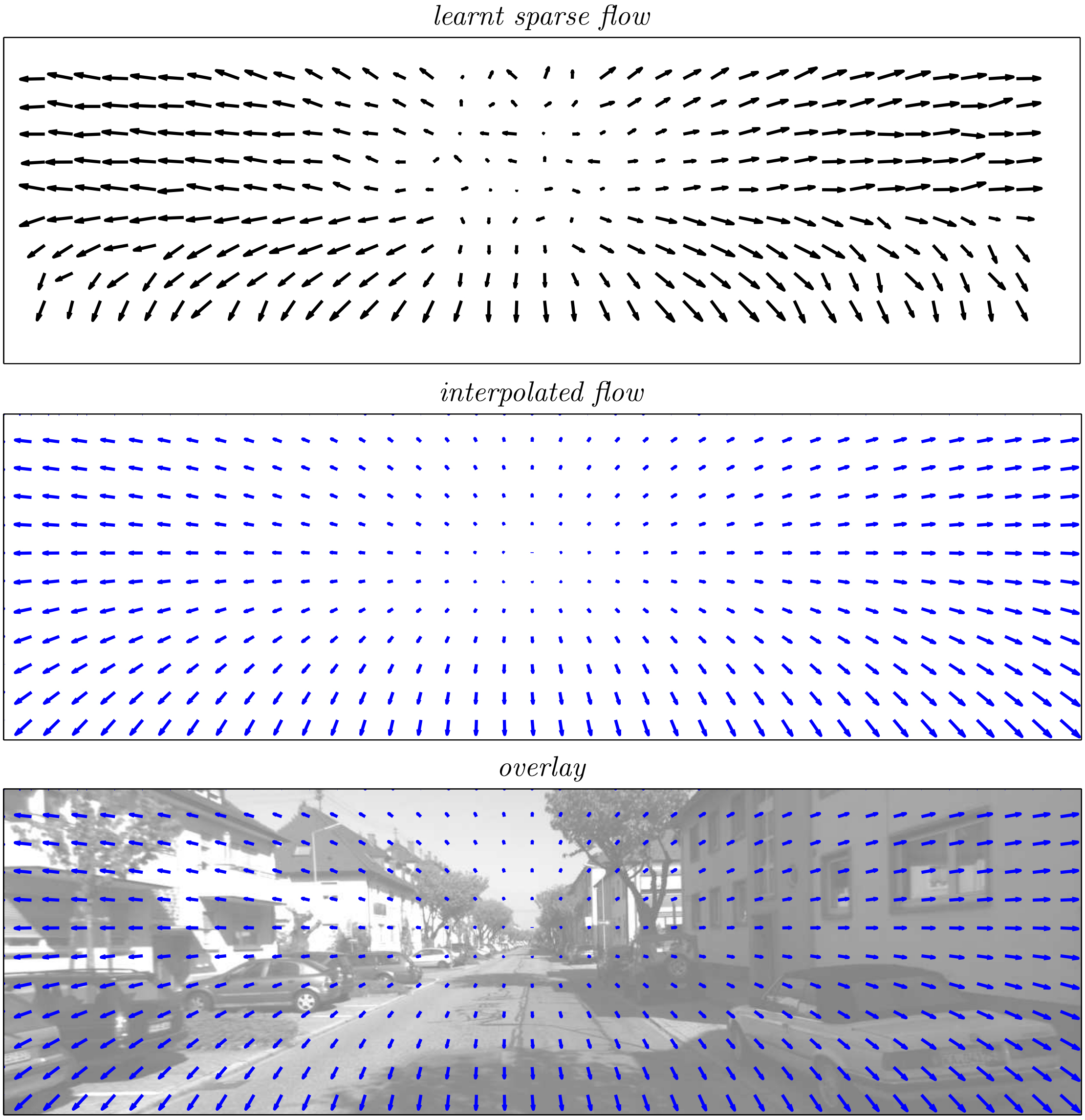
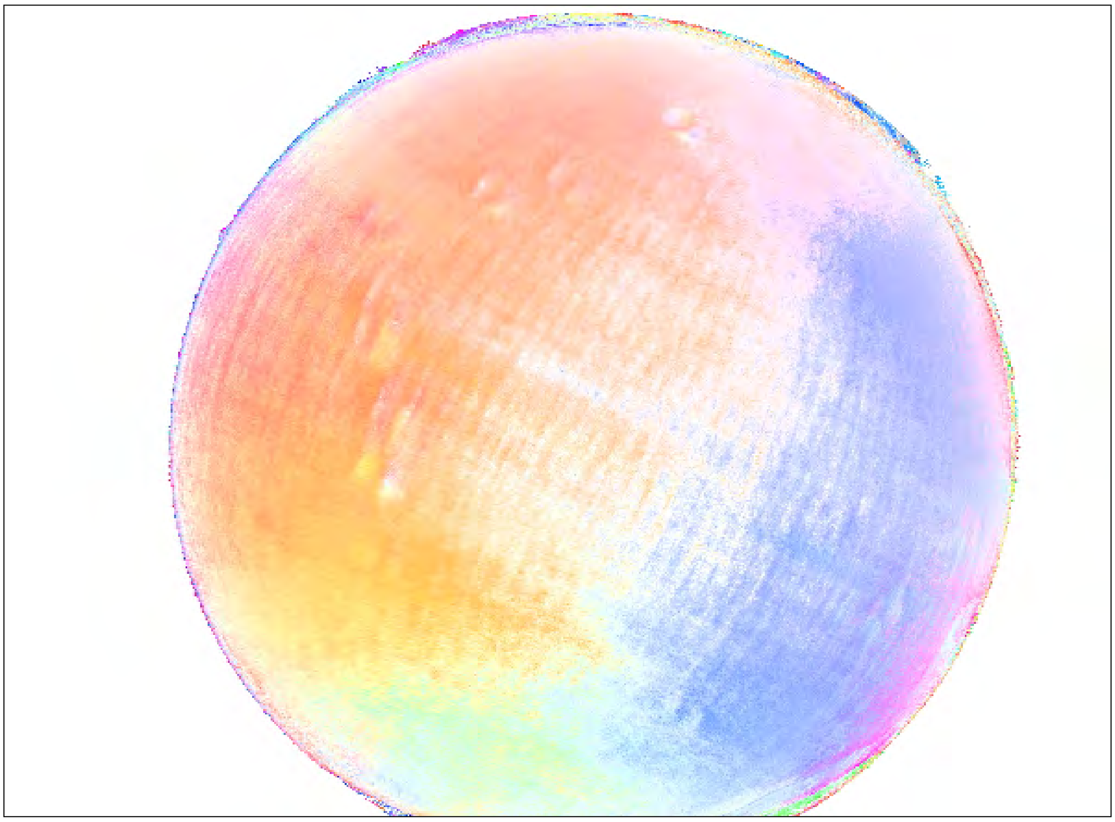
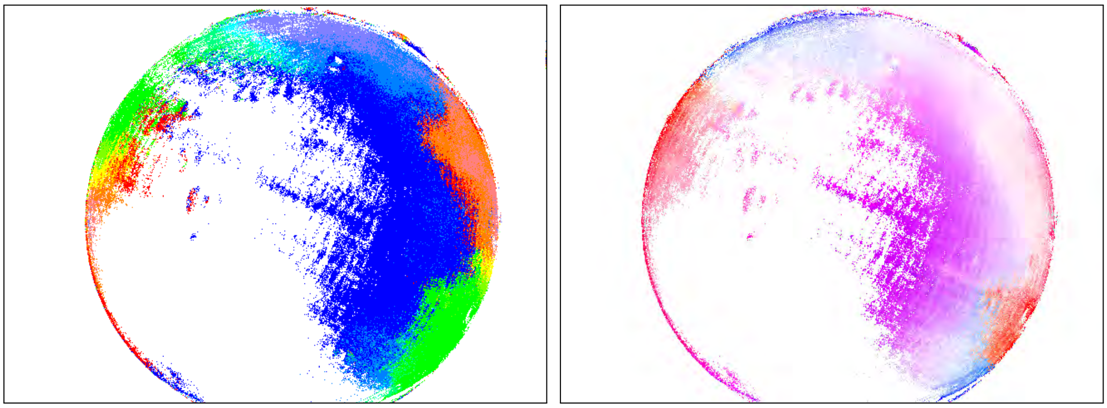


Figure 5.7: **Interpolated sparse flow for `Kitti-Odo`:** Based on a sparse set of flow vectors learnt via TCA (top) a bi-quadratic parametric model is fitted. The fitted model may then be used to interpolate the optical flow at non-seed pixel locations as shown in (bottom) the overlay of interpolated sparse flow vectors in blue and the same set of interpolated flow vectors w/o background (middle). Note: Flow vectors are scaled by a factor of 1.5 for visualisation purposes. See text for details.

Experiments on `GUOmni`: In the experiments presented so far, TCA was applied to images originating from cameras with standard geometry (≈pinhole). In this experiment, I show that we may learn average flow for cameras with fisheye lenses as well. Figure 5.8 shows a learnt dense flow map and associated confidence maps for sequence `GUOmni` after 10,000 processed frames. Observe how the average flow is clearly different from the ones presented for standard cameras previously. Objects observed through the fisheye lens are subject to severe distortions. While the camera is mainly moved forward through an office environment, the main directions of motion are approximately $k \cdot 45$ degrees with k being integer valued, as can be seen from the colour coded flow.



dense average flow map



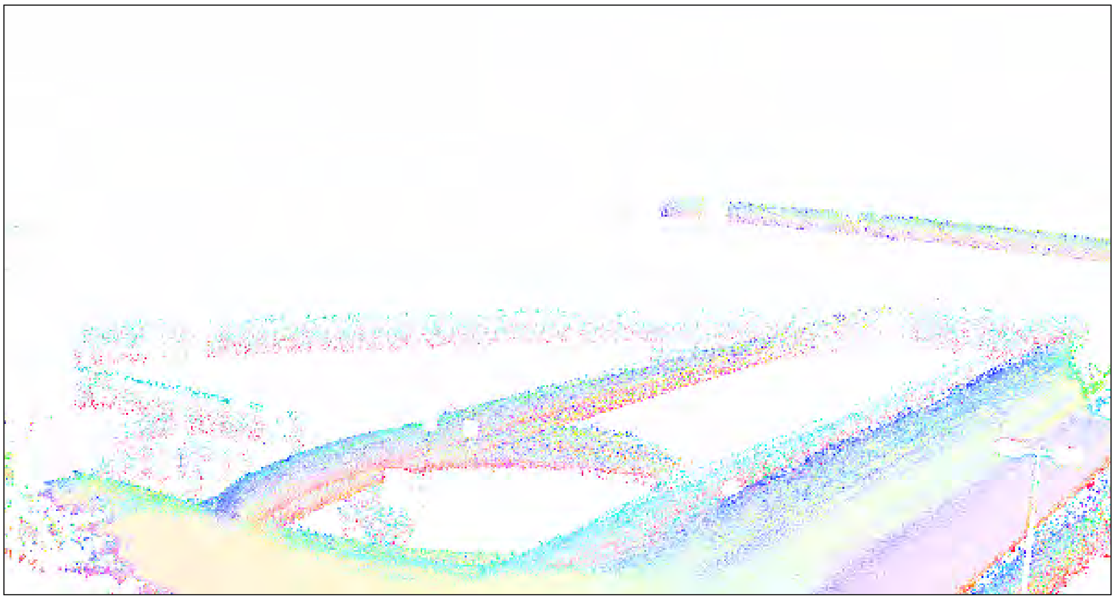
eigenvector for smaller eigenvalue

eigenvector for larger eigenvalue

Figure 5.8: **Dense flow and confidence maps for `GUOmni`:** (top) Colour coded dense flow map after 12000 processed frames, (bottom) colour coded eigenvector corresponding to the (left) smaller and (right) larger eigenvalue of the accumulator's covariance matrix, respectively. Colour coding according to the Middlebury (Baker et al. 2011) standard. Note: The eigenvectors are rescaled for visualisation purposes. Figure only interpretable when viewed in colour.

Experiments on `Virat`: The presented approach may of course be applied to static cameras as well. Figure 5.9 shows results for sequence `Virat`, taken from the Virat data set (Oh et al. 2011). Here, the camera observes a parking lot. The average motion maps clearly show the typical inbound and outbound traffic paths of the parking lot. Learning of average motion maps for static cameras is of special interest in surveillance scenarios, where typical and abnormal behaviour is to be detected.

dense average flow map



eigenvector for smaller eigenvalue *eigenvector for larger eigenvalue*

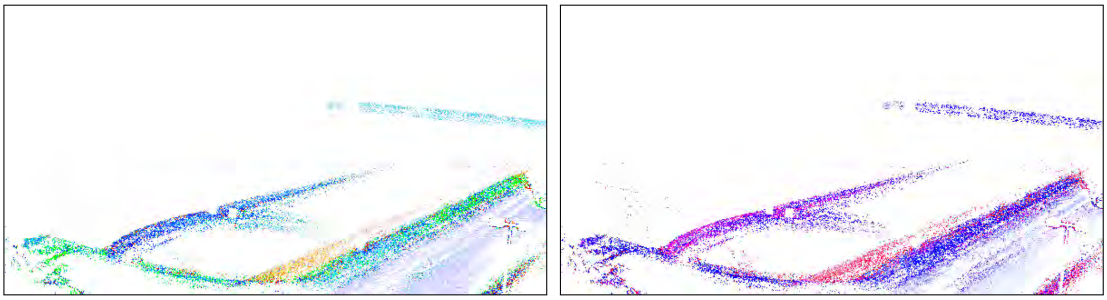


Figure 5.9: **Dense flow and confidence maps for `Virat`:** (top) Colour coded dense flow map after 10,000 processed frames, (bottom) colour coded eigenvector corresponding to the (left) smaller and (right) larger eigenvalue of the accumulator's covariance matrix, respectively. Colour coding according to the Middlebury (Baker et al. 2011) standard. Note: The eigenvectors are rescaled for visualisation purposes. Figure only interpretable when viewed in colour.

5.3.2 Interim Conclusion

To summarise, the first set of experiments shows that average motion maps can be learnt by just looking at the temporal change of single pixels. These maps may subsequently be used as a prior for other algorithms such as estimating instantaneous motion (optical flow between two frames) or to detect abnormal object motion in surveillance setups. Furthermore, the learnt average motion allows to detect non-self-motion, e.g., to identify other moving objects within the scene as they deviate from the learnt average flow. From the results for sequences `GUCar` and `Kitti-Odo` we may conclude that a moving platform mainly drives straight ahead, rotational motion due to turning manoeuvres seems to be averaged out. The question remains how then to learn the average motion observed, e.g., in a turning manoeuvre? Or more generally, how multiple dominant but different motions can be learnt? To this end, next, I will show that the accumulator may encode multiple dominant motion. Then I will present a somewhat more general approach to learn and represent multiple types of motion based on TCA in combination with a latent variable model.

5.3.3 Learning Multiple Motions

Within the results presented so far, at each seed pixel a *single* dominant motion was expected (forward motion). This raises the questions what the method learns when multiple dominant motions exist.

Conceptually, the accumulator is not restricted to encode a single dominant motion only, but may represent multiple motions. As an example, regard Fig. 5.10, where TCAM is applied to the stream of the first camera of sequence `GUBo1616`. For two selected seed pixels, it can be seen that the accumulator captures two dominant motions which are typically observed at a traffic junction. However, to extract these motion vectors, the analysis of the accumulator needs to be adapted. Instead of summarising the accumulator by its first and second order moments, a kind of cluster analysis to separate the peaks in the accumulator is needed.

It should be noted that the accumulator will only represent multiple dominant motions, given that they are observed equally often within the regarded scene. If this is not the case, those motion vectors which are observed less frequently will be averaged out, due to the same reasons as described in Sec. 5.2.1. Therefore, I propose a latent variable model, in which different motions are explicitly learnt and represented as shown next.

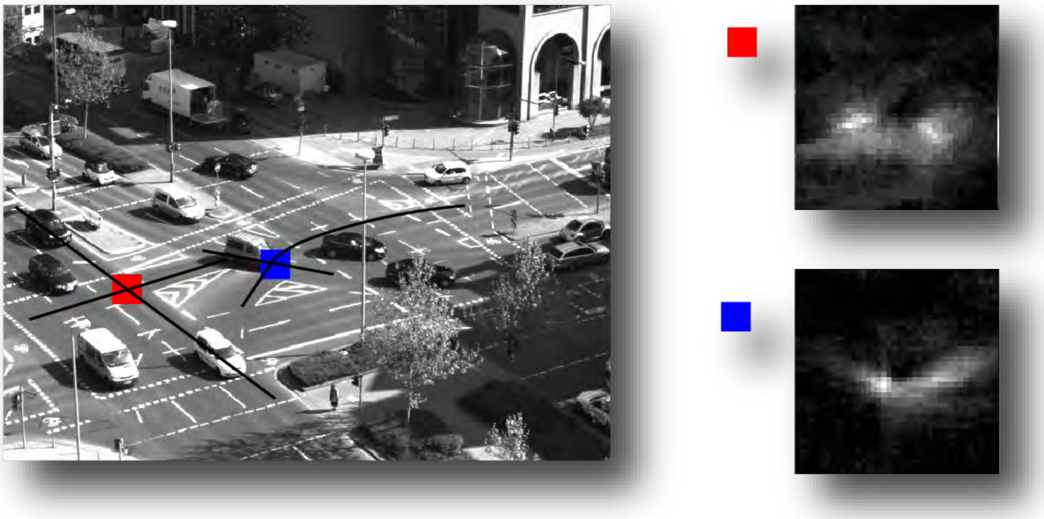


Figure 5.10: **Learning multiple motions:** An accumulator may encode multiple dominant motions, here shown for two selected seed pixels. See text for details.

5.4 Motion as a Hidden Variable

So far, we have seen how average motion maps may be learnt via TCA. For a camera which is mainly moved straight ahead, typical average motion maps such as those shown for sequences `GUCar` and `Kitti-Odo` evolve.

In the following, I will show that TCAM may also learn the characteristic motion fields observed in left and right turning manoeuvres. Furthermore, I will show that the same principle of detecting and matching events allows to infer instantaneous (differential) image motion as well. Specifically, the yaw (pitch, roll) rate (rotation about the downward facing Y (X,Z) axis of the camera coordinate system, Z axis towards the image plane) may be inferred by analysing the correspondence samples, i.e., the sets $\Omega_{\boldsymbol{x}_i}$ available within two time steps.

5.4.1 Optical Flow Structure

It is well known that the instantaneous image motion vector $(\boldsymbol{U}(x,y), \boldsymbol{V}(x,y))^T$ of a general rigid 3-D scene can be modelled as (Irani and Anandan 1998; Longuet-Higgins 1984; Longuet-Higgins and Prazdny 1980):

$$\begin{pmatrix} \boldsymbol{U}(x,y) \\ \boldsymbol{V}(x,y) \end{pmatrix} = \begin{bmatrix} \frac{\tau_Z x - \tau_X}{Z} - \omega_Y + \omega_Z y - \omega_Y x^2 + \omega_X xy \\ \frac{\tau_Y y - \tau_Y}{Z} + \omega_X - \omega_Z x - \omega_Y xy + \omega_X y^2 \end{bmatrix}, \tag{5.11}$$

where $(\boldsymbol{U}(x,y), \boldsymbol{V}(x,y))^T$ denotes the image motion vector at location (x,y), $\tau_{\{X,Y,Z\}}$ denote translational velocities in direction X, Y or Z, and $\omega_{\{X,Y,Z\}}$ denote the angular velocities about the three axes, respectively, with $x = X/Z$ and $y = Y/Z$. From Eq. 5.11 we see that the optical flow at any pixel within the image is a superposition of two independent flow fields: a flow field which only depends on the translational velocities and a flow field which only depends on the angular velocities. Consider the case where the camera pitch and roll rates are zero (or sufficiently small with $\omega_X, \omega_Z \approx 0$), then from Eq. 5.11 we obtain:

$$\begin{pmatrix} \boldsymbol{U}(x,y) \\ \boldsymbol{V}(x,y) \end{pmatrix} \approx \begin{bmatrix} \frac{\tau_Z x - \tau_X}{Z} - \omega_Y - \omega_Y x^2 \\ \frac{\tau_Y y - \tau_Y}{Z} - \omega_Y xy \end{bmatrix}. \tag{5.12}$$

Next, assume that for the translational velocities it holds that $|\tau_{\{X,Y,Z\}}| \ll Z$, then the translational velocities will vanish and we obtain:

$$\begin{pmatrix} \boldsymbol{U}(x,y) \\ \boldsymbol{V}(x,y) \end{pmatrix} \approx \begin{bmatrix} -\omega_Y - \omega_Y x^2 \\ -\omega_Y xy \end{bmatrix} = \begin{bmatrix} -\omega_Y - \omega_Y (\frac{X}{Z})^2 \\ -\omega_Y \frac{X}{Z} \frac{Y}{Z} \end{bmatrix}. \tag{5.13}$$

From Eq. 5.13 we see that the (scaled) camera yaw rate corresponds to the observed image motion of pixels where the pixel's corresponding scene point lies at infinity (or sufficiently far away from the camera with $Z \to \infty$):

$$\begin{pmatrix} \boldsymbol{U}(x,y) \\ \boldsymbol{V}(x,y) \end{pmatrix} \approx \begin{bmatrix} -\omega_Y \\ 0 \end{bmatrix}. \tag{5.14}$$

Similarly, if we assume that the yaw and roll rates are zero, the camera pitch is given as:

$$\begin{pmatrix} \boldsymbol{U}(x,y) \\ \boldsymbol{V}(x,y) \end{pmatrix} \approx \begin{bmatrix} 0 \\ -\omega_X \end{bmatrix}. \tag{5.15}$$

Assuming that the yaw and pitch rates are zero, the camera roll is given as:

$$\begin{pmatrix} \boldsymbol{U}(x,y) \\ \boldsymbol{V}(x,y) \end{pmatrix} \approx \begin{bmatrix} \omega_Z \\ -\omega_Z \end{bmatrix}. \tag{5.16}$$

From these derivations, we see that the yaw rate (pitch/roll) is the same for all pixels, given that the corresponding scene depth is sufficiently large. Based on this finding, we may now formulate an inference scheme for the yaw (pitch/roll) rate based on TCA. For a single seed pixel, recall that we may not learn the instantaneous (between-frame) motion, as the set of matched events is typically far too noisy. Therefore, matched events are accumulated over time and form a strong hypothesis for the *average* flow. This also implies that we may not infer the current yaw (pitch/roll) rate, based on a single seed pixel. However, from Eq. 5.15 we see that for all seed pixels with large associated depth values, the true correspondence and hence the yaw rate is given by $-\omega_Y$. Hence we may accumulate matched events spatially over a set of seed pixels which will result in the average yaw rate observed. Specifically, I build the *mother accumulator* as follows. The mother accumulator is of the same dimensions as the standard accumulator described in Sec. 2.1. At time $t+1$, all entries in the mother accumulator are set to 0. I then add up all correspondence sample sets $\Omega_{\boldsymbol{x}_i}$ and build the *mother accumulator* $\boldsymbol{A}_m$ for time step $t+1$. Given that the yaw rate is the dominant image motion (at the regarded seed pixels), the mother accumulator will show a distinct maximum at the location of the true (but typically scaled) yaw differential. Pixels at infinity are those for which the average flow is small; they can be determined by first learning the average flow map and the local flow dispersion (see experiments for details). Next, I will present results of inferring the yaw rate for sequence `GUCar`.

5.4.2 Learning Hidden Flow Field Variables

As has been explained in Sec. 5.4, if we assume that the yaw rate is the dominant image motion, then the *mother accumulator* $\boldsymbol{A}_m$ for time step $t+1$ will show a distinct peak, marking the location of the true (but typically scaled) yaw differential.

I determine a set of pixels, where their corresponding 3-D point will lie approximately at infinity as follows: I subdivide the image in 4 sectors by two diagonal lines, which pass through the focus of expansion. The upper sector can well be used for estimation of the horizontal shift and hence the yaw rate, since the rotational part does not contribute significantly in vertical direction. Figure 5.11 visualises the selected pixels for some sample images of sequence `GUCar`, where the car turns left, goes straight ahead and turns right. Figure 5.11 also shows the associated mother accumulators. It can be seen that left and right turns show up as a distinct peak within the accumulator in the opposite turn direction as expected. To extract the actual yaw rate, I pool the accumulator (=sum up) vertically to obtain a more robust estimate. When vertical motion is not the dominant motion, or if only a few number of selected pixels show an event, the mother accumulator will be scattered, as no strong hypothesis may be formed by the set of matched events. However, this can easily be detected, e.g. by an eigenvalue analysis of the accumulator's covariance matrix. If the accumulator is scattered, its associated eigenvalues should be large.

I found that left and right turns can be detected with high confidence (by means of the accumulator statistics as described), given that the analysis is carried out on a sufficiently large number of active pixels, which approximately lie at infinity. Forward and backward motion of the platform cannot directly be read off the mother accumulator, as it will be scattered for both types of motion.

While the presented results show that TCA can in principle be used to learn instantaneous motion, it remains future work to analyse this approach in more detail.

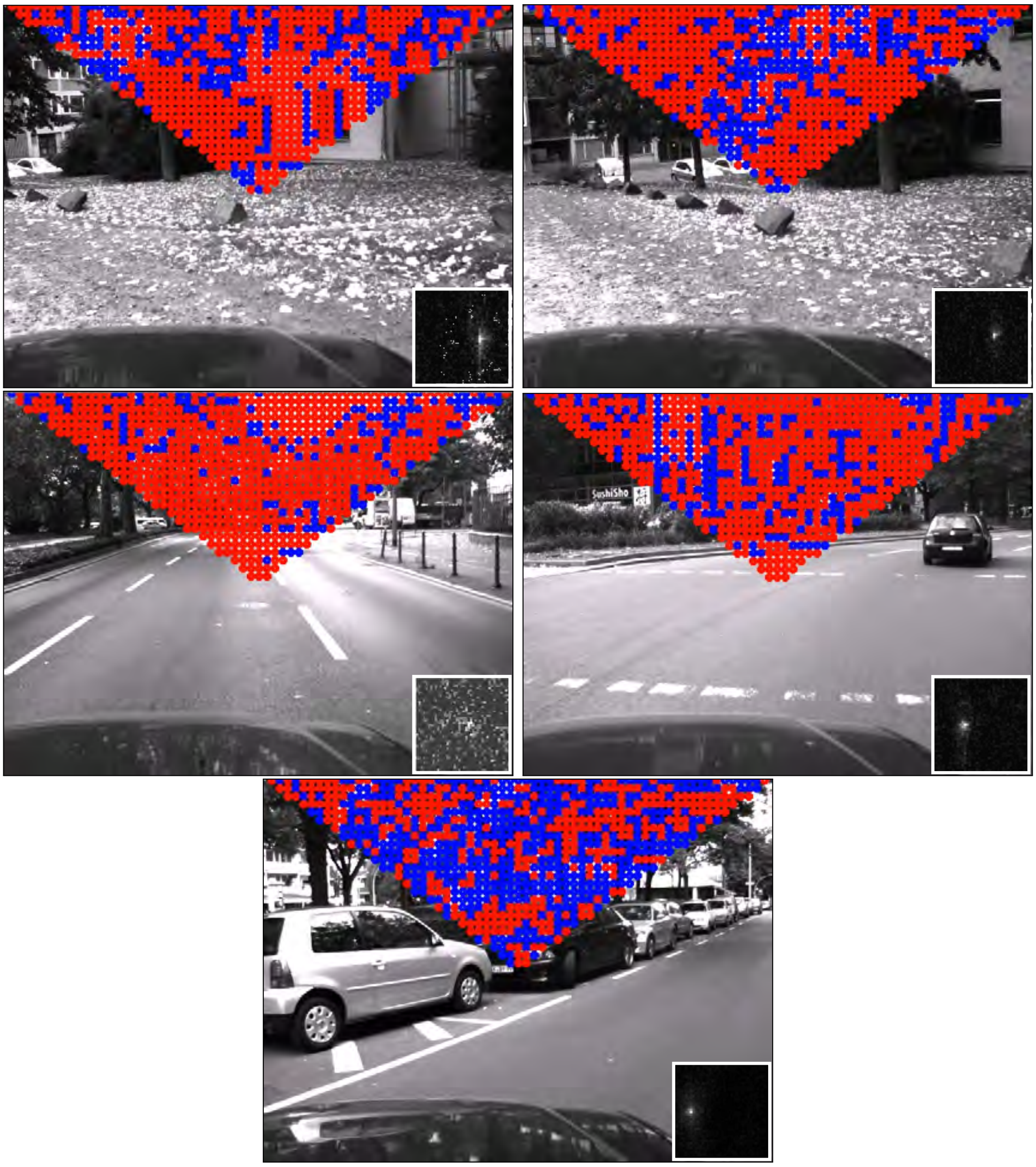


Figure 5.11: **Inferring current yaw rate via TCA:** (top left and right) Car turns left, (middle left) goes straight, (middle right and bottom) and turns right. From pixels within the upper triangle, the yaw rate between time steps t and $t+1$ is inferred. At each time step only the pixels showing an event are taken into consideration, here marked blue (sic!). At each time step a mother accumulator is build (as shown in the bottom right of each image) by summing up all $\Omega_{\boldsymbol{x}_i}$. The mother accumulator then encodes the current yaw rate as can be deduced from the peaks in the accumulator array. See text for details.

5.4.3 Motion as a Latent Variable

The optical flow observed by a camera mounted on a car depends on the car's self-motion. Assuming the car drives through a static scene we may differentiate between three general types of motion: forward motion, left turn and right turn. As has been shown previously, for sequences where forward motion (of the camera platform) dominates, characteristic flow fields evolve which can be learnt via TCA. All other types of motion are essentially averaged out. In the following, I present a latent variable model based on which the characteristic motion fields observed in left and right turning manoeuvres can be learnt as well.

The basic idea is to train separate instances of TCA, one for each motion type (forward, left and right motion). The actual platform motion represents the state of the ternary hidden variable $\mathcal{M} = \{f, l, r\}$. In order to train the TCA instances, we need to infer the state of $\mathcal{M}$ which indicates which of the TCA instances is to be updated.

I infer the state of $\mathcal{M}$ between consecutive frames via the method of *phase correlation.* To this end, I crop a patch of dimension $w \times h = 2^8 \times 2^7$ pixels approximately centred at the focus of expansion at time $t-1$ and t , respectively. The translational offset $(\triangle x, \triangle y)$ (yaw, pitch) between the patches is then estimated in the frequency domain exploiting the Fourier shift theorem. See (Kuglin and Hines 1975b) and (Eisenbach, Mertz, et al. 2013) for details. We may infer $\mathcal{M}$ also based on the mother accumulator. However, this remains future work.

Figure 5.12 visualises the relative pose change as estimated via phase correlation for the case of forward, left and right motion. The blue crosshair marks the image centre, while the red crosshair is moved according to the estimated motion. The TCA instance to be updated is then selected based on thresholding the estimated offset in x direction ($\triangle x$ = yaw rate):

$$\begin{cases} \triangle x < -5 & \text{update right turn model,} \\ |\triangle x| \leq 5 & \text{update forward motion model,} \\ \triangle x \quad > 5 & \text{update left turn model.} \end{cases} \tag{5.17}$$

Next, I apply TCA with parameter setting: $T_e = 30$, $T_m = 10$, accumulator size 40×40 pixels for the forward motion model and accumulator size 90×90 pixels for left and right turns. The spatial extend of the accumulators for the left and right turn need to be larger than those for the forward motion, as the observed optical flow is considerably larger. This is due to building walls which are very close to the camera during turning manoeuvres.

Figure 5.14 visualises the learnt sparse flow for a right turn. From the orientation of the learnt flow vectors, it can be seen that all of them capture a displacement to the left as expected. Figure 5.14 also shows the interpolated flow which clearly represents the characteristic flow field for a right turn. Figure 5.13 visualises the results obtained for a left turn. From these, we may draw similar conclusions as for the right turn.

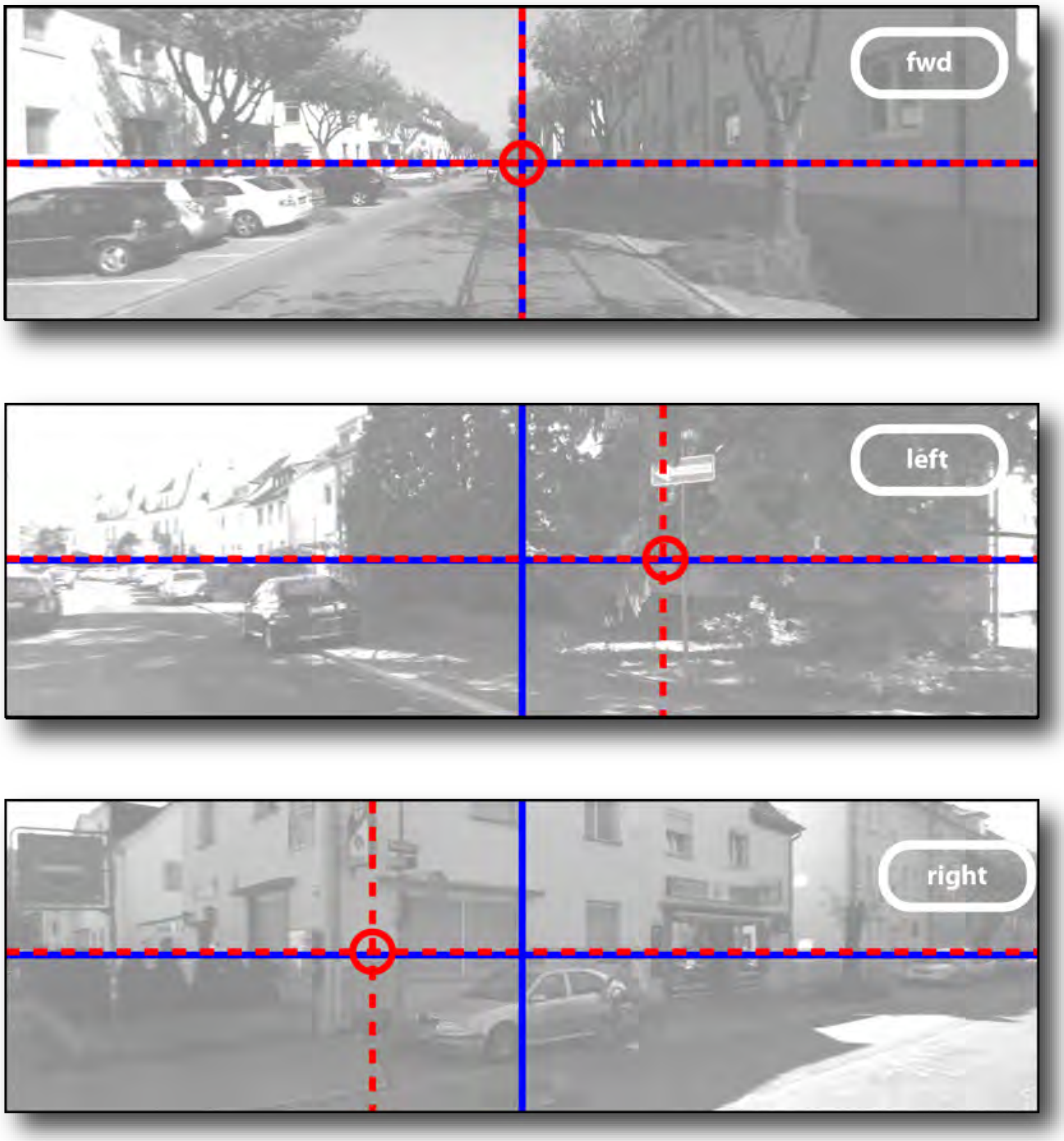


Figure 5.12: **Motion as a latent variable :** The optical flow observed by a camera mounted on a car depends on the car's self motion. Assuming the car drives through a static scene we may differentiate between three general types of motion: (top) forward motion, (middle) left turn, (bottom) right turn. We may then learn the associated average optical flow by training three independent TCA instances. The solid blue lines mark the image centre while the dashed red lines mark the lateral motion (=yaw rate) between consecutive time steps. The lateral motion is here estimated via the method of phase correlation and is used to select the TCA instance. See text for details.

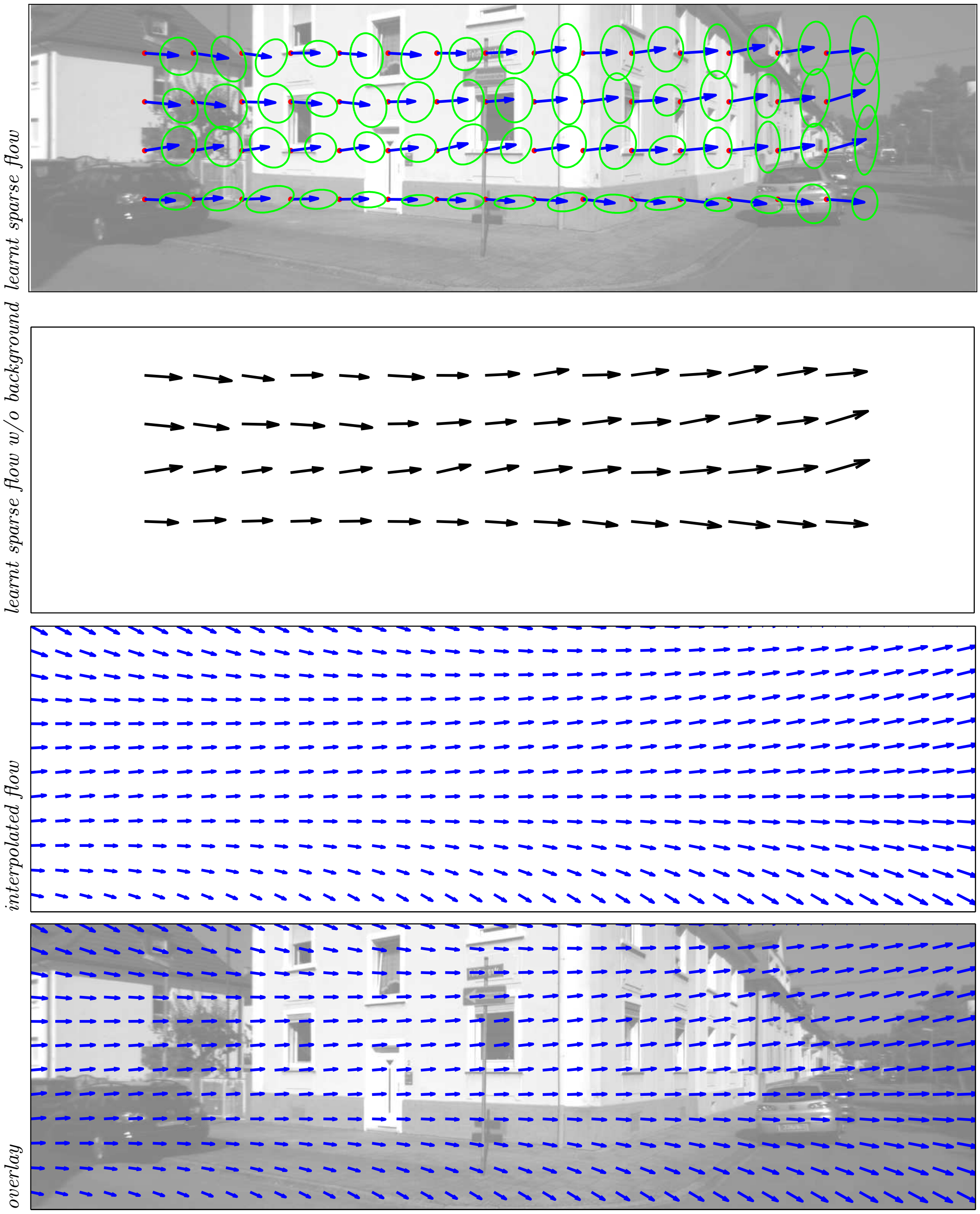


Figure 5.13: **Learnt and interpolated sparse flow for `Kitti-Odo` (left turn):** The TCA model is only updated, when the motion variable $\mathcal{M}$ indicates a left turn. Best viewed in colour. See text for details.

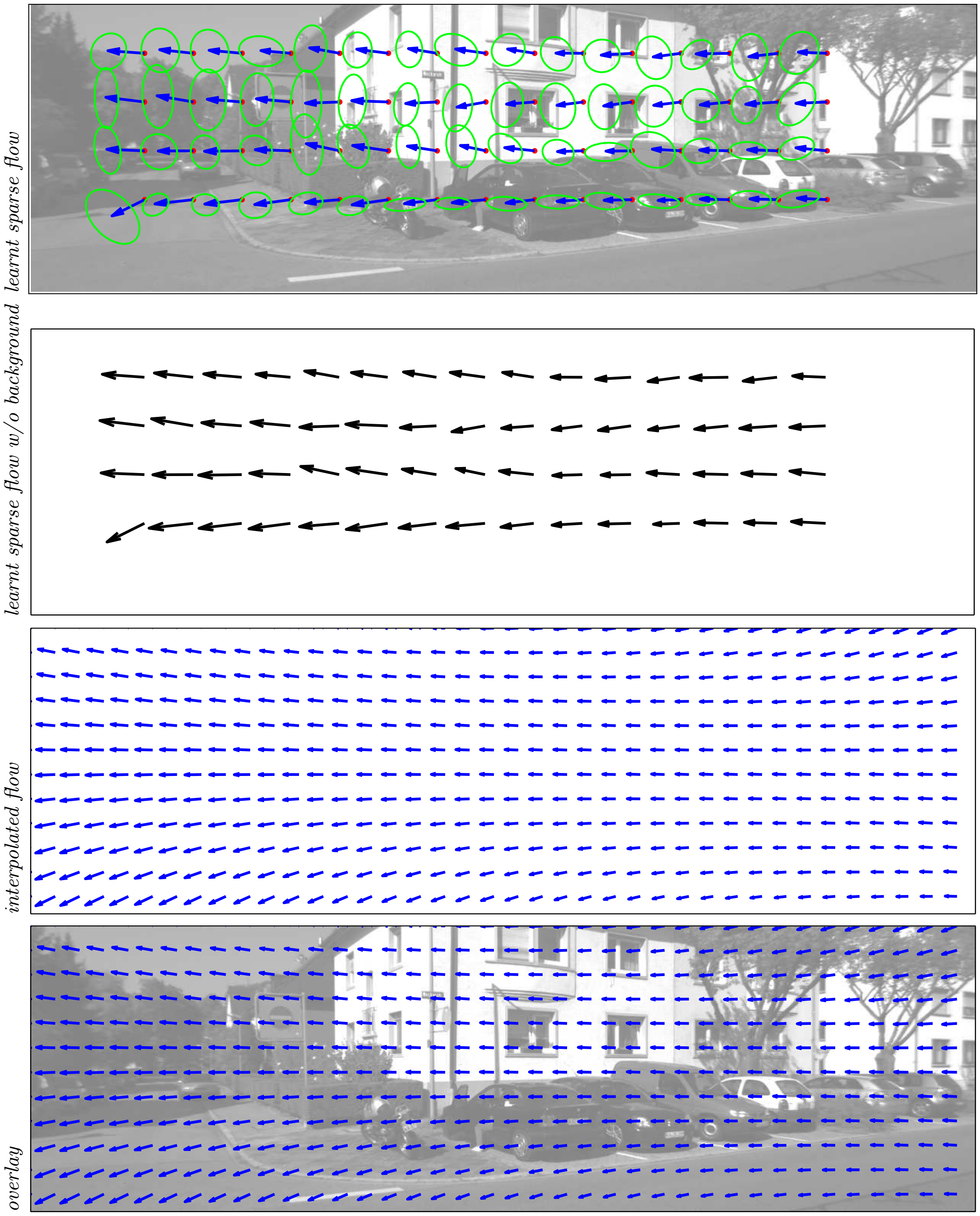


Figure 5.14: **Learnt and interpolated sparse flow for `Kitti-Odo` (right turn):** The TCA model is updated, when the motion variable $\mathcal{M}$ indicates a right turn. Best viewed in colour. See text for details.

5.5 Summary and Conclusion

I presented the applicability of TCA to motion analysis. Experimental results show that the method is able to successfully learn the distribution of motion correspondences unsupervised without explicitly computing the optical flow.

I showed that a sparse set of learnt average motion vectors can be interpolated by means of fitting the parameters of a bi-quadratic model to the set of learnt average motion vectors. The scheme takes the uncertainties encoded in the learnt correspondence distribution explicitly into account.

I demonstrated that TCA can also be used to perform inference on global hidden motion variables, e.g., the yaw rate, by pooling the set of matched events over many pixels lying approximately at infinity.

The approach can be applied to static and moving cameras and can handle different camera models in a principled manner.

Based on a mixture model of TCA experts, I demonstrated that the three dominant types of motion observed by a moving camera in a natural environment, i.e., left, right and forward motion can be learnt autonomously. This leads to three characteristic average motion fields.

Part II

Learning Global Transformations

6 Learning Global Mappings

In the following, I develop and analyse a method to learn global transformations between pairs of views in an unsupervised manner. This is in contrast to TCA described previously, where correspondences were learnt on a per pixel basis.

My approach is based on a linear model known as *Canonical Correlation Analysis* (CCA) (Hotelling 1936). Given a large set of training image pairs, which are related by an unknown (=latent) spatial transformation, I learn this transformation in a feature free (=no keypoint detection or matching) manner. The learnt transformation is represented implicitly in terms of pairs of learnt basis vectors and does neither use nor require an analytic/parametric expression for the latent mapping.

It is important to note that this work is not about estimating the parameters of a known class of transformations but about learning *the* transformation itself.

6.1 Introduction

Let us now regard the problem of learning *global* spatial transformations between pairs of images, which I denote as *mappings* in the following.

In its most general form, for two images I_i and I_j the mapping function describes the pixel to pixel mapping $\boldsymbol{y_j} = \boldsymbol{f}(\boldsymbol{x}_i)$, where $\boldsymbol{x}_i$ and $\boldsymbol{y}_j$ are the spatial 2-D coordinates of corresponding pixels in images I_i and I_j, respectively. Inferring the mapping function $\boldsymbol{f}$ is the fundamental problem in many vision tasks, e.g., in stereo vision or motion estimation (Stiller and Konrad 1999). In Sec. 2, we have seen that we may sample the mapping function $\boldsymbol{f}$ by the detection and matching of spatial keypoints. This leads to sparse or dense sample sets, e.g., by means of a disparity map or an optical flow map (cf. Sec. 2.2). However, due to the potential complexity of natural scenes, inferring a fully functional description of the mapping function is usually infeasible. Instead, the true mapping is approximated by fitting a specific parametric model. Fitting of these models may be done in a number of different ways. A common approach to determine a geometric mapping for a pair of images is based on the spatial feature matching pipeline (cf. Sec. 2) by fitting the parametric model to a set of pixel correspondences.

In computer vision, a prominent parametric model is the *projective linear group*, which defines 2-D planar transformations (Hartley and Zisserman 2004). Let $\boldsymbol{x}_i$ be the 2-D pixel coordinate of a pixel in camera $\mathcal{C}_i$ and let $\tilde{\boldsymbol{x}}_i = \begin{pmatrix} x_1 & x_2 & w \end{pmatrix}_i^T$ be its homogeneous representation. In the following we assume $w = 1$. The *projective* transformation

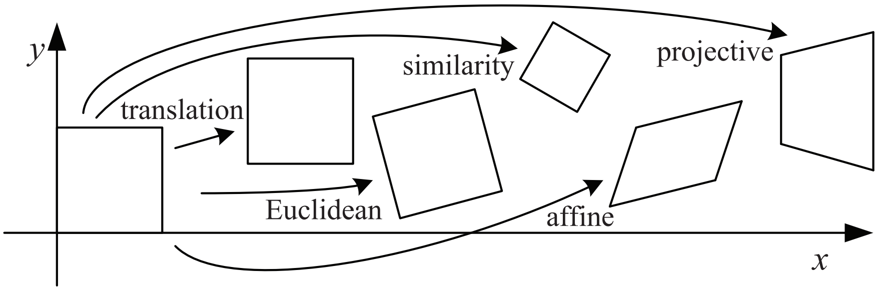


Figure 6.1: **2-D spatial transformations:** Overview of 2-D spatial transformations which can be modelled via $\boldsymbol{H}$. See text for details. Image source: (Szeliski 2010, p. 36).

linearly maps coordinates $\tilde{\boldsymbol{x}}_i$ to coordinates $\tilde{\boldsymbol{y}}_j$ in $\mathcal{C}_j$ such that $\tilde{\boldsymbol{x}}_i \leftrightarrow \tilde{\boldsymbol{y}}_j$ holds according to (ibid., p. 33):

$$\tilde{\boldsymbol{y}}_j = \begin{pmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{pmatrix} \tilde{\boldsymbol{x}}_i = \boldsymbol{H}\tilde{\boldsymbol{x}}_i. \tag{6.1}$$

Here, $\boldsymbol{H}$ is an arbitrary non-singular matrix, also denoted as *homography.* By constraining the structure of $\boldsymbol{H}$, i.e., reducing its number of free parameters, we obtain *affine*, *similarity*, *euclidean*, and *translation* transformations (cf. Fig. 6.1). Inferring an unknown projective transformation then amounts to estimate the parameters $\boldsymbol{\theta} = \{h_{11}, .., h_{33}\}$. While the homography has 9 parameters, in its most general form it only has 8 degrees of freedom due to the fact that a projective transformation is only uniquely defined up to a scale factor (ibid.).

Now assume that for a set of P correspondences $\{\tilde{\boldsymbol{x}}_i \leftrightarrow \tilde{\boldsymbol{y}}_j\}_P$ fulfilling Eq. 6.1 the transformation $\boldsymbol{H}$ is sought. For each pixel correspondence (in homogeneous coordinates) we have (ibid., p. 88):

$$\tilde{\boldsymbol{y}}_j \times \boldsymbol{H}\tilde{\boldsymbol{x}}_i = \boldsymbol{0}. \tag{6.2}$$

Carrying out the cross product in Eq. 6.2 (assuming $w = 1$) and rearranging, we may rewrite Eq. 6.2 for the p'th correspondence as (ibid., p. 89):

$$\begin{bmatrix} \boldsymbol{0}_3^T & -\tilde{\boldsymbol{x}}_i^T & (y_2)_j\tilde{\boldsymbol{x}}_i^T \\ \tilde{\boldsymbol{x}}_i^T & \boldsymbol{0}_3^T & -(y_1)_j\tilde{\boldsymbol{x}}_i^T \\ -(y_2)_j\tilde{\boldsymbol{x}}_i^T & (y_1)_j\tilde{\boldsymbol{x}}_i^T & \boldsymbol{0}_3^T \end{bmatrix} \begin{pmatrix} \boldsymbol{h}^1 \\ \boldsymbol{h}^2 \\ \boldsymbol{h}^3 \end{pmatrix} = \boldsymbol{0} = \boldsymbol{A}_p\boldsymbol{h}, \tag{6.3}$$

where $\boldsymbol{h}^{\{1,2,3\}}$ denote the rows of $\boldsymbol{H}$ (as a column vector), which when concatenated form column vector $\boldsymbol{h}$. In Eq. 6.3 only two rows of $\boldsymbol{A}_p$ are linearly independent. If we stack $\boldsymbol{A}_p$ for all correspondences we arrive at the homogeneous equation system:

$$\begin{bmatrix} \boldsymbol{A}_1 \\ \boldsymbol{A}_2 \\ \vdots \\ \boldsymbol{A}_P \end{bmatrix} \boldsymbol{h} = \boldsymbol{A}\boldsymbol{h} = \boldsymbol{0}. \tag{6.4}$$

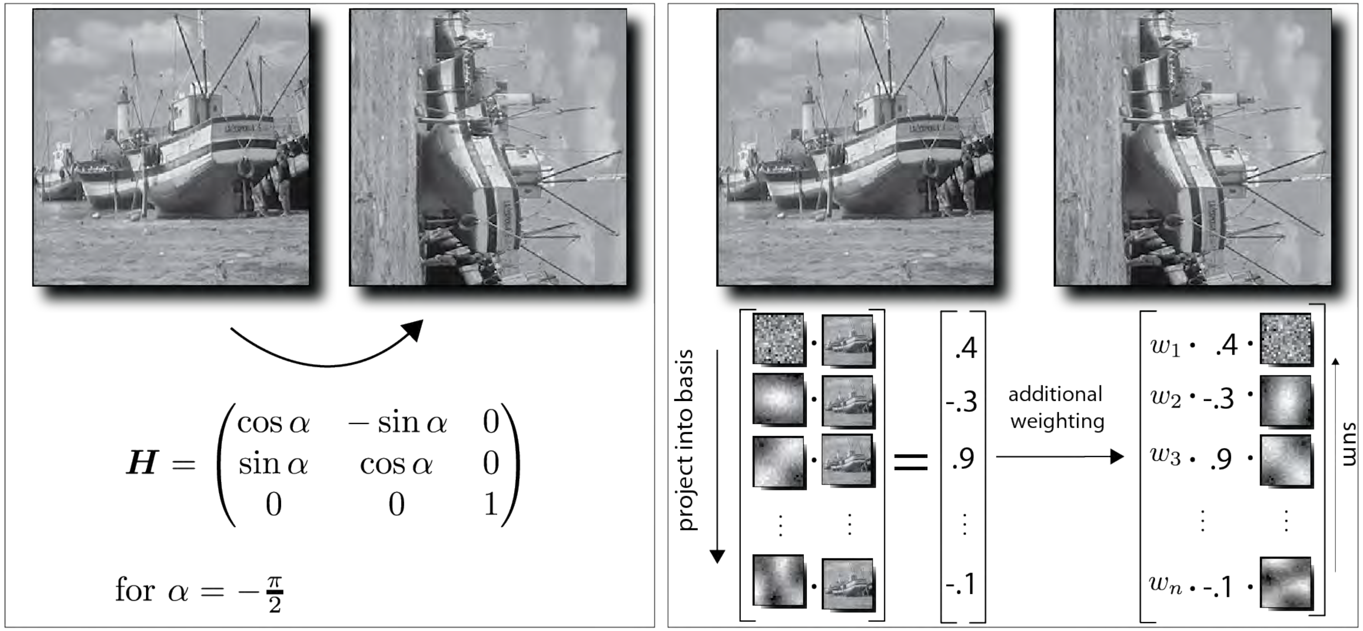
Similar to the estimation of the fundamental matrix (cf. Sec. 2.2), Eq. 6.4 may be solved (in a least squares sense) via the singular value decomposition of $\boldsymbol{A}$. This method is also known as the *direct linear transform* algorithm in the literature (ibid.).

Let us now return to the problem of inferring an unknown mapping between two views $\mathcal{C}_i$ and $\mathcal{C}_j$. If a parametrised functional representation in the form of $\boldsymbol{H}$ is sought, we may identify a set of pixel correspondences among the views and then solve for the parameters of the functional form. Obviously, prior to infer the parameters, we need to select a specific functional form (affine, similarity...) or have to infer it as well, via a model selection approach.

In contrast, I consider learning of global spatial transformations between pairs of views in a *feature free*, *non-parametric* and *unsupervised* manner. In my approach, I never compute spatial features and I make no assumption about the functional form of the underlying mapping. Furthermore, I do not rely on a topological relation between the pixel signals which is implicitly done when estimating $\boldsymbol{H}$. It is important to note that I consider learning of arbitrary but *fixed* transformations, i.e., the transformation is assumed to be the same over a large set of training images. This is not the case for the parametric approach described earlier, where a single image pair suffices to estimate the *parameters* of the mapping function. However, also note that this work is not about estimating the parameters of a known class of transformations but about learning *the* transformation itself.

6.1.1 Mapping as a Linear Projection

The fundamental difference to the common functional/geometrical representation of a spatial mapping function is that I represent the mapping by means of a linear projection. From now on, instead of corresponding pixels, I will regard corresponding images or patches. I will now let $\boldsymbol{x}_i, \boldsymbol{y}_j \in \mathbb{R}^{R \cdot C}$, with $\boldsymbol{x}_i \leftrightarrow \boldsymbol{y}_j$ being a corresponding pair of *vectorised* images (=2-D array rearranged to a column vector), and *not* single pixels as in the first part of the thesis. In order to keep the notation uncluttered, I usually drop indices i and j and $\boldsymbol{x}$ will always denote an image of view $\mathcal{C}_i$ and $\boldsymbol{y}$ will always denote



$$\boldsymbol{H} = \begin{pmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

for $\alpha = -\frac{\pi}{2}$

Figure 6.2: **Representation of transformations:** (left) Parametric vs. (right) tensor based representations of transformations. In a parametric approach we estimate the parameters of the sought transformation, here a rotation by -90 degrees. The transformation is encoded by means of a geometric coordinate transform $\boldsymbol{H}$. In the tensor based approach, a transformation is encoded by means of basis vectors and is applied by means of a basis change. See text for details.

an image of view $\mathcal{C}_j$. Let $\boldsymbol{F} \in \mathbb{R}^{R \cdot C \times N}$ be a matrix of N *basis* vectors. I then define the mapping of $\boldsymbol{x}$ to $\boldsymbol{y}$ (and vice versa) according to:

$$\boldsymbol{y} = \boldsymbol{F}_{\boldsymbol{yx}}^{T} \boldsymbol{x}, \tag{6.5}$$

$$\boldsymbol{x} = \boldsymbol{F}_{\boldsymbol{xy}}^{T} \boldsymbol{y}. \tag{6.6}$$

While applying a mapping via Eq. 6.6 is a *linear* operation, it is important to note that the actual spatial transformation *encoded* by $\boldsymbol{F}_{\boldsymbol{yx}}$ does *not* need to be linear. Figure 6.2 visualises the general difference between the geometric and the tensor based approach to apply a spatial transformation.

The approach developed in the following estimates $\boldsymbol{F}_{\boldsymbol{yx}}$ and $\boldsymbol{F}_{\boldsymbol{xy}}$, based on the method of *Canonical Correlation Analysis* (CCA), which was introduced by Hotelling in 1936 (Hotelling 1936) (cf. Fig. 6.3). I found that CCA is capable of learning general, even non-parametric, transformations of visual data. Specifically, given a large set of pairs of training images, the mappings $\boldsymbol{F}_{\boldsymbol{yx}}$ and $\boldsymbol{F}_{\boldsymbol{xy}}$ are learnt and encoded by means of two sets of basis vectors, which are obtained via CCA. Recall that I consider learning of arbitrary but *fixed* transformations, i.e., the transformation is assumed to be the same over the whole training set. While CCA may be applied to a dataset containing a whole class of transformations, the learnt basis vectors will then constitute an invariant representation of the data under the transformations observed. This has been shown in (Bethge et al. 2007), where a rotation invariant representation for 2-D images is learnt with a variant of CCA. As expected, CCA then learns eigenfunctions of the rotation operator. My approach differs from (ibid.) as my goal is to learn and apply an unknown transformation. When learning an invariant basis for a *class* of transforms, the knowledge about the particular transformation at hand is lost in the CCA model. This is due to the fact that CCA does not have additional hidden variables to 'switch' between the different transforms, hence we may not predict the outcome of this specific transformation to new data. Restricting the training data to contain a single transformation, as I do here, could be considered a limitation in practice; however, it reveals a fundamental mechanism that applies when the transform parameters are kept fixed. To represent a parameterised *set* of transformations, a suitable mixture model is needed which allows to switch between different CCA 'experts'. In Sec. 6.7 , I will give a preliminary example for such a mixture model. However, general learning of multiple transformations via CCA is considered future work.

As will be shown in Sec. 6.5, CCA extracts features (basis vectors) that capture the hidden transformation and generates a linear mapping tensor that allows to apply a learnt transformation to previously unseen data. This task is sometimes denoted as 'analogy making' in the literature (Memisevic and Hinton 2010). From the CCA representation, the rank of the shared signal, i.e., the part of the signal visible within both views may be determined and an analysis of the energy distribution among the basis vectors allows to identify the spatial footprint of the shared signal.

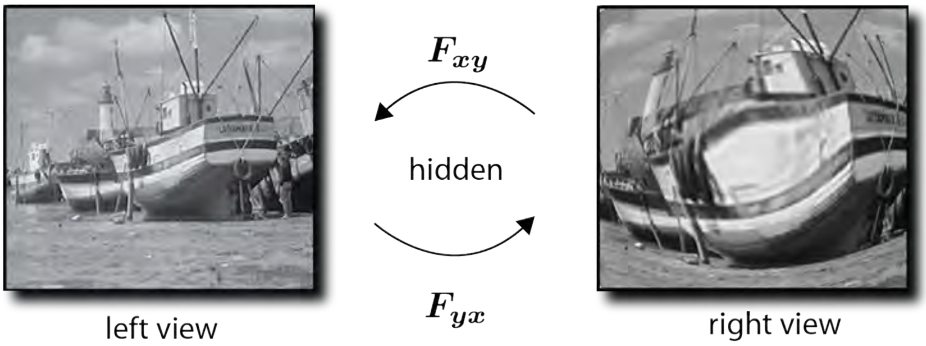


Figure 6.3: **Learning hidden transformations:** I am interested in learning spatial transformations between pairs of views. Transformations considered here are general nonlinear, non-parametric transformations. Learning is based on *Canonical Correlation Analysis* applied to a large training set of image pairs, which were subject to an arbitrary but *fixed* transformation. See text for details.

In Sec. 6.6.2.1 I will show that the basic learning scheme may also be used to generate correspondence priors for binocular camera setups. Specifically, I determine a global mapping on a coarse scale of the given video streams of cameras $\mathcal{C}_i$ and $\mathcal{C}_j$. I then determine pixel correspondences on a coarse scale, via learning the inter-image transformation, employing CCA. Subsequently, I project those correspondences to the original resolution and obtain a correspondence prior: for each spatial location in the spatial domain of $\mathcal{C}_i$, regions of high probability containing the true correspondence within $\mathcal{C}_j$ are determined. Such correspondence priors may then be plugged into probabilistic and energy based formulations of specific vision applications.

6.2 Related Work

Canonical Correlation Analysis is a well-established statistical method to model/infer the *relationship* between paired data (rather than modelling the *content* of a single source, like, e.g., PCA does). CCA has been used in quite different disciplines, among them economics, statistical signal processing and climatology (Barnett and Preisendorfer 1987; Scharf and Mullis 2000).

Applications in computer vision range from depth estimation (Borga and Knutsson 1999) to image and action classification (T.-K. Kim et al. 2007; T. Kim et al. 2007). Although there is interesting vision-related work based on CCA, CCA as a tool seems to be familiar to a part of the vision community only. In (Borga and Knutsson 1999), Borga and Knutsson combine the phase-based approach to disparity estimation with CCA to estimate depth maps in semi-transparent stereo images. Specifically, CCA is used to generate adaptive linear combinations of quadrature filters in order to be able to estimate multiple disparities at a given image location, which would not be possible when using a standard phase-based approach alone. Furthermore, Borga (Borga 1998) presented the relationships of CCA and PCA (including other linear models) showing that their solutions are given by solving a generalised eigenvalue problem. In (Johansson et al. 2001), a corner orientation detector based on CCA is developed, which is invariant to the actual corner angle and intensity. In (T.-K. Kim et al. 2007; T. Kim et al. 2007), CCA is applied in high level applications like image and action classification. Specifically, a similarity measure based on CCA is derived and used within a discriminative learning framework.

In (Loy et al. 2009), a framework based on CCA is developed, which addresses tasks like activity modelling or finding the spatiotemporal topology within multi-camera networks. While I share a similar application domain, the fundamental difference to my approach is that they try to identify corresponding regions among *non-overlapping* cameras that show similar activity.

In (Donner et al. 2006), an active appearance model search algorithm based on CCA is proposed, where object characteristics are learnt and subsequently used for search. In (Kobayashi 2014), a regularised variant of CCA is developed and is used in a partial pattern matching application. Specifically, the optimisation problem addressed by CCA is constrained by i) a smoothness and ii) a sparseness constraint. Results show that the constraints help in identifying shared partial patterns within two views. However, the smoothness constraint introduces a topological assumption on the input data which does not need to be made when using classic CCA (hence, general permutations can be learnt).

In the machine learning community, CCA is used in labelling and dimensionality reduction tasks, to name a few (Dhillon et al. 2011; Lampert and Krömer 2010; Rai and Daumé 2009). In (Bach and Jordan 2005), a probabilistic interpretation of CCA (PCCA) is presented. However, note that also the classic formulation of CCA is based on a statistical signal model. This has been extensively studied by Scharf and colleagues in their work on CCA in the signal detection and estimation community (A. Pezeshki et al. 2006; Scharf and Mullis 2000; Scharf and John Thomas 1998). However, their work seems to be less known in the computer vision and machine learning community.

Variants and extensions of classic CCA include *kernel CCA* (Fyfe and Pei Ling Lai 2000; Hardoon et al. 2004; Pei Ling. Lai and Fyfe 2000; C. Wang 2007) and recently *deep CCA* (Andrew et al. 2013). In (Klami et al. 2013), a Bayesian interpretation of CCA is given and shown to be equivalent to a model known as *inter-battery factor analysis.*

In this work, I will stick to the classic statistical formulation of CCA and present an analysis of how the shared signal among pairs of views is represented by CCA and how a learnt transformation can be applied in a statistically optimal way by means of an associated MMSE estimator, in analogy to (Scharf and Mullis 2000; Scharf and John Thomas 1998).

My work can be considered as a relational feature learning approach, where the goal is to extract features which model the *relationship* of the paired data, rather than modelling their *content*. This type of feature learning has found attention both in the area of computational neuroscience and in the machine learning/machine vision communities (see (Memisevic 2013) for a recent overview). Recent models include the *Gated Boltzmann Machine* (Memisevic and Hinton 2007) and the *Gated Autoencoder* (Memisevic 2008), which mainly differ by means of the score function which is optimised. In these types of models, relational features are represented by means of filters/basis vectors obtained, e.g., by maximum likelihood training via contrastive divergence on a rather large set of training images. Once trained, the transformation of a given image pair is encoded by means of the activation of (binary) hidden variables, which select a sparse subset of the learnt filters to represent the observed transformation. It is important to note that these models are able to learn and represent a manifold of transformations at the same time. For example, by training a classifier on the activation pattern of the hidden variables, it is possible to distinguish between, say, different rotations, as the activation pattern of the hidden variables will differ. Compared to this, the CCA approach is less powerful when it comes to learn multiple transformations at the same time. However, the processing structure is much simpler and merely amounts to perform a singular value decomposition, compared to rather sophisticated iterative optimisation schemes of the previously mentioned models.

To summarise, the tasks addressed in the following are learning a representation of a fixed unknown geometric mapping between images $\boldsymbol{x}$ and $\boldsymbol{y}$ from a rather large training set, which allows to predict $\boldsymbol{x}$ from $\boldsymbol{y}$ and vice versa. The mapping may be arbitrary complicated, even non-parametric, and not necessarily one-to-one. Furthermore, I identify the area which is 'seen' by both images, which may be inferred based on the rank of the shared signal.

6.3 Learning Transformations with CCA

6.3.1 Observation Model

I regard pairs of vectorised images $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^{R \cdot C}$, which are arbitrarily transformed observations of a vectorised source signal $\boldsymbol{s}$, corrupted by additive zero mean Gaussian noise, represented by vectors $\boldsymbol{v_x}$ and $\boldsymbol{v_y}$. The observation model is then:

$$\boldsymbol{x} = \phi(\boldsymbol{s}) + \boldsymbol{v_x}, \tag{6.7}$$

$$\boldsymbol{y} = \theta(\boldsymbol{s}) + \boldsymbol{v_y}. \tag{6.8}$$

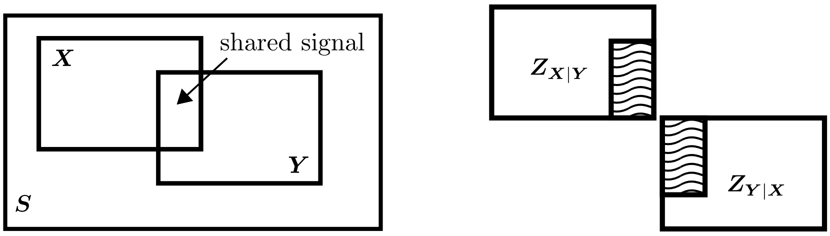


Figure 6.4: **Observation model:** (left) An image pair $\boldsymbol{x}, \boldsymbol{y}$ is given as two partially overlapping observations of a source signal $\boldsymbol{s}$. In the present illustration, the geometrical pixel-to-pixel relation is depicted as a simple spatial shift, but the model is valid for *any* geometrical transformation. (right) Definition of the 'shared signal' (textured), which represents the part of the source signal which is visible within both observations.

Note that ϕ and θ are *not* the transformations I want to learn. Instead, I am interested in learning the transformations $\boldsymbol{F_{xy}}, \boldsymbol{F_{yx}} \in \mathbb{R}^{R \cdot C \times N}$ which map $\boldsymbol{x}$ to $\boldsymbol{y}$ and vice versa, according to:

$$\hat{\boldsymbol{x}} = \begin{bmatrix} | & | & | & | \\ \boldsymbol{f}^1_{\boldsymbol{xy}} & \boldsymbol{f}^2_{\boldsymbol{xy}} & \cdots & \boldsymbol{f}^N_{\boldsymbol{xy}} \\ | & | & | & | \end{bmatrix}^T \boldsymbol{y} = \boldsymbol{F}^T_{\boldsymbol{xy}} \boldsymbol{y}, \tag{6.9}$$

$$\hat{\boldsymbol{y}} = \begin{bmatrix} | & | & | & | \\ \boldsymbol{f}^1_{\boldsymbol{yx}} & \boldsymbol{f}^2_{\boldsymbol{yx}} & \cdots & \boldsymbol{f}^N_{\boldsymbol{yx}} \\ | & | & | & | \end{bmatrix}^T \boldsymbol{x} = \boldsymbol{F}^T_{\boldsymbol{yx}} \boldsymbol{x}. \tag{6.10}$$

Recall that the mapping functions $\boldsymbol{F_{yx}}$ and $\boldsymbol{F_{xy}}$ are to be understood as a set of basis vectors which are *linear* functions of the pixel's signal *values*. Therefore, the mapping functions $\boldsymbol{F_{yx}}$ and $\boldsymbol{F_{xy}}$ are *not* identical to the parametric geometrical transformation that maps coordinates (locations) in one image to coordinates in the other image. Utilising this formulation for transformations allows to represent simple transformations like shifts and rotations, but can also deal with arbitrarily complex nonlinear mappings, including a complete random permutation of the pixel locations.

I define the *shared signals* $\boldsymbol{z_{y|x}}$ and $\boldsymbol{z_{x|y}}$, which are the parts of the source signal visible within both images, given by:

$$\boldsymbol{z_{x|y}} = \boldsymbol{x} \cap \boldsymbol{F}^T_{\boldsymbol{xy}} \boldsymbol{y}, \tag{6.11}$$

$$\boldsymbol{z_{y|x}} = \boldsymbol{y} \cap \boldsymbol{F}^T_{\boldsymbol{yx}} \boldsymbol{x}, \tag{6.12}$$

where operator $\cap$ extracts the common part of two given signals. Figure 6.4 illustrates the observation model and the shared signal for a translation transformation in both spatial directions. Observe that for a translation, the shared signal cannot be of full rank, i.e., there are parts within $\boldsymbol{x}$ which are not visible within $\boldsymbol{y}$ and vice versa. In general, the shared signal will only be of full rank, given that the transformation is a full *permutation*, which is an invertible transformation by definition.

The idea to learn the transformations $\boldsymbol{F_{yx}}$ and $\boldsymbol{F_{xy}}$ is now the following: The true (hidden) transformations $\boldsymbol{F}_{\boldsymbol{xy}}^T\boldsymbol{y}$ and $\boldsymbol{F}_{\boldsymbol{yx}}^T\boldsymbol{x}$ should maximise the correlation of the shared signals $\mathrm{Corr}(\boldsymbol{z}_{\boldsymbol{y}|\boldsymbol{x}}, \boldsymbol{z}_{\boldsymbol{x}|\boldsymbol{y}})$ among all possible transformations. In the following, I show that the latent mappings $\boldsymbol{F_{yx}}$ and $\boldsymbol{F_{xy}}$ may be obtained as an MMSE estimator, which is defined based on two sets of basis vectors obtained by CCA.

6.3.2 Canonical Correlation Analysis

Next, I will give a brief overview of CCA. For a thorough introduction see (Mardia et al. 1980).

Let $\mathbf{x} \in \mathbb{R}^L$ and $\mathbf{y} \in \mathbb{R}^P$ be two random vectors. CCA determines a separate set of basis vectors (=filters) for each random vector, such that the correlations between $\mathbf{x}$ and $\mathbf{y}$, when projected onto the basis vectors are mutually maximised (ibid.). For a single pair of basis vectors $\mathbf{u}$ and $\mathbf{v}$, the projections are given as:

$$a = \mathbf{u}^T\mathbf{x}, \tag{6.13}$$

$$b = \mathbf{v}^T\mathbf{y}. \tag{6.14}$$

Using the above definitions and assuming $\mathbb{E}[\mathbf{x}] = \mathbb{E}[\mathbf{y}] = 0$, the (Pearson) correlation ρ between a and b can be written as:

$$\begin{aligned}
\rho(a,b) &= \frac{\mathbb{E}[ab]}{\sqrt{\mathbb{E}[aa]\,\mathbb{E}[bb]}}, \\
&= \frac{\mathbb{E}\left[(\mathbf{u}^T\mathbf{x})(\mathbf{v}^T\mathbf{y})\right]}{\sqrt{\mathbb{E}\left[(\mathbf{u}^T\mathbf{x})(\mathbf{u}^T\mathbf{x})\right]\mathbb{E}\left[(\mathbf{v}^T\mathbf{y})(\mathbf{v}^T\mathbf{y})\right]}}, \\
&= \frac{\mathbb{E}\left[(\mathbf{u}^T\mathbf{x})(\mathbf{y}^T\mathbf{v})\right]}{\sqrt{\mathbb{E}\left[(\mathbf{u}^T\mathbf{x})(\mathbf{x}^T\mathbf{u})\right]\mathbb{E}\left[(\mathbf{v}^T\mathbf{y})(\mathbf{y}^T\mathbf{v})\right]}}, \\
&= \frac{\mathbf{u}^T\mathbb{E}\left[\mathbf{x}\mathbf{y}^T\right]\mathbf{v}}{\sqrt{\mathbf{u}^T\mathbb{E}\left[\mathbf{x}\mathbf{x}^T\right]\mathbf{u}\mathbf{v}^T\mathbb{E}\left[\mathbf{y}\mathbf{y}^T\right]\mathbf{v}}}, \\
&= \frac{\mathbf{u}^T\mathbf{C}_{xy}\ \mathbf{v}}{\sqrt{\mathbf{u}^T\mathbf{C}_{xx}\ \mathbf{u}\mathbf{v}^T\mathbf{C}_{yy}\ \mathbf{v}}},
\end{aligned} \tag{6.15}$$

where $\boldsymbol{C}_{\bullet\bullet}$ are the respective covariance matrices and $\mathbf{u}$ and $\mathbf{v}$ are the sought unknown basis vectors maximising the correlation. Note that Eq. 6.15 does not depend on the actual scaling of $\mathbf{u}$ or $\mathbf{v}$. Therefore, the optimisation problem addressed by CCA can formally be defined as:

$$\underset{\mathbf{u},\mathbf{v}}{max}\ \mathbf{u}^T\mathbf{C}_{xy}\ \mathbf{v} \qquad \text{s.t.} \quad \mathbf{u}^T\mathbf{C}_{xx}\ \mathbf{u} = \mathbf{v}^T\mathbf{C}_{yy}\ \mathbf{v} = 1. \tag{6.16}$$

Following the derivations of (Anderson 1958) (or similarly (Borga 1998)) the constrained optimisation problem from Eq. (6.16) can be written as the following Lagrangian:

$$F(\mathbf{u}, \mathbf{v}; \lambda_u, \lambda_v) = \mathbf{u}^T \mathbf{C}_{xy}\ \mathbf{v} - \lambda_u(\mathbf{u}^T \mathbf{C}_{xx}\ \mathbf{u} - 1) - \lambda_v(\mathbf{v}^T \mathbf{C}_{yy}\ \mathbf{v} - 1), \tag{6.17}$$

with λ_x and λ_y being the Lagrange multipliers. From the partial derivatives of Eq. 6.17 with respect to $\mathbf{u}$ and $\mathbf{v}$, we see that $\lambda_u = \lambda_v = \lambda$ and finally obtain:

$$\mathbf{C}_{xx}^{-1}\mathbf{C}_{xy}\mathbf{C}_{yy}^{-1}\mathbf{C}_{yx}\mathbf{u} = \lambda^2\mathbf{u}, \tag{6.18}$$

and

$$\mathbf{C}_{yy}^{-1}\mathbf{C}_{yx}\mathbf{C}_{xx}^{-1}\mathbf{C}_{xy}\mathbf{v} = \lambda^2\mathbf{v}. \tag{6.19}$$

Equations 6.18 and 6.19 define a generalised eigenvalue problem. The pair of basis vectors corresponding to the largest eigenvalue maximises the (canonical) correlation of the projected data. With each of the remaining eigenvalues, a pair of basis vectors is associated, such that bases $\boldsymbol{U} \in \mathbb{R}^{K\times R}$ and $\boldsymbol{V} \in \mathbb{R}^{K\times R}$ with $K = \min(L, P)$ and $R = \min(rank(\boldsymbol{x}), rank(\boldsymbol{y}))$ are obtained, which contain one basis vector per column. In the following, I assume that the *input* signals are of full rank such that $K = R$. However, depending on the transformation, the shared signal will in general not be of full rank. It can be shown (Mardia et al. 1980) that the projections $a_k = \mathbf{u}_k^T\mathbf{x}$ and $b_k = \mathbf{v}_k^T\mathbf{y}$ are uncorrelated which implies that $\mathbf{u}_i^T\mathbf{C}_{xx}\mathbf{u}_j = 0$ and $\mathbf{v}_i^T\mathbf{C}_{yy}\mathbf{v}_j = 0$ for $i \neq j$.

In practice, CCA may be applied via the SVD. Here, I adopt the CCA formulation from (Scharf and Mullis 2000) which is given as follows. Random vectors $\mathbf{x}$ and $\mathbf{y}$ are represented in canonical coordinates $\boldsymbol{a}$ and $\boldsymbol{b}$ according to:

$$\begin{bmatrix}\boldsymbol{a}\\ \boldsymbol{b}\end{bmatrix} = \begin{bmatrix}\mathbf{F}^T & \mathbf{0}\\ \mathbf{0} & \mathbf{G}^T\end{bmatrix}\begin{bmatrix}\mathbf{C}_{xx}^{-1/2} & \mathbf{0}\\ \mathbf{0} & \mathbf{C}_{yy}^{-1/2}\end{bmatrix}\begin{bmatrix}\mathbf{x}\\ \mathbf{y}\end{bmatrix}, \tag{6.20}$$

with $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{R}^K$, i.e., $\boldsymbol{a}$ and $\boldsymbol{b}$ are the coefficients of the CCA basis vectors, obtained by projecting the original data. The canonical correlations are then given as the main diagonal of the covariance matrix $\mathbf{C}_{\boldsymbol{ab}}$ of the projections $\boldsymbol{a}$ and $\boldsymbol{b}$, which is a diagonal matrix by definition (Mardia et al. 1980). In Eq. 6.20, first a whitening transform is applied to $\mathbf{x}$ and $\mathbf{y}$, based on the whitening matrices $\mathbf{C}_{xx}^{-1/2}$ and $\mathbf{C}_{yy}^{-1/2}$. Next, the whitened vectors are transformed with orthogonal matrices $\mathbf{F}$ and $\mathbf{G}$ to obtain their canonical form. Matrices $\mathbf{F}$ and $\mathbf{G}$ are given by the SVD of the so called coherence matrix $\mathbf{C} = \mathbf{C}_{xx}^{-1/2}\mathbf{C}_{xy}\mathbf{C}_{yy}^{-T/2}$, such that:

$$\mathrm{svd}(\mathbf{C}) = \mathbf{FKG}^T. \tag{6.21}$$

Here, matrix $\mathbf{K}$ is the (diagonal) canonical correlation matrix with:

$$\mathbf{K} = \mathbb{E}\{\boldsymbol{ab}^T\} = \mathbf{C}_{\boldsymbol{ab}} = \mathbf{F}^T\mathbf{CG}. \tag{6.22}$$

Bases $\mathbf{U}$ and $\mathbf{V}$ are given as:

$$\mathbf{U} = \mathbf{C}_{xx}^{-T/2}\mathbf{F}, \tag{6.23}$$

$$\mathbf{V} = \mathbf{C}_{yy}^{-T/2}\mathbf{G}. \tag{6.24}$$

The canonical correlations measure the correlations of the canonical coordinates and thus the cosine of the angle between coordinates $\boldsymbol{a}(i)$ and $\boldsymbol{b}(i)$ (Scharf and Mullis 2000).

CCA can also be performed via QR decomposition or SVD of both data matrices without having to estimate covariance matrices from data (see (Anderson 1958; Mardia et al. 1980) for detailed derivations). Canonical coordinates may also be extracted in a recursive manner, as shown in (Ali Pezeshki et al. 2003), where a neural network implementation of CCA is presented.

6.4 Application of CCA to Transformation Learning

As we have seen, the objective of CCA is to determine sets of basis vectors $\boldsymbol{U}$ and $\boldsymbol{V}$, which I denote as *latent mappings*. The correlation of the paired data is then maximised in the *latent* space, i.e., when projected onto the CCA bases. Intuitively, the correlation in the latent space is maximised when the transformation between $\mathbf{x}$ and $\mathbf{y}$ has been compensated perfectly such that $\mathbf{x}$ and $\mathbf{y}$ appear to be (approximately) identical. Therefore, the CCA basis vectors have to represent the mapping from $\mathbf{x}$ to $\mathbf{y}$ and vice versa. Inferring $\mathbf{x}$ from $\mathbf{y}$, i.e., to apply the learnt transformation on $\mathbf{x}$, then amounts to project $\mathbf{x}$ into the latent space via $\mathbf{U}^T\mathbf{x}$ and to apply the inverse mapping of $\mathbf{V}$ to obtain an estimate of $\mathbf{y}$.

It turns out that the minimum mean-squared error estimator of $\mathbf{x}$ from $\mathbf{y}$ (and vice versa) in canonical coordinates is given by (Scharf and Mullis 2000; Scharf and John Thomas 1998):

$$\hat{\mathbf{x}} = \boldsymbol{F}_{\boldsymbol{xy}}^T\mathbf{y} = \mathbf{C}_{xx}^{1/2}\mathbf{F}\mathbf{K}\mathbf{G}^T\mathbf{C}_{yy}^{-1/2}\mathbf{y}, \tag{6.25}$$

$$\hat{\mathbf{y}} = \boldsymbol{F}_{\boldsymbol{yx}}^T\mathbf{x} = \mathbf{C}_{yy}^{1/2}\mathbf{G}\mathbf{K}\mathbf{F}^T\mathbf{C}_{xx}^{-1/2}\mathbf{x}. \tag{6.26}$$

Note that the canonical correlations $\mathbf{K}$ in Eq. 6.25 and Eq. 6.26 are weighting each basis vector and are a measure of the importance of the basis vector to explain the hidden transformation. Assume that both input images are of full rank, such that $rank\{\boldsymbol{x}\} = rank\{\boldsymbol{y}\} = K$. As has been explained previously, depending on the latent mapping, the shared signal will in general not be of full rank and we have $rank\{\boldsymbol{z}_{\boldsymbol{y}|\boldsymbol{x}}\}, rank\{\boldsymbol{z}_{\boldsymbol{x}|\boldsymbol{y}}\} \leq rank\{\boldsymbol{x}\} = rank\{\boldsymbol{y}\}$. The rank of the shared signal is therefore given as the rank of the canonical correlation matrix $\mathbf{K}$.

In practice, we obtain the bases $\boldsymbol{U}$ and $\boldsymbol{V}$ by applying CCA to a large set of training images (see Sec. 6.5 for details). A learnt transformation may then be applied to previously unseen data via Eq. 6.25 and Eq. 6.26.

6.5 Learning Global Transformations on Natural Images

In the following, I present results for learning various transformations based on the proposed approach. Prior learning a transformation, we need to set up data matrices $\mathbf{X}$ and $\mathbf{Y}$. The i-th columns $\mathbf{X}_{:,i}$ and $\mathbf{Y}_{:,i}$ are assumed to be generated according to the observation model in Eqs. 6.7 and 6.8, and are assumed to be related via the hidden mappings $\boldsymbol{F_{xy}}$ and $\boldsymbol{F_{yx}}$ according to Eq. 6.9 and Eq. 6.10. To setup the data matrices, we obviously have to identify corresponding image pairs. This could be done by first learning pixel-to-pixel correspondences and then to crop patches centred on the pixel correspondences. However, in the following I will generate corresponding image pairs manually. Specifically, I learn transformations on patch pairs, cropped from natural images of the *Berkeley Segmentation Database* (BSD300) (Martin et al. 2001) which consists of 300 natural images. For a specific transformation, I generate a training set as follows: The first view is generated by randomly selecting G square patches of dimension $2D \times 2D$ out of the pool of natural images. Then, I apply the transformation to each patch, which generates the second view. Finally, the centre part of dimension $D \times D$ of both patches will be cropped and used as a training example. This guarantees that both patches are completely covered by a part of a natural image, such that the training data has full rank. I discard patches where the signal variance is small (=homogeneous areas). Finally, I obtain data matrices $\boldsymbol{X}, \boldsymbol{Y} \in \mathbb{R}^{D \cdot D \times G}$ which contain training views column wise. Note that for each data set, the transformation is held fixed. Throughout the experiments for each training set, I generate $G = 10,000$ square patches of size $D \times D = 21 \times 21$ pixels.

The classes of transformations considered here are:

- translation in x direction by d_x pixels,
- translation in y direction by d_y pixels,
- in plane rotation by α degrees,
- scaling by factor γ,
- general nonlinear transformations,
- random permutations,
- combinations of these.

After the data matrices have been generated, I add i.i.d. zero-mean Gaussian noise with standard deviation σ separately to every image in order to simulate observation noise.

6.5.1 CCA Basis Vectors

Let us now visually inspect the bases **U** and **V**, which we obtain when applying CCA to the training data. To this end, I reshape the basis vectors and visualise them as 2-D images. Each basis vector is independently rescaled to the interval $[0, 1]$. Note that I rescale the basis vectors *only* for reasons of a better visualisation.

As the training images are of dimensions $D \times D = 21 \times 21$ pixels (and assuming that the input data is of full rank), CCA will return 441 basis vectors for the first view and the second view, respectively. For reasons of clarity, I will only visualise the first 25 pairs of basis vectors for each learnt transformation.

We will first regard a rotation transformation by $\alpha = 90$ degrees and the influence of the observation noise on the obtained basis vectors. Figure 6.5 visualises the basis vectors for four different noise standard deviations, i.e., for $\sigma \in [0, 1, 4, 10]$. While we never expect to observe noise free data in practice, it is still interesting to see how CCA represents the transformation, compared to the noise afflicted data. In the noise free case, the basis vectors tend to develop a localised structure around a single pixel or a small pixel subset. If noise is added to the training data, the basis vectors develop a low-pass like characteristic. The 90 degree rotation is clearly reflected within each pair of basis vectors; each pair of basis vectors is related by a 90 degree rotation. While we may distinguish between the noise free and noise afflicted case only by visual inspection, the learnt basis vectors for the different noise levels appear very similar. Note that the filters may have an opposite polarity (black areas become white and vice versa) but still represent the same basis vector in terms of direction. Next, let us inspect the influence of the amount of training data on the learnt basis vectors.

Figure 6.6 visualises the basis vectors obtained for training set sizes of 500, 2,500, 5,000 and 10,000 image pairs with a fixed noise level $\sigma = 1$. For 500 images, the basis vectors seem to be of a random structure, which lets us conclude that the basis vectors do not capture the transformation. For 2,500 image pairs, the basis vectors develop the expected low pass characteristic but appear to be noisy. The basis vectors become smoother for 5,000 images and 10,000 image pairs. From a theoretical point of view, the required minimum number of *linearly independent* training pairs is given by the dimensions of the input data. However, for a practical application, we conclude that the amount of training data should be a multiple of this theoretical lower limit in order to attenuate the influence of data noise on the basis vectors.

For other transformations, we basically see the same influence of observation noise and sample size on the learnt basis vectors, as was shown for the 90 degree rotation. In the following, I will therefore only present results for a fixed observation noise of $\sigma = 1$ and 10,000 training image pairs.

Figure 6.7 visualises learnt basis vectors for translations in x and y direction, rotations and for a combination of these transformations. Again, the basis vectors tend to develop a low-pass like structure, and each pair of basis vectors is related by the respective transformation. In contrast to the 90 degree rotation, a translation transformation will always lead to shared signals $\boldsymbol{z}_{\boldsymbol{y}|\boldsymbol{x}}$ and $\boldsymbol{z}_{\boldsymbol{x}|\boldsymbol{y}}$ which cannot be of full rank. Obviously, this is due to the fact that for a translation, there are signal parts in the right view, which are not visible in the left view and vice versa. This is also the case for rotations, when the rotation angle is *not* an integer multiple of $\frac{\pi}{2}$. The fact that the shared signal is not of full rank can also be seen from the learnt basis vectors. The filters contain areas in which the filters are inactive, i.e., the filter weights are close to 0 (greyish colour in the plots). This is due to the fact that none of the transformations in Fig. 6.7 is a full permutation and can thus not be inverted.

Based on the results presented so far, we conclude that basis vectors obtained by CCA are able to differentiate between shared and non-shared signal parts by means of non-zero and close to zero filter values. I will further discuss this in Sec. 6.5.3.

Figure 6.8 visualises the basis vectors for a vertical split, a mirror, a quarter split and a random permutation transformation. Similar as for the previously regarded linear transformations, the CCA basis vectors clearly reflect the respective transformation. As all transformations are full permutations, each pixel in the left view has a unique corresponding pixel in the right view. Hence, the shared signals are of full rank and the basis vectors cover the whole spatial extend. The basis vectors obtained for the random permutation are particularly interesting: While the filters for all transformations regarded so far developed a low-pass characteristic, this is only the case for the basis vectors of the left view.

So far, we have only regarded linear transformations of the input data. Recall that the process of projecting data onto the CCA basis is a linear operation. However, the *basis vectors* may represent arbitrary nonlinear transformations. Next, let us inspect the basis vectors obtained, when CCA is applied to image pairs that are related by a *nonlinear* transformation. Figure 6.9 visualises basis vectors for four more nonlinear transformations; a 'foveated' transformation represented by a nonlinear radially symmetric scaling, scaling by a factor of 2, and a nonlinear (quadratic) scaling in x direction. For the first three transformations, the shared signal $\boldsymbol{z}_{\boldsymbol{y}|\boldsymbol{x}}$ is of full rank. Therefore, the basis vectors of the left view cover the whole spatial extend. However, the shared signal $\boldsymbol{z}_{\boldsymbol{x}|\boldsymbol{y}}$ is not of full rank. Hence, the basis vectors of the right view contain close to 0 values.

So far, we have seen that CCA represents a hidden transformation by means of low-pass like basis vectors. Besides linear (affine) transformations, CCA also seems to be able to extract nonlinear transformations of the input data. While these findings are solely based on a visual inspection of the basis vectors, next, I will analyse the behaviour of the canonical correlations and show that these indicate the rank of the shared signal. While we may easily infer the shared signal by visual inspection of the basis vectors, there is no direct way to actually compute the shared dimensions. In order to infer the coordinates of the shared dimensions, I will regard the *summed energy filters* in Sec. 6.5.3.

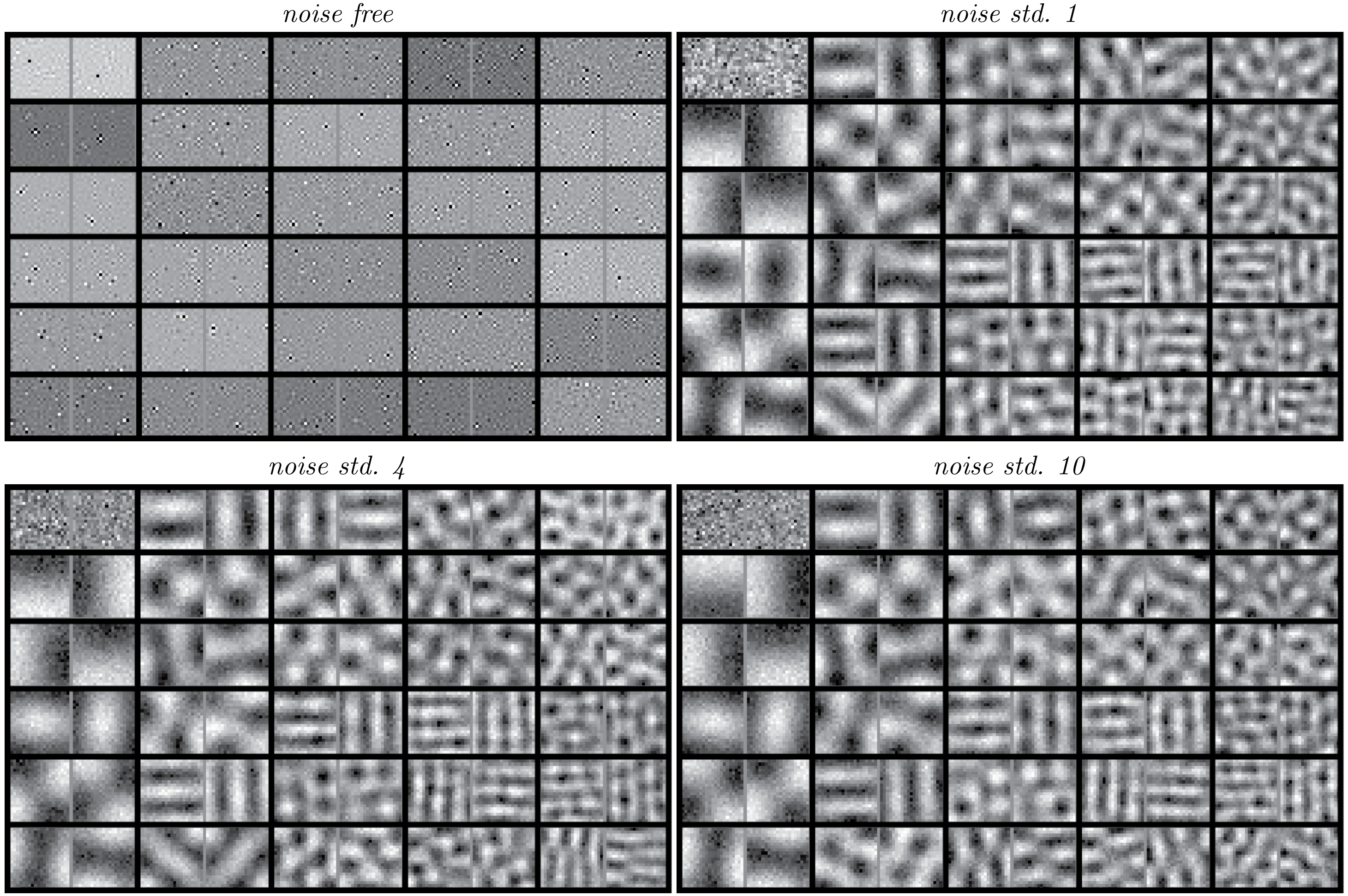
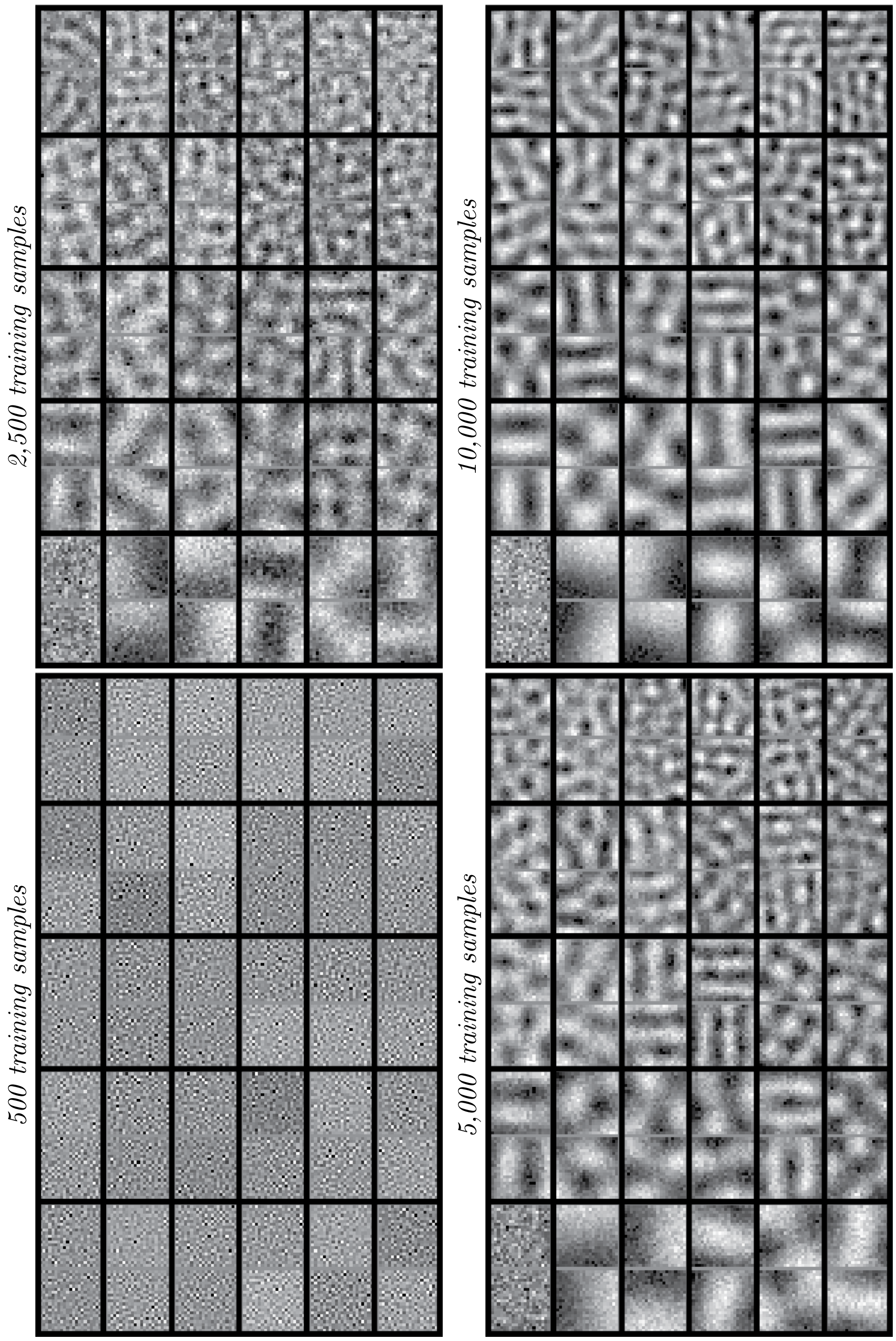


Figure 6.5: **Learning a 90 degree rotation:** while varying the noise standard deviation. Within each image, the first 25 basis vectors obtained by CCA are visualised (column-major ordering). The i-th pair of basis vectors visualises $\mathbf{U}_{:,i}$ and $\mathbf{V}_{:,i}$ reshaped to form 2-D images of dimensions $D \times D$. See text for details.



500 training samples

2,500 training samples

5,000 training samples

10,000 training samples

Figure 6.6: **Learning a 90 degree rotation:** while varying the amount of training data. See text for details.

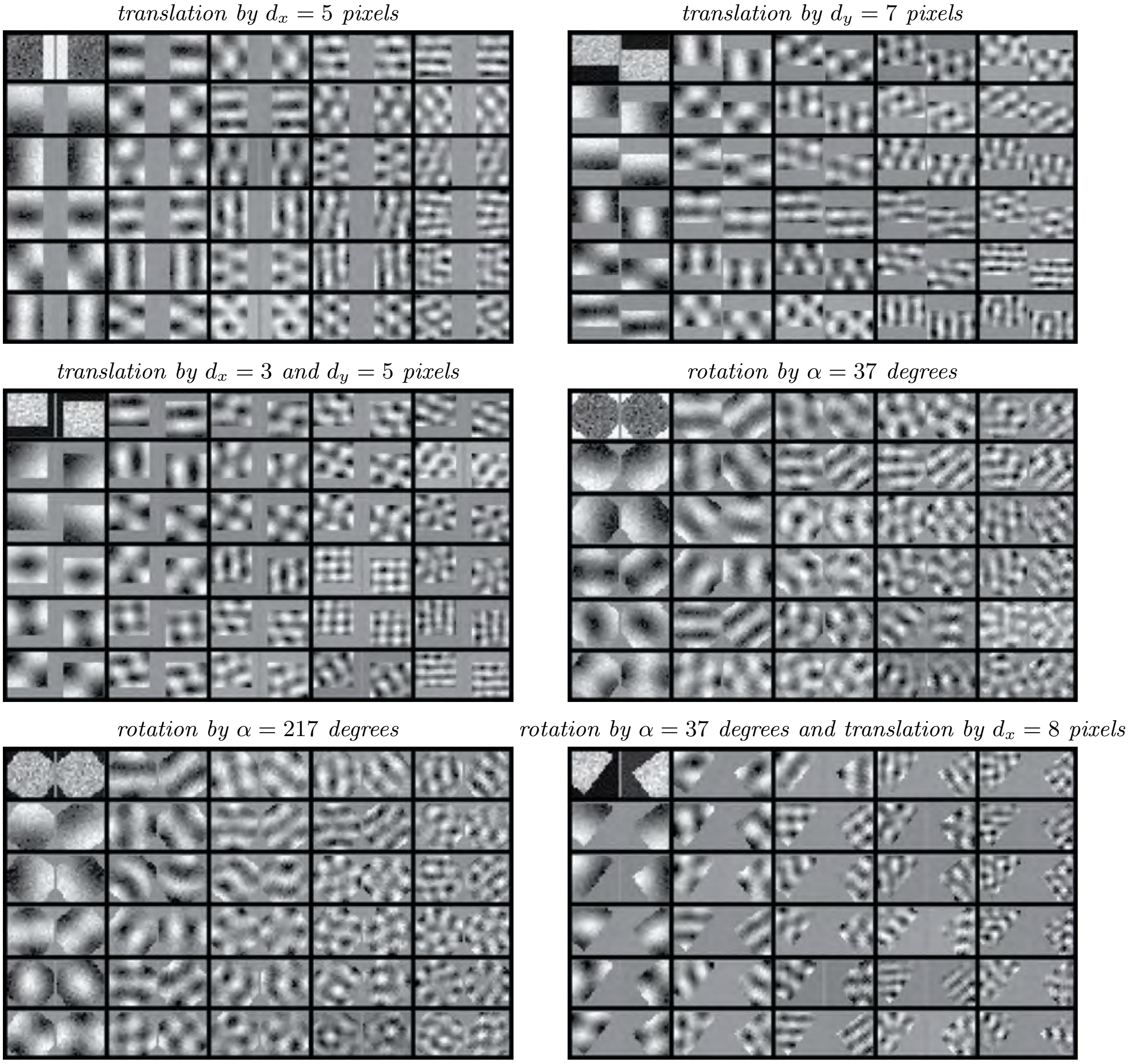

translation by $d_x = 5$ pixels
translation by $d_y = 7$ pixels
translation by $d_x = 3$ and $d_y = 5$ pixels
rotation by $\alpha = 37$ degrees
rotation by $\alpha = 217$ degrees
rotation by $\alpha = 37$ degrees and translation by $d_x = 8$ pixels

Figure 6.7: **Learning affine transformations:** CCA basis vectors obtained when the hidden transformation is a translation and/or rotation transformation. See text for details.

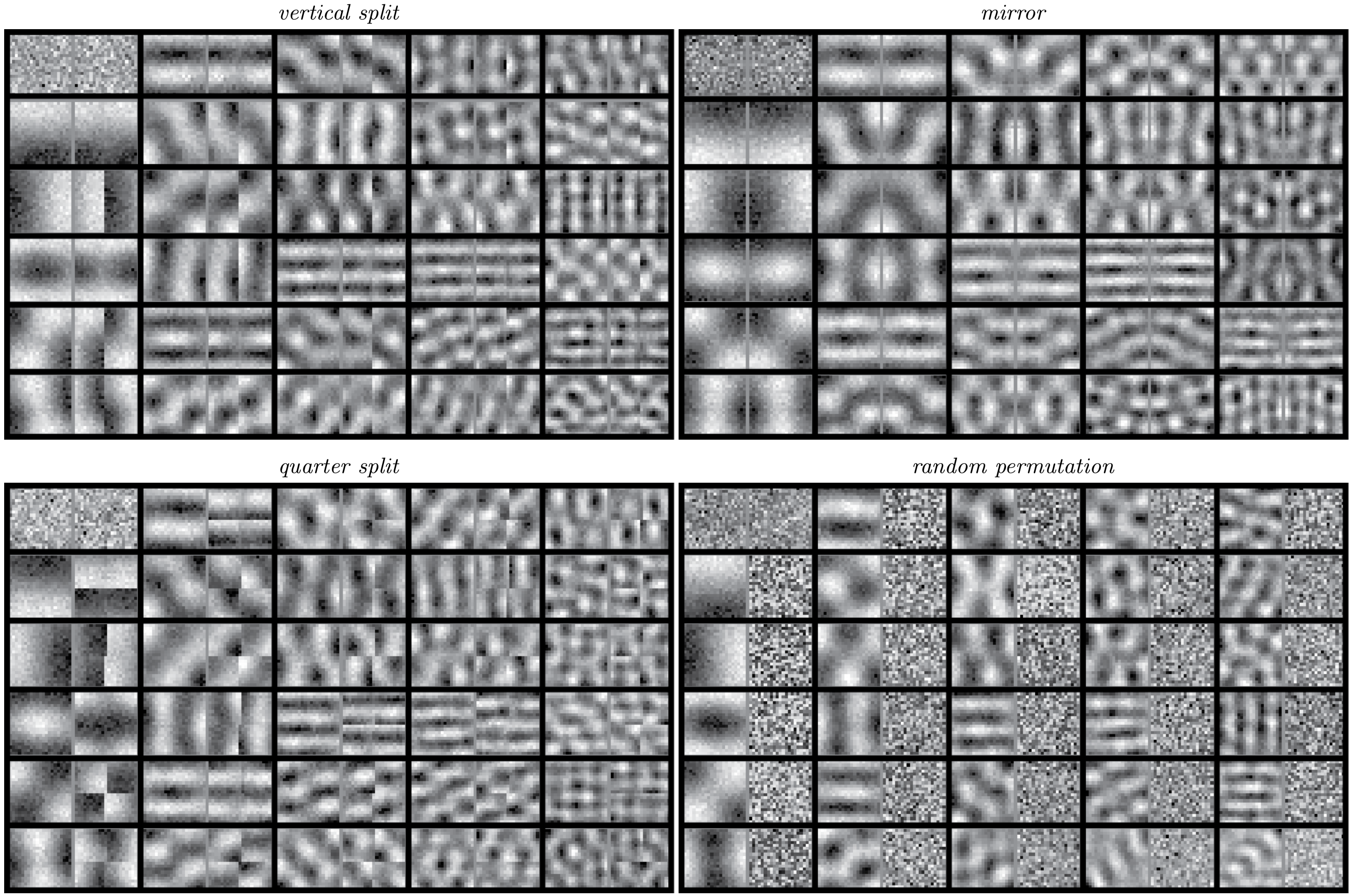


Figure 6.8: **Learning permutations:** CCA basis vectors obtained when the hidden transformation is a full permutation. See text for details.

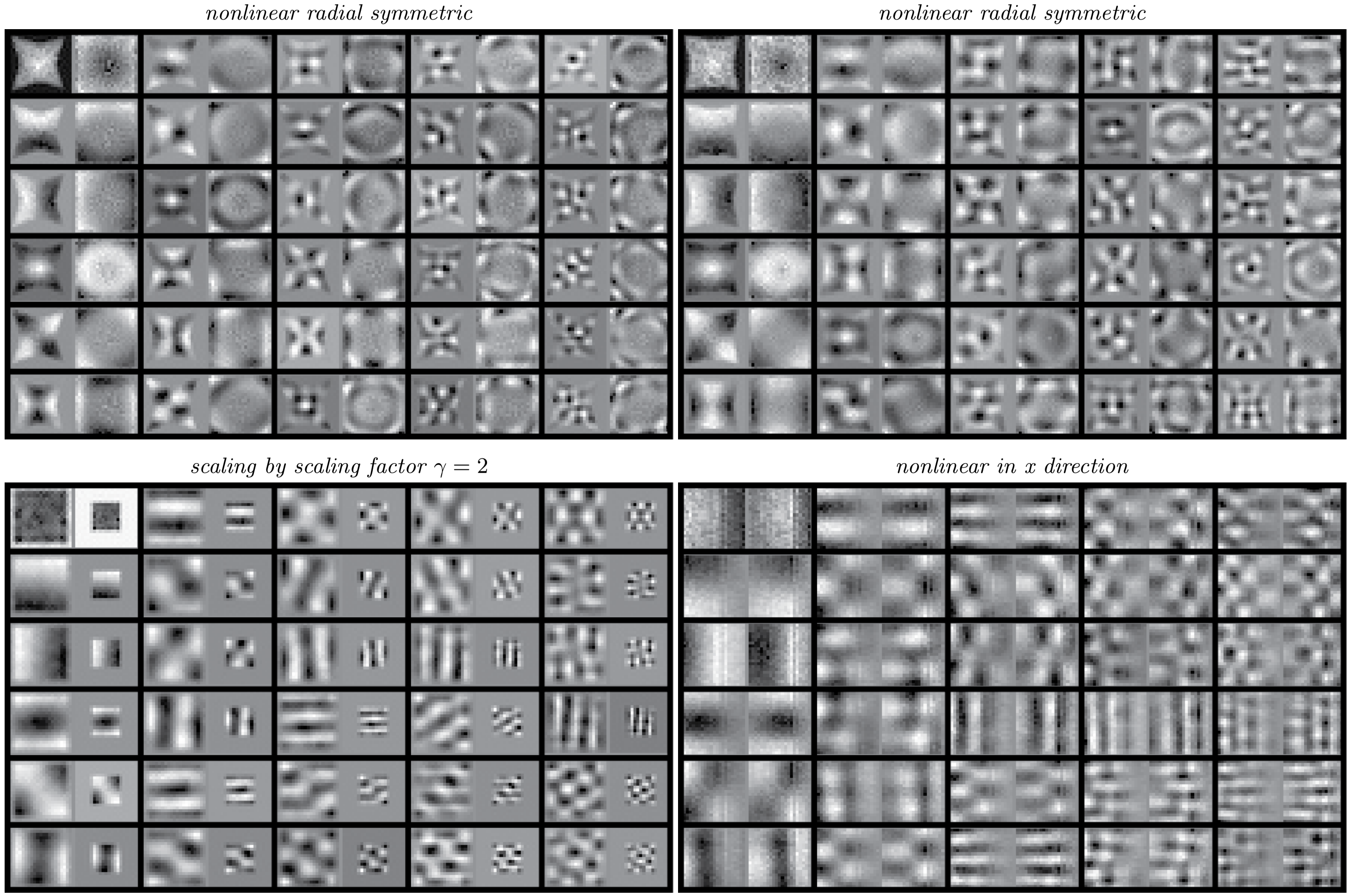

nonlinear radial symmetric
nonlinear radial symmetric
scaling by scaling factor $\gamma = 2$
nonlinear in x direction

Figure 6.9: **Learning nonlinear transformations:** CCA basis vectors obtained when the hidden transformation is a nonlinear transformation. See text for details.

6.5.2 Behaviour of the Canonical Correlations

Let us now regard the canonical correlations, which are associated with each pair of basis vectors. Recall that the canonical correlation associated with the i-th pair of basis vectors represents the empirical correlation of the input data, when projected onto the i-th pair of basis vectors given by $\mathrm{Corr}(\mathbf{U}_i^T\mathbf{x}, \mathbf{V}_i^T\mathbf{y})$. I expect the canonical correlations to be large, when the pair of basis vectors represent a shared dimension. Likewise, I expect the canonical correlations to be small, when the pair of basis vectors represent a non-shared dimension. Therefore, I expect the canonical correlations to indicate the rank of the shared signal.

In the following, I will analyse the canonical correlations for the transformations presented in Sec. 6.5.1. As the training data is of dimension $21 \times 21 = 441$, there are 441 canonical correlations assuming that the training data is of full rank. In order to study the influence of observation noise on the canonical correlations, I will vary the noise level within $\sigma = [0, 1, 2, 4, 6, 9]$.

Let us now regard the canonical correlations for a rotation transformation by $\alpha = 90$ degrees, which are visualised in Fig. 6.10 (top left). In the noise free case, all correlations are 1. This is to be expected, as the 90 degree rotation constitutes a full permutation, i.e., each pixel in the left view has a unique corresponding pixel in the right view and CCA is able to learn basis vectors which perfectly align the input data. When noise is added to the training data, the correlations decline nonlinearly. The more noise is added to the training data, the steeper the descent is. Let us now compare the correlations of the 90 degree rotation to a transformation where the shared signal is not of full rank. Figure 6.10 (top right) visualises the canonical correlations for a translation transformation by $d_x = 5$ pixels. For this transformation, the number of shared dimensions is bounded by $(D \cdot D) - (d_x \cdot D) = 441 - 5 \cdot 21 = 336$. For the noise free case, the correlations are one for the shared dimensions and drop abruptly for the first non-shared dimension. Then, for the non-shared dimensions, the correlations decline linearly. When noise is added to the training data, the correlations decline nonlinearly for the shared dimensions and again linearly for the non-shared dimensions, where the inflection point of the curve marks the true rank of the shared signal. Furthermore, it is interesting to note that the *slower* the correlations decline in the shared dimensions, the *faster* they decline in the non-shared dimensions. When we compare these results to the 90 degree rotation, we see that the qualitative course of the correlations is very similar, when regarding only the shared dimensions.

For the remaining plots in Fig. 6.10, we basically see the same behaviour of the correlations for shared and non-shared dimensions; the canonical correlations decline nonlinearly in shared dimensions and linearly in non-shared dimensions.

We may ask why the correlations for non-shared dimensions do not vanish. Recall that I generated the training data from a large set of natural images. As natural images do not vary randomly but most of the time smoothly, correlations will be found among non-shared regions as well. Clearly, the distribution of the canonical correlations depend on the statistics of the source signal. For example, if a transformation would be learnt on, say, white noise images, then the correlations would be zero for all non-shared dimensions, as a white noise signal has zero cross correlation by definition. However, this can only be seen when the transformation is learnt on a very large dataset, such that the empirical cross correlations completely vanish.

In Fig. 6.11, I visualise the canonical correlations for the split and mirror transformations. As these transformations are again full permutations, the canonical correlations are very similar to those of the 90 degree rotation.

Let us now regard the canonical correlations for nonlinear transformations of the input data, as visualised in Fig. 6.12. For the noise free case, we see a clear difference in the behaviour of the correlations, when compared to those of the linear transformations: The correlations for the noise free case decline over the shared dimensions. While the decline is very small for the scaling transformation and the nonlinear scaling in x direction, the plots for the 'foveated' transformation clearly show a nonlinear decline over the shared dimensions. However, when noise is added to the training data, the qualitative course of the correlations is similar to the ones of the linear transformations. As real world training data will always contain noise, we expect the correlations to decline nonlinearly in practice.

I am now interested in identifying the shared/non-shared dimensions. As the correlations are given in canonical coordinates, there is no direct 1-to-1 mapping to the original coordinates (=pixels). Therefore, I will regard the *summed energy filters* as described in the next section.

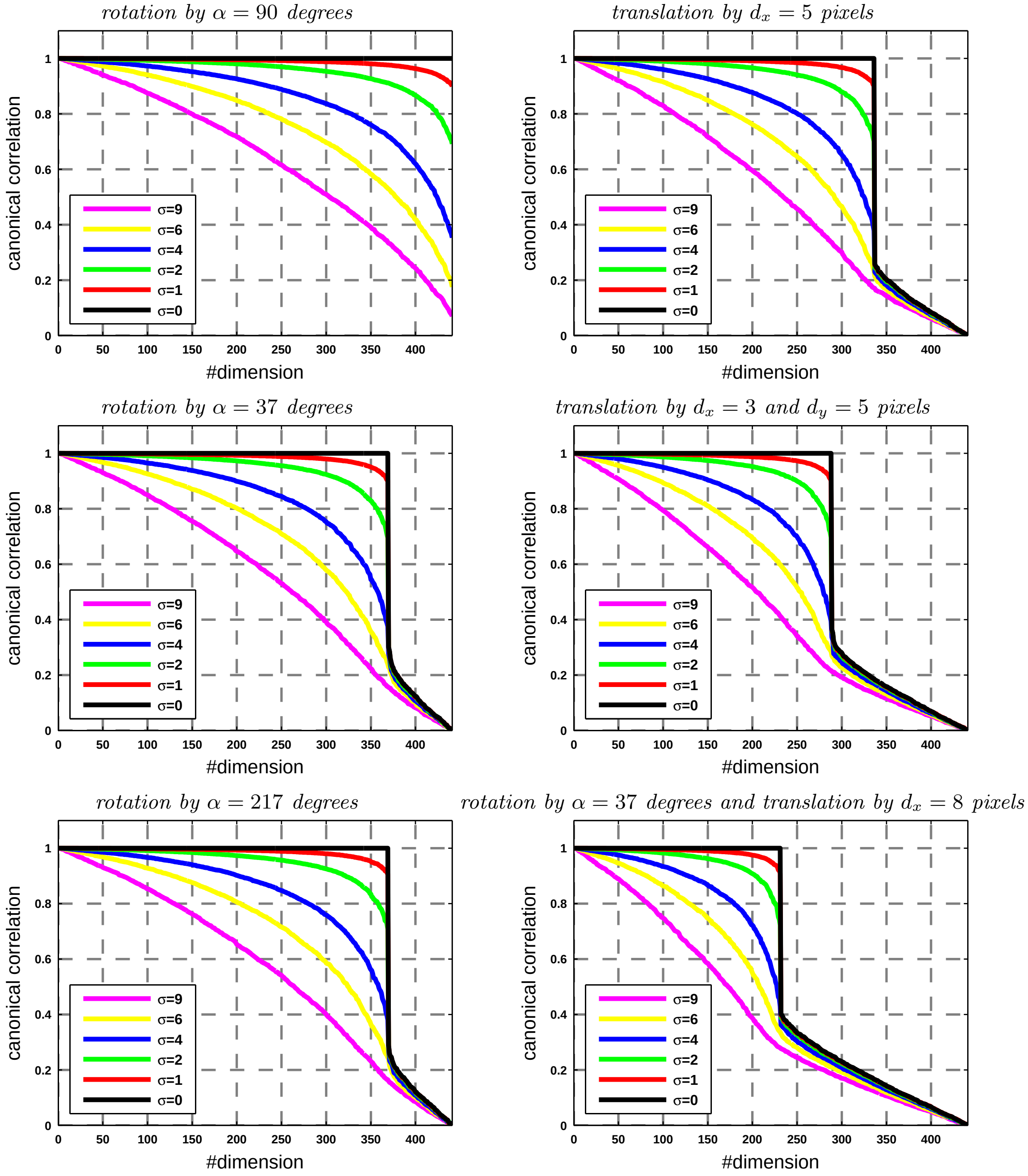


Figure 6.10: **Canonical correlations for translated and rotated views:** (within each plot) Each curve shows the canonical correlation for a specific signal noise level $\sigma = \{0, 1, 2, 4, 6, 9\}$ under an i.i.d. zero-mean Gaussian noise model. Similar results are obtained for other types of transformations. See text for details. Figure only interpretable when viewed in colour.

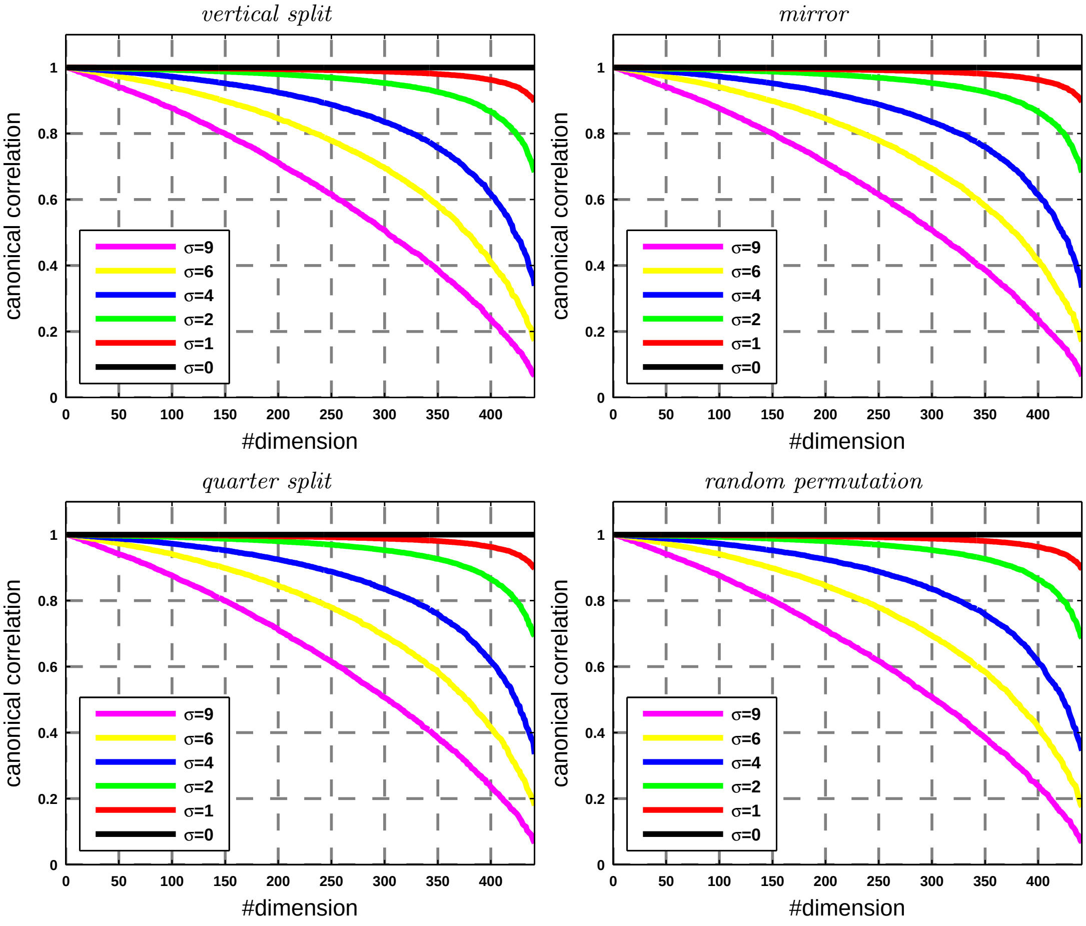


Figure 6.11: **Canonical correlations for full permutations:** (within each plot) Each curve shows the canonical correlation for a specific signal noise level $\sigma = \{0, 1, 2, 4, 6, 9\}$ under an i.i.d. zero-mean Gaussian noise model. Similar results are obtained for other types of transformations. See text for details. Figure only interpretable when viewed in colour.

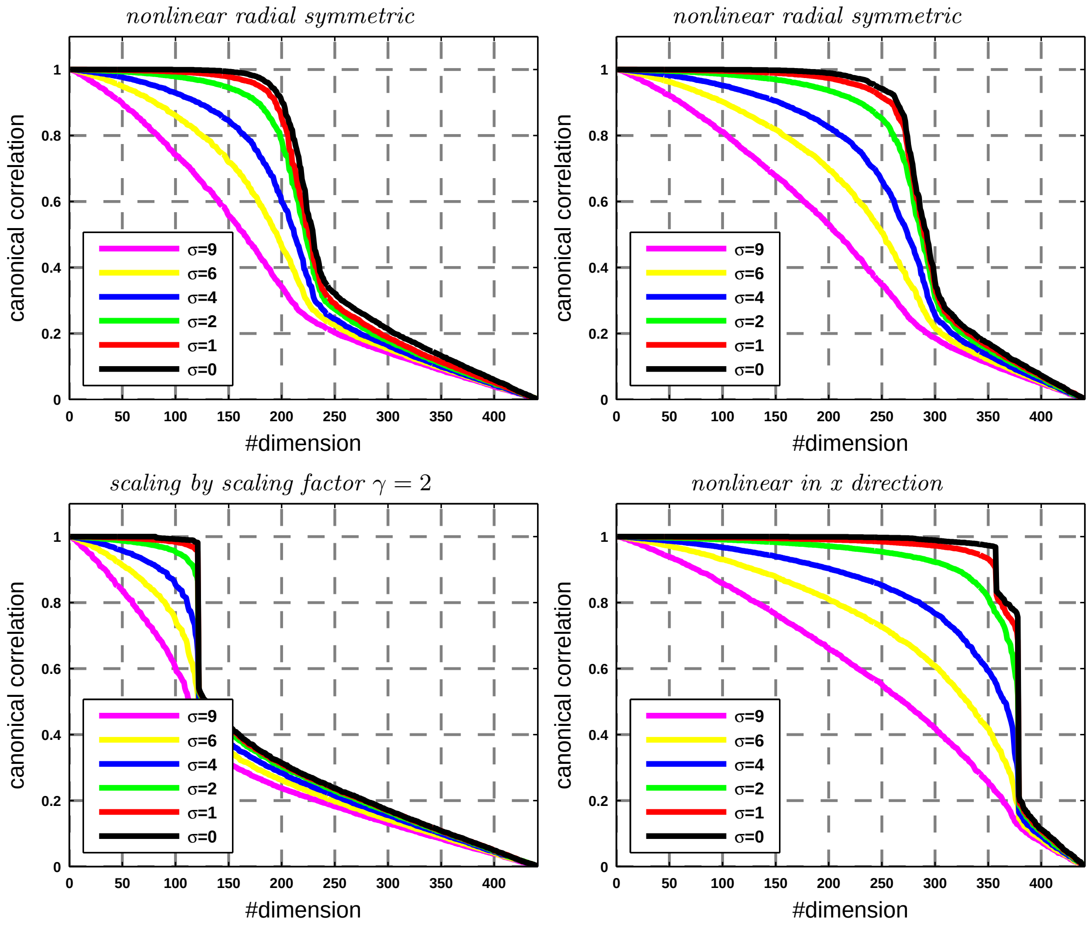


Figure 6.12: **Canonical correlations for nonlinear transformations:** (within each plot) Each curve shows the canonical correlation for a specific signal noise level $\sigma = \{0, 1, 2, 4, 6, 9\}$ under an i.i.d. zero-mean Gaussian noise model. Similar results are obtained for other types of transformations. See text for details. Figure only interpretable when viewed in colour.

6.5.3 Inferring Shared Dimensions

While the rank of the shared signal indicates the number of pixels (dimensions) which are shared among both views, there is no direct way to identify those pixels. This is due to the fact that a CCA basis vector is not to be understood as a mapping from one pixel to another, but that the basis vector may spend energy over the whole spatial domain. In order to infer the common 'footprint' of the shared signal in the original coordinates, I propose to analyse *where* the basis vectors spend energy.

Let $\mathbf{U}_n = [\mathbf{u}_1, .., \mathbf{u}_n]$ and $\mathbf{V}_n = [\mathbf{v}_1, .., \mathbf{v}_n]$ be matrices consisting of the first n basis vectors of bases $\mathbf{U}$ and $\mathbf{V}$, respectively. I define the *summed energy filter* $\boldsymbol{e}^n_{\mathbf{U}}$ and $\boldsymbol{e}^n_{\mathbf{V}}$ for both bases $\mathbf{U}$ and $\mathbf{V}$ as:

$$\boldsymbol{e}^n_{\mathbf{U}} = \mathrm{diag}\{\mathbf{U}_n \mathbf{U}_n^T\}, \tag{6.27}$$

$$\boldsymbol{e}^n_{\mathbf{V}} = \mathrm{diag}\{\mathbf{V}_n \mathbf{V}_n^T\}. \tag{6.28}$$

Then, $\boldsymbol{e}^n_{\mathbf{U}}$ and $\boldsymbol{e}^n_{\mathbf{V}}$ will contain the sum of the squared filter weights (=energy) per dimension. If n is set to the true rank of the shared signal, one observes that the basis vectors spend energy only among the shared dimensions. Due to numerical inaccuracies, basis vectors may spend small amounts of energy within non-shared regions as well.

For an intuitive understanding of the summed energy filters, regard Fig. 6.13. The figure shows the first and last 3 basis vectors, as well as $\boldsymbol{e}^n_{\mathbf{U}}$ and $\boldsymbol{e}^n_{\mathbf{V}}$ for a translation transformation by $d_x = 3$ and $d_y = 7$ pixels. The summed energy filters are shown for $n = 231$ (true rank of the shared signal) and $n = 441$ (all basis vectors). It can be seen that the first 3 filters spend energy within shared dimensions and no energy in non-shared regions (=grey areas). The last 3 basis vectors spend energy in non-shared dimensions as expected. When the summed filters are determined up to the true rank of the shared signal, the shared and non-shared dimensions are clearly separated (black/white) while they merge when all filters are summed.

In order to estimate the true rank of the shared signal, I regard the first and second order derivative of the canonical correlations curve, or rather the discrete approximation of these. Strictly speaking, I regard the difference quotient of first and second order of the canonical correlation curve as it is a discrete function.

Recall that my empirical results suggested that the correlations drop nonlinearly in shared dimensions and linearly in non-shared dimensions, where the coordinate of the inflection point of the correlation curve marks the true rank. Figure 6.14 visualises the first and second order derivative of the canonical correlation curve for rotations, translations and the mirror transformation. For the 90 degree rotation and the mirror transformations, the derivatives are close to 0 over the whole domain. This is to be expected, as the shared signal is of full rank (cf. Sec. 6.5.2).

For the remaining rotations and translation transformations, we see that the first derivative has a global minimum, and that the second derivative changes its sign, thus indicating an inflection point. Throughout these plots, we may detect a global minimum

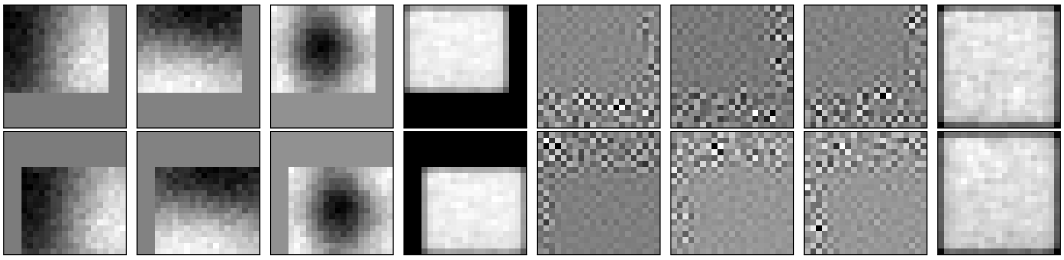


Figure 6.13: **Energy distribution among dimensions for the noise free case:** (first to third column) First three basis vectors with maximum canonical correlation for (top) left view and (bottom) right view. (fourth column) Summed energy filters $e^n_{\mathbf{U}}$ and $e^n_{\mathbf{V}}$ which only spend energy among the shared dimensions. (fifth to seventh column) Last three basis vectors with small canonical correlation and summed energy filters. See text for details.

in the first derivative and a sign change in the second derivative up to a noise level of $\sigma = 4$. For larger noise levels the plots need to be regarded on a different scale.

Figure 6.15 visualises the first and second order derivative of the canonical correlation curve for nonlinear transformations of the training data for which we see similar results as for the linear transformations. Note that the plot limits for the 'foveated' transformation are much smaller than for the other transformations, in order to make the plot interpretable. In fact, the derivatives appear very noisy, and it is questionable whether we may determine the true rank for these kind of transformations in practice. However, we may still estimate a conservative lower bound on the number of shared dimensions, which suffices to at least get an approximation of the spatial footprint of the shared signal.

Figure 6.16 visualises the summed energy filters for various transformations. The rank of the shared signal, up to which the filters are summed, has been determined according to the global minima of the first derivative of the canonical correlation curve. It can be seen that we may infer the spatial footprint of the shared signal based on the proposed approach.

As will be shown in the following, if we apply a learnt transformation based on the full CCA bases (including the ones which spend energy in the non-shared dimensions), an interesting side effect is that the predictors (Eqs. 6.25,6.26) effectively extrapolate within the non-shared regions.

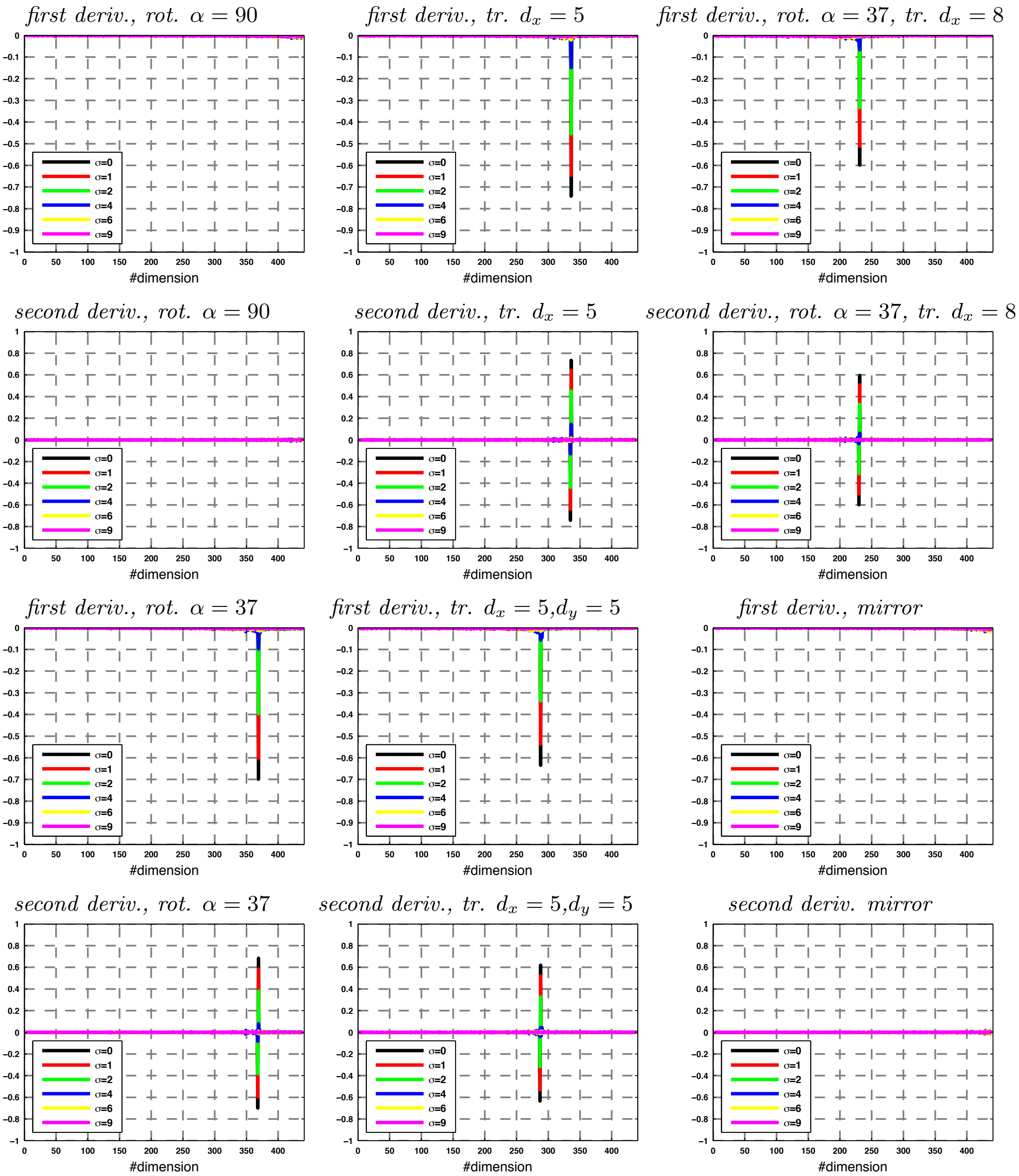


Figure 6.14: **Approximated first and second order derivative of the canonical correlation curve:** for rotations, translations and general permutations. See text for details. Figure only interpretable when viewed in colour.

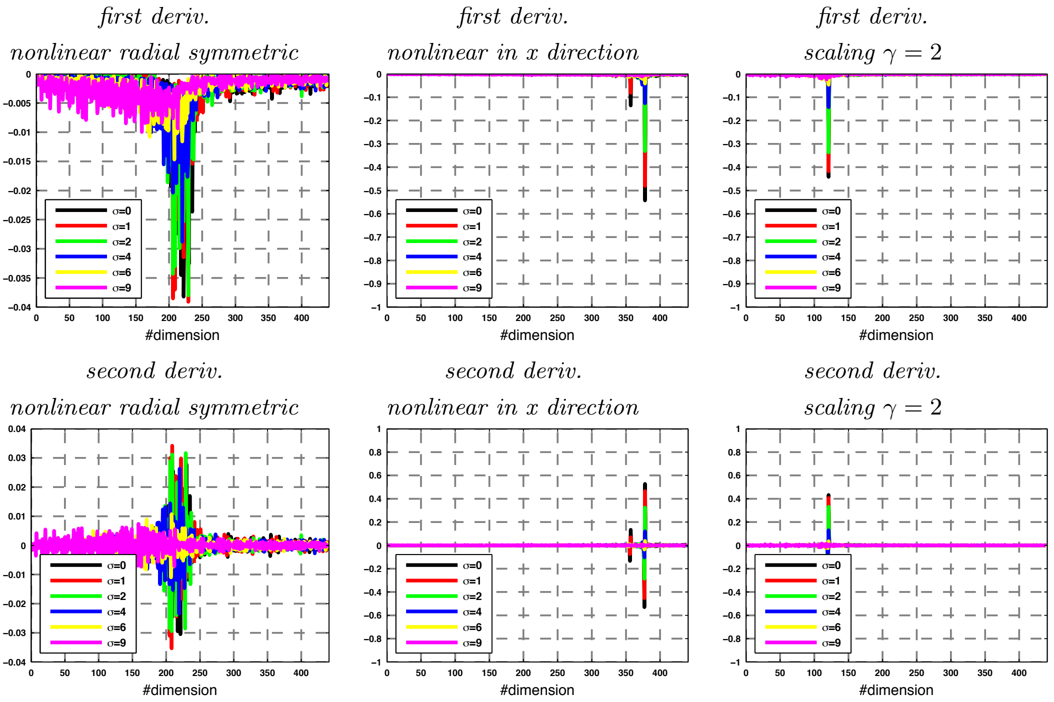


Figure 6.15: **Approximated first and second order derivative of the canonical correlation curve:** for nonlinear transformations. See text for details. Figure only interpretable when viewed in colour.

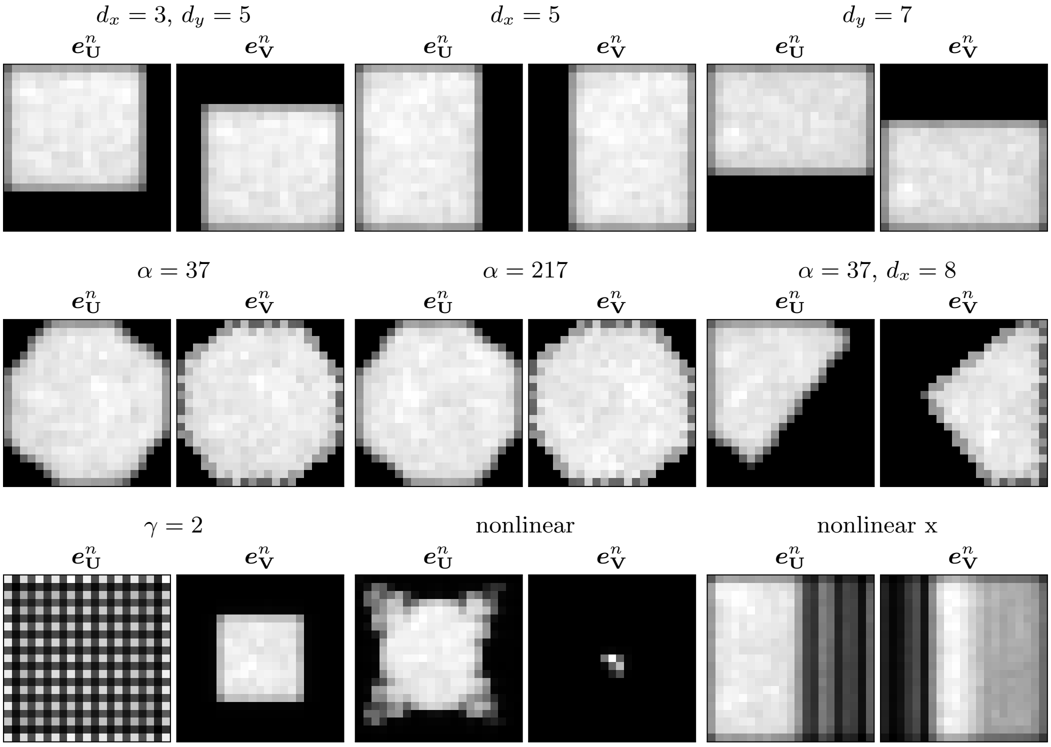


Figure 6.16: **Summed energy filters up to estimated signal rank:** Summed energy filters $e^n_{\mathbf{U}}$ and $e^n_{\mathbf{V}}$ summed up to the estimated rank of the shared signal for various transformations. See text for details.

6.5.4 Predictions

So far we have seen that CCA represents spatial transformations by means of transformation specific basis vectors. Based on the canonical correlations, we may infer the rank of the shared signal and determine the spatial footprint of the shared signal.

Next, I will apply the learnt transformations, based on the MMSE estimators given in Eq. 6.25, and Eq. 6.26. It is important to note that I apply the transformations to *previously unseen* data. The only assumption I make is that the image statistics of the training data and previously unseen data are similar. This is the case here, as I learn and apply the transformations based on natural images. In the following, I apply the learnt transformations to face images of the Olivetti database (ATT-Laboratories-Cambridge n.d.). For each transformation, I show examples for the predictions of $\hat{\mathbf{x}}$ and $\hat{\mathbf{y}}$ when interpreting the given input image as $\mathbf{y}$ or $\mathbf{x}$, respectively.

Figures 6.17 and 6.18 show predictions for rotation and translation transformations. In the upper half images of the figure, it can be seen that the predicted images are non-zero in the non-shared dimensions. It is important to note that this is actually not noise, but that the model extrapolates within non shared dimensions. This becomes clearer when we regard input images with a black border. This is shown in the lower half images of Fig. 6.17 and Fig. 6.18, where I set a two pixel wide black border around the input images. It can be seen that the non-shared regions are not filled with noise but by the texture close to the image borders (= homogeneous black) as expected.

From the predictions of the rotation transformations, it can be seen that this type of transformation is considerable more difficult to represent, as there is no 1-to-1 correspondence between the pixels and the model needs to interpolate.

Figure 6.19 shows predictions for the split and mirror transformations. As all these transformations are full permutations, the predictions are very accurate. It is interesting to note that the left and right predictions are identical. This is to be expected, as the input image always represents a frontal (non-split/mirrored) face.

Figure 6.20 shows predictions for nonlinear transformations. Note how the model is able to zoom in and zoom out the input images. The foveated transformation, especially the right predictions, clearly show that the method is able to represent nonlinear transformations. The left predictions show the nonlinear behaviour as well, but are harder to interpret. Again, we see that the model extrapolates in the non-shared regions only when there is evidence that the image extends over the borders.

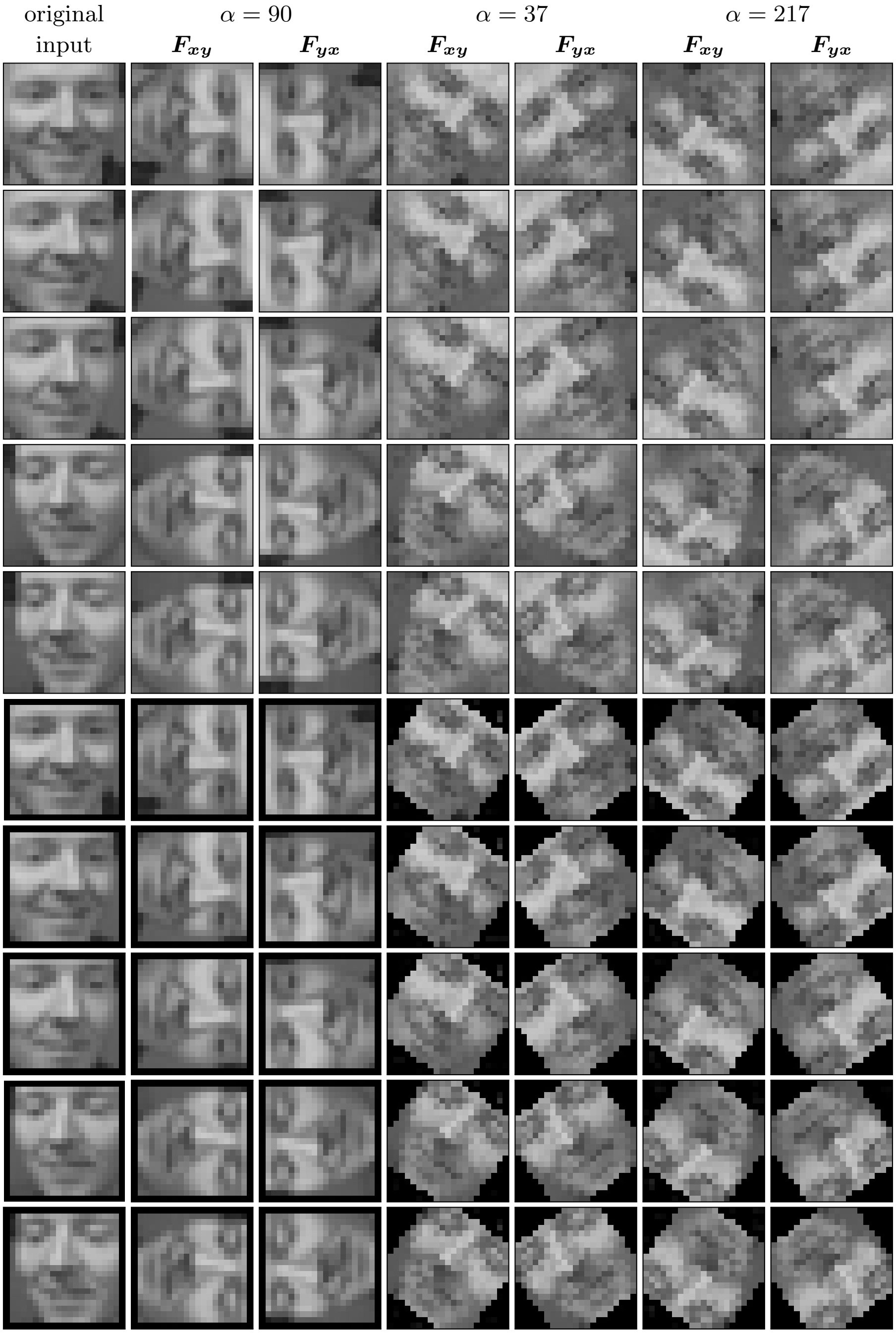


Figure 6.17: **Learnt rotations applied to previously unseen data:** See text for details.

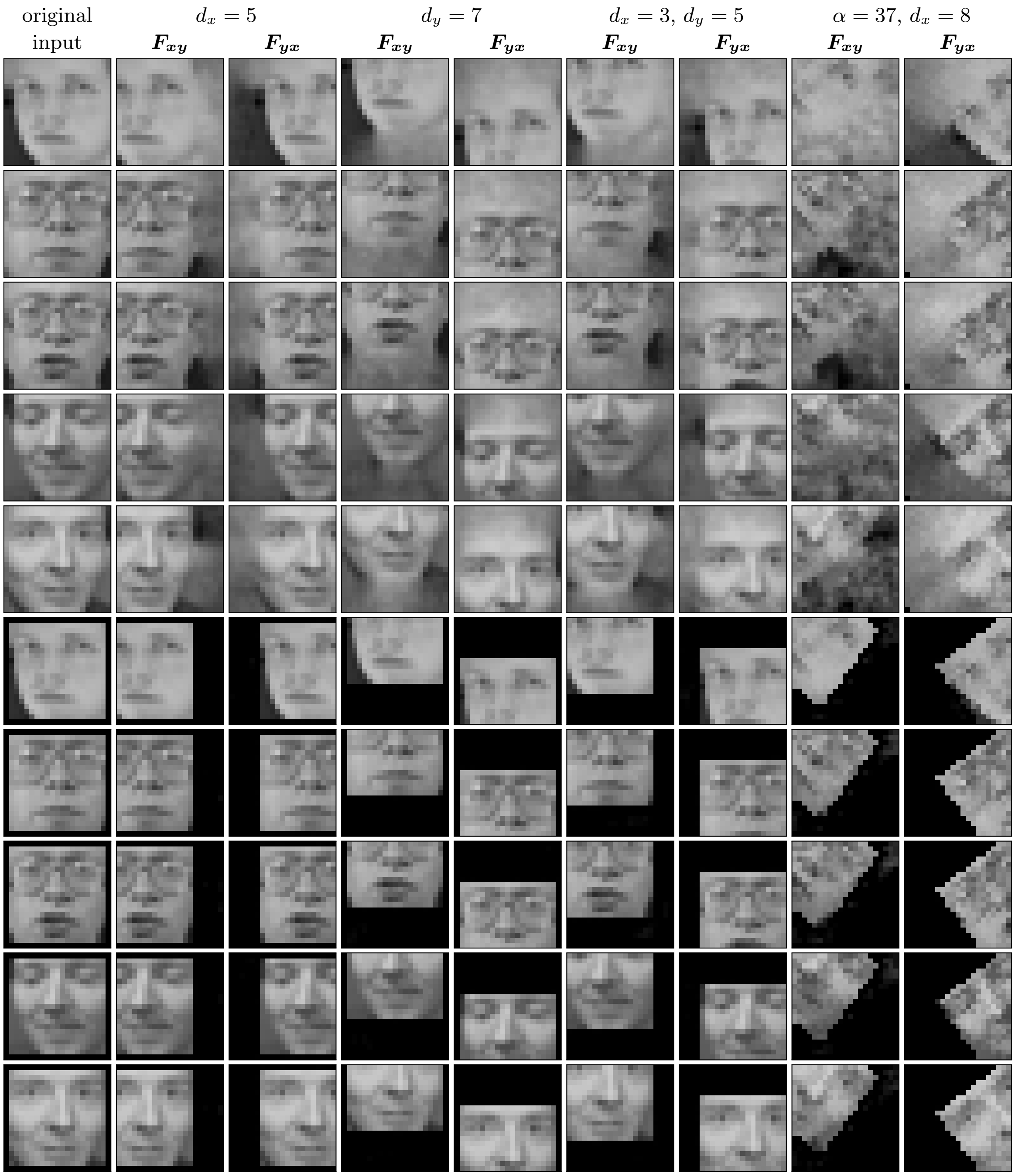


Figure 6.18: **Learnt translations and rotations applied to previously unseen data:** See text for details.

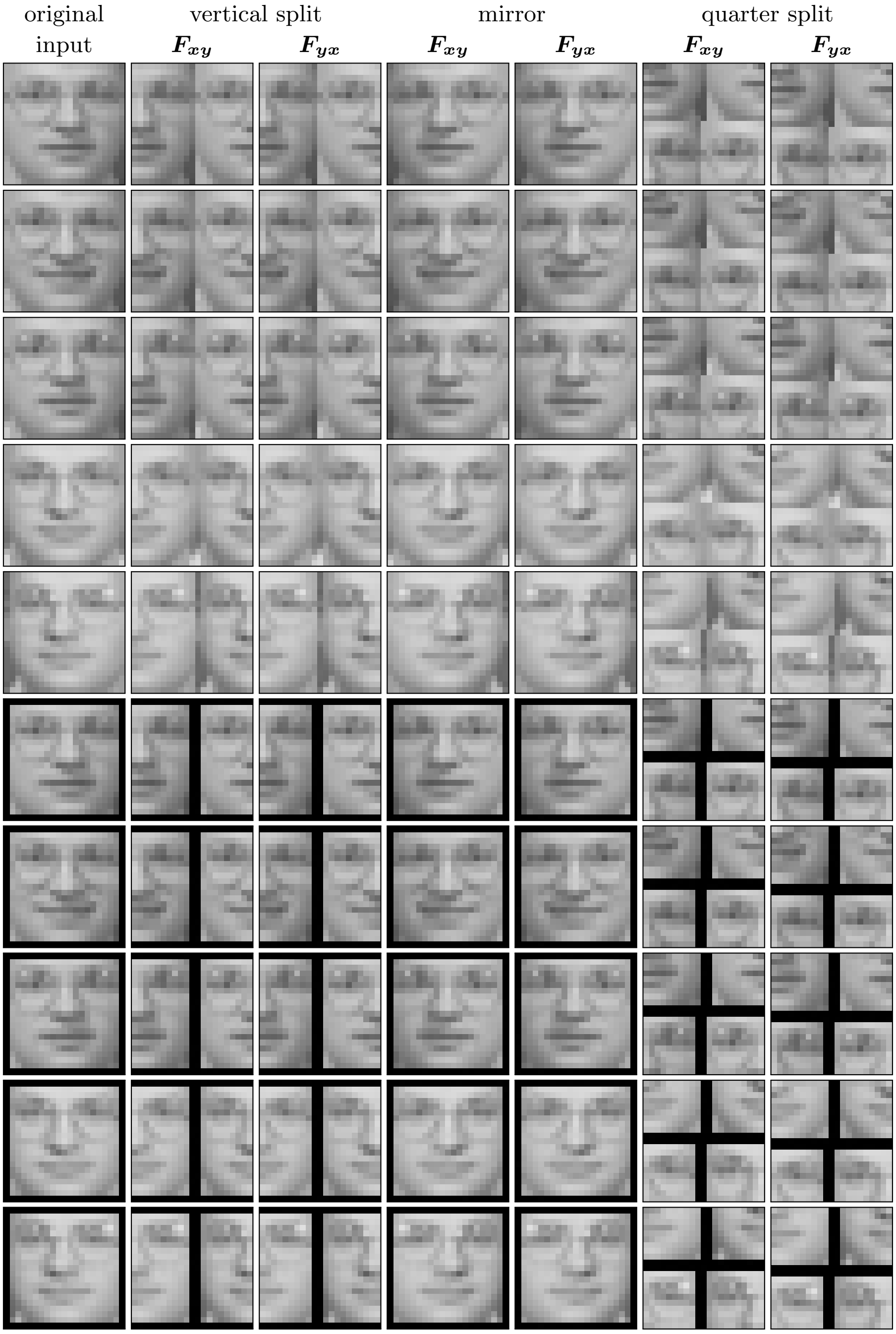


Figure 6.19: **Learnt permutations applied to previously unseen data:** See text for details.

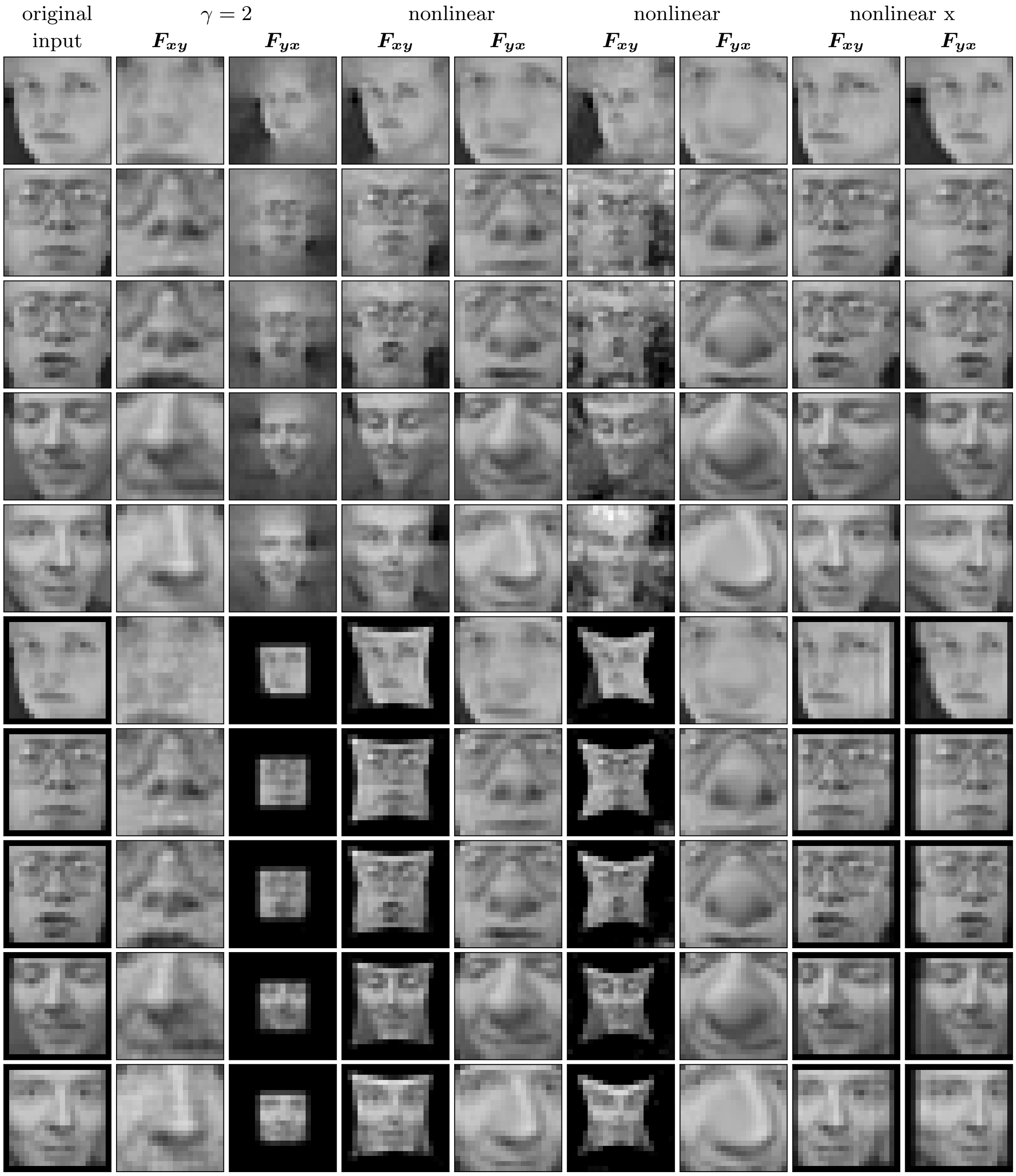


Figure 6.20: **Learnt nonlinear transformations applied to previously unseen data:** See text for details.

6.6 Correspondence Priors for Binocular Camera Setups

Within the previous sections, we have seen that CCA is able to extract and represent linear and nonlinear spatial transformations by means of two sets of basis vectors. While the experiments were based on manually generated training data, I will now present a CCA based approach to estimate correspondence priors in real-world binocular camera setups.

My approach is split into two stages. Within the first stage, I learn the inter-image transformation for a pair of cameras $\mathcal{C}_i$ and $\mathcal{C}_j$ (with fixed relative orientation) based on CCA. This analysis usually has to be done in a multi-scale framework, depending on the given image resolution, as applying CCA directly to full resolution images may be computationally prohibitive. In the second stage, I employ the learnt transformation and predict for a given pixel in $\mathcal{C}_i$ its corresponding region within $\mathcal{C}_j$. I denote these regions as correspondence prior.

6.6.1 Method

It may appear more as a theoretical possibility than as an actually feasible approach to learn the transformation between paired data via CCA. Applying CCA directly on natural images, even of rather low spatial resolutions, say, above 256×256 pixels, is computationally prohibitive. While online/recursive algorithms for CCA exist (Ali Pezeshki et al. 2003), I follow a different route; I apply CCA within a multi-resolution framework, where the transformation between the views $\mathcal{C}_i$ and $\mathcal{C}_j$ of a binocular image stream is determined on a coarse scale only. Having learnt the transformation on the coarse scale, I can predict for a given pixel on the fine scale in $\mathcal{C}_i$, which pixel in a low resolution version of $\mathcal{C}_j$ corresponds to it. Subsequently, the predicted correspondence is reprojected to the original resolution. The reprojection obviously results in a loss of resolution (cf. TCA on the pyramid, Sec. 3.7), and we cannot find pixel-to-pixel correspondences on the fine scale. Instead, I determine a region which contains the true correspondence with high probability. This region can compactly be described by means of spatial moments (second order statistics) and will be denoted as *correspondence prior*.

6.6.2 Learning the Inter-Image Transformation

In the first phase, I learn the transformation between $\mathcal{C}_i$ and $\mathcal{C}_j$ as follows. Let T be the number of images in a temporal segment of the binocular image stream. The algorithm starts with generating the two data matrices $\mathbf{X} \in \mathbb{R}^{N \times T}$ and $\mathbf{Y} \in \mathbb{R}^{N \times T}$, where each column in $\mathbf{X}$ and $\mathbf{Y}$ correspond to a subsampled version of the original image, respectively. During subsampling, I keep the aspect ratio of the original resolution such that $\frac{R}{\gamma} \cdot \frac{C}{\gamma} = N$, where γ is the subsampling factor.

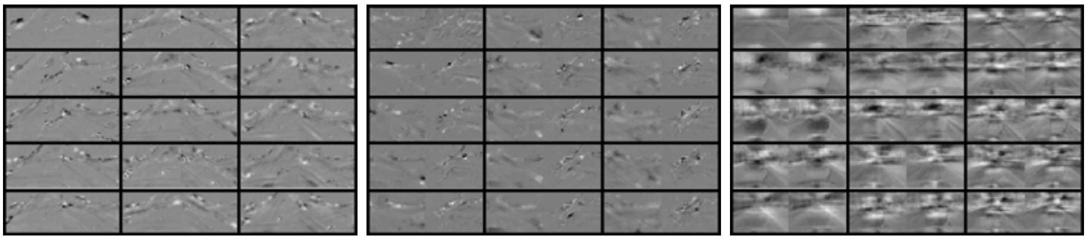


Figure 6.21: **Learning transformations in real world setups:** First 9 pairs of basis vectors determined by CCA for each of the three sequences (`GUBo1616`, `GUBo1606` and `GUCar`) used within the experiments. See text for details.

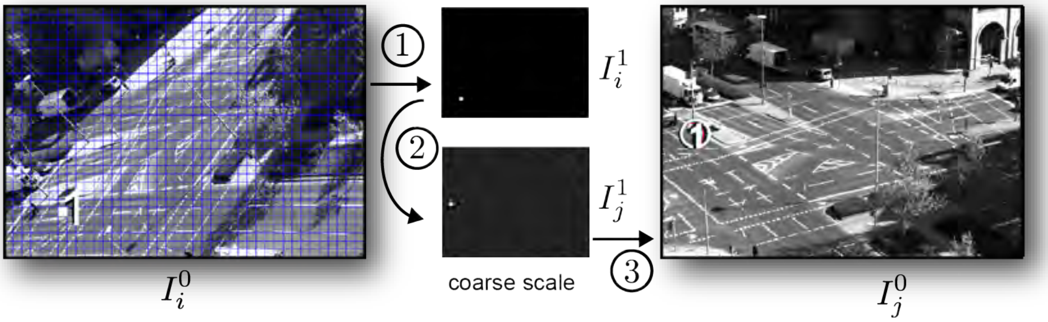


Figure 6.22: **Generation of correspondence priors:** Visual description of how correspondence priors are generated, once the transformation between two views has been learnt via CCA. See Sec. 6.6.2.1 for details. Best viewed in colour.

6.6.2.1 Prediction

Within the multi-resolution framework, I use the predictor equations from Sec. 6.4 to generate correspondence priors as follows. Let (x, y) be the spatial coordinates of a pixel in $\mathcal{C}_i^0$, where the superscript 0 denotes the original resolution. Next, determine the pixels' coordinates within the low resolution as $(u, v) = (\frac{x}{\gamma}, \frac{y}{\gamma})$, and generate a binary image $I_i^1 \in \{0, 1\}^{\frac{R}{\gamma} \times \frac{C}{\gamma}}$ of the same size as the low resolution, where the pixel at (u, v) is set to 1 (see step 1 in Fig. 6.22). Then apply the transformation learnt by CCA to the binary image via Eq. 6.25 and obtain the predicted image $I_j^1 \in \mathbb{R}^{\frac{R}{\gamma} \times \frac{C}{\gamma}}$. Obviously, when there is a one-to-one pixel correspondence and the transformation has perfectly been determined, the predicted image will be binary again, where a single pixel is set to 1, marking the corresponding pixel. However, this is only the case when the hidden transformation is a permutation. This will only rarely be the case for real world setups and we typically obtain predictions that encode regions of high probability containing the corresponding pixel (see step 2 in Fig. 6.22). Interpreting the predicted images as

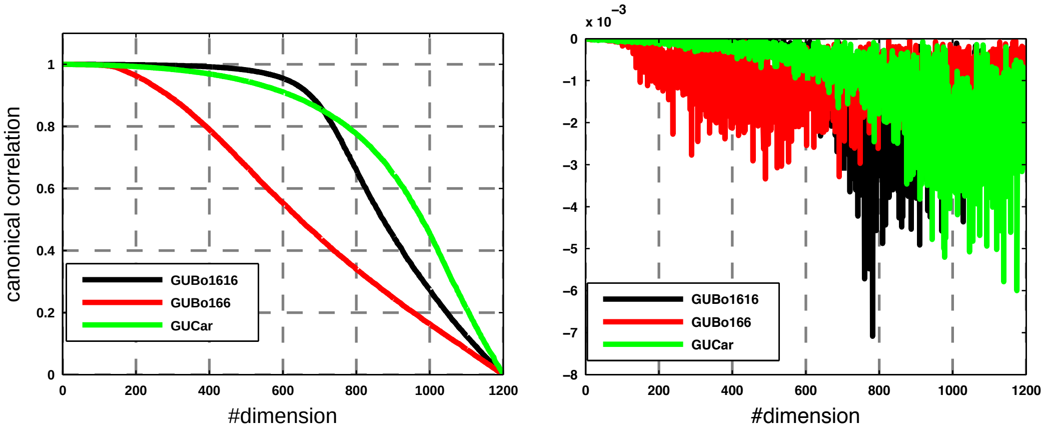


Figure 6.23: **Canonical correlation curves and corresponding derivatives:** for sequences `GUBo1616`, `GUBo1606` and `GUCar`. Figure only interpretable when viewed in colour.

an empirical bivariate correspondence distribution over the spatial image coordinates, I encode the prediction by means of a 2 × 2 covariance matrix $\mathbf{C}_p$ (cf. Sec. 3.5). Given that the eigenvalues of $\mathbf{C}_p$ are both small, I consider a correspondence to exist. Finally, the correspondence prior to the pixel at $(x, y) \in \mathcal{C}_i^0$ in the second view is given as the covariance error ellipse of $\mathbf{C}_p$, projected onto I_j^0 (see step 3 in Fig. 6.22).

Using the proposed method, I learn transformations for sequences `GUBo1616`, `GUBo1606` and `GUCar`. Recall that these sequences have a spatial resolution of 480 × 640 pixels. I learn the transformations as described in 6.6.2 using a subsampling factor of 16 (with bicubic interpolation), i.e., I apply CCA on data matrices $\mathbf{X}, \mathbf{Y}$ of size $(30 \cdot 40) \times 10{,}000$. I perform CCA via SVD on the data matrices which takes roughly 7 seconds on a single 2.6 GHz CPU.

Figure 6.21 shows the first 9 pairs of CCA basis vectors for each of the scenes. It can be seen that for all of the three setups, the basis vectors are not random but develop a structure that is characteristic for the setup. However, the basis vectors appear to be substantially different from the ones presented in Sec. 6.5.1. Figure 6.23 shows the canonical correlations and the respective first derivatives. Figure 6.24 visualises the log of the respective summed energy filters. The plots of the canonical correlations for `GUBo1616` and `GUBo1606` are similar to the ones of the nonlinear transformations shown in Sec. 6.5.2. The correlations for `GUBo1616` decline slower than for `GUBo1606`, indicating that the shared signal for sequence `GUBo1616` has a larger spatial footprint, which indeed is the case. The correlations for sequence `GUCar` are more similar to the translation/rotation transformations, and indicate a large overlap of the input views, as is the case for this sequence. The summed energy filters in Fig. 6.24 are determined as described in Sec. 6.5.3. The filters may appear counter intuitive when compared to

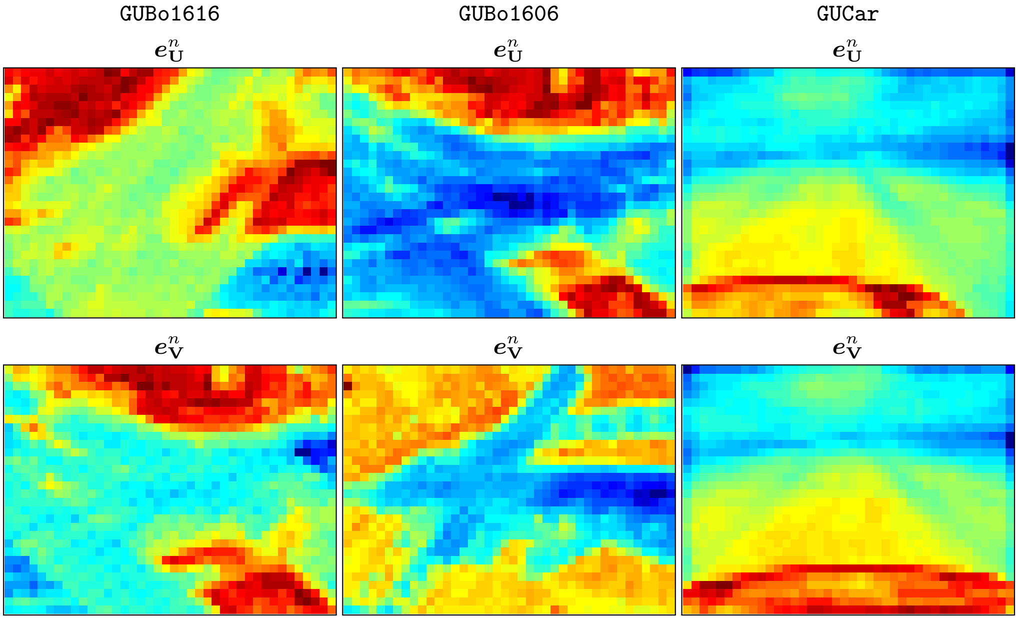


Figure 6.24: **Summed energy filters:** for sequences `GUBo1616`, `GUBo1606` and `GUCar`. See text for details. Figure only interpretable when viewed in colour.

the ones in Sec. 6.5.3. In fact, we may not directly read off shared and non-shared dimensions. I believe this is due to the fact that the training data sample the spatial domain relatively sparse, compared to the hand generated data. Then, the static parts of the scene seem to dominate the overall structure of the summed energy filter. This can bee seen from the dark red regions of the filters. Regard the summed energy filters for `GUCar`. Here, the engine hood is a constant structure and hence dominates the filter. In order to find overlapping regions, I do not regard the summed energy filters but only regard the correspondence priors as described in the following.

Finding Correspondence Regions For a pixel location in $\mathcal{C}_i$, I generate a correspondence prior in $\mathcal{C}_j$ (and vice versa), as explained in Sec. 6.6.2.1. Figure 6.25 shows examples of such correspondence priors for sequences `GUBo1616`, `GUBo1606` and `GUCar`. It is important to note that correspondence priors can also be generated for pixels which lie within i) static parts of the scene, but more importantly ii) for pixels which lie within homogeneous parts of the scene. This can be seen for the selected pixel number 2 in `GUBo1616` or pixel number 4 in `GUCar` in Fig. 6.25. Obviously, this is possible since the model learns a global transformation between the views.

The correspondence regions shown in Fig. 6.25 are the covariance error ellipses for a confidence level of 99% containing the true correspondence. If a pixel is not visible within both views, the correspondence prior exhibits high uncertainty, indicated by a large error ellipse, as both eigenvalues of the priors' covariance matrix would be large.

Finding Overlapping Regions As the approach returns correspondence priors with a confidence measure given by a covariance matrix, we can easily find regions visible within both views as follows: For all pixels of one of the two views we generate correspondence priors and store the sum of the eigenvalues of the associated covariance matrices. Observing that the correspondence prior will be the same for all pixels on the original scale which fall within the same pixel on the subsampled view, we do not have to compute $R \cdot C$ correspondence priors but only $\frac{R}{\gamma} \cdot \frac{C}{\gamma}$. This is simply the number of pixels of the subsampled view. Finally, we obtain a confidence map of the size of the original resolution, where low values indicate high confidence in that there is a corresponding pixel in the second view.

Figure 6.26 shows such confidence maps for all of the three camera setups. It can be seen that the proposed approach can find overlapping regions within binocular camera setups even when the views are heavily twisted and considerably differ in scale.

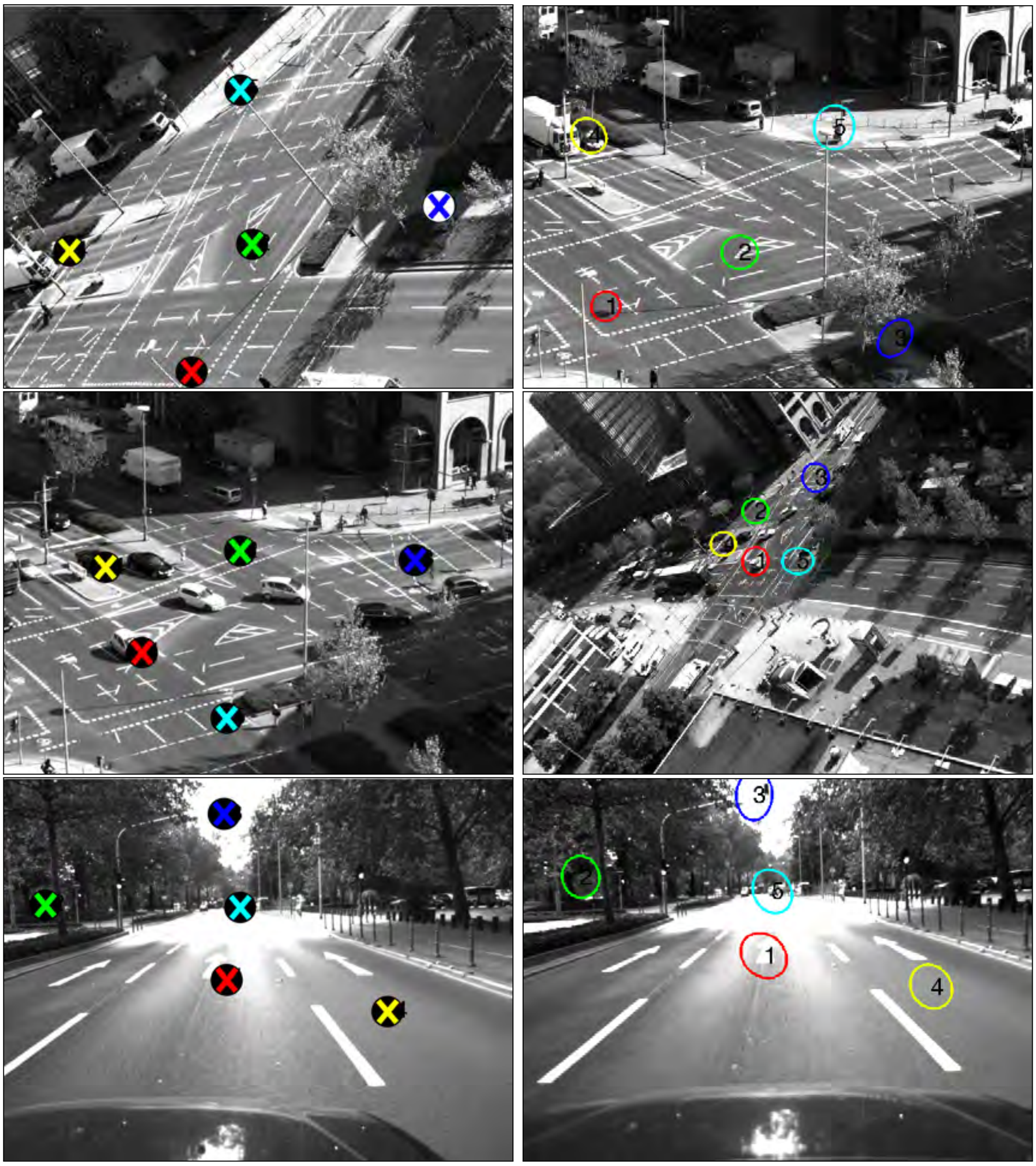


Figure 6.25: **Correspondence priors for real world setups:** Within each row for sequences `GUBo1616`, `GUBo1606` and `GUCar`: for selected pixels within the first view (left), regions of high probability containing the true corresponding pixel in second view (right) are determined based on the proposed approach. See Fig. 6.22 and text for details. Best viewed in colour.

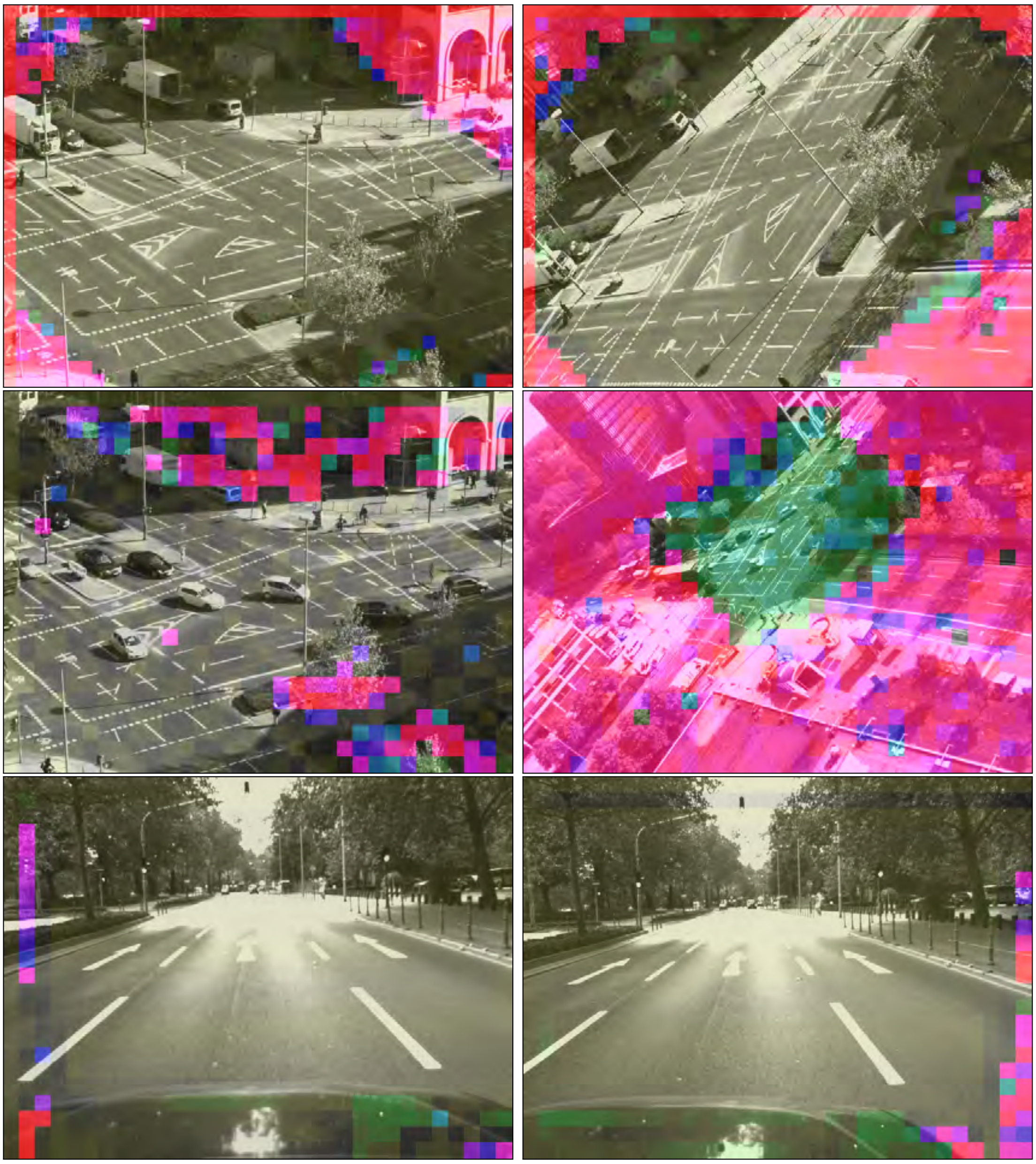


Figure 6.26: **Correspondence maps for real world setups:** Within each column for sequences `GUBo1616`, `GUBo1606` and `GUCar`: (left) left and (right) right view of the binocular camera setup overlaid with a confidence map encoding regions of high probability of being visible in both views. Regions coloured in shades of purple have a high probability of *not* being visible in the opposite view. See text for details. Best viewed in colour.

6.7 Learning Multiple Transformations

As has been discussed in Sec. 6.2, the standard CCA model can only learn and represent a single global transformation. If CCA is applied to a dataset containing multiple transformations, the CCA basis vectors turn into an invariant representation of the data under the given classes of transformations. This is shown exemplarily in Fig. 6.27. The basis vectors shown were obtained for a dataset containing multiple rotation and translation transformations, respectively. As can be seen, the filters develop a transformation invariant representation.

In the following, I present an EM/k-means style algorithm, which allows to learn multiple transformations based on a mixture of independent CCA models. Certainly, I am not the first to investigate CCA mixture models, e.g., in (Viinikanoja et al. 2010), a variational Bayesian CCA mixture model is presented. An algorithm similar to the ones presented by me is given in (Fern et al. 2005) for an application in the earth science community. However, their distance measure used to assign data points to CCA experts differs from mine, as will be discussed in the following.

6.7.1 Approach

Assume we are given two data matrices $\mathbf{X}, \mathbf{Y}$ of N vectorised training images. Each pair of images is assumed to be related by *one* out of K hidden transformations $\boldsymbol{F}_{\boldsymbol{xy}}^{k}, k = 1, .., K$, according to Eq. 6.6. My goal is to learn K independent CCA models, each of which is an expert in explaining transformations $\boldsymbol{F}_{\boldsymbol{yx}}^{k}$ and $\boldsymbol{F}_{\boldsymbol{xy}}^{k}$.

The n-th image pair is assigned to the expert minimising the expected squared error between the input images $\boldsymbol{x}_i$ and $\boldsymbol{y}_i$ and their predictions $\hat{\mathbf{x}}_i$ and $\hat{\mathbf{y}}_i$ according to:

$$\begin{aligned} k &= \underset{k}{argmin} ||\boldsymbol{x}_n - \hat{\mathbf{x}}_n||_2^2 + ||\boldsymbol{y}_n - \hat{\mathbf{y}}_n||_2^2, \\ &= \underset{k}{argmin} ||\boldsymbol{x}_n - \boldsymbol{F}_{\boldsymbol{xy}}^{k}\boldsymbol{y}_n||_2^2 + ||\boldsymbol{y}_n - \boldsymbol{F}_{\boldsymbol{yx}}^{k}\boldsymbol{x}_n||_2^2. \end{aligned} \tag{6.29}$$

Based on Eq. 6.29 we may now define the energy function to be minimised as:

$$\mathbf{Q} = \sum_n \sum_k \gamma_{nk} \left\{ ||\boldsymbol{x}_n - \boldsymbol{F}_{\boldsymbol{xy}}^{k}\boldsymbol{y}_n||_2^2 + ||\boldsymbol{y}_n - \boldsymbol{F}_{\boldsymbol{yx}}^{k}\boldsymbol{x}_n||_2^2 \right\}. \tag{6.30}$$

Here, γ_{nk} is a binary variable which is 1 iff the n-th image pair is assigned to expert k. Transformations $\boldsymbol{F}_{\boldsymbol{xy}}^{k}$ and $\boldsymbol{F}_{\boldsymbol{yx}}^{k}$ are determined as the MMSE estimators given in Eq. 6.25 and Eq. 6.26. This is in contrast to the work by (ibid.), where per class linear regression models are determined in canonical coordinates to perform the assignment.

We may minimise Eq. 6.30 based on the usual k-means style iterative scheme as follows (cf. (Bishop 2006)). Initially, all image pairs are assigned randomly to one out of the k experts. Then, in the first phase, each of the k CCA models is trained on its

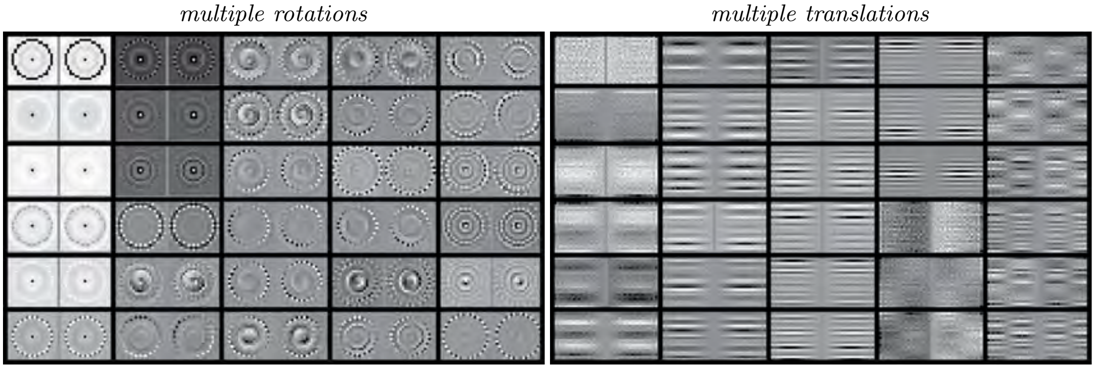


Figure 6.27: **Transformation invariant basis:** CCA basis vectors obtained for data sets containing multiple transformations. See text for details.

assigned image pairs. In the second phase, each image pair is reassigned to the expert which minimises Eq. 6.30. This scheme is repeated until i) no image pairs are reassigned or ii) the energy function does not further decreases than a suitable small threshold ϵ. Clearly, the estimated minimum is likely to be a local minimum, as the optimisation scheme does not guarantee to find the global minimum.

In Fig. 6.28, Fig. 6.29 and Fig. 6.30, I show results for splitting various transformations. It can be seen that the proposed method is able to extract the different classes of transformations. I have found that the scheme is sensitive to the initial assignment of image pairs to CCA experts, and the splitting not always succeeds (=weak local minimum). However, it remains future work to explore the splitting of multiple transformations via CCA in more detail.

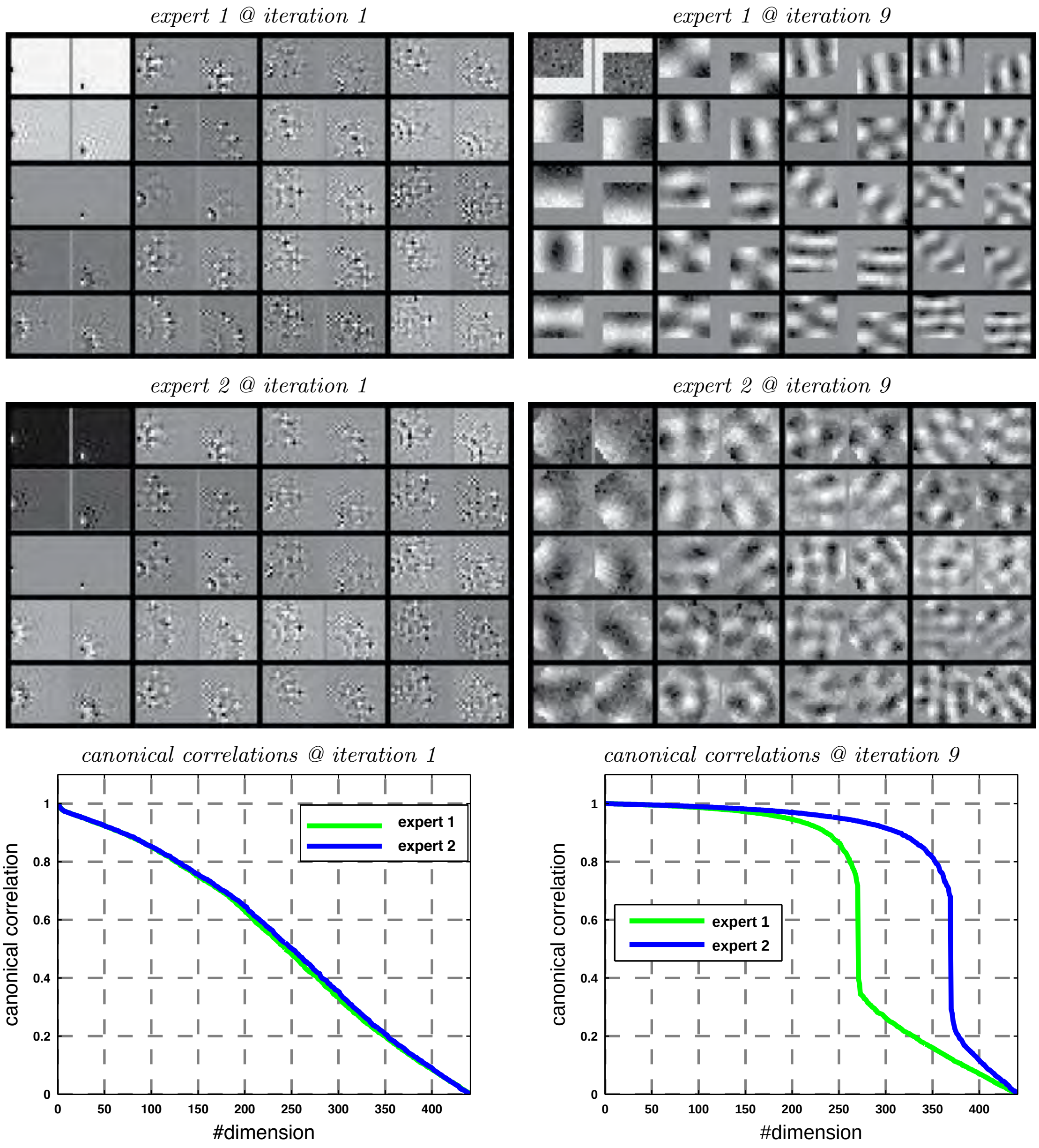


Figure 6.28: **Splitting transformations:** Splitting a translation and rotation transformation

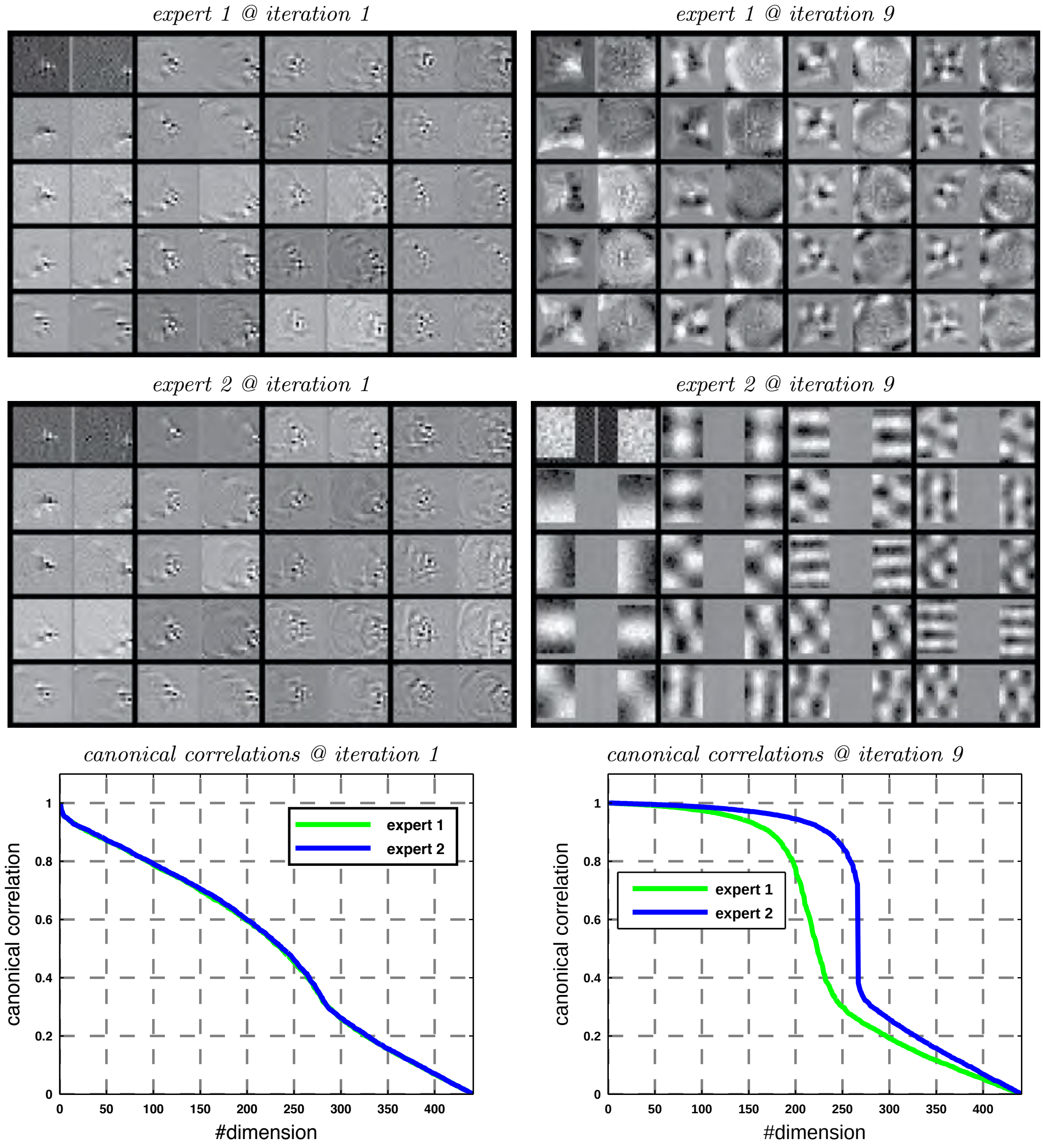


Figure 6.29: **Splitting transformations:** Splitting a translation and a nonlinear transformation.

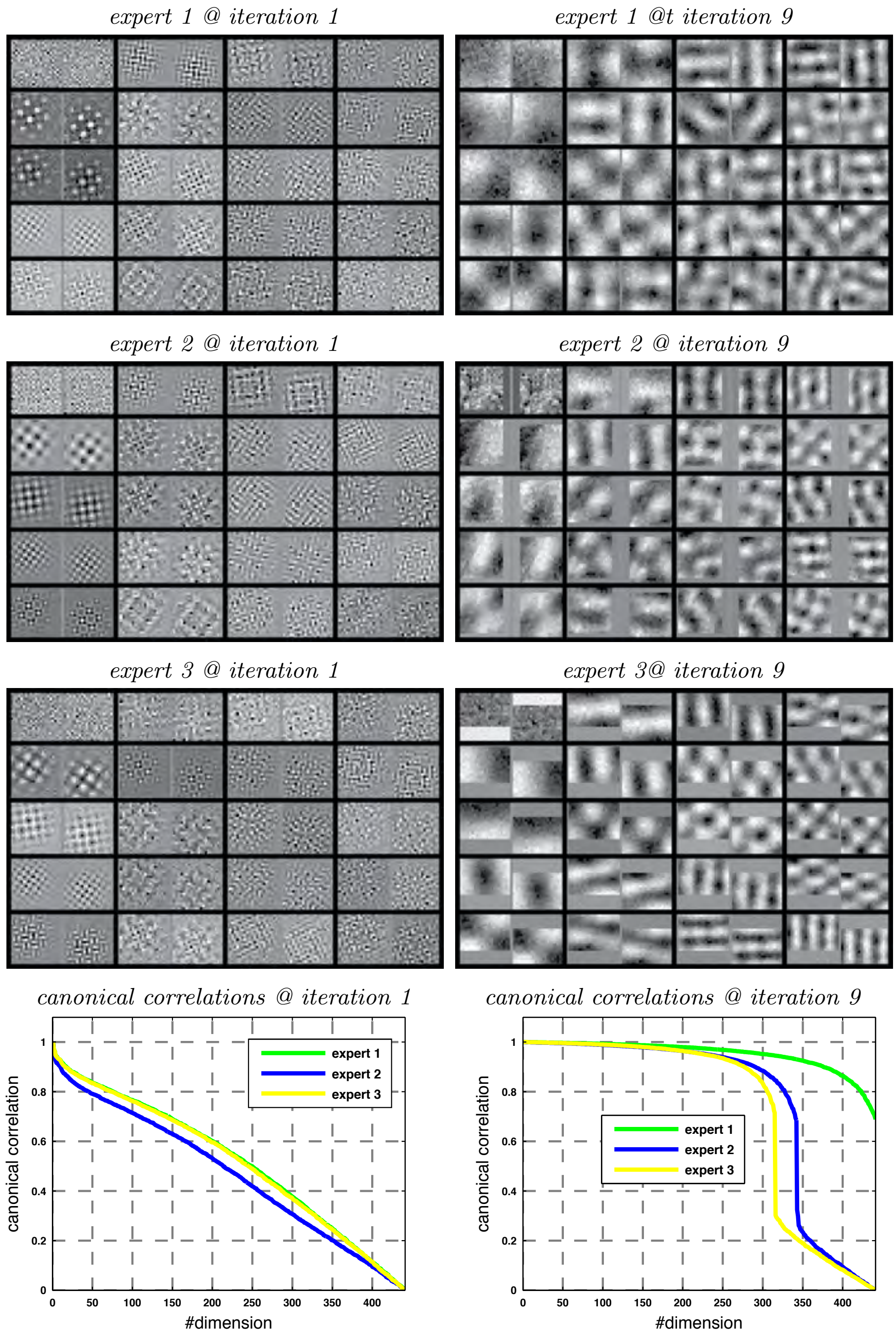


Figure 6.30: **Splitting transformations:** Splitting two translations and one rotation transformation.

6.8 Summary and Conclusion

I presented a feature free approach to learn latent global mappings between pairs of views, which is based on the method of *Canonical Correlation Analysis* (CCA). Mappings are learnt and represented by means of two sets of basis vectors. In contrast to a classic parametric approach, in which parameters of a transformation are estimated, say, a rotation angle, I learn *the* transformation itself.

I assume that the training image pairs are related by a single fixed transformation. Only then the CCA representation will encode the hidden transformation. If the training data contains multiple different transformations (of the same class, say, rotations) the CCA representation turns into a transformation invariant representation, i.e., the eigenfunctions of the transformation operator. Based on a straight forward mixture model of CCA experts, I showed that multiple transformations contained within a single training data set may be split and learnt.

A learnt transformation can be applied to previously unseen data via a MMSE estimator. Applying a learnt transformation in this way is a linear operation. However, the basis vectors obtained by CCA may encode arbitrary nonlinear transformations.

Experimental results validate the presented approach and show that rather simple CCA-type processing structures are able to learn and represent general, nonlinear spatial transformations in an unsupervised manner.

7 Summary and Outlook

7.1 Summary

As we have seen, the correspondence problem is at the core of many low-level computer vision tasks and is a precursor to high-level processes. The correspondence problem may be studied locally or globally. In a local approach, individual pixel correspondences are sought, while in a global approach a holistic mapping function is sought. In this thesis, I introduced two approaches to learn local and global correspondence relations in an unsupervised manner.

In the first part of the thesis, I proposed the method of *Temporal Coincidence Analysis* (TCA) to learn pixel correspondences in binocular camera setups. The classic approach is to compute pixel correspondences in two images, based on the spatial feature matching pipeline. This pipeline consists of three stages: i) the detection of spatial keypoints in both images ii) their description by means of a descriptor and iii) the matching of keypoints across the views. In contrast, in TCA I only regard the temporal information of single pixels. Based on the detection and matching of temporal events, I estimate a correspondence distribution over a long image stream. Correspondences are never computed explicitly, only the evidence for a correspondence relation by means of matched events is collected over time. As we have seen, TCA is theoretically justified and is not a heuristic; the learning scheme is derived as a statistical model for matching pairs of independent signal channels. In this model, the basic principle of detecting and matching temporal events in the grey value signal is cast as a temporal update scheme of an associated posterior distribution. The posterior distribution may also be approximated by replacing the posterior update with a simple threshold test. The correspondence distribution then turns into and an accumulator array.

A pixel correspondence in the classic sense and the correspondence distribution which my method learns are identical, given that the scene depth structure is approximately static. If the scene depth varies, the correspondence may vary along the epipolar line. While TCA has no build in knowledge about the epipolar geometry, the correspondence distribution will encode parts of the epipolar ray, given that the scene depth is regularly sampled.

I encode a correspondence distribution by means of a set of attributes. Among them, the distribution's first and second order statistics and its entropy. These attributes allow to monitor the progress of the learning process and to assess the uncertainty about the correspondence estimate. These are important entities and allow to propagate uncertainties to higher level processes which operate on the learnt correspondences.

As we have seen, TCA builds on the detection and matching of temporal events, which are determined from the grey value signal of single pixels. As in any appearance based approach, matching of events across different cameras may suffer from illumination differences, different camera transfer functions etc.. I therefore model the *Grey Value Transfer Function* (GVTF) which maps grey values observed in a camera $\mathcal{C}_i$ to its corresponding grey value in a camera $\mathcal{C}_j$. I proposed a method to learn the GVTF from a set of learnt correspondences, which is robust to spatial uncertainties in the correspondence estimates. To this end, a comparagram, i.e., a 2-D histogram of grey values, extracted from corresponding pixels is build. As has been shown, it is important to only add those grey value pairs to the comparagram for which the local signal around the regarded pixels is rather homogeneous. Otherwise, even small inaccuracies in the correspondence estimate may lead to severe outliers in the comparagram. The GVTF is then estimated as a low order polynomial fitted to the data within the comparagram.

I demonstrated the applicability of TCA in a series of simulations as well as for real world camera setups, both for stereo and motion correspondences. Regarding the stereo case, it was shown that TCA can handle setups in which the camera views are translated and/or rotated w.r.t. each other or show a large scale difference. The cameras may be static or moving. The only assumption that is made is that the relative orientation of the cameras is fixed and that the video streams are synchronised.

I showed that TCA can also learn the distribution of motion correspondences in monocular video streams without computing the optical flow. I apply TCA as for the stereo case without changing the learning process in any way. Experimental results validate that TCA can learn motion correspondences for various camera setups, including static and moving cameras. I demonstrated that a sparse set of learnt average motion vectors can be interpolated by means of fitting the parameters of a bi-quadratic model to the set of learnt average motion vectors. The scheme takes the uncertainties encoded in the learnt correspondence distributions explicitly into account.

Besides average motion correspondences, TCA can also be used to infer global hidden motion variables, e.g., the yaw rate of a moving camera. To this end, I pool the set of matched events over many pixels lying approximately at infinity. The matched events will then form a cluster, encoding the true yaw rate. Furthermore, I introduced a mixture model of TCA experts, which is able to learn the characteristic motion maps observed by a moving camera during left, right and forward motion.

I also showed that TCA is naturally applied in a coarse to fine manner on a, say, Gaussian pyramid of the input video streams. Correspondence distributions are then learnt on a coarse scale, allowing to limit the spatial extend of the distribution on a fine scale. A correspondence learnt on a coarse scale can then be reprojected on the fine scale to form a correspondence prior.

In the second part of the thesis, I presented an approach to learn global image transformations in an unsupervised manner. In a classic approach to determine global transformations, usually the parameters of a specific class of transformations, say, rotations are estimated based on a given set of spatial correspondences (features). A prominent class of parametric models is given by the projective linear group. In contrast to this, I present a feature free approach in which a transformation is not encoded by means of a parametric model but by a mapping tensor which may represent arbitrary nonlinear transformations. The approach is based on a method known as *Canonical Correlation Analysis* (CCA). Given a large set of image pairs, which are related by a single fixed transformation, I showed that CCA extracts pairs of basis vectors which encode the sought transformation implicitly. The learnt transformation can be applied to previously unseen data based on a MMSE estimator. While this is a linear operation, the learnt basis vectors may encode arbitrary nonlinear transformations.

A limitation of CCA is that only single transformations may be learnt. Due to the lack of additional latent variables, CCA may not switch between several different transformations. If the given training data contains multiple transformations of the same class, the basis vectors extracted by CCA constitute an invariant basis for the respective transformation operator.

A further limitation of CCA is that it is not possible to directly infer the shared signal, i.e., the parts of the signal which are visible within both views irrespective of the transformation. In order to extract the spatial footprint of the shared signal, I determine the summed energy filters.

I also demonstrated that CCA can be used to generate correspondence priors in real-world binocular camera setups. To this end, I learn a global transformation between the input views on a coarse scale, apply the learnt transformation to a binary image marking the regarded pixel, apply the learnt transformation to the binary image and finally reproject the predicted image to the fine scale of the second view.

To conclude, in this thesis I presented two learning based approaches addressing the important correspondence problem in computer vision. Both approaches are unsupervised and show that local and global correspondence relations can be *learnt* instead of being *computed*.

7.2 Outlook

The methods presented in this thesis show that local and global correspondence relations can be learnt unsupervised. My work somewhat contrasts the still lasting trend in computer vision research to regard specific problems in a rather isolated setting based on single images or short sequences. As an example, regard the Middlebury benchmark (Baker et al. 2011). This is the de facto standard to benchmark stereo and optical flow algorithms but neglects temporal information as it only operates on pairs of images.

However, in multi-camera scenarios, e.g., in surveillance networks, in driver assistance systems or robotics applications, very long image sequences are available. While we may apply pure spatial approaches separately to each image pair in the image stream, it seems to be a waste of information and computational resources not to make use of the temporal regularities of natural video sequences. It should be noted that there are benchmark suites with rather long image sequences compared to the Middlebury benchmark, e.g., the `KITTI` benchmark suite (Geiger et al. 2012) (sequences with up to several thousand frames) which contains renowned benchmark data for algorithms in the area of driver assistance systems.

Clearly, there is a large compound of work in the field of spatiotemporal image analysis. However, I believe that processing of say, hours or days of video data makes it necessary to develop new processing methodologies.

Of special interest are learning based approaches which are able to autonomously adapt to the scenarios in which they are applied. This allows to build large scale systems which simply cannot be parameterised due to their complexity. The approaches developed in this thesis are a step towards the direction of autonomous learning systems. I showed that correspondence relations can be learnt unsupervised, and correspondence relations are the basis to many other vision problems.

Clearly, the methods proposed in this thesis can be explored further and can be extended in many ways. Foremost, I believe that the combination of TCA with state-of-the-art spatial feature based approaches is highly attractive. TCA could be used as a prior generator, which learns the scene specific correspondence relations, which are used by a spatial approach to constrain the search space.

In a robotics or automotive application, it would be interesting to combine the learning of average motion maps with the robot's/car's ego-motion data. This allows to build a model based on which we could predict the expected optical flow or the platform motion. In turn, this allows to detect abnormal motion.

Further extensions of TCA include, among others, the automatic placement of seed pixels, the automatic selection of the source signal noise level, the automatic selection of the event threshold.

The mixture model of CCA experts could be extended and further analysed in order to learn multiple global transformations at the same time.

Appendix A

Video Sequences Overview

In this thesis, I will present results for various different image sequence data, which is shortly presented here.

A.0.1 Binocular Sequences

See Figs. A.1 and A.3 for sample frames of the following sequences.

Sequence `GUCar`

- **Setup:** Moving cameras, mounted on top of a car. Baseline of roughly 30 cm.
- **Camera settings:** Uncalibrated views but same focal lengths, resolution of 640×480 pixels @ 30 FPS, $\approx$ 50,000 frames in total.
- **Scene content:** The car drives through an urban environment and on a highway.
- **Source:** Recorded by our group.

Sequence `GUBo1616`

- **Setup:** Static cameras on a roof top. Cameras are rotated w.r.t. each other.
- **Camera settings:** Uncalibrated views with same focal lengths. Resolution of 640×480 pixels @ 30 FPS, $\approx$ 12,000 frames in total.
- **Scene content:** Cameras observe a traffic junction.
- **Source:** Recorded by our group.

Sequence `GUBo1606`

- **Setup:** Static cameras on a roof top. Cameras are rotated w.r.t. each other, large scale difference.

- **Camera settings:** Uncalibrated views with different focal lengths. Resolution of 640 × 480 pixels @ 30 FPS, ≈ 12,000 frames in total.
- **Scene content:** Cameras observe a traffic junction.
- **Source:** Recorded by our group.

Sequence `GUOmni`

- **Setup:** Moving cameras mounted on a robot. Cameras are rotated w.r.t. each other.
- **Camera settings:** Uncalibrated views with fisheye optics. Resolution of 640×480 pixels @ 30 FPS, ≈ 30,000 frames in total.
- **Scene content:** The robot drives through an office environment.
- **Source:** Recorded by our group.

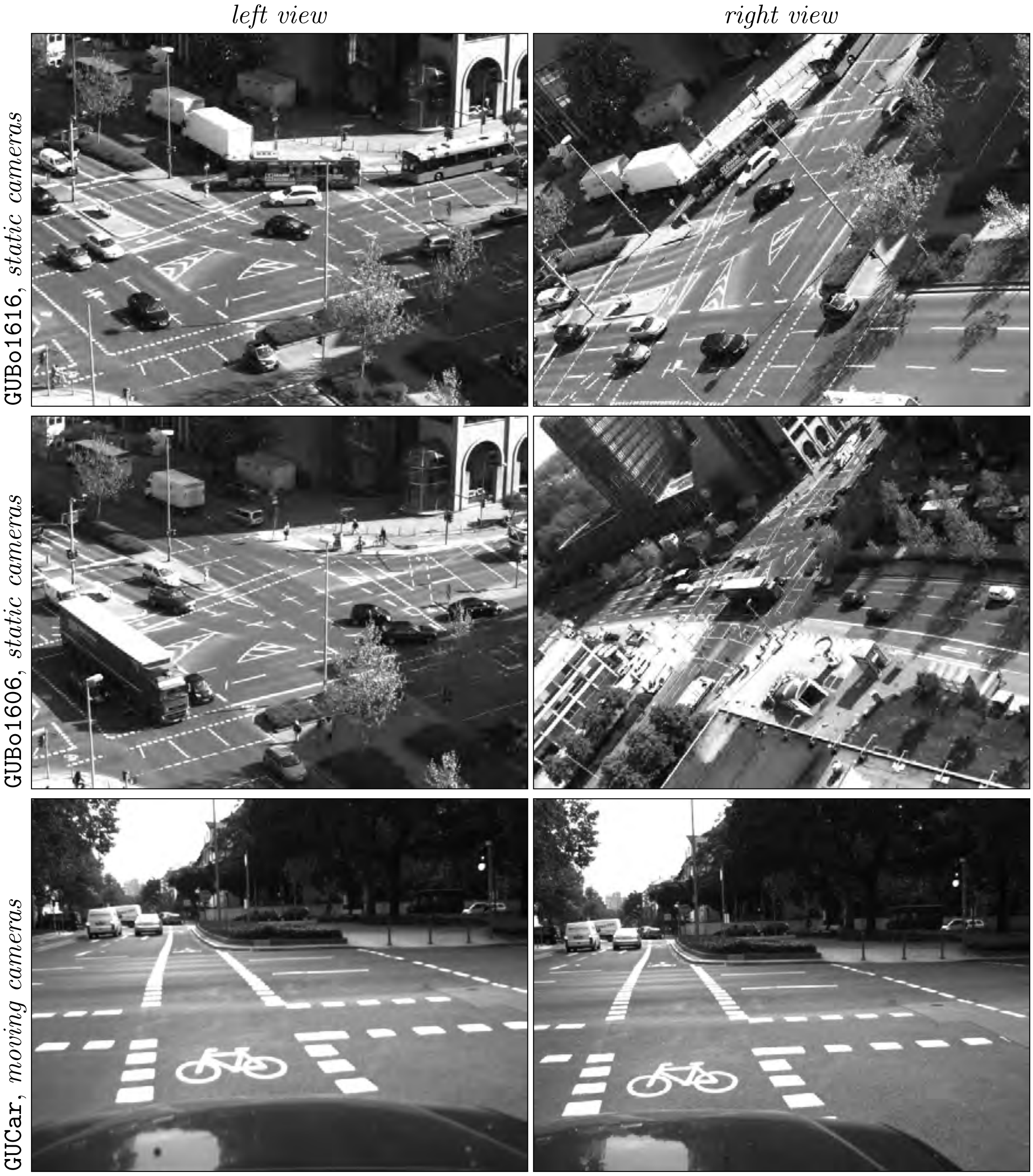

left view
right view
`GUBo1616`, *static cameras*
`GUBo1606`, *static cameras*
`GUCar`, *moving cameras*

Figure A.1: **Exemplary frames for binocular sequences:** These sequences are used within Sec. 4 and Sec. 5. See text for details.

A.0.2 Monocular Sequences

See Fig. A.2 for sample frames of the following sequences.

Sequence `Forrest`

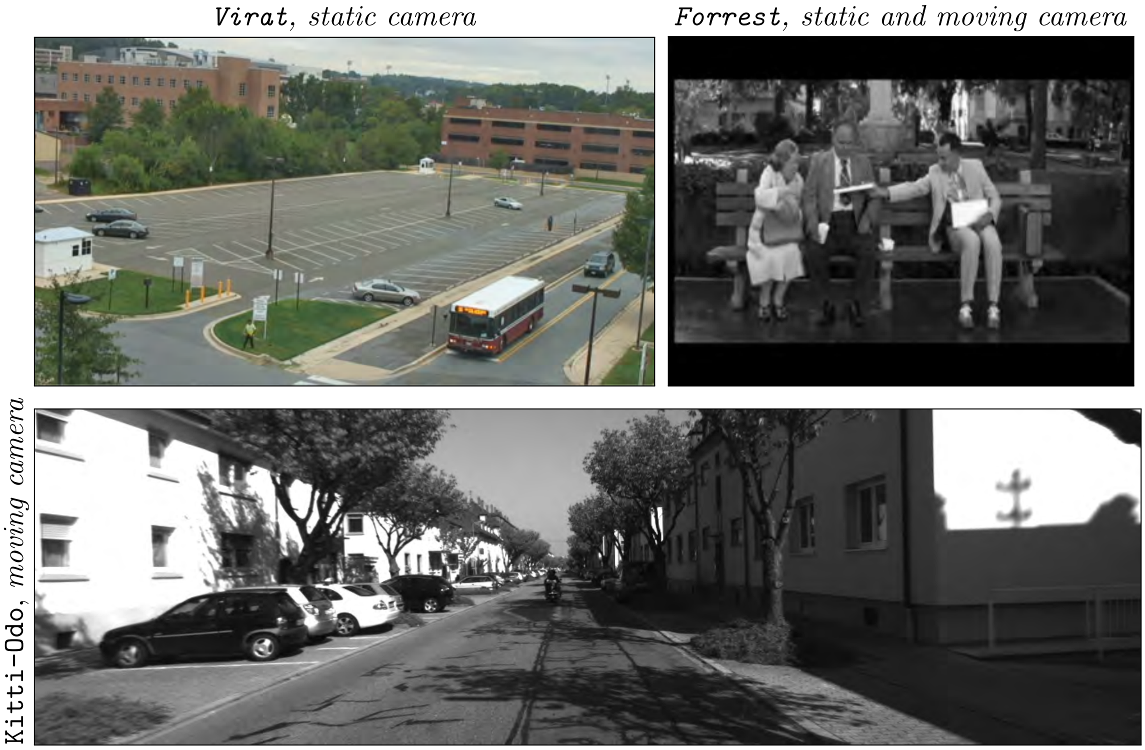
- **Setup:** Static and moving camera.
- **Camera settings:** Uncalibrated view. Resolution of 1440 × 1080 pixels @ 24 FPS, ≈ 5,000 frames in total.
- **Scene content:** Trailer of the **Forrest Gump** movie.
- **Source:** ©Paramount Pictures.

Sequence `Kitti-Odo`

- **Setup:** Moving camera mounted on a car.
- **Camera settings:** Calibration data available but not used. Resolution of 1241 × 376 pixels @ 30 FPS, ≈ 4,500 frames in total
- **Scene content:** Car drives through an urban environment.
- **Source:** KITTI Benchmark (ibid.).

Sequence `Virat`

- **Setup:** Static camera mounted on a roof top.
- **Camera settings:** Calibration data available but not used. Resolution of 1281 × 789 pixels @ 25 FPS, ≈ 5,000 frames in total.
- **Scene content:** Camera observes a parking lot.
- **Source:** VIRAT Benchmark (Oh et al. 2011).

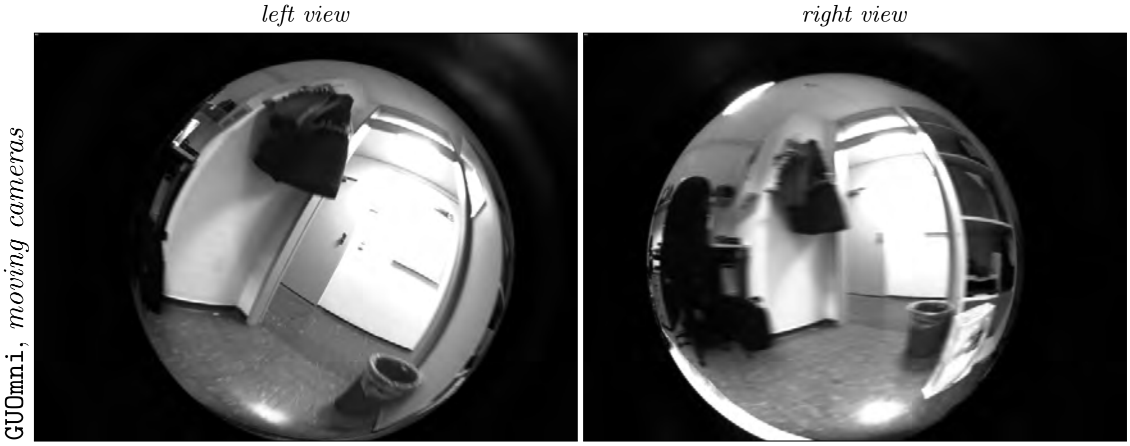


`Virat`, *static camera*

`Forrest`, *static and moving camera*

`Kitti-Odo`, *moving camera*

Figure A.2: **Exemplary frames for monocular sequences:** These sequences are used within Sec. 4 and Sec. 5. See text for details.



left view

right view

`GUOmni`, *moving cameras*

Figure A.3: **Exemplary frames for binocular sequences:** This sequence is used within Sec. 5. See text for details.

Bibliography

Anderson, Theodore. *An introduction to multivariate statistical analysis.* Wiley, 1958 (cit. on pp. 177 sq.).

Andrew, Galen, Raman Arora, Jeff Bilmes, and Karen Livescu. „Deep Canonical Correlation Analysis“. In: *International Conference on Machine Learning* (2013) (cit. on p. 173).

ATT-Laboratories-Cambridge. *Olivetti Face Dataset* (cit. on p. 197).

Bach, Francis and Michael Jordan. *A probabilistic interpretation of canonical correlation analysis.* Tech. rep. Department of Statistics, University of California, Berkeley, 2005 (cit. on p. 173).

Baker, Simon, Daniel Scharstein, John Lewis, Stefan Roth, Michael Black, and Richard Szeliski. „A database and evaluation methodology for optical flow“. In: *International Journal of Computer Vision* (2011) (cit. on pp. 13 sq., 106, 135 sq., 143 sq., 148, 151, 153, 217).

Ballard, Dana H. „Generalizing the Hough transform to detect arbitrary shapes“. In: *Pattern recognition* 13.2 (1981), pp. 111–122 (cit. on p. 35).

Ballard, Dana and Christopher Brown. *Computer vision.* Prentice Hall, 1982 (cit. on p. 143).

Barnett, Tim and Rudolph Preisendorfer. „Origins and levels of monthly and seasonal forecast skill for United States surface air temperatures determined by canonical correlation analysis“. In: *Monthly Weather Review* 115 (1987), pp. 1825–1850 (cit. on p. 172).

Bay, Herbert, Tinne Tuytelaars, and Luc Van Gool. „Surf: Speeded up robust features“. In: *European Conference on Computer Vision.* Springer, 2006, pp. 404–417 (cit. on p. 17).

Benosman, Ryad, Sio Hoi Ieng, Paul Rogister, and Christoph Posch. „Asynchronous event-based Hebbian epipolar geometry“. In: *Neural Networks* 22.11 (2011), pp. 1723–1734 (cit. on p. 35).

Bethge, Matthias, Sebastian Gerwinn, and Jakob H Macke. „Unsupervised learning of a steerable basis for invariant image representations“. In: *Human Vision and Electronic Imaging*. SPIE. 2007, pp. 64920C–64920C (cit. on p. 171).

Bigun, Josef. *Vision with direction*. Springer, 2006 (cit. on p. 82).

Bishop, Christopher. *Pattern recognition and machine learning*. Springer, 2006 (cit. on pp. 49, 209).

Borga, Magnus. „Learning multidimensional signal processing“. PhD thesis. Linköping University, 1998 (cit. on pp. 173, 177).

Borga, Magnus and Hans Knutsson. „Estimating multiple depths in semi-transparent stereo images“. In: *Scandinavian Conference on Image Analysis*. Vol. 1. 1999, pp. 127–134 (cit. on p. 172).

Brooks, Michael, Wojciech Chojnacki, Darren Gawley, and Anton van den Hengel. „What value covariance information in estimating vision parameters?“ In: *International Conference on Computer Vision*. Vol. 1. IEEE. 2001, pp. 302–308 (cit. on p. 93).

Burt, Peter and Edward Adelson. „The Laplacian pyramid as a compact image code“. In: *Communications* 31.4 (1983), pp. 532–540 (cit. on p. 90).

Calonder, Michael, Vincent Lepetit, Christoph Strecha, and Pascal Fua. „Brief: Binary robust independent elementary features“. In: *European Conference on Computer Vision*. Springer, 2010, pp. 778–792 (cit. on p. 17).

Carneiro, João, Sio-Hoi Ieng, Christoph Posch, and Ryad Benosman. „Event-based 3D reconstruction from neuromorphic retinas“. In: *Neural Networks* 45 (2013), pp. 27–38 (cit. on p. 35).

Cheng, Eric and Massimo Piccardi. „Matching of objects moving across disjoint cameras“. In: *International Conference on Image Processing*. IEEE. IEEE, 2006, pp. 1769–1772 (cit. on p. 77).

Clarot, Pierre, Erhan B Ermis, P-M Jodoin, and Venkatesh Saligrama. „Unsupervised camera network structure estimation based on activity“. In: *International Conference on Distributed Smart Cameras*. IEEE, 2009, pp. 1–8 (cit. on p. 37).

Coates, Adam, Honglak Lee, and Andrew Ng. „An analysis of single-layer networks in unsupervised feature learning“. In: *International Conference on Artificial Intelligence and Statistics* (2011) (cit. on p. 136).

Conrad, Christian, Alvaro Guevara, and Rudolf Mester. „Learning multi-view correspondences from temporal coincidences“. In: *Computer Vision and Pattern Recognition Workshops.* IEEE, 2011 (cit. on p. 5).

Conrad, Christian, Matthias Mertz, and Rudolf Mester. „Contour-Relaxed Superpixels“. In: *Energy Minimization Methods in Computer Vision and Pattern Recognition.* Springer. 2013, pp. 280–293 (cit. on p. 5).

Conrad, Christian and Rudolf Mester. „Learning geometrical transforms between multi camera views using Canonical Correlation Analysis“. In: *British Machine Vision Conference.* 2012 (cit. on p. 5).

— „Learning Multi-view Correspondences via Subspace-Based Temporal Coincidences“. In: *Scandinavian Conference on Image Analysis.* Springer, 2013, pp. 456–467 (cit. on p. 5).

— „Learning Rank Reduced Mappings using Canonical Correlation Analysis.“ In: *Statistical Signal Processing Workshop.* IEEE, 2016 (cit. on p. 5).

— „Learning Relative Photometric Differences of Pairs of Cameras.“ In: *Advanced Video and Signal based Surveillance.* IEEE, 2015 (cit. on p. 5).

Cover, Thomas and Joy Thomas. *Elements of information theory.* 2nd ed. John Wiley & Sons, 2009 (cit. on pp. 46, 73).

Criminisi, Antonio, Andrew Blake, Carsten Rother, Jamie Shotton, and Philip Torr. „Efficient dense stereo with occlusions for new view-synthesis by four-state dynamic programming“. In: *International Journal of Computer Vision* 71.1 (2007), pp. 89–110 (cit. on p. 75).

Davis, John. *Combining error ellipses.* Tech. rep. Harvard CXC memo, 2007 (cit. on p. 74).

Dederscheck, David, Thomas Müller, and Rudolf Mester. „Illumination invariance for driving scene optical flow using comparagram preselection“. In: *Intelligent Vehicles Symposium.* IEEE, 2012, pp. 742–747 (cit. on pp. 75, 77).

Delbrück, Tobi, Bernabé Linares-Barranco, Eugenio Culurciello, and Christoph Posch. „Activity-driven, event-based vision sensors“. In: *International Symposium on Circuits and Systems.* IEEE, 2010, pp. 2426–2429 (cit. on p. 34).

Dhillon, Paramveer, Dean Foster, and Lyle Ungar. „Multi-View Learning of Word Embeddings via CCA.“ In: *Neural Information Processing Systems.* Vol. 24. 2011, pp. 199–207 (cit. on p. 173).

Dickmanns, Ernst Dieter, Reinhold Behringer, Dirk Dickmanns, Thomas Hildebrandt, Markus Maurer, Frank Thomanek, and Joachim Schiehlen. „The seeing passenger car'VaMoRs-P'“. In: *Intelligent Vehicles.* IEEE. 1994, pp. 68–73 (cit. on p. 1).

Donner, Rene, Michael Reiter, Georg Langs, Philipp Peloschek, and Horst Bischof. „Fast active appearance model search using canonical correlation analysis“. In: *Pattern Analysis and Machine Intelligence* (2006) (cit. on p. 173).

Duda, Richard and Peter Hart. „Use of the Hough transformation to detect lines and curves in pictures“. In: *Communications* 15.1 (1972), pp. 11–15 (cit. on p. 35).

Eisenbach, Jens, Christian Conrad, and Rudolf Mester. „A Temporal Scheme for Fast Learning of Image-Patch Correspondences in Realistic Multi-camera Setups“. In: *Computer Vision and Pattern Recognition Workshops.* IEEE, 2013 (cit. on p. 5).

Eisenbach, Jens, Matthias Mertz, Christian Conrad, and Rudolf Mester. „Reducing camera vibrations and photometric changes in surveillance video“. In: *International Conference on Advanced Video and Signal Based Surveillance.* IEEE. 2013, pp. 69–74 (cit. on p. 159).

Ellis, Tim, Dimitrios Makris, and James Black. „Learning a multi-camera topology“. In: *International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance.* IEEE, 2003, pp. 165–171 (cit. on p. 18).

Ermis, Erhan. „Activity based geometry independent features for information processing in heterogeneous camera networks“. PhD thesis. Boston University, 2010 (cit. on pp. 35 sq.).

Ermis, Erhan, Venkatesh Saligrama, Pierre Jodoin, and Janusz Konrad. „Abnormal behavior detection and behavior matching for networked cameras“. In: *International Conference on Distributed Smart Cameras.* IEEE. 2008, pp. 1–10 (cit. on p. 35).

Felsberg, Michael, Fredrik Larsson, Johan Wiklund, Niclas Wadstromer, and Jörgen Ahlberg. „Online learning of correspondences between images“. In: *Pattern Analysis and Machine Intelligence* 35.1 (2013), pp. 118–129 (cit. on p. 34).

Fern, Xiaoli, Carla Brodley, and Mark Friedl. „Correlation clustering for learning mixtures of canonical correlation models“. In: *International conference on data mining*. Vol. 119. Society for Industrial Mathematics. SIAM, 2005, p. 439 (cit. on p. 209).

Fischler, Martin and Robert Bolles. „Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography“. In: *Communications* 24.6 (1981), pp. 381–395 (cit. on pp. 11, 36).

Förstner, Wolfgang. „A feature based correspondence algorithm for image matching“. In: *International Archives of Photogrammetry and Remote Sensing* 26.3 (1986), pp. 150–166 (cit. on p. 16).

— *Statistische Verfahren für die automatische Bildanalyse und ihre Bewertung bei der Objekterkennung und-vermessung*. Beck, 1991 (cit. on pp. 2, 16, 70, 72, 81).

Förstner, Wolfgang and Eberhard Gülch. „A fast operator for detection and precise location of distinct points, corners and centres of circular features“. In: *Intercommission Workshop on Fast Processing of Photogrammetric Data*. ISPRS, 1987, pp. 281–305 (cit. on p. 15).

Forsyth, David and Jean Ponce. *Computer vision: a modern approach*. Prentice Hall, 2002 (cit. on p. 7).

Franke, Uwe, Clemens Rabe, Hernán Badino, and Stefan Gehrig. „6d-vision: Fusion of stereo and motion for robust environment perception“. In: *Pattern Recognition*. Springer, 2005, pp. 216–223 (cit. on p. 17).

Fyfe, Colin and Pei Ling Lai. „ICA using kernel canonical correlation analysis“. In: *International Workshop on Independent Component Analysis and Blind Signal Separation*. 2000 (cit. on p. 173).

Gall, Juergen, Angela Yao, Nima Razavi, Luc Van Gool, and Victor Lempitsky. „Hough forests for object detection, tracking, and action recognition“. In: *Pattern Analysis and Machine Intelligence* 33.11 (2011), pp. 2188–2202 (cit. on p. 35).

Geiger, Andreas, Philip Lenz, and Raquel Urtasun. „Are we ready for autonomous driving? The KITTI vision benchmark suite“. In: *Computer Vision and Pattern Recognition*. 2012 (cit. on pp. 218, 222).

Golub, Gene and Charles van Loan. *Matrix computations*. Vol. 3. Johns Hopkins University Press, 2012 (cit. on pp. 11, 73, 82).

Grimson, Eric, Chris Stauffer, Raquel Romano, and Lily Lee. „Using adaptive tracking to classify and monitor activities in a site“. In: *Computer Vision and Pattern Recognition.* IEEE, 1998, pp. 22–29 (cit. on p. 136).

Grossberg, Michael and Shree Nayar. „Determining the camera response from images: What is knowable?“ In: *Pattern Analysis and Machine Intelligence* 25.11 (2003), pp. 1455–1467 (cit. on p. 77).

— „What can be known about the radiometric response from images?“ In: *European Conference on Computer Vision.* Springer, 2002, pp. 393–413 (cit. on p. 77).

Guevara, Alvaro, Christian Conrad, and Rudolf Mester. „Boosting segmentation results by contour relaxation“. In: *International Conference on Image Processing.* IEEE, 2011, pp. 1405–1408 (cit. on p. 5).

— „Curvature oriented clustering of sparse motion vector fields“. In: *Southwest Symposium on Image Analysis and Interpretation.* IEEE, 2012, pp. 161–164 (cit. on pp. 5, 139).

Hafner, David, Oliver Demetz, and Joachim Weickert. *Why Is the Census Transform Good for Robust Optic Flow Computation?* Springer, 2013 (cit. on p. 76).

Haralick, Robert. „Propagating covariance in computer vision“. In: *Performance Characterization in Computer Vision.* Springer, 2000, pp. 95–114 (cit. on p. 93).

Hardoon, David, Sandor Szedmak, and John Shawe-Taylor. „Canonical correlation analysis: An overview with application to learning methods“. In: *Neural Computation* 16.12 (2004), pp. 2639–2664 (cit. on p. 173).

Harris, Chris and Mike Stephens. „A combined corner and edge detector.“ In: *Alvey Vision Conference.* Vol. 15. 1988, p. 50 (cit. on pp. 15, 81).

Hartley, Richard and Andrew Zisserman. *Multiple view geometry in computer vision.* Vol. 2. Cambridge University Press, 2004 (cit. on pp. 7, 10 sq., 21, 126, 167 sqq.).

Herdtweck, Christian and Cristobal Curio. „Monocular heading estimation in non-stationary urban environment“. In: *Conference on Multisensor Fusion and Integration for Intelligent Systems.* IEEE, 2012 (cit. on p. 136).

Hirschmüller, Heiko and Daniel Scharstein. „Evaluation of stereo matching costs on images with radiometric differences“. In: *Pattern Analysis and Machine Intelligence* 31.9 (2009), pp. 1582–1599 (cit. on p. 76).

Hoover, Wayne. *Algorithms For Confidence Circles and Ellipses.* US Department of Commerce, National Oceanic and Atmospheric Administration, National Ocean Service, 1984 (cit. on p. 74).

Hordley, Steven. „Scene illuminant estimation: past, present, and future“. In: *Color Research & Application* 31.4 (2006), pp. 303–314 (cit. on p. 77).

Horn, Berthold and Brian Schunck. „Determining optical flow“. In: *Artificial intelligence* 17.1 (1981), pp. 185–203 (cit. on p. 135).

Hotelling, Harold. „Relations between two sets of variates“. In: *Biometrika* (1936) (cit. on pp. viii, 4, 167, 171).

Hough, Paul. *Method and means for recognizing complex patterns.* US Patent 3,069,654. 1962 (cit. on p. 35).

Hu, Min, Saad Ali, and Mubarak Shah. „Learning motion patterns in crowded scenes using motion flow field.“ In: *International Conference on Pattern Recognition.* 2008, pp. 1–5 (cit. on p. 136).

Humenberger, Martin, Stephan Schraml, Christoph Sulzbachner, Ahmed Nabil Belbachir, Agoston Srp, and Ferenc Vajda. „Embedded fall detection with a neural network and bio-inspired stereo vision“. In: *Computer Vision and Pattern Recognition Workshops.* 2012 (cit. on p. 35).

Irani, Michal and P Anandan. „Video indexing based on mosaic representations“. In: *Proceedings of the IEEE* 86.5 (1998), pp. 905–921 (cit. on p. 155).

Jähne, Bernd. *Digitale Bildverarbeitung.* 7th ed. Springer, 2012 (cit. on pp. 72, 75, 81, 90).

Javed, Omar, Zeeshan Rasheed, Khurram Shafique, and Mubarak Shah. „Tracking across multiple cameras with disjoint views“. In: *International Conference onComputer Vision.* IEEE, 2003, pp. 952–957 (cit. on p. 76).

Javed, Omar, Khurram Shafique, and Mubarak Shah. „Appearance modeling for tracking in multiple non-overlapping cameras“. In: *Computer Vision and Pattern Recognition.* Vol. 2. IEEE, 2005, pp. 26–33 (cit. on p. 76).

Johansson, Björn, Magnus Borga, and Hans Knutsson. „Learning corner orientation using canonical correlation“. In: *Symposium on Image Analysis.* SSAB. 2001, pp. 89–92 (cit. on p. 173).

Kay, Steven. *Fundamentals of Statistical Signal Processing: Estimation Theory.* 1st ed. Prentice Hall, 1993 (cit. on p. 85).

Kim, Tae-Kyun, Josef Kittler, and Roberto Cipolla. „Discriminative learning and recognition of image set classes using canonical correlations“. In: *Pattern Analysis and Machine Intelligence* 29.6 (2007), pp. 1005–1018 (cit. on pp. 172 sq.).

Kim, T.K., S.F. Wong, and R. Cipolla. „Tensor canonical correlation analysis for action classification“. In: *CVPR.* IEEE, 2007 (cit. on pp. 172 sq.).

Klami, Arto, Seppo Virtanen, and Samuel Kaski. „Bayesian canonical correlation analysis“. In: *Machine Learning Research* 14.1 (2013), pp. 965–1003 (cit. on p. 173).

Kobayashi, Takumi. „S 3 CCA: Smoothly Structured Sparse CCA For Partial Pattern Matching“. In: *International Conference on Pattern Recognition.* 2014 (cit. on p. 173).

Kogler, Jürgen, Christoph Sulzbachner, and Wilfried Kubinger. „Bio-inspired stereo vision system with silicon retina imagers“. In: *Computer Vision Systems.* Springer, 2009, pp. 174–183 (cit. on p. 35).

Kuglin, C.D. and D.C. Hines. „The phase correlation image alignment method“. In: *International Conference of Cybernetics and Society.* 1975, pp. 163–165 (cit. on pp. viii, 4).

— „The phase correlation image alignment method“. In: *Proc. Int. Conf. Cybernetics and Society, Sept. 1975.* 1975, pp. 163–165 (cit. on pp. 135, 159).

Lai, Pei Ling. and Colin Fyfe. „Kernel and nonlinear canonical correlation analysis“. In: *Neural Systems* 10.5 (2000), pp. 365–378 (cit. on p. 173).

Lampert, Christoph and Oliver Krömer. „Weakly-paired maximum covariance analysis for multimodal dimensionality reduction and transfer learning“. In: *European Conference on Computer Vision.* Springer, 2010, pp. 566–579 (cit. on p. 173).

Laptev, Ivan. „On space-time interest points“. In: *International Journal of Computer Vision* 64.2-3 (2005), pp. 107–123 (cit. on p. 17).

Le, Quoc, Will Zou, Serena Yeung, and Andrew Ng. „Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis“. In: *Computer Vision and Pattern Recognition.* 2011, pp. 3361–3368 (cit. on p. 136).

Lee, Lily, Raquel Romano, and Gideon Stein. „Monitoring activities from multiple video streams: Establishing a common coordinate frame“. In: *Pattern Analysis and Machine Intelligence* 22.8 (2000), pp. 758–767 (cit. on p. 17).

Longuet-Higgins, Christopher. „The visual ambiguity of a moving plane“. In: *Biological Sciences* (1984) (cit. on p. 155).

Longuet-Higgins, Christopher and Kvetoslav Prazdny. „The interpretation of a moving retinal image“. In: *Biological Sciences* (1980) (cit. on p. 155).

Lowe, David. „Distinctive image features from scale-invariant keypoints“. In: *International Journal of Computer Vision* 60.2 (2004), pp. 91–110 (cit. on p. 17).

— „Object recognition from local scale-invariant features“. In: *International Conference on Computer Vision.* Vol. 2. IEEE, 1999, pp. 1150–1157 (cit. on p. 17).

Loy, Chen, Tao. Xiang, and Shaogang Gong. „Multi-camera activity correlation analysis“. In: *Computer Vision and Pattern Recognition.* IEEE, 2009, pp. 1988–1995 (cit. on p. 173).

Lucas, Bruce and Takeo Kanade. „An iterative image registration technique with an application to stereo vision.“ In: *International Joint Conference on Artificial Intelligence.* Vol. 81. 1981, pp. 674–679 (cit. on pp. 81, 135).

Mahowald, Misha. „The silicon retina“. In: *An Analog VLSI System for Stereoscopic Vision.* Springer, 1994, pp. 4–65 (cit. on p. 34).

Mann, Steve. „Comparametric equations with practical applications in quantigraphic image processing“. In: *Image Processing* 9.8 (2000), pp. 1389–1406 (cit. on p. 77).

Mann, Steve and Richard Mann. „Quantigraphic imaging: Estimating the camera response and exposures from differently exposed images“. In: *Computer Vision and Pattern Recognition.* Vol. 1. IEEE. 2001 (cit. on p. 78).

Mardia, Kanti., John Kent, and John Bibby. *Multivariate analysis.* Academic Press, 1980 (cit. on pp. 176 sqq.).

Marr, David. *Vision: A computational investigation into the human representation and processing of visual information.* MIT Press, 1982 (cit. on pp. 1, 89).

Marr, David and Tomaso Poggio. „A computational theory of human stereo vision“. In: *Biological Sciences* 204.1156 (1979), pp. 301–328 (cit. on p. 12).

Martin, David, Charles Fowlkes, Doron Tal, and Jitendra Malik. „A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics“. In: *International Conference on Computer Vision.* Vol. 2. IEEE, 2001, pp. 416–423 (cit. on p. 179).

Matthies, Larry, Mark Maimone, Andrew Johnson, Yang Cheng, Reg Willson, Carlos Villalpando, Steve Goldberg, Andres Huertas, Andrew Stein, and Anelia Angelova. „Computer Vision on Mars“. In: *International Journal of Computer Vision* 75.1 (2007), pp. 67–92 (cit. on p. 1).

McCall, Joel and Mohan Trivedi. „Video-based lane estimation and tracking for driver assistance: survey, system, and evaluation“. In: *Intelligent Transportation Systems* 7.1 (2006), pp. 20–37 (cit. on p. 1).

Mead, Carver and Misha Mahowald. „A silicon model of early visual processing“. In: *Neural networks* 1.1 (1988), pp. 91–97 (cit. on p. 34).

Memisevic, Roland. „Learning to relate images“. In: *Pattern Analysis and Machine Intelligence* (2013) (cit. on pp. 136, 174).

— „Non-linear latent factor models for revealing structure in high-dimensional data“. PhD thesis. University of Toronto, 2008 (cit. on p. 174).

Memisevic, Roland and Geoffrey Hinton. „Learning to represent spatial transformations with factored higher-order Boltzmann machines“. In: *Neural Computation* 22.6 (2010), pp. 1473–1492 (cit. on p. 171).

— „Unsupervised learning of image transformations“. In: *Computer Vision and Pattern Recognition.* IEEE. 2007 (cit. on p. 174).

Mendel, Jerry. *Lessons in estimation theory for signal processing, communications, and control.* Pearson Education, 1995 (cit. on p. 140).

Mester, Rudolf. „Orientation estimation: Conventional techniques and a new non-differential approach“. In: *European Signal Processing Conference.* Vol. 2. 2000, pp. 921–924 (cit. on p. 81).

Mester, Rudolf, Til Aach, and Lutz Dümbgen. „Illumination-invariant change detection using a statistical colinearity criterion“. In: *German Conference on Pattern Recognition.* Springer, 2001, pp. 170–177 (cit. on p. 76).

Mester, Rudolf and Christian Conrad. „When patches match – a statistical view on matching under illumination variation“. In: *International Conference on Pattern Recognition.* 2014 (cit. on p. 76).

Mikolajczyk, Krystian. *Graf dataset.* 2014. URL: `http://www.robots.ox.ac.uk/~vgg/research/affine/` (cit. on p. 16).

Mikolajczyk, Krystian and Cordelia Schmid. „A performance evaluation of local descriptors“. In: *Pattern Analysis and Machine Intelligence* 27.10 (2005), pp. 1615–1630 (cit. on p. 17).

Mitsunaga, Tomoo and Shree Nayar. „Radiometric self calibration“. In: *Computer Vision and Pattern Recognition.* Vol. 1. IEEE, 1999 (cit. on p. 77).

Moravec, Hans. *Obstacle avoidance and navigation in the real world by a seeing robot rover.* Tech. rep. Robotics Institute, Carnegie Mellon University, 1980 (cit. on p. 15).

Mühlich, Matthias and Rudolf Mester. „Unbiased errors-in-variables estimation using generalized eigensystem analysis“. In: *European Conference on Computer Vision.* Springer, 2004 (cit. on p. 11).

Müller, Thomas, Clemens Rabe, and Uwe Franke. „Dense6D - Position und Bewegung robust für jeden Bildpunkt“. In: *Workshop Fahrerassistenzsysteme.* 2011 (cit. on p. 17).

Oh, Sangmin, Anthony Hoogs, Amitha Perera, Naresh Cuntoor, Chia-Chih Chen, Jong Taek Lee, Saurajit Mukherjee, JK Aggarwal, Hyungtae Lee, Larry Davis, et al. „A large-scale benchmark dataset for event recognition in surveillance video“. In: *Computer Vision and Pattern Recognition.* 2011 (cit. on pp. 152, 222).

Papert, Seymour. *The summer vision project.* Tech. rep. MIT, 1966 (cit. on p. 1).

Pezeshki, A., L.L. Scharf, J.K. Thomas, and B.D. Van Veen. „Canonical coordinates are the right coordinates for low-rank Gauss–Gauss detection and estimation“. In: *Signal Processing* 54.12 (2006), pp. 4817–4820 (cit. on pp. ix, 173).

Pezeshki, Ali, Mahmood Azimi-Sadjadi, and Louis Scharf. „A network for recursive extraction of canonical coordinates“. In: *Neural Networks* 16.5 (2003), pp. 801–808 (cit. on pp. 178, 202).

Porikli, Fatih. „Inter-camera color calibration by correlation model function“. In: *International Conference on Image Processing.* Vol. 2. IEEE, 2003, pp. II–133 (cit. on p. 77).

Rai, Piyush and Hal Daumé. „Multi-Label Prediction via Sparse Infinite CCA.“ In: *Neural Information Processing Systems.* 2009, pp. 1518–1526 (cit. on p. 173).

Roberts, Richard, Christian Potthast, and Frank Dellaert. „Learning general optical flow subspaces for egomotion estimation and detection of motion anomalies“. In: *Computer Vision and Pattern Recognition.* IEEE, 2009 (cit. on p. 136).

Rosten, Edward and Tom Drummond. „Machine learning for high-speed corner detection“. In: *European Conference on Computer Vision.* Springer, 2006, pp. 430–443 (cit. on p. 15).

Rosten, Edward, Reid Porter, and Tom Drummond. „Faster and better: A machine learning approach to corner detection“. In: *Pattern Analysis and Machine Intelligence* 32.1 (2010), pp. 105–119 (cit. on p. 15).

Rublee, Ethan, Vincent Rabaud, Kurt Konolige, and Gary Bradski. „ORB: an efficient alternative to SIFT or SURF“. In: *International Conference on Computer Vision.* IEEE, 2011, pp. 2564–2571 (cit. on p. 17).

Scharf, Louis and Clifford Mullis. „Canonical coordinates and the geometry of inference, rate, and capacity“. In: *Signal Processing* 48.3 (2000), pp. 824–831 (cit. on pp. 172 sq., 177 sq.).

Scharf, Louis and John Thomas. „Wiener filters in canonical coordinates for transform coding, filtering, and quantizing“. In: *Signal Processing* 46.3 (1998), pp. 647–654 (cit. on pp. 173, 178).

Scharstein, D. and Richard Szeliski. „A taxonomy and evaluation of dense two-frame stereo correspondence algorithms“. In: *International Journal of Computer Vision* (2002) (cit. on p. 14).

Schmid, Cordelia, Roger Mohr, and Christian Bauckhage. „Evaluation of Interest Point Detectors“. In: *International Journal of Computer Vision* 37.2 (2000), pp. 151–172 (cit. on pp. 15 sq.).

Shi, Jianbo and Carlo Tomasi. „Good features to track“. In: *Computer Vision and Pattern Recognition.* IEEE, 1994, pp. 593–600 (cit. on pp. 15, 81).

Sinha, Sudipta, Marc Pollefeys, and Leonard McMillan. „Camera network calibration from dynamic silhouettes“. In: *Computer Vision and Pattern Recognition.* Vol. 1. IEEE. 2004, pp. I–195 (cit. on p. 18).

Snavely, Noah, Steven Seitz, and Richard Szeliski. „Modeling the world from internet photo collections“. In: *International Journal of Computer Vision* 80.2 (2008), pp. 189–210 (cit. on p. 13).

— „Photo tourism: exploring photo collections in 3D“. In: *Graphics* 25.3 (2006), pp. 835–846 (cit. on p. 13).

Stein, Fridtjof. „Efficient computation of optical flow using the census transform“. In: *German Conference on Pattern Recognition.* Springer, 2004, pp. 79–86 (cit. on p. 76).

Stiller, Christoph and Janusz Konrad. „Estimating motion in image sequences“. In: *Signal Processing Magazine* (1999) (cit. on p. 167).

Sun, Deqing, Stefan Roth, and Michael Black. „Secrets of optical flow estimation and their principles“. In: *Computer Vision and Pattern Recognition.* IEEE, 2010 (cit. on p. 135).

Sun, Deqing, Stefan Roth, John Lewis, and Michael Black. „Learning optical flow“. In: *European Conference on Computer Vision.* Springer, 2008, pp. 83–97 (cit. on p. 136).

Svoboda, Tomáš, Daniel Martinec, and Tomáš Pajdla. „A convenient multicamera self-calibration for virtual environments“. In: *PRESENCE: teleoperators and virtual environments* 14.4 (2005), pp. 407–422 (cit. on p. 18).

Szeliski, Richard. *Computer vision: algorithms and applications.* Springer, 2010 (cit. on pp. 7, 14 sq., 80, 90, 168).

Szlávik, Zoltán. „Matching in Videoimages: Camera Registration and CNN Based Feature Extraction“. PhD thesis. Hungarian Academy of Sciences, 2006 (cit. on p. 35).

Szlávik, Zoltán, László Havasi, and Tamás Szirányi. „Estimation of common groundplane based on co-motion statistics“. In: *Image Analysis and Recognition.* Springer, 2004, pp. 347–354 (cit. on p. 35).

— „Geometrical scene analysis using co-motion statistics“. In: *Advanced Concepts for Intelligent Vision Systems.* Springer. 2007, pp. 968–979 (cit. on p. 35).

Szlávik, Zoltán and Tamás Szirányi. „Bayesian estimation of common areas in multi-camera systems“. In: *International Conference on Image Processing.* IEEE, 2006, pp. 1045–1048 (cit. on p. 35).

Szlávik, Zoltán, Tamás Szirányi, and László Havasi. „Video Camera Registration using Accumulated Co-Motion Maps“. In: *Photogrammetry and Remote Sensing* 61.5 (2007), pp. 298–306 (cit. on pp. 35 sq.).

Szlávik, Zoltán, Tamás Szirányi, László Havasi, and Csaba Benedek. „Optimizing of searching co-motion point-pairs for statistical camera calibration“. In: *International Conference on Image Processing.* Vol. 2. IEEE. 2005, pp. II–1178 (cit. on pp. 35 sq.).

Therrien, Charles W. *Discrete random signals and statistical signal processing.* Prentice Hall, 1992 (cit. on p. 52).

Thrun, Sebastian, Wolfram Burgard, and Dieter Fox. *Probabilistic robotics.* MIT Press, 2005 (cit. on p. 17).

Torr, Philip and Andrew Zisserman. „Feature based methods for structure and motion estimation“. In: *Vision Algorithms: Theory and Practice* (2000), pp. 278–294 (cit. on p. 12).

Triggs, Bill. „Detecting keypoints with stable position, orientation, and scale under illumination changes“. In: *European Conference on Computer Vision.* Springer, 2004, pp. 100–113 (cit. on p. 15).

— „Joint feature distributions for image correspondence“. In: *International Conference on Computer Vision.* Vol. 2. IEEE, 2001, pp. 201–208 (cit. on p. 34).

Triggs, Bill, Philip McLauchlan, Richard Hartley, and Andrew Fitzgibbon. „Bundle adjustment a modern synthesis“. In: *Vision algorithms: theory and practice.* Springer, 2000, pp. 298–372 (cit. on p. 12).

Tuytelaars, Tinne and Krystian Mikolajczyk. „Local invariant feature detectors: a survey“. In: *Foundations and Trends® in Computer Graphics and Vision* 3.3 (2008), pp. 177–280 (cit. on pp. 15, 17).

van den Hengel, Anton, Anthony Dick, Henry Detmold, Alex Cichowski, and Rhys Hill. „Finding camera overlap in large surveillance networks“. In: *Asian Conference on Computer Vision.* Springer, 2007 (cit. on p. 3).

— „Finding camera overlap in large surveillance networks“. In: *Asian Conference on Computer Vision*. Springer, 2007, pp. 375–384 (cit. on p. 18).

van den Hengel, Anton, Anthony Dick, and Rhys Hill. „Activity topology estimation for large networks of cameras“. In: *Advanced Video and Signal Based Surveillance*. IEEE, 2006, pp. 44–44 (cit. on p. 18).

Viinikanoja, Jaakko, Arto Klami, and Samuel Kaski. „Variational Bayesian mixture of robust CCA models“. In: *Machine Learning and Knowledge Discovery in Databases*. Springer, 2010, pp. 370–385 (cit. on p. 209).

Vogel, Christoph, Stefan Roth, and Konrad Schindler. „An Evaluation of Data Costs for Optical Flow“. In: *German Conference on Pattern Recognition*. Vol. 8142. Springer, 2013, pp. 343–353 (cit. on p. 76).

Wang, Chong. „Variational Bayesian approach to canonical correlation analysis“. In: *Neural Networks* 18.3 (2007), pp. 905–910 (cit. on p. 173).

Wang, Heng, Muhammad Muneeb Ullah, Alexander Klaser, Ivan Laptev, and Cordelia Schmid. „Evaluation of local spatio-temporal features for action recognition“. In: *British Machine Vision Conference*. 2009 (cit. on p. 17).

Wang, Xiaogang, Kinh Tieu, and Eric Grimson. „Correspondence-free activity analysis and scene modeling in multiple camera views“. In: *Pattern Analysis and Machine Intelligence* 32.1 (2010), pp. 56–71 (cit. on p. 17).

Wexler, Yonatan, Andrew Fitzgibbon, and Andrew Zisserman. „Learning epipolar geometry from image sequences“. In: *Computer Vision and Pattern Recognition*. Vol. 2. IEEE, 2003, pp. II–209 (cit. on p. 34).

Wright, John and Robert Pless. „Analysis of persistent motion patterns using the 3d structure tensor“. In: *Workshops on Application of Computer Vision*. Vol. 2. IEEE. 2005, pp. 14–19 (cit. on p. 135).

Zabih, Ramin and John Woodfill. „Non-parametric local transforms for computing visual correspondence“. In: *European Conference on Computer Vision*. Springer, 1994, pp. 151–158 (cit. on p. 76).

Zhang, Zhengyou, Rachid Deriche, Olivier Faugeras, and Quang-Tuan Luong. „A robust technique for matching two uncalibrated images through the recovery of the unknown epipolar geometry“. In: *Artificial intelligence* 78.1 (1995), pp. 87–119 (cit. on p. 36).

CURRICULUM VITAE

PERSONAL DETAILS

- Christian Conrad, born in Künzelsau, Germany.

ACADEMIC AND PROFESSIONAL EXPERIENCE

- *Carl Zeiss Group, R&D division,* since 01.02.2015, Sensor and algorithm development in the area of optical metrology.

- *Research Assistant,* from 01.06.2009 to 31.01.2015, Goethe-Universität Frankfurt, Computer Science and Math Department, VSI Group headed by Prof. Dr.-Ing. Rudolf Mester.

EDUCATION

- *M.Sc. in Computer Science*, Technische Universität Darmstadt, *Thesis:* Graph Cut Inferenz für Low Level Vision in Modellen höherer Ordnung, supervised by Prof. Stefan Roth, Ph.D., 2009.

- *B.Sc. in Computer Science*, Technische Universität Darmstadt, *Thesis:* Aufgabenverteilung in Multi-Roboter-Systemen, supervised by Prof. Dr. Oskar von Stryk, 2006.

PUBLICATIONS (PEER-REVIEWD)

- C. Conrad and R. Mester „Learning Rank Reduced Mappings using Canonical Correlation Analysis.“ In: *IEEE Statistical Signal Processing Workshop (SSP)*, Palma de Mallorca, Spain, June 2016.

- C. Conrad and R. Mester „Learning Relative Photometric Differences of Pairs of Cameras.“ In: *Proc. of the IEEE International Conference on Advanced Video and Signal based Surveillance (AVSS)*, Karlsruhe, Germany, 2015.

- R. Mester and C. Conrad. „When patches match - a statistical view on matching under illumination variation.“ In: *Proc. of the IEEE International Conference on Pattern Recognition (ICPR)*, Stockholm, Sweden, 2014.
- C. Conrad, M. Mertz, and R. Mester. „Contour-relaxed Superpixels.“ In: *Energy Minimization Methods in Computer Vision and Pattern Recognition (EMMCVPR)*, Lund, Sweden, 2013.
- J. Eisenbach, M. Mertz, C. Conrad, and R. Mester. „Reducing Camera Vibrations and Photometric Changes in Surveillance Video.“ In *Proc. of the IEEE International Conference on Advanced Video and Signal-Based Surveillance (AVSS)*, Kraków, Poland, 2013.
- J. Eisenbach, C. Conrad, and R. Mester. „A temporal scheme for fast learning of image-patch correspondences in realistic multi-camera setups.“ In: *Proc. of the Workshops of the IEEE International Conference Computer Vision and Pattern Recognition (WCVPR).* Portland, USA, 2013.
- C. Conrad and R. Mester. „Learning Multi-View Correspondences via Subspace-Based Temporal Coincidences.“ In: *Proc. of the Scandinavian Conferences on Image Analysis (SCIA)*, Espoo, Finland, 2013.
- C. Conrad, and R. Mester. „Learning Geometrical Transforms between Multi Camera Views using Canonical Correlation Analysis.“ In: *Proc. of the British Machine Vision Conference (BMVC)*, Surrey, United Kingdom, 2012.
- A. Guevara, C. Conrad, and R. Mester. „Curvature oriented clustering of sparse motion vector fields.“ In: *Proc. of the IEEE Southwest Symposium on Image Analysis & Interpretation (SSIAI)*, Santa Fe, USA, 2012.
- R. Memisevic, C. Conrad. „Stereopsis via Deep Learning.“ In: *Deep Learning Workshops at the Neural Information Processing Systems, (WNIPS)*, Granada, Spain, 2011.
- A. Guevara, C. Conrad, and R. Mester. „Boosting segmentation results by contour relaxation.“ In: *Proc. of the IEEE International Conference on Image Processing (ICIP)*, Brussels, Belgium, 2011.
- C. Conrad, A. Guevara, and R. Mester. „Learning multi-view correspondences from temporal coincidences.“ In: *Proc. of the Workshops of the IEEE International Conference Computer Vision and Pattern Recognition (WCVPR).* Colorado Springs, USA, 2011.