Learning Data-Driven Representations for Robust Monocular Computer Vision Applications

Dissertation
der Mathematisch-Naturwissenschaftlichen Fakultät
der Eberhard Karls Universität Tübingen
zur Erlangung des Grades eines
Doktors der Naturwissenschaften
(Dr. rer.-nat.)

vorgelegt von
Dipl.-math. Christian Joachim Herdtweck
aus Stuttgart

Tübingen
2013

Tag der mündlichen Qualifikation:	26.08.2013
Dekan:	Prof. Dr. Wolfgang Rosenstiel
1. Gutachter:	Prof. Dr. Heinrich H. Bülthoff
2. Gutachter:	Prof. Dr. Andreas Schilling

Contents

Zusammenfassung

Die vorliegende Arbeit stellt drei neue datengetriebene Darstellungen von Bildern und Bildsequenzen für Anwendungen im Bereich des maschinellen Bildverstehens vor. Die ersten beiden Darstellungen finden ihre Anwendungen in der Interpretation von Bildsequenzen, die von einem bewegten Fahrzeug mit einer einzelnen Kamera aufgenommen werden. Durch Projektion des optischen Flusses zwischen aufeinanderfolgenden Bildern in einen erlernten Subraum von Flussvektoren wird die Behandlung von fehlenden Beobachtungen, Fehlern in der Flussberechnung, Fluss von bewegten Objekten und anderen Verletzungen des Interpretationsmodells auf natürliche Weise ermöglicht. Der Anteil des optischen Flusses, der durch die Eigenbewegung der Kamera erzeugt wurde, kann durch eine erlernte lineare Abbildung auf Rotation und Vorwärtsbewegung des Fahrzeugs abgebildet werden.

Die zweite Repräsentation nutzt Ergebnisse der Objekterkennung und Statistik auf zirkulären Variablen um den Ansichtswinkel auf Objekte zu schätzen und in Form einer multimodalen Verteilung darzustellen. Dies erlaubt, die Mehrdeutigkeiten im Zusammenhang zwischen Aussehen und Orientierung eines Objekts korrekt der weitern Verarbeitung zuzuführen. Eine zeitliche Integration solcher Verteilungen durch einen Partikelfilter wird vorgestellt, die eine konsistente Zustandsverfolgung von Ansichtswinkeln ermöglicht. Es wird weiter gezeigt, dass die Verfolgung von Position, Orientierung, Geschwindigkeit und Radeinschlag eines Fahzeugs von einer bewegten Kamera aus verbessert werden kann, indem man mehrdeutige Ansichtswinkelschätzungen hinzufügt.

Um die Erzeugung einer ganzheitlichen 'Quintessenz' (engl.: 'gist') eines Bildes geht es im letzten Teil der Arbeit. Laut Forschungsergebnissen zur menschlichen Wahrnehmung entsteht diese Darstellung im menschlichen Gehirn innerhalb weniger hundert Millisekunden und bildet die Basis für die weitere Verarbeitung des Gesehenen. Dies wird durch einen Algorithmus nachgebildet, der Ergebnisse von Oberflächenorientierungsschätzung, Objekterkennern, Szenentypklassifikatoren, und Schätzungen von Kamerahöhe und -Neigung mit erlerntem Vorwissen kombiniert, indem er iterativ eine Auswahl von Teilergebnissen zusammenstellt, die konsistent zueinander sind. In mehreren Experimenten wird gezeigt, dass der Horizont im Bild Teil dieser Darstellung ist und untersucht welche Informationsquellen zu seiner Schätzung in Mensch und Computer benutzt werden.

Beiträge zu dieser Arbeit

Mit Ausnahme der im folgenden aufgeführten Beiträge wurden alle Ergebnisse und die dafür nötigen Programme von Christian Herdtweck entworfen, erzeugt und beschrieben. Ideen, Umsetzung und Ergebnisse wurden mit den jeweiligen Betreuern diskutiert. Der Teil der Arbeit, der in Teil I enthalten ist, wurde von Cristóbal Curio betreut, die Arbeit aus Teil II wurde von Christian Wallraven betreut.

Kapitel „Flow Subspaces for Self-Motion"

Das hier beschriebene generative Modell zur Beschreibung von optischem Fluss, eine Methode zum Trainieren des Modells sowie die Erweiterung der Arbeit von Irani [1999] auch auf längere Bildsequenzen stammen aus der Arbeit Roberts et al. [2009]. Diese Ideen wurden in dieser Arbeit aufgegriffen, erweitert und getestet.

Kapitel „Application: Estimating the Focus Of Expansion"

Der Text dieses Kapitels stammt aus einer Veröffentlichung, die von Christian Herdtweck und Cristóbal Curio zusammen geschrieben wurde. Die hier präsentierten Ergebnisse wurden von Christian Herdtweck erzeugt, mit Ausnahme des beschriebenen effizienten Algorithmus zur numerischen Bestimmung des "Focus of Expansion" durch Faltung, sowie eine Implementierung dieses Algorithmus, die aus einer früheren Arbeit von Cristóbal Curio stammen.

CHAPTER 1

Introduction

Getting computers to understand images in a way we humans do has been the goal of computer vision for almost five decades now (Kropatsch [2008]). Greatly underestimated in the beginning due to the apparent ease with which we humans interpret visual input, the field has attracted a plethora of research (at least 326 labs in 2005[1], having produced at least several ten thousand publications[2]). As a result, researchers can now boast with impressive results enabling self-driving cars (Kammel et al. [2008], Urmson et al. [2008]), real-time simultaneous localization and mapping (SLAM, Sibley et al. [2010]), image retrieval (Jégou et al. [2008]), accurate tracking of body parts (Pons-Moll et al. [2011], Hasler et al. [2009]), face recognition (Li and Jain [2011]), and object scene understanding (Felzenszwalb et al. [2010]), to mention just a few.

This progress is due to a deeper understanding of computer vision tasks, which led to improvements in feature extraction, processing mechanisms, representations of image content and interpretation frameworks. It would not have been possible without the great contributions from the fields of statistics and machine learning (e.g. Schölkopf and Smola [2002]), as well as advances in sensor and computer hardware. Insights from human perception research (e.g. Bülthoff and Edelman [1992]) have also inspired many new approaches.

[1]according to `http://www.cs.cmu.edu/~cil/v-groups.html`

[2]according to `http://academic.research.microsoft.com`, when listing all publications from Computer Vision

However, most applications are still severely restricted in their applicability. Dealing with the great variability of visual input that we humans process every day, is still a big challenge.

1.1. The Importance of Data-Driven Representations and Learning

While all of these factors are important, many researchers agree that the crucial step in the design of modern computer vision systems is the choice of a suitable representation of the visual input. A simple, direct mapping from simple features like edges, corners or optical flow vectors to the final interpretation is not possible for most tasks. Therefore, in current computer vision approaches, an appropriate intermediate representation of the input is formed, which can then be interpreted in a second step.

The importance of representation, i.e., the way how visual information is presented to a system that interprets it further, is exemplified in Fig. 1.1, Fig. 1.2 and Fig. 1.3. The exact same information, which is the number of photons that hit the camera sensor in a certain time and position, is represented in two different ways: first as height in Fig. 1.1, then as pixel luminance in Fig. 1.3, with Fig. 1.2 as intermediate step to clarify the transition. Since our human interpreting system is used to the second representation, it is intuitively able to correctly interpret Fig. 1.3 while the same information in Fig. 1.1 cannot be interpreted.

Representation in this example only specifies the way the information is encoded. A similar example can be found in Marr [1982], where representation also plays a major role: 1,2,3,4,... or I,II,III,IV,... are just different representations of the same concept of numbers. In the more general sense it also includes a specification of which part of the information to encode. An example for the diversity of representations, can be found in the field of object recognition, one of the core areas of computer vision. Related to the question, how objects are represented in the human brain (3-dimensional structures as suggested in Biederman [1987] or view-based as argued in Bülthoff and Edelman [1992]), propositions of how to encode the appearance of an object include all of the following: a collection of 3-dimensional parts (geons, Biederman [1987]), an arrangement of highly-specific interest points (Lowe [1999]), the response to a cascade of simple intensity filters (Viola and Jones [2001]), a set of histograms of oriented gradients (Dalal and Triggs [2005a]), which can also be required to have certain parts (Felzenszwalb et al. [2010]), or

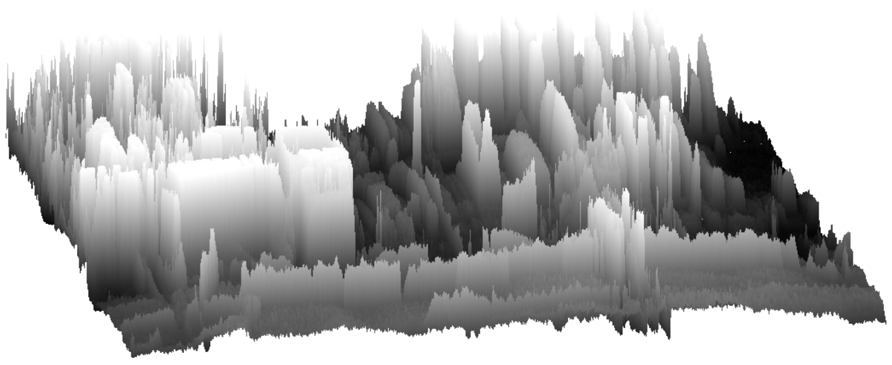


Figure 1.1.: Changing the representation of an image destroys the human ability to intuitively interpret it.

regions of homogeneous intensity, color and texture (Cao and Fei-Fei [2007]), to mention just a few.

Forming most of these representations is only possible through new methods in machine-learning and statistics. In the early days of computer vision, representations were built on the basis of deterministic rules inspired by physical laws and observations of biological vision system behavior. Extraction of geons, for example, was defined by a deterministic arrangement of lines and curves, interpretation of image contents based on them included finding pre-defined patterns. Modern representations, on the other hand, are based on general probabilistic models that are instantiated by abstracting from example data. In a way, the task of abstracting from observations has partly moved from human researchers to the computer. Where humans have specified heuristics and formulae from observations to build computer vision systems, algorithms now partly create their own representations from training data. Explicit world knowledge is still contained in these models through their structure and choice of training data, but statistics play an increasingly important role in forming the representations. Building and using such representations, which are created by abstracting from training data through machine learning algorithms, are the goal of this thesis. They are referred to as 'data-driven', in contrast to approaches and representations based on deterministic rules. Using such data-driven approaches enables interpretations of visual input, that a human researcher may never have thought of.

A decisive advantage of using data-driven methods is their robustness. Capturing the structure in data allows approaches to find commonalities and identify sources of variance. The resulting representations can thus tolerate noise from various sources, ranging from sensor noise to semantic inconsistencies. This is what allows computer vision

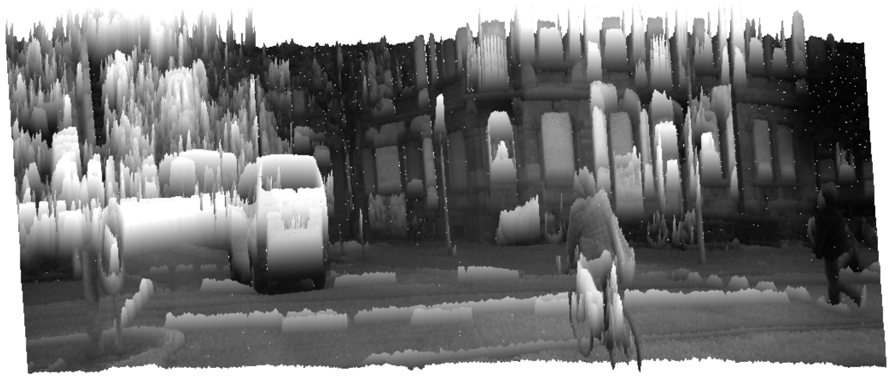


Figure 1.2.: Only with the proper representation, ... (continued in Fig. 1.3)

to leave the realm of controlled industrial applications and laboratory simulations and venture into real-life applications that feature clutter, uncontrolled lighting conditions, complex geometry and motion patterns, as well as a huge variety of scene types.

Applying computer vision methods in an automotive context is especially challenging. It requires not only a robust estimation of a represenation of the observed world, but also fast methods to calculate it and stable filtering approaches to maintain it. Using estimates for autonomous driving on potentially busy streets demands for excellent performance, since estimation errors can have dire consequences. On the other hand, increased car autonomy could offer great benefits given the increasing number of cars, allowing to decrease environment pollution, traffic jams and the number of often fatal accidents. Car manufacturers therefore fund an ever greater body of research, e.g. Viola and Jones [2001], Enzweiler and Gavrila [2011], Badino et al. [2009], Yao et al. [2012].

Another reason for the increasing number of machine-learning approaches, besides advances in computer hardware, is the availability of more data sets that provide the necessary samples and ground truth annotations for training. Capturing or collecting large amounts of images and labeling them is tedious work, but by sharing it with the scientific community research greatly benefits from these efforts. One example is again the field of object detection, which has established data sets for training and comparison (e.g. Everingham et al. [2012], Griffin et al. [2007]) 8 years ago, resulting in a great increase in the number and performance of detection approaches. Similar trends can be observed for more general processing of static (e.g. LabelMe Russell et al. [2007a]) and dynamic (e.g. Kitti Geiger et al. [2012]) imagery. Interpreting data of either of these two data sets is challenging for a variety of tasks, due to the considerable variety of image geometry, lighting conditions and semantic content, as well as large amounts of

Figure 1.3.: (continued from Fig. 1.2) ..., image processing becomes tractable.

clutter and occlusions. While LabelMe focuses on the variety of object and scene content for static scene analysis, Kitti is intended for dynamic vision tasks like self-motion estimation, object tracking, simultaneous localization and mapping (SLAM), and structure from motion. It provides not only images with object labels, but includes stereo images, ground truth data on observer motion and 3D object position, as well as 3D point clouds from laser scans. Being able to work with such data sets could lead to improved methods for more robust static and dynamic scene interpretation, allowing more interesting applications of computer vision in every-day life.

1.2. Finding New Representations

Given the importance of representation and learning, this thesis focuses on new representations for computer vision tasks that are driven by machine learning methods, replacing mostly deterministic processing of visual input. Three different examples to describe aspects of scene content in a new way are presented together with machine-learning methods to extract them from cluttered, real-world image data. The resulting estimates are used for typical tasks in dynamic and static computer vision.

Estimating self-motion from monocular input is an ill-posed problem since observed correspondences in consecutive images are caused by a mixture of several sources: scene geometry, self-motion and object motion. Assuming a roughly static scene, most current solutions (e.g. Lategahn et al. [2012]) infer both scene geometry and self-motion together using elaborate geometric reasoning. In Roberts et al. [2009], a different representation of flow is presented that implicitly includes prior knowledge on geometry, allowing computations to focus on self- and object motion. The underlying generative probabilistic model allows an automatic identification of deviations from the geometric prior and object motion, which can then be ignored in self-motion estimation. This representation is extended in two ways in this work: firstly, the probabilistic model is generalized and secondly embedded in a mixture of expert system. Evaluation on a new data set of challenging urban traffic scenarios shows that this results in a greater range of application and improves monocular self-motion estimation performance.

Having estimated one's own motion, predicting future states of other objects is necessary for navigation. Object orientation has long been used in object state representations (e.g. Koller et al. [1993]) since it restricts possible object motion. However, this variable is hardly ever measured directly, it is mostly inferred from object position estimates in successive frames. On the other hand, multi-view object detection (e.g. Felzenszwalb et al. [2010]) has created a variety of implicit and explicit ways to roughly estimate object orientation. The second new representation to be presented in this work is the inclusion of object orientation estimates in form of multi-modal distributions on a continuous circular space into state estimates of observed objects. A new learning-based approach for object orientation estimation from static images is introduced. Based on research in object detection and circular statistics, is provides a natural way to deal with ambiguities in object appearance. The resulting multi-modal estimates, as well as object position estimates are combined in a mixed circular and linear dynamic filtering framework for object state estimation. Incorporating monocular self-motion estimates described earlier allows filtering from a moving platform.

For the third representation, one needs to 'take a step back' from task-specific computer vision methods and look at vision in a more general sense. Early computer vision systems (e.g. Hanson and Riseman [1978], Marr [1982]) addressed a wide variety of tasks simultaneously that have afterwards been investigated in isolation. Having made considerable progress in most of them, a current trend is to consider combinations of tasks again, which are closely connected like object categorization and segmentation (Leibe et al. [2006]). Results on one aspect of scene context can prior other algorithms or limit the search space. This is in accordance with findings on human scene perception. Within a few hundred milliseconds, early visual processes seem to form a rough but consistent general-purpose representation of visual input, called the *gist* of a scene (Greene and Oliva [2009b]), which is the basis for further visual processing. The third representation proposed in this work is a computational equivalent of this gist. It is an attempt to form a consistent holistic representation of a single static image using appropriate priors and combined inference on several aspects of an image. Results from object detection (Dalal and Triggs [2005b]), classification of scene type (Oliva and Torralba [2001]), surface types (Hoiem et al. [2005]) and statistical prior knowledge are combined with viewpoint estimation, using each algorithm's output to bias interpretation of the others. A perceptual study investigates the presence and formation of horizon estimates in the gist.

To summarize, the main contributions of this thesis are:

- An extension and analysis of a generative subspace model for optical flow representation
- Thorough evaluation of the subspace model for monocular self-motion estimation
- Embedding of the self-motion estimation approach in a mixture of experts
- Application of the subspace model to estimation of the focus of expansion
- Estimation of object orientation using representations based on object detection results
- Representing orientation knowledge in form of a multi-modal distribution on a continuous circular space
- Adaptation of random regression forests and particle filters for viewpoint regression and filtering
- Combination of monocular self-motion estimation and object orientation estimation in a dynamic filtering framework
- A method for extracting the gist, a general-purpose representation of static images
- Perceptual and computational experiments on horizon estimation in human and machine

1.3. Thesis Structure

This thesis is structured as follows: In chapter 2 standard methods are explained and generalized that are needed in the remainder of the thesis.

Part I contains the main body of contributions of this thesis, which is on dynamic scene interpretation from a moving platform. The subspace representation of optical flow fields for self-motion estimation is derived, extended and discussed in chapter 3. A further extension using a mixture of experts system is presented and evaluated in chapter 4. In chapter 5 the flow representation is used to find the focus of expansion, a low-level representation of heading.

Object orientation is the second major topic of part I. A learning-based approach for estimating object orientation from the output of a part-based object detector using random

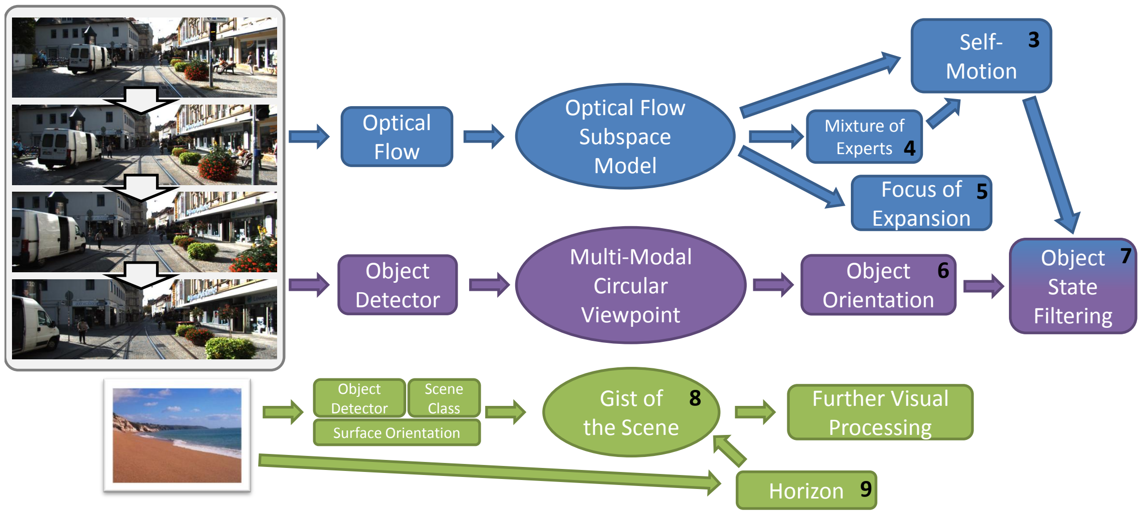


Figure 1.4.: Overview over the thesis structure for chapters 3 to 9. From inputs (**left**) three new representations (**middle**) are formed and used for various computer vision tasks (**right**). The black numbers in boxes are the chapter numbers where the applications are discussed.

regression forests and circular statistics is described in chapter 6. Adding orientation to representations of object state improves tracking of objects from a moving platform, which is shown in chapter 7.

In part II, chapter 8 introduces the perceptual gist of a scene, together with a system to create a similar representation in the computer. Whether horizon is part of this gist and how it is formed in humans is investigated in psychophysical and computational experiments discussed in chapter 9

A summary of main findings and pointers to future work are given in chapter 10. An overview over the structure of chapters 3 to 9 is shown in Fig. 1.4.

CHAPTER 2

Methods

Solving any interesting task in computer vision and machine learning requires a 'toolbox' of methods, which solve general computational tasks like regression, approximation, clustering, and filtering. The necessary set of methods for this thesis is presented in this chapter. It is mainly intended to be a collection of independent sections that can be referred to in the next chapters. There are two exceptions: firstly, the standard methods for kernel density estimation and clustering have been generalized to usage with variables that have linear and circular dimensions. Secondly, a small but important change in the derivation of the particle filter is introduced that is needed for an 'insertion' of particles that are sampled from measurement distributions. These are extensions of standard 'equipment' and therefore also explained here.

The chapter is organized as follows: after introducing notation used in this thesis, a collection of basic equations is provided. The next section (sec. 2.3) is a short introduction to methods in circular statistics that are used in chapter 6 and chapter 7. Two methods for regression using Gaussian processes and random regression forests are treated in sec. 2.4. Contained in sec. 2.5 is an introduction to kernel density estimation for the approximation of unknown probability densities in general, and for circular variables in particular. An efficient method for finding maxima in kernel density estimates is derived here, as well. Clustering is the topic of sec. 2.6, again specializing on two methods, namely using a mixture of Gaussian distributions and the mean shift procedure. Finally,

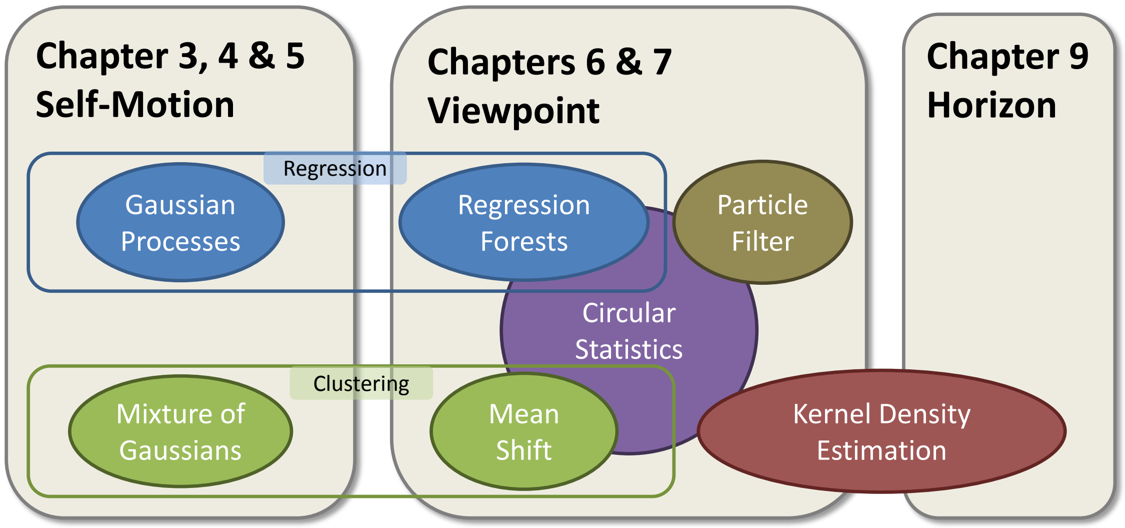


Figure 2.1.: Overview over the usage of the methods presented in this chapter. Gaussian process regression is used in chapter 3, chapter 4, and chapter 5, clustering is performed in these chapters with a mixture of Gaussians. In chapter 6 and chapter 7 all methods are based on circular statistics. Regression here is done using random regression forests, mean shift serves as clustering method. Kernel density estimation is used in chapter 6 and chapter 7 for (partly) circular variables, in chapter 9 for linear variables.

sec. 2.7 introduces particle filtering and describes a variation used in the last chapter. The usage of the methods presented here is summarized in Fig. 2.1.

2.1. Notation

Throughout this thesis the following notation will be used:

- scalar variables are denoted by standard math font, e.g. x
- Vectors of variables are written in bold, e.g. $\boldsymbol{x}$ or $\boldsymbol{\mu}$
- Matrices are written in bold and upper case, e.g. $\boldsymbol{A}$ or $\boldsymbol{\Sigma}$
- Deterministic variables, random variables and realizations of random variables are not distinguished
- Approximations to variables usually have a scalar integer variable as index, e.g. $p_N \approx p$
- Samples from a distribution are $\boldsymbol{x}_1 \sim p(\boldsymbol{x})$
- The Gaussian distribution is $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, evaluations of the density at a position $\boldsymbol{x}$ is $\mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$

- In general, parameters to a function are written as subscript or behind a ';', e.g. $I_0(x)$ or $p_{\mathsf{VM}}(x;\mu,\kappa)$
- Proportionality, i.e., equality up to a constant, is denoted by '$\propto$'
- $\mathcal{R}^d$ is the space of real-valued vectors of length d, $\mathcal{N} = \{1, 2, 3, \ldots\}$ is the set of positive integers (excluding 0), and $\mathcal{Z} = \ldots, -2, -1, 0, 1, 2, \ldots$ is the set of all integers
- $E(\boldsymbol{x})$ is the expectation of a random variable $\boldsymbol{x}$
- $\operatorname{cov}(\boldsymbol{x})$ is the covariance operator on random variables $\boldsymbol{x}$
- $|A|$ is the number of elements in the set A
- $|\boldsymbol{A}|$ is the determinant of matrix $\boldsymbol{A}$
- $\mathbf{1}_N$ is the identity matrix of size $N \times N$
- If not specified otherwise, scalar indices are written in lower case and range from 1 to their uppercase letter, e.g., $i = 1, \ldots, I$, $n = 1, \ldots, N$, etc.
- Closed intervals are denoted by $[a, b]$, while half-open or open intervals use parentheses, e.g. $[a, b)$

2.2. Basic Equations

There are a few non-trivial formulae that will be used in the following. They can be found in any text book on statistics or machine learning, e.g., Bishop [2006], and are therefore only mentioned here without further explanations.

One of them is Bayes' rule which expresses the *posterior* belief for a random variable $\boldsymbol{x}$ given observations $\boldsymbol{y}$, which is $p(\boldsymbol{x}|\boldsymbol{y})$, in terms of a *prior* belief $p(\boldsymbol{x})$, the probability of measurements $p(\boldsymbol{y})$, and the *likelihood* for the observation $p(\boldsymbol{y}|\boldsymbol{x})$ as follows:

$$p(\boldsymbol{x}|\boldsymbol{y}) = p(\boldsymbol{y}|\boldsymbol{x})\frac{p(\boldsymbol{x})}{p(\boldsymbol{y})} \quad . \tag{2.1}$$

The Gaussian distribution for a d-dimensional variable, given mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma} \in \mathcal{R}^{d\times d}$ has the density

$$\mathcal{N}(\boldsymbol{x};\boldsymbol{\mu},\boldsymbol{\Sigma}) = p_{\mathsf{gauss}}(\boldsymbol{x};\boldsymbol{\mu},\boldsymbol{\Sigma}) = (2\pi)^{-d/2}|\boldsymbol{\Sigma}|^{-1/2}\exp\Big((\boldsymbol{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\Big) \quad . \tag{2.2}$$

In one dimension one usually specifies the standard deviation σ, resulting in

$$\mathcal{N}(x;\mu,\sigma) = p_{\text{gauss}}(x;\mu,\sigma) = \frac{1}{\sqrt{2\pi}\sigma}\exp\left(\frac{(\boldsymbol{x}-\mu)^2}{2\sigma^2}\right) \quad . \tag{2.3}$$

2.3. Basic Circular Statistics

Most variables and values encountered in science are defined on linear spaces like the continuous space $\mathcal{R}^d$ or the 1-dimensional discrete space $n \in \mathcal{Z}$. These linear variables have an infinite support, which only sometimes is restricted (e.g., to be positive). They accurately describe quantities like time and often position. However, there are quantities that do not fit within this schematic. Yaw orientation, for example, is restricted to the finite interval $[0, 2\pi]$ radians or $[0°, 360°]$, where 0 and 2π or $360°$ describe the same value. For distances on the earth surface greater than a few hundred kilometers one has to take into account the curvature of the earth, which results in spherical coordinates. The angular component of polar coordinates and phase information for periodic data is also best modeled as circular quantity.

The circular nature of these quantities can often be neglected, but this is not always possible. Even a simple operation like taking the mean is different in circular spaces, as described in an example in Fig. 2.2. Circular analogues for operations like mean, variance and correlation, as well as concepts like distributions and statistical tests can be defined on circular spaces in a theoretically rigorous way. Some of these are described in this section for a circular space with one-dimensional variables ranging from 0 to 2π. A complete treatment of this topic, including generalizations to higher-dimensional spaces (e.g., a sphere or torus) can be found in Fisher [1995]. Chapters in Zar [2010] and Bishop [2006] are also devoted to circular statistics.

Usage of circular statistics in machine learning applications is becoming more popular, which can be seen in an increasing body of literature (e.g. Razavian et al. [2011], Banerjee et al. [2005], Wallace and Dowe [2000], Di Marzio et al. [2011], Rasmussen and Nickisch [2010]). In this thesis, circular variables are used in chapter 6 and chapter 7.

2.3.1. Circular Mean and Concentration

For defining the mean and analogues to the variance for circular data, values $x_1, \dots, x_N$ are interpreted as angles defining points

$$e^{ix_n} = \cos(x_n) + i\sin(x_n) \tag{2.4}$$

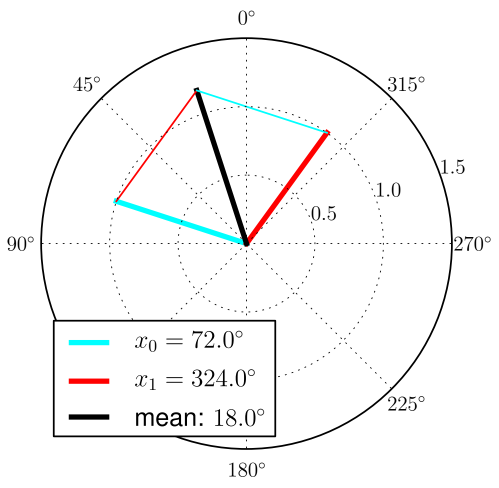


Figure 2.2.: Calculating the circular mean. Intuitively, two unit vectors with given angles relative to the 0 direction are added. The angle of the resulting line is the mean, while the length, divided by the number of vectors, is the mean resultant length. The result differs fundamentally from the linear mean, which would be $206°$.

on a unit circle in the space $\mathcal{C}$ of complex numbers, where i is the imaginary unit ($i^2 = -1$). The following derivations could be repeated in the same way using the 2-dimensional linear space $\mathcal{R}^2$ instead of $\mathcal{C}$, as done in Fisher [1995]. Then the *circular mean* is the angle of the complex mean of these numbers:

$$\bar{x} = \arg\left(\frac{1}{N}\sum_{n=1}^{N} e^{ix_n}\right) \quad . \tag{2.5}$$

This process is visualized in Fig. 2.2.

The length

$$\bar{r} = \mathrm{abs}\left(\frac{1}{N}\sum_{n=1}^{N} e^{ix_n}\right) \in [0,1] \tag{2.6}$$

of the complex mean is called the *mean resultant length*. It is used to define various analogs to the linear variance, describing the variance or concentration or dispersion of circular data. Various such measures exist, some bounded to a fixed interval, some only with a lower bound like the linear variance.

In chapter 6, estimates $\kappa_N = \mathrm{C}(x_1, \ldots, x_N)$ of the *circular concentration* κ of a von Mises-distribution are used. This unimodal distribution is described in the next subsection, but the formula for estimating κ from samples is given here (c.f. Fisher [1995], page 88).

The maximum-likelihood estimate κ_{ML} for κ is defined by

$$\frac{I_1(\kappa_{\mathsf{ML}})}{I_0(\kappa_{\mathsf{ML}})} = \bar{r} \tag{2.7}$$

where I_0 and I_1 are the modified Bessel functions (see e.g. Abramowitz and Stegun [1970]) of degree 0 and 1, respectively. Since this equation is not solvable in general, approximations are given by Taylor expansions

$$\kappa_{\mathsf{ML}} \approx \begin{cases} 2\bar{r} + \bar{r}^3 + 5\bar{r}^5/6 & \text{if} \quad \bar{r} < 0.53 \\ -0.4 + 1.39\bar{r} + 0.43\,(1-\bar{r})^{-1} & \text{if} \quad 0.53 \leq \bar{r} < 0.85 \\ (\bar{r}^3 - 4\bar{r}^2 + 3\bar{r})^{-1} & \text{if} \quad \bar{r} \geq 0.85 \quad . \end{cases} \tag{2.8}$$

For small sample sizes ($N \leq 15$) and small mean resultant lengths ($\bar{r} < 0.7$), κ_{ML} overestimates the true value of κ. For those cases, a correction is available. In summary, the circular concentration for samples is estimated as

$$\kappa \approx \kappa_N = \begin{cases} \kappa_{\mathsf{ML}} & \text{if} \quad N > 15 \text{ or } \bar{r} \geq 0.7 \\ \max\left(0, \kappa_{\mathsf{ML}} - 2(N\kappa_{\mathsf{ML}})^{-1}\right) & \text{if} \quad \kappa_{\mathsf{ML}} < 2 \\ (N-1)^3(N^3+N)^{-1}\kappa_{\mathsf{ML}} & \text{if} \quad \kappa_{\mathsf{ML}} \geq 2 \end{cases} \tag{2.9}$$

in this work.

2.3.2. Wrapped Gaussian and von Mises Distribution

There exist several analogs to the Gaussian distribution for circular data, none of which has all the properties of the Gaussian on linear spaces like scalability ($\mathcal{N}(a(\mu - x); \mu, \sigma^2) = \mathcal{N}(\mu - x; \mu, (a\sigma)^2)$, which often simplifies calculations.

Two distributions are used in this chapter, was well as in chapter 6 and chapter 7, both are symmetric and specified by a mean and a measure of concentration or variance. The first is the *von Mises* distribution, which is obtained by conditioning a two-dimensional linear Gaussian to the unit circle (Bishop [2006]). This also "[...] corresponds to the distribution of the angle, x, of a circular pendulum in a uniform field (at angle μ) subjected to thermal fluctuations, with κ representing the ratio of field strength to temperature. " (from Wallace and Dowe [2000], p76 top left). The density in one dimension is

$$p_{\mathsf{VM}}(x; \mu, \kappa) = (2\pi I_0(\kappa))^{-1} \exp\left(\kappa \cos(x - \mu)\right) \tag{2.10}$$

for mean $\mu \in [0, 2\pi]$ and concentration $\kappa \in [0, \infty)$. The von Mises distribution has a single maximum for $x = \mu$ and falls off symmetrically from that point. For large concentrations $\kappa \gg 1$ the distribution tends to a Gaussian distribution with variance $\frac{1}{\kappa}$, converging for $\kappa \to \infty$ towards a delta distribution in μ. For small concentrations $\kappa \ll 1$ the von Mises distribution tends to the uniform distribution on the circle

$$p_{\text{uni}}(x) = \frac{1}{2\pi} \quad . \tag{2.11}$$

Examples for mean $\mu = 0$ and various concentrations κ are plotted in Fig. 2.4 (left). Generalizations to several dimensions are the multivariate von Mises distribution or the von Mises-Fisher distribution family on the sphere (Fisher [1995]), both of which are not used in this work.

A convenient analog to the Gaussian distribution is the *wrapped Gaussian* distribution. In general, the wrapped version of a one-dimensional function $f : \mathcal{R} \to \mathcal{R}$ is

$$f_{\text{wrap}}(x) = \sum_{k \in \mathcal{Z}} f(x - 2\pi k) \tag{2.12}$$

The intuition for this is visualized in Fig. 2.3 for a Gaussian distribution. The wrapped Gaussian inherits its uni-modality and symmetry from the linear Gaussian. Examples for a wrapped Gaussian with mean $\mu = 0$ and different standard deviations σ are visualized in Fig. 2.4 (right).

2.4. Regression Methods

Regression is a very general technique to describe the relationship between a set of independent variables $\boldsymbol{g}$ and a dependent variable h. The aim is usually a function $f : \boldsymbol{g} \to h$ that describes this relationship in a mathematical formula. This function is often estimated from training sample pairs $(\boldsymbol{g}, h)$ with a model specifying how to generalize from these known pairs. By limiting the regression function to a certain family of functions with limited degrees of freedom (e.g. linear functions or polynomials), regression automatically regularizes the training input, thus ensuring favorable properties like smoothness and avoiding over-fitting.

An easy example is linear regression, where the aim is to find the best values $\boldsymbol{a}$ and b such that the function $f(\boldsymbol{g}) = \boldsymbol{a}\boldsymbol{g} + b$ matches the training samples best. This can be

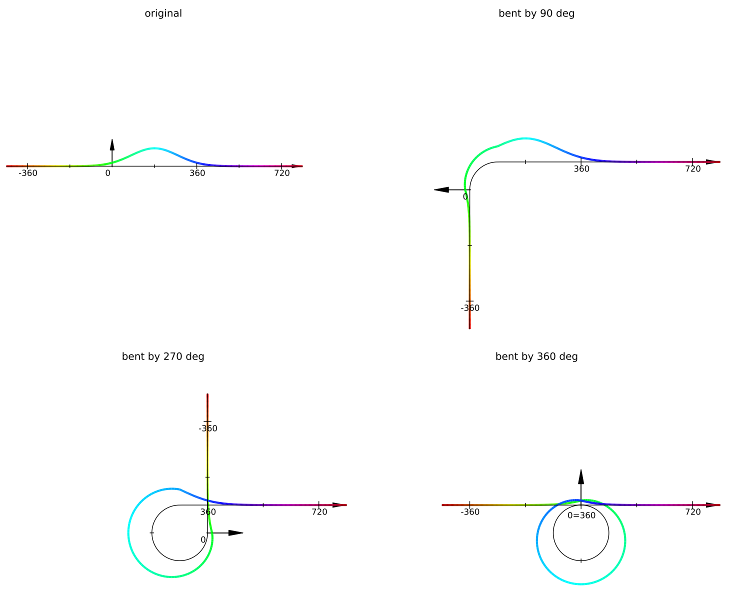


Figure 2.3.: Wrapping a Gaussian. A Gaussian function on a linear axis (**top left**) is wrapped in 90 degree increments (**from left to right**) to form a wrapped Gaussian (**bottom right**).

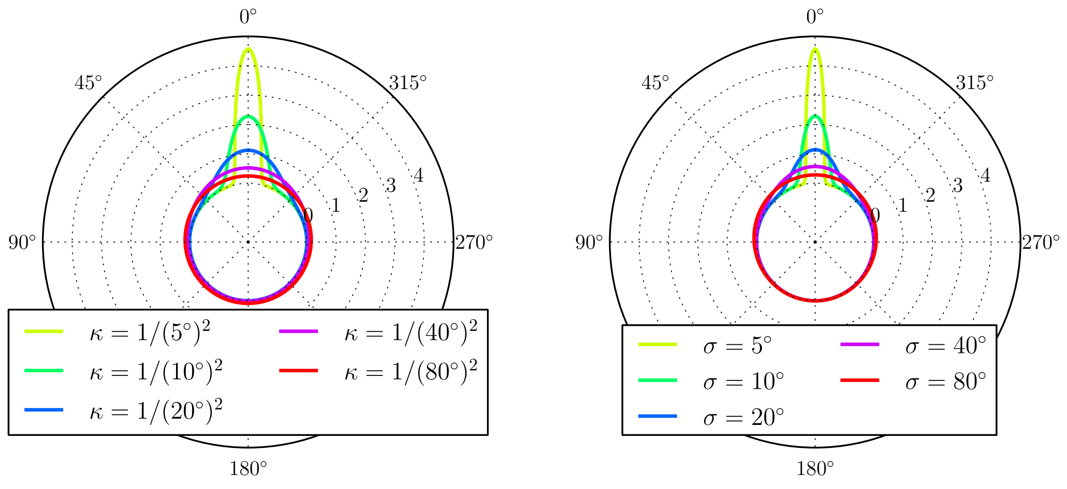


Figure 2.4.: Von Mises (**left**) and wrapped Gaussian distributions (**right**) with mean $\mu = 0$ and varying parameters for standard deviation σ and concentration κ. Actual values for κ are calculated by converting degrees to radians, and taking the inverse square.

solved using least squares methods, which minimize

$$\sum_{n=1}^{N} ||f(\boldsymbol{g}_n) - h_n||^2 \overset{!}{=} min \tag{2.13}$$

given sample pairs $(\boldsymbol{g}_1, h_1), \ldots, (\boldsymbol{g}_n, h_N)$. A popular algorithm for solving this task in presence of noisy training data is iteratively re-weighted least squares Holland and Welsch [1977]. This is used in chapter 3.

Two other methods used in this thesis are presented in the following.

2.4.1. Gaussian Process Regression

Gaussian processes are generalizations of Gaussian distributions to functions. While a realization (or sample) of a Gaussian distribution is a scalar or vector, realizations of Gaussian processes are functions $\boldsymbol{f}$ that can be multi-variate. Like Gaussian distributions, Gaussian processes have a mean and a covariance, both of which are functions. The mean is a function of one, the covariance of two variables. Most covariance functions have hyper-parameters, specifying, for example, the smoothness of realizations.

Applied to the task of regression, one tries to find hyper-parameters and a realization $\boldsymbol{f}$ of a Gaussian process that gives the maximum likelihood for given training sample pairs $(\boldsymbol{g}, \boldsymbol{h})$.

One of the advantages of using probabilistic method for regression is that every estimate can be augmented with a likelihood, which represents a confidence of the model in the given estimate. This can be used when combining different estimates.

Since the treatment of Gaussian processes is beyond the scope of a thesis, the reader is referred to the literature (e.g. Rasmussen and Williams [2006]) for a more detailed explanation. Gaussian process regression is used in chapter 3, chapter 4 and chapter 5.

2.4.2. Random Regression Forests

Random regression forests (Breiman [2001]) are a very general regression method, meaning that they are able to represent a wide variety of functions, including multi-variate linear and non-linear functions, possibly also containing a limited number of discontinuities. Calculating a function value $\boldsymbol{f}(\boldsymbol{g})$ from a regression forest is very fast.

A regression forest consists of T trees, each being a set of nodes and connections between them. Each node is connected to a *parent* and two *child* nodes, with the exception of one *root* node per tree that has no parent and *leaf* nodes that have no children. In each non-leaf node a binary test is saved, and each leaf node contains a distribution of output values of some form.

Regression in a regression forest, i.e. calculation of $\boldsymbol{f}(\boldsymbol{g})$ proceeds by gradually localizing the input vector $\boldsymbol{g}$ with respect to training samples and inferring $\boldsymbol{f}(\boldsymbol{g})$ from training samples with similar input vectors. To do so, the binary test stored in the first tree's root node is evaluated with $\boldsymbol{g}$. If the outcome is 0, regression proceeds to the left child of the root node, if the outcome is 1, the right child node is used. This corresponds to a localization of $\boldsymbol{g}$ in one of two parts of the input space.This procedure is repeated at the left and right child node until a leaf-node is reached, whose contents gives an estimate of the function value $\boldsymbol{f}(\boldsymbol{g})$.

This procedure is repeated for each tree in the forest, resulting in T estimates for $\boldsymbol{f}(\boldsymbol{g})$, which are then combined to form a single estimate with an optional certainty.

For training each tree of the regression forests using training samples $(\boldsymbol{g_1}, \boldsymbol{f_1}), ..., (\boldsymbol{g_N}, \boldsymbol{f_N})$ one first selects random subsets of the whole training sample set $H \subset \{1, \ldots, N\}$. For defining the first tree's root node, a random selection of binary tests is created. Each test is evaluated on each training sample in the training set H, resulting in a split of output values $\{f_i, i \in H\}$ into two disjoint sets $H_L = \{f_n : n \in H, b_r(g_n) = 0\}$ and $H_R = \{f_n : n \in H, b_r(g_n) = 1\}$. Each test thus defines a split of the input space into two parts. One of these tests is then chosen based on some criterion, often the information gained by the split. Denoting by IG the information gain and by $\mathcal{H}$ the entropy, the corresponding fomulae are

$$IG = \mathcal{H}(H) - \frac{|H_L|}{|H|}\mathcal{H}(H_L) - \frac{|H_R|}{|H|}\mathcal{H}(H_R) \tag{2.14}$$

$$= \log(|\operatorname{cov}(H)|) - \frac{|H_L|}{|H|}\log(|\operatorname{cov}(H_L)|) - \frac{|H_R|}{|H|}\log(|\operatorname{cov}(H_R)|) \quad . \tag{2.15}$$

Training of the left and right child is done in the same way using H_L and H_R instead of H, respectively. This way, the training set (and with it the input space) is consecutively split, until either a maximum tree depth is reached, or the information gained by splitting does not grow any more because the remaining f_n are all very similar or too small in number. An example of a training process is visualized in Fig. 2.5.

Since the number of free parameters of even a single regression tree scales with the number of nodes, regression in a single tree is prone to over-fitting. By combining estimates from different trees this problem is avoided, given that the trees are sufficiently

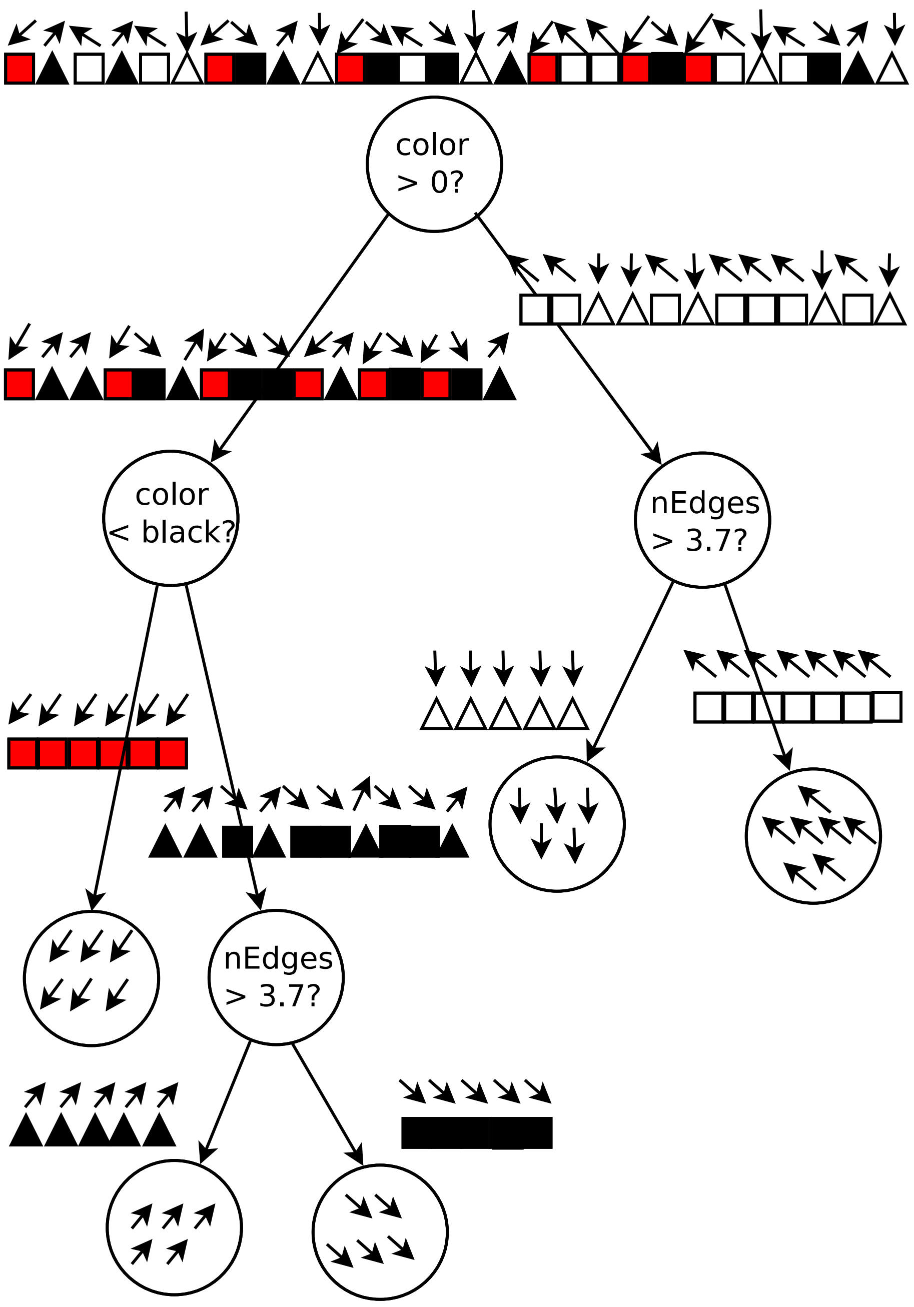


Figure 2.5.: Example of training a simple random regression tree. The set of all training samples (squares and triangles) with their ground truth labels (orientations indicated by arrows), is used to find a binary test for the root node (top). The best split of ground truth orientations is reached by splitting white and colored shapes. For the left child one now has to find a good way to split orientations of all the colored squares and triangles, resulting again in color as feature. The left child of this node now does not need to find another split since the training data that reaches it is already very homogeneous (arrows to the lower left), so a leaf is created with those orientations. This procedure is repeated for all other right child nodes until all the training data is contained in a leaf.

different. This is why different training subsets H are selected for each tree, and why binary tests for each node are chosen in a suboptimal fashion.

Regression trees are not necessarily balanced, although good splitting criteria favor an approximately equal splitting of training data at each node. Newer studies (Gall and Lempitsky [2009], Bernard et al. [2012]) select the training subsets H based on cross-validation performance of already trained trees. A theoretical analysis of random regression forests is given in Biau [2012]. Random forests are used for regression in chapter 6 and chapter 7.

2.5. Kernel Density Estimation

Density estimation has the goal of estimating an unknown probability density function. In contrast to regression, however, one is not given pairs of positions and associated values, but samples from the function, i.e. a set $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{\boldsymbol{N}}\}$ of values with values occurring more frequently at positions where the function values are higher.

One way to form a piecewise constant approximation of a density is by calculating a histogram of samples $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{\boldsymbol{N}}$. Often, however, a continuous estimate is needed. Kernel density estimation is a way to create such a continuous approximation to a probability density. It represents the density as a sum of kernel functions k approximating the density values by a local mean of kernel values. It has various applications including tracking (Comaniciu et al. [2003]). In this thesis, kernel density estimation is used to interpret output of random regression forests in chapter 6. Maxima of kernel density estimation are calculated in chapter 6 and chapter 7.

Given N random samples $\boldsymbol{x}_n$ from an unknown density function, and a kernel k one can imagine replacing each sample $\boldsymbol{x}_n$ with the given kernel k centered at $\boldsymbol{x}_n$, summing up the resulting function and normalizing for a probability density. In formulae this is given by

$$f(\boldsymbol{x}) = \text{const} \sum_{n=1}^{N} k(\boldsymbol{x} - \boldsymbol{x}_n) \quad . \tag{2.16}$$

with a constant chosen such that $\int_x f(\boldsymbol{x}) d\boldsymbol{x} = 1$. Evaluated at any $\boldsymbol{x}$ the result is the mean over kernel function values evaluated at positions that depend on the distance of the corresponding samples to $\boldsymbol{x}$.

Kernel functions k typically have global maximum in the origin and decrease monotonically in all directions. Further properties of the kernel function determine properties of the kernel density estimate. Continuous or differentiable kernels, for example, result in continuous or differentiable f, respectively. The speed with which the kernel function falls off determines the rate of change of the kernel density.

A typical choice for the kernel k is the Gaussian with bandwidth = covariance matrix $\boldsymbol{B}$ and factor const $= 1/N$:

$$k_1(\boldsymbol{x}) = (2\pi)^{-d/2}|\boldsymbol{B}|^{-1/2}\exp\left(-\frac{1}{2}\boldsymbol{x}^T B^{-1}\boldsymbol{x}\right) \quad . \tag{2.17}$$

2.5.1. Application to Weighted Samples in a Mixed Circular and Linear Space

In this thesis, kernel density estimation is applied to weighted samples that are defined on either a linear or a circular space. However, we originally had planned to apply it on mixed space $\mathbb{X}$ of circular and linear variables, and therefore derived equations for this general case. Since the purely linear and purely circular case are special cases of the mixed case, the following derivation applies to both problems. Application of circular methods to kernel density estimation can also be found in the literature (e.g. Di Marzio et al. [2011])

For applying kernel density estimation to the mixed space $\mathbb{X}$, a mixture of kernels is used, a multivariate Gaussian for the linear dimensions and a multivariate wrapped Gaussian in the circular dimensions. Before deriving equations for this case, we simplify notation for adding multiples of 2π to the circular dimensions of samples $\boldsymbol{x} \in \mathbb{X}$. To do so, we denote by d the number of circular variables and by $i_1, \ldots, i_d$ their indices in elements $\boldsymbol{x} \in \mathbb{X}$. We now define the space $\hat{\mathbb{Z}}_d$, which is isomorphic to $\mathcal{Z}^d$, the space of all integer-valued vectors of size d. The difference between elements $\hat{\boldsymbol{k}} \in Y_d$ and $\boldsymbol{k} \in \mathcal{Z}^d$ is that $\hat{\boldsymbol{k}}$ has the same number of elements as $\boldsymbol{x} \in \mathbb{X}$ but has entries set only for the dimensions in which $\mathbb{X}$ is circular. Vector entries for linear dimensions are 0. This means that the i-th entry of $\hat{\boldsymbol{k}}$ is

$$\hat{\boldsymbol{k}}^{(\mathrm{i})} = \begin{cases} \boldsymbol{k}^{(\mathrm{j})} & \mathrm{i}{=}i_j \\ 0 & i{\notin}\ \{i_1, \ldots, i_d\} \end{cases} \quad . \tag{2.18}$$

This way, we can add different multiples of 2π to the circular dimensions of samples $\boldsymbol{x} \in \mathbb{X}$ as $\boldsymbol{x} + 2\pi\hat{\boldsymbol{k}}$ without affecting the linear dimensions.

We can now define partly-wrapped Gaussian kernels k with bandwidth matrix $\boldsymbol{B}$ on this mixed space $\mathbb{X}$ as

$$k(\boldsymbol{x}) = \text{const} \sum_{\hat{\boldsymbol{k}} \in \hat{\mathcal{Z}}_d} \exp\left(-\frac{1}{2}(\boldsymbol{x} - 2\pi\hat{\boldsymbol{k}})^T \boldsymbol{B}^{-1} (\boldsymbol{x} - 2\pi\hat{\boldsymbol{k}})\right) \quad , \tag{2.19}$$

with normalization constant const chosen such that $\int_{\mathbb{X}} k(\boldsymbol{x}) d\boldsymbol{x} = 1$

If the covariance matrix $\boldsymbol{B}$ contains no covariances between linear and circular dimensions, this formulation is equivalent to splitting the space $\mathbb{X}$ into linear and circular dimensions $\boldsymbol{x} = \left[\boldsymbol{x}^{(\text{lin})}, \boldsymbol{x}^{(\text{circ})}\right]^T$ and defining the kernel as a product of a regular and a wrapped Gaussian. However, the more general equation 2.19 will be used in the following.

As a further extension of the standard formulation, samples can have an associated weight that specifies an importance of each sample relative to the others. Putting all together, the kernel density estimate given samples $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N \in \mathbb{X}$ with weights $w_1, \ldots, w_N$ is

$$\begin{aligned} f(\boldsymbol{x}) &= \frac{1}{N} \sum_{n=1}^{N} w_n k(\boldsymbol{x} - \boldsymbol{x}_n) \\ &= \frac{\text{const}}{N} \sum_{n=1}^{N} w_n \sum_{\hat{\boldsymbol{k}} \in \hat{\mathcal{Z}}_d} \exp\left(-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{x}_n - 2\pi\hat{\boldsymbol{k}})^T \boldsymbol{B}^{-1} (\boldsymbol{x} - \boldsymbol{x}_n - 2\pi\hat{\boldsymbol{k}})\right) \quad . \end{aligned} \tag{2.20}$$

2.5.2. Finding Local Maxima

For evaluation, local maxima of a kernel density estimate f are needed in chapter 6 and chapter 7. One could of course use general optimization techniques for this purpose. However, since these estimates have to be found very efficiently, we derive in this subsection an iterative gradient ascent procedure for finding local maxima for the special form of f, which can be efficiently computed for several initializations in parallel. The equations bear some resemblance to mean shift clustering (Comaniciu and Meer [1999]), but have two important differences which are pointed out in the corresponding section below (sec. 2.6.2).

A local maximum of f fulfills $\frac{\partial f}{\partial \boldsymbol{x}} = 0$. Calculating the partial derivative of equation 2.20 and abbreviating

$$v_{n,k} = \exp\left(-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{x}_n - 2\pi\hat{\boldsymbol{k}})^T \boldsymbol{B}^{-1} (\boldsymbol{x} - \boldsymbol{x}_n - 2\pi\hat{\boldsymbol{k}})\right) \tag{2.21}$$

yields

$$0 \overset{!}{=} \frac{\partial f}{\partial \boldsymbol{x}} \tag{2.22}$$

$$= \frac{\text{const}}{N} \sum_{n=1}^{N} w_n \sum_{\hat{\boldsymbol{k}} \in \hat{\mathcal{Z}}_d} \exp\Big(-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{x}_n - 2\pi\hat{\boldsymbol{k}})^T \boldsymbol{B}^{-1}(\boldsymbol{x} - \boldsymbol{x}_n - 2\pi\hat{\boldsymbol{k}})\Big) \times$$

$$\times \frac{\boldsymbol{B}^{-1}(\boldsymbol{x} - \boldsymbol{x}_n - 2\pi\hat{\boldsymbol{k}})}{2} \tag{2.23}$$

$$= \frac{\text{const}}{N} \sum_{n=1}^{N} w_n \sum_{\hat{\boldsymbol{k}} \in \hat{\mathcal{Z}}_d} v_{n,k} \frac{\boldsymbol{B}^{-1}(\boldsymbol{x} - \boldsymbol{x}_n - 2\pi\hat{\boldsymbol{k}})}{-2} \tag{2.24}$$

$$\Leftrightarrow \frac{\text{const}}{N} \sum_{n=1}^{N} w_n \sum_{\hat{\boldsymbol{k}} \in \hat{\mathcal{Z}}_d} v_{n,k} \frac{\boldsymbol{B}^{-1}\boldsymbol{x}}{-2} = \frac{\text{const}}{N} \sum_{n=1}^{N} w_n \sum_{\hat{\boldsymbol{k}} \in \hat{\mathcal{Z}}_d} v_{n,k} \frac{\boldsymbol{B}^{-1}(\boldsymbol{x}_n - 2\pi\hat{\boldsymbol{k}})}{-2} \tag{2.25}$$

$$\Leftrightarrow \boldsymbol{x} = \frac{\sum_{n=1}^{N} w_n \sum_{\hat{\boldsymbol{k}} \in \hat{\mathcal{Z}}_d} v_{n,k}(\boldsymbol{x}_n - 2\pi\hat{\boldsymbol{k}})}{\sum_{n=1}^{N} w_n \sum_{\hat{\boldsymbol{k}} \in \hat{\mathcal{Z}}_d} v_{n,k}} \quad . \tag{2.26}$$

Note that this is not a closed-form solution for the local maximum $\boldsymbol{x}$ since expression 2.21 for $v_{n,k}$ still contains the unknown $\boldsymbol{x}$. However, equations 2.21 and 2.26 define an iterative procedure that converges towards a local optimum of the density 2.20. Starting this procedure at different positions (e.g., every 10th sample) will result in a list of local optima. A visualization of traces of $\boldsymbol{x}$ can be found in Fig. 2.6.

In practice, it can be beneficial to first evaluate the kernel density f on a regular grid spanning the whole space $\mathcal{C}$, checking for local maxima in the resulting values, and using these as starting points for iteration. This is feasible if the space is not too high-dimensional and the kernel bandwidth, which defines the grid spacing, is not too small.

In post-processing, local minima can be removed by comparing density values at the local optima with values around it (e.g. $\boldsymbol{x} \pm \boldsymbol{B} \times \boldsymbol{1}$). The remaining local maxima can then be unified to unique optima using, for example, a few iterations of mean shift clustering. For computations with reasonable bandwidths ($< \pi/2$) in the circular dimensions, the infinite sums over all integers $\mathcal{Z}$ can be replaced by corresponding finite sums over $\{-k_{\max}, -k_{\max}+1, \ldots, k_{\max}-1, k_{\max}\}$, with $k_{\max} = 3$. Sudderth [2006] reports that even $k_{\max} = 1$ is enough for standard deviations $\sigma < \frac{\pi}{3}$. Whether $k_{\max}$ was chosen big enough can be tested by comparing contributions from terms with $+2\pi k_{\max}$ and $-2\pi k_{\max}$ to the overall influence.

To describe this iterative procedure in words: initialized to some position $\boldsymbol{x}$, all samples $\boldsymbol{x}_n$ in the vicinity of $\boldsymbol{x}$ are found and their center of gravity is calculated, using the kernel to weight close samples more than far samples. Then $\boldsymbol{x}$ is shifted into that local center

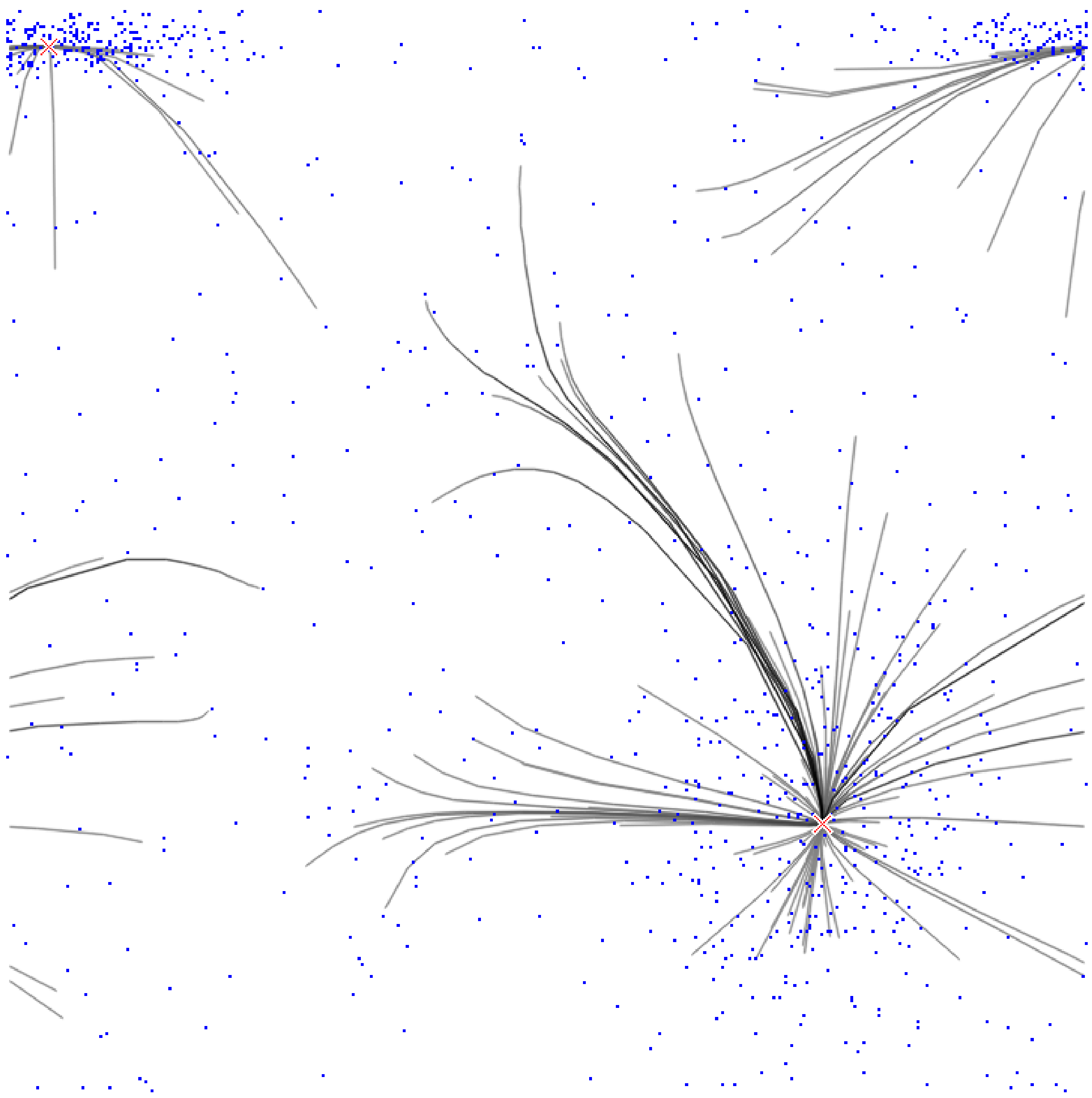


Figure 2.6.: Kernel density maximization in a mixed circular and linear space. Blue dots are samples x_n, grey lines show traces of a few x, their points of convergences marked with red crosses. The space is circular in horizontal direction and linear in the vertical, which can be seen, by the fact that traces starting at the lower left point left, then continue on the right side .

of gravity and the weighted center of gravity around that position is calculated. This is repeated until the center of gravity is so close to $\boldsymbol{x}$ that changes are smaller than a threshold. This will almost always result in $\boldsymbol{x}$ located in a local maximum of sample density, except if $\boldsymbol{x}$ starts at an equilibrium point between two or more denser clusters, such that their influences on it cancel. However, this is case is extremely unlikely since influences would have to cancel out perfectly in order to keep $\boldsymbol{x}$ stationary.

2.6. Clustering

Clustering is the task of assigning each sample $\boldsymbol{x}_n$ to a single index $c_n \in [1, \ldots, C]$, such that the distance between samples with the same index is as small as possible, and the distance between samples with different indices is as large as possible. Such a division of data into several distinct clusters is used, for example. in data compression and for identifying different subgroups of participants in statistical analysis.

The two most prominent approaches to clustering are hierarchical clustering and k-means clustering. In applications in this thesis, however, more powerful clustering approaches were needed, which are presented in the following. Both are intended for application on mixed circular and linear spaces by using wrapped Gaussian distributions.

2.6.1. Gaussian Mixture Model

A Gaussian mixture model (GMM, or Mixture of Gaussians MoG) is a probability distribution, whose density function is a weighted sum of several Gaussian distributions:

$$p(\boldsymbol{x}) = (2\pi)^{-d/2} \sum_{c=1}^{C} w_c |\boldsymbol{\Sigma}_c|^{-1/2} \exp\left((\boldsymbol{x} - \boldsymbol{\mu_c})^T \boldsymbol{\Sigma}_c^{-1} (\boldsymbol{x} - \boldsymbol{\mu_c})^T \right)$$

$$\text{where} \quad \sum_{c=1}^{C} w_c = 1 \quad .$$

Means $\boldsymbol{\mu}_c$ that are far enough apart and covariance matrices $\boldsymbol{\Sigma}_c$ with small eigenvalues give rise to a clustering of samples, if one assigns every sample to the Gaussian c that has the highest probability at it's position. Vice versa, a clustering where each cluster can be approximated by a Gaussian distribution (i.e. has a roughly ellipsoidic shape), can be described by a Gaussian mixture model. Representing a data set in this way allows an easy interpretation of the data: weights w_c specify the relative importance of clusters, means μ_m are the centers of each cluster and covariance matrices $\boldsymbol{\Sigma}_c$ determine orientation and spread of clusters. In this sense, finding a Gaussian mixture model solves a similar problem to kernel density estimation: finding a representation of a density function given samples from that density.

Finding such a representation for a given set of data requires an iterative procedure that is derived, for example, in Bishop [2006]. An implementation of it is being used in sec. 4.1.

The procedure was generalized during this thesis to spaces $\mathcal{C}$ of mixed linear and circular dimensions, similar to kernel density maximization above. Mixture models for circular data using von Mises distributions instead of wrapped Gaussians were also considered. However, the resulting algorithm was slower than using mean shift clustering and not as flexible, and has therefore been replaced with the latter. It is, however, used to cluster condensed depth profiles in chapter 4 on a mutli-dimensional linear space.

2.6.2. Anisotropic Mean Shift Clustering

Mean shift is a mode-seeking procedure that has been applied to tracking, image filtering, segmentation and clustering (Fukunaga and Hostetler [1975], Comaniciu and Meer [2002, 1999]). It is related to kernel density estimation, since it also approximates the local sample density.

The great advantage over most other clustering methods - at least for the purposes of this thesis - is, that the number C of clusters does not have to be defined in advance. The only parameter needed is a kernel function which includes a length scale. Cluster number and length scale are closely related, but with the important difference that specifying C requires knowledge about the structure of the data, while the length scale is more directly related to the structure of the space itself, which can be more easily be determined. Compared to Gaussian mixture models, mean shift clustering makes less assumptions about the form of clusters, which has been argued in Comaniciu and Meer [2002] to limit the applicability of Gaussian mixture models to image processing. Mean shift clustering is used for evaluation in chapter 7.

The basic mean shift algorithm resembles the procedure for finding a maximum in the kernel density described in section sec. 2.5.2, except that samples $\boldsymbol{s}_n$ and potential maxima $\boldsymbol{x}$ are identical. The algorithm runs in parallel on all samples $\boldsymbol{s}_n$ and moves them towards their common (local) concentration maximum until a certain convergence criterion is fulfilled (e.g., motion falls below a threshold).

Formally, let $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N$ be samples in $\mathcal{R}^D$. Again a kernel density specified as

$$f(\boldsymbol{x}) = \frac{1}{N} \sum_{n=1}^{N} k_{\{B\}}(\boldsymbol{x} - \boldsymbol{x}_n) \tag{2.27}$$

is used, where K is a kernel function that depends on a bandwidth matrix $\boldsymbol{B}$. A common assumption in mean shift clustering is, that the kernel is radially symmetric and thus only depends on a scalar *profile function* k and a single bandwidth parameter b:

$$K_b^s(\boldsymbol{x}) = \frac{\text{const}_k}{b^D} k\left(\frac{||\boldsymbol{x}||^2}{b}\right) \tag{2.28}$$

where $||\cdot||$ is the euclidean norm in $\mathcal{R}^D$ and k is defined on R^+. The constant is chosen such that $\int_{\mathcal{R}^D} K(\boldsymbol{x}) d\boldsymbol{x} = 1$. Examples include the Gaussian (normal) kernel k^N and the Epanechnikov kernel k^E:

$$k^N(x) = \exp(-\frac{x}{2}) \tag{2.29}$$

$$k^E(x) = \begin{cases} 1 - x & \text{if} \quad 0 \leq x \leq 1 \\ 0 & \text{otherwise} \end{cases} \tag{2.30}$$

with appropriate constants (c.f. n Comaniciu and Meer [2002]).

In our application, however, the radially-symmetric kernels are not appropriate since the different dimensions of the input space have large differences in scaling, so the kernel would be dominated by those dimensions that have a larger scaling. We therefore follow the example of Wang et al. [2004] and replace the Euclidean norm with a Mahalanobis distance which is defined by a diagonal matrix $\boldsymbol{B} = \text{diag}(\boldsymbol{b}^2)$ with a scaling vector $\boldsymbol{b} = (b_1, \ldots, b_D)$ which allows a different scaling for each dimension:

$$K_{\boldsymbol{b}}(\boldsymbol{x}) = |B|^{-1/2} \text{const}_k \, k(\boldsymbol{x}^T \boldsymbol{B}^{-1} \boldsymbol{x}) \tag{2.31}$$

$$= \frac{\text{const}_k}{\prod_{d=1}^{D} b_d} k\left(\sum_{d=1}^{D} \frac{x_{(d)}^2}{b_d^2}\right) \quad , \tag{2.32}$$

where $x_{(d)}$ is the dth component of the vector $\boldsymbol{x}$. This can easily be generalized to full covariance matrices $\boldsymbol{B}$, but since in this thesis the matrix only compensates for the different scaling of spaces, equations are derived for the diagonal matrix case only. Note also that the bandwidth parameter describes two values here: it specifies the bandwidth of the kernel as well as determines the scaling of the anisotropic distance functions. One could, in general, disentangle those two roles, e.g., for automatic bandwidth selection (see Wang et al. [2004], Comaniciu et al. [2001]). The resulting density estimate with the Gaussian kernel is

$$f(\boldsymbol{x}) = \text{const} \sum_{n=1}^{N} \exp\left(\sum_{d=1}^{D} \frac{(x_{n,(d)} - x_{(d)})^2}{-2b_d^2}\right) \tag{2.33}$$

In analogy to sec. 2.5.1, this kernel can be extended to spaces $\mathcal{C}$ which are a mix of linear and circular dimensions using wrapped Gaussian kernels. Also, samples have associated weights in applications in this thesis. This results in the mean shift density

$$f(\boldsymbol{x}) = \text{const} \sum_{n=1}^{N} w_n \sum_{\hat{\boldsymbol{k}} \in \hat{\mathcal{Z}}_d} \exp\left(\sum_{d=1}^{D} \frac{(x_{n,(d)} - x_{(d)} - 2\pi \hat{\boldsymbol{k}}_{(d)})^2}{-2b_d^2}\right) \quad . \tag{2.34}$$

As with kernel density maximization (sec. 2.5.2), the mean shift procedure is derived by setting $0 \overset{!}{=} \frac{\partial f}{\partial \boldsymbol{x}}$, resulting in an equation for $\boldsymbol{x}$, which still contains $\boldsymbol{x}$ in form of weighting factors v_n:

$$\frac{\partial f}{\partial \boldsymbol{x}} \overset{!}{=} 0 \qquad \Leftrightarrow \qquad \boldsymbol{x} = \frac{\sum_{n=1}^{N} w_n v_n \boldsymbol{x}_n}{\sum_{n=1}^{N} w_n v_n} \tag{2.35}$$

$$\text{with} \qquad v_n = \sum_{\hat{\boldsymbol{k}} \in \hat{\mathcal{Z}}_d} \exp\left(\sum_{d=1}^{D} \frac{(x_{n,(d)} - x_{(d)} - 2\pi \hat{\boldsymbol{k}}_{(d)})^2}{-2b_d^2}\right) \quad . \tag{2.36}$$

This can also be written as

$$\boldsymbol{x}_{\text{new}} - \boldsymbol{x}_{\text{old}} = \frac{\sum_{n=1}^{N} w_n v_n (\boldsymbol{x}_n - \boldsymbol{x}_{\text{old}})}{\sum_{n=1}^{N} w_n v_n} \tag{2.37}$$

which explains the name 'mean shift': $\boldsymbol{x}$ is shifted by a weighted mean of differences to samples $\boldsymbol{x}_n$. Repeatedly applying this equation will move $\boldsymbol{x}$ towards a local maximum in sample density. This equation is now applied to all samples $\boldsymbol{x} = \boldsymbol{x}_n$ in parallel, moving them all at the same time towards their local density optima. A visualization of this procedure can be found in Fig. 2.7.

One can show for linear dimensions that this procedure converges to local density maxima (Comaniciu and Meer [2002]), and that, when using the Gaussian kernel, successive updates will have angular differences of less than $180°$, which means that convergence cannot occur in a zig-zag fashion and should therefore be quick. It is also reported in Comaniciu and Meer [2002] that in practical applications the Epanechnikov kernel leads to faster (but maybe less accurate) convergence.

For post-processing one can select one representative from each created maximum. Assigning all the original samples that have converged in the same optimum to the re-

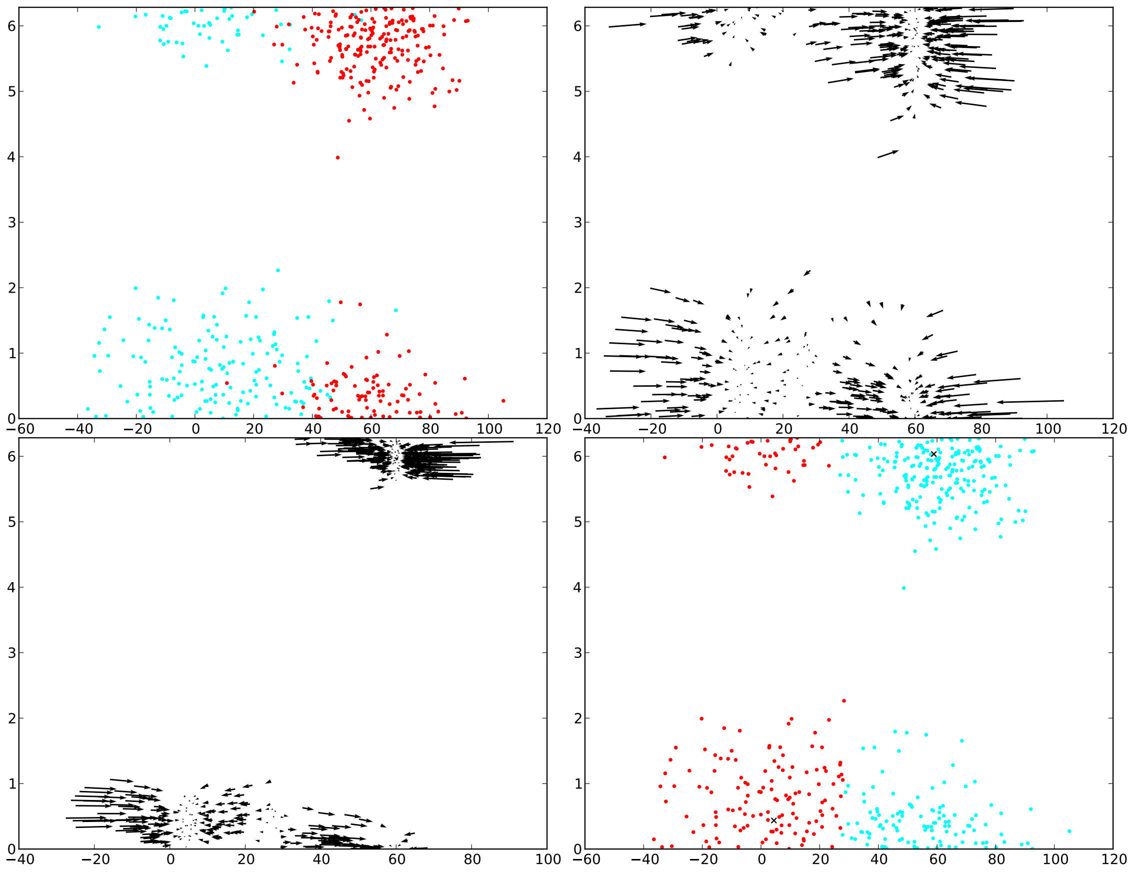


Figure 2.7.: Mean shift clustering procedure on a mixed circular and linear space with different scaling. In this example, the horizontal space dimension is linear, with sample positions ranging from around -40 to 100, while the vertical dimension is circular with the usual range $[0, 2\pi]$. Sample positions are created by sampling from two Gaussian distributions, one centered at position $[5, 0.5]^T$ with diagonal standard deviation $[19, 0.7]^T$, the other at $[60, 6.0]^T$ with diagonal standard deviation $[16, 0.6]^T$, shown in the **top left** plot. The bandwidth of the mean shift procedure is defined by $b_1 = 10$ and $b_2 = 1$. The first two steps of mean shift clustering are shown in the **top right** and **bottom left** plots, with arrows pointing from the old to the updated sample positions. The final clustering result, obtained from only 8 iterations, is shown in the **bottom right** plot.

spective representative then defines a clustering of the input samples. From the number and original positions of samples converged to a certain cluster, one can create a representation similar to the one obtained from Gaussian mixture clustering: the points of convergence are cluster centers, the ratio of samples converged to that center defines the importance of that cluster. If the clusters can be assumed to be unimodal, one can even calculate a covariance of the original positions that have converged to each cluster center, representing the spread and orientation of the cluster in the feature space.

2.7. Particle Filter

All methods presented in this chapter so far are mostly applicable to single images or values derived from them. Often, however, one deals with dynamically changing data, obtained from a stream of images. Most real-world properties change continuously over time, so estimates obtained from neighboring images should also vary continuously. However, estimates can be erroneous due to imperfections in models, image noise or computational reasons like convergence in different local optima.

Dynamic filtering is a simple but efficient way to overcome errors obtained from single images by enforcing temporal consistency onto estimates. Individual estimates are modeled as measurements of an unknown state, accumulating them together with a system model that describes how the state changes over time. Particle filters and extensions of the Kalman filter (Kalman [1960]) are the most popular filters in research on dynamic estimation. While the Kalman filter assumes Gaussian noise for measurements and estimated state, and requires a linear state transition model, particle filter methods use a Monte-Carlo representation of their knowledge of the world state, allowing multiple hypotheses in the filtering density, non-linear state transition models and non-Gaussian noise in measurements. They belong to the family of Monte-Carlo Markov Chain (MCMC) or sequential Monte-Carlo methods (Doucet, Arnaud and de Freitas, Nando and Gordon [2001]). Well-known particle filters are the bootstrap filter and the Condensation (conditional density propagation) algorithm Isard and Blake [1998], with applications including target tracking in clutter, on-line control of autonomous platforms, financial data analysis and many more. A general introduction into Monte-Carlo methods for dynamic filtering is provided in Doucet, Arnaud and de Freitas, Nando and Gordon [2001], a popular shorter overview is provided in Arulampalam et al. [2002]. Convergence results for particle filters are provided in Crisan and Doucet [2002].

Both the Kalman Filter and the particle filter are based on the same principle, which is derived in the first subsection. The second subsection derives the standard particle filter following the publication "A Tutorial on Particle Filters for Online Nonlinear / Non-Gaussian Bayesian Tracking" by Arulampalam and colleagues from 2002 (Arulampalam et al. [2002]). A modification is suggested in the third subsection, followed by a short treatment of particle degeneracy.

The derivations in this section are general enough to be valid for linear and circular variables or mixtures of those. Two applications of these equations for a single circular and a mixture of circular and linear variables are given in chapter 6 and chapter 7.

2.7.1. Bayesian Optimal Filtering

A filter retains a model about the state of a certain variable in form of a probability distribution (the *filtering density*). This distribution is updated by measurements made of the true state and by prediction of changes of that variable over time. This way the filter tries to 'keeps track' of that variable's true state, by trying to keep a maximum in the probability distribution close to the true state of that variable.

Given an unknown state $\boldsymbol{x}_t \in \mathcal{R}^d$ that changes with time, one seeks a series of estimates $\boldsymbol{x}_{0:t} = (\boldsymbol{x}_0, \ldots, \boldsymbol{x}_t)$ representing the belief about that state at every discrete time step up to time t. In order to do so, one further assumes some uncertain knowledge about how that state evolves over time in form of a conditional probability

$$\boldsymbol{x}_{t+1} \sim p(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t) \tag{2.38}$$

and how the state relates to measurements

$$\boldsymbol{y}_t \sim p(\boldsymbol{y}_t|\boldsymbol{x}_t) \quad . \tag{2.39}$$

Often, both these distributions are given as deterministic functions with additive noise

$$\boldsymbol{x}_{t+1} = \boldsymbol{f}_t(\boldsymbol{x}_t, \boldsymbol{u}_t) + \boldsymbol{v}_t \tag{2.40}$$

$$\boldsymbol{y}_t = \boldsymbol{h}_t(\boldsymbol{x}_t, \boldsymbol{u}_t) + \boldsymbol{e}_t \quad , \tag{2.41}$$

where $\boldsymbol{v}_t$ and $\boldsymbol{e}_t$ are zero-mean random variables with independent Gaussian distributions and $\boldsymbol{f}_t, \boldsymbol{h}_t$ are continuous functions. The variable $\boldsymbol{u}_t$ comprises control signals and parameters and is omitted in the following (appearing always as additional variable behind the conditional sign).

This fulfills the requirements of a Markov process, meaning that the state estimate $\boldsymbol{x}_t$ at time t does not depend on states other than $\boldsymbol{x}_{t-1}$, and measurements y_t are conditionally independent given the process $(\boldsymbol{y}_t \perp\!\!\!\perp \boldsymbol{y}_s|\boldsymbol{x}_{0:t}\forall s < t)$

The density, which summarizes knowledge from all measurements $\boldsymbol{y}_{1:t} = (\boldsymbol{y}_1, \dots, \boldsymbol{y}_t)$ up to time t, is denoted $p(\boldsymbol{x}_t|\boldsymbol{y}_{1:t})$. To obtain formulae for an iterative estimation of this filtering density, marginalization is first applied, expressing the prediction density (equation 2.38) in terms of distributions given above and the previous state, which results in the Chapman-Komolgorov equation

$$p(\boldsymbol{x}_{t+1}|\boldsymbol{y}_{1:t}) = \int_{\mathcal{R}^d} p(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t)p(\boldsymbol{x}_t|\boldsymbol{y}_{1:t})dx_t \quad . \tag{2.42}$$

The measurement update equation 2.39 can also be expressed in terms of these distributions using the Markov property and Bayes' theorem, as well as the same marginalization trick:

$$\begin{aligned} p(\boldsymbol{x}_t|\boldsymbol{y}_{1:t}) = p(\boldsymbol{x}_t|\boldsymbol{y}_t, \boldsymbol{y}_{1:t-1}) &= \frac{p(\boldsymbol{y}_t|\boldsymbol{x}_t, \boldsymbol{y}_{1:t-1})p(\boldsymbol{x}_t|\boldsymbol{y}_{1:t-1})}{p(\boldsymbol{y}_t|\boldsymbol{y}_{1:t-1})} \\ &= \frac{p(\boldsymbol{y}_t|\boldsymbol{x}_t)p(\boldsymbol{x}_t|\boldsymbol{y}_{1:t-1})}{\int_{\mathcal{R}^d} p(\boldsymbol{y}_t|\boldsymbol{x}_t)p(\boldsymbol{x}_t|\boldsymbol{y}_{1:t-1})dx_t} \quad . \end{aligned} \tag{2.43}$$

This motivates the conceptual approach to implement an iterative procedure that takes the posterior density $p(\boldsymbol{x}_{t-1}|\boldsymbol{y}_{1:t-1})$ of the previous time step, predicts a prior density $p(\boldsymbol{x}_t|\boldsymbol{y}_{1:t-1})$ using equation 2.42 and updates that by measurements using equation 2.43 to get the posterior $p(\boldsymbol{x}_t|\boldsymbol{y}_{1:t})$. Iteration starts with a given initialization prior $p(\boldsymbol{x}_0)$.

Assuming all probabilities involved are Gaussian and functions $\boldsymbol{h}_t$ and $\boldsymbol{g}_t$ are linear, one can give explicit formulae for mean and covariance of the resulting filtering density, which results in the well-known Kalman filter Kalman [1960]. However, assuming Gaussian distributions is not appropriate for applications in this thesis since measurements follow a multi-modal distribution and the state prediction equations are non-linear. We also want to allow more general filtering densities describing two different hypothesis for the same variable, for example.

2.7.2. General Particle Filter

To derive the particle filter, we continue following Arulampalam et al. [2002], where filter equations are derived not for the filtering density $p(\boldsymbol{x}_t|\boldsymbol{y}_{1:t})$, but for densities describing the whole time-span $p(\boldsymbol{x}_{0:t}|\boldsymbol{y}_{1:t})$. These are in a later step simplified to expressions for the filtering density at the current time point $p(\boldsymbol{x}_t|\boldsymbol{y}_{1:t})$. In the next subsection this derivation will be repeated with a modification of our own, that appears minor in equations, but is important in applications.

The key idea of the particle filter is to work with a Monte-Carlo approximation of the density, i.e., a set of $N \gg 1$ weighted samples $\boldsymbol{x}_{0:t,1}, \ldots, \boldsymbol{x}_{0:t,N}$ (called *particles*):

$$p(\boldsymbol{x}_{0:t}|\boldsymbol{y}_{1:t}) \approx p_N(\boldsymbol{x}_{0:t}|\boldsymbol{y}_{1:t}) = \sum_{n=1}^{N} w_t^{(n)} \delta(\boldsymbol{x}_{0:t} - \boldsymbol{x}_{0:t}^{(n)}) \tag{2.44}$$

where the *weights* $w_t^{(n)}$ fulfill

$$\sum_{n=1}^{N} w_t^{(n)} = 1, \qquad \text{and} \qquad w_t^{(n)} \geq 0 \, \forall n \quad , \tag{2.45}$$

and δ is the Dirac delta distribution. Note that each particle $\boldsymbol{x}_{0:t}^{(n)}$ represents a whole trajectory in time.

To be mathematically more precise, the particles and weights form a random measure. This implies that common notions associated with (density) functions, like “smoothness” or “multi-modality”, cannot easily be applied to such particle distributions. They can, however, be applied to a probability distribution, that could produce the given particles as samples with high probability. One way to obtain such a distribution is kernel density estimation (see sec. 2.5) which 'replaces' the delta distribution by a smooth kernel function with a small bandwidth that depends on the sample distances.

Assuming a known approximation to the posterior density at time point $t-1$

$$p(\boldsymbol{x}_{0:t-1}|\boldsymbol{y}_{1:t-1}) \approx p_N(\boldsymbol{x}_{0:t-1}|\boldsymbol{y}_{1:t-1}) = \sum_{n=1}^{N} w_{t-1}^{(n)} \delta(\boldsymbol{x}_{0:t-1}, \boldsymbol{x}_{0:t-1}^{(n)}) \tag{2.46}$$

one would like to obtain samples from the posterior density $p(\boldsymbol{x}_{0:t}|\boldsymbol{y}_{1:t})$ which includes the new time point t. However, the expressions for prediction and update as given in equations 2.42 and 2.43 are usually intractable or hard to sample from. Therefore, one resorts to *importance sampling*. Instead of sampling from the posterior, samples $\boldsymbol{x}_{0:t,1}, \ldots, \boldsymbol{x}_{0:t,N}$ are drawn from a *proposal distribution* (or *importance distribution*) $q(\boldsymbol{x}_{0:t}|\boldsymbol{y}_{0:t})$, that has the same or a greater support than the posterior density (i.e., $p(\boldsymbol{x}_{0:t}|\boldsymbol{y}_{1:t}) > 0 \Rightarrow q(\boldsymbol{x}_{0:t}|\boldsymbol{x}_{0:t}) > 0$). The error made by sampling from the wrong distribution is corrected for by weighting the resulting samples with *importance weights*

$$w_t^{(n)} = \frac{p(\boldsymbol{x}_{0:t}^{(n)}|\boldsymbol{y}_{1:t})}{q(\boldsymbol{x}_{0:t}|\boldsymbol{y}_{1:t})} \quad , \tag{2.47}$$

resulting in an approximation

$$p(\boldsymbol{x}_{0:t}|\boldsymbol{y}_{1:t}) \approx p_N(\boldsymbol{x}_{0:t}|\boldsymbol{y}_{1:t}) = \sum_{n=1}^{N} w_t^{(n)} \delta(\boldsymbol{x}_{0:t} - \boldsymbol{x}_{0:t}^{(n)}) \tag{2.48}$$

which has been shown in Crisan and Doucet [2002] to be a valid approximation (distribution converges almost certainly to the correct distribution as N tends to infinity). This has the additional advantage that one needs to evaluate the posterior density only up to a constant, since the importance weights can easily be normalized to a sum of 1 afterwards.

Assuming the proposal distribution was chosen such that the denominator is easy to evaluate, it remains to derive an expression for the numerator. This is done repeating the steps in equations 2.43 and 2.42, thus combining prediction and measurement in one step:

$$\begin{aligned} p(\boldsymbol{x}_{0:t}|\boldsymbol{y}_{1:t}) = p(\boldsymbol{x}_{0:t}|\boldsymbol{y}_t, \boldsymbol{y}_{0:t-1}) &\overset{\dagger}{=} \frac{p(\boldsymbol{y}_t|\boldsymbol{x}_{0:t}, \boldsymbol{y}_{0:t-1}) p(\boldsymbol{x}_{0:t}|\boldsymbol{y}_{0:t-1})}{p(\boldsymbol{y}_t|\boldsymbol{y}_{0:t-1})} \\ &\overset{\ddagger}{=} \frac{p(\boldsymbol{y}_t|\boldsymbol{x}_{0:t}, \boldsymbol{y}_{0:t-1})}{p(\boldsymbol{y}_t|\boldsymbol{y}_{0:t-1})} p(\boldsymbol{x}_t|\boldsymbol{x}_{0:t-1}, \boldsymbol{y}_{0:t-1}) p(\boldsymbol{x}_{0:t-1}|\boldsymbol{y}_{0:t-1}) \\ &\overset{\Diamond}{=} \frac{p(\boldsymbol{y}_t|\boldsymbol{x}_{0:t})}{p(\boldsymbol{y}_t|\boldsymbol{y}_{0:t-1})} p(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}) p(\boldsymbol{x}_{0:t-1}|\boldsymbol{y}_{0:t-1}) \\ &\propto p(\boldsymbol{y}_t|\boldsymbol{x}_{0:t}) p(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}) p(\boldsymbol{x}_{0:t-1}|\boldsymbol{y}_{0:t-1}) \end{aligned} \tag{2.49}$$

where Bayes' theorem is applied in †, the chain rule for conditional densities in ‡, and the Markov property in $\Diamond$.

The (unnormalized) importance weights (equation 2.47) therefore are

$$\tilde{w}_t^{(n)} \propto \frac{p(\boldsymbol{x}_{0:t}^{(n)}|\boldsymbol{y}_{1:t})}{q(\boldsymbol{x}_{0:t}^{(n)}|\boldsymbol{y}_{1:t})} \tag{2.50}$$

$$= \frac{p(\boldsymbol{y}_t|\boldsymbol{x}_{0:t}^{(n)}) p(\boldsymbol{x}_t^{(n)}|\boldsymbol{x}_{t-1}^{(n)}) p(\boldsymbol{x}_{0:t-1}^{(n)}|\boldsymbol{y}_{0:t-1})}{q(\boldsymbol{x}_{0:t}^{(n)}|\boldsymbol{y}_{1:t})} \quad . \tag{2.51}$$

Note that particle values are used both in front and behind the conditional sign "|" since these equations are derived for particle trajectories, and the new state $\boldsymbol{x}_t^{(n)}$ therefore depends on $\boldsymbol{x}_{t-1}^{(n)}$ and no other $\boldsymbol{x}_t^{(m)}, m \neq n$.

If the proposal density q was chosen such that it factorizes

$$q(\boldsymbol{x}_{0:t}|\boldsymbol{y}_{1:t}) = q(\boldsymbol{x}_t|\boldsymbol{x}_{0:t-1}, \boldsymbol{y}_{1:t}) q(\boldsymbol{x}_{0:t-1}|\boldsymbol{y}_{1:t-1}) \tag{2.52}$$

the weight update 2.51 can be simplified to

$$\tilde{w}_t^{(n)} = \frac{p(\boldsymbol{y}_t|\boldsymbol{x}_{0:t}^{(n)})p(\boldsymbol{x}_t^{(n)}|\boldsymbol{x}_{t-1}^{(n)})p(\boldsymbol{x}_{0:t-1}^{(n)}|\boldsymbol{y}_{0:t-1})}{q(\boldsymbol{x}_{0:t}^{(n)}|\boldsymbol{y}_{1:t})} \tag{2.53}$$

$$= \frac{p(\boldsymbol{y}_t|\boldsymbol{x}_{0:t}^{(n)})p(\boldsymbol{x}_t^{(n)}|\boldsymbol{x}_{t-1}^{(n)})}{q(\boldsymbol{x}_t^{(n)}|\boldsymbol{x}_{0:t-1},\boldsymbol{y}_{1:t})} \frac{p(\boldsymbol{x}_{0:t-1}^{(n)}|\boldsymbol{y}_{0:t-1})}{q(\boldsymbol{x}_{0:t-1}^{(n)}|\boldsymbol{y}_{1:t-1})} \tag{2.54}$$

$$= \frac{p(\boldsymbol{y}_t|\boldsymbol{x}_{0:t}^{(n)})p(\boldsymbol{x}_t^{(n)}|\boldsymbol{x}_{t-1}^{(n)})}{q(\boldsymbol{x}_t^{(n)}|\boldsymbol{x}_{0:t-1},\boldsymbol{y}_{1:t})} w_{t-1}^{(n)} \quad , \tag{2.55}$$

finally allowing a recursive update.

Only now, one makes the simplifying step to consider particles $\boldsymbol{x}_t$ for the current time only, instead of trajectories $\boldsymbol{x}_{0:t}$ that also 'live' in time. This simplification is often necessary since with time the sampling of trajectories becomes more and more time-consuming, and an update of estimates for much earlier time steps is usually not required. To express weights for particles $\boldsymbol{x}_t^{(n)}$ given weighted particles $\boldsymbol{x}_{t-1}^{(n)}$, the weight update equation is obtained just like 2.55 above, with the additional assumption that the proposal distribution q depends on the previous state $\boldsymbol{x}_{t-1}$ (as opposed to the whole history $\boldsymbol{x}_{0:t-1}$) and the current measurement: $q = q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1},\boldsymbol{y}_t)$. This results in the new weights

$$\tilde{w}_t^{(n)} = \frac{p(\boldsymbol{y}_t|\boldsymbol{x}_t^{(n)})p(\boldsymbol{x}_t^{(n)}|\boldsymbol{x}_{t-1}^{(n)})}{q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}^{(n)},\boldsymbol{y}_t)} w_{t-1}^{(n)} \quad , \tag{2.56}$$

which are then re-normalized to $w_t^{(n)} = \tilde{w}_t^{(n)} / \sum_{m=1}^{N} \tilde{w}_t^{(m)}$. The approximation to the new posterior density is then given as

$$p(\boldsymbol{x}_t|\boldsymbol{y}_{1:t}) \approx p_N(\boldsymbol{x}_t|\boldsymbol{y}_{1:t}) = \sum_{n=1}^{N} w_t^{(n)} \delta(\boldsymbol{x}_t - \boldsymbol{x}_t^{(n)}) \quad . \tag{2.57}$$

A convenient choice for q is the $q = p(\boldsymbol{x}_t|\boldsymbol{x}_{t-1})$, which simplifies sampling to $\boldsymbol{x}_t^{(n)} = \boldsymbol{f}(\boldsymbol{x}_{t-1}^{(n)}) + \boldsymbol{v}_{t-1}$ and results in importance weights

$$\tilde{w}_t^{(n)} = p(\boldsymbol{y}_t|\boldsymbol{x}_t^{(n)}) w_{t-1}^{(n)} \quad . \tag{2.58}$$

This last equation is used in the *bootstrap* filter, one of the simplest and most efficient particle filter methods. However, since the proposal density is independent of the measurements, this method is often not the best choice.

The derivation of particle filter equations up to this point was based on Arulampalam et al. [2002].

2.7.3. Sample Insertion

In some applications it is beneficial to maintain multiple modes in the filtering density, corresponding to two or more competing hypotheses about the state of the variable. Although particle filters can in general handle multi-modal densities, they usually quickly converge to one mode and track that. If in the beginning the wrong mode receives more support by measurements, this may result in a consistent but wrong filtering result.

To encourage the survival of several modes in applications where not only a single measurement $\boldsymbol{y}_t$ but a whole distribution of measurements is available, we modify the derivations from Arulampalam et al. [2002] described above. We suggest using the proposal density

$$q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}, \boldsymbol{y}_t) = (1-\alpha)\, p(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}) + \alpha\, p(\boldsymbol{x}_t|\boldsymbol{y}_t) \quad , \tag{2.59}$$

which replaces a given ratio $\alpha \in [0, 1]$ of particles with new particles drawn from the current measurement distribution. If the state of the variable $\boldsymbol{x}_t$ is ambiguous, then the measurement distribution will reflect this by having multiple modes. Sampling from $p(\boldsymbol{x}_t|\boldsymbol{y}_t)$ will create samples from all modes which will give support to multiple modes in the sample distribution given a large enough value of α. On the other hand, small values of \alpha i.e., a smaller number of samples from the measurement, will usually only be enough to slightly influence the main mode of the filtering distribution, not to maintain further modes. The optimal choice of this ratio depends on the reliability of measurements and the probability of getting ambiguous measurements in the concrete application. A similar procedure, but without the following modifications to the filter, has been used in Isard and Blake [1998], Curio [2004].

Insertion or replacement of particles might also be desirable to increase the variance of particle positions (as opposed to their weights) by inserting samples from a uniform or other state-independent prior distribution. This could, for example, decrease the chance that a scattered measurement is completely 'lost' because no sample is close enough to benefit from its contribution.

With the particle filter deduced above, the proposal distribution 2.59 will almost certainly result in zero-weight particles for secondary modes since in the weight update equation 2.56 the factor $p(\boldsymbol{x}_t^{(n)}|\boldsymbol{x}_{t-1}^{(n)})$ appears. This means that significant weight is only given to the new particle $\boldsymbol{x}_t^{(n)}$ if it happens to be close to the prediction given its old position $\boldsymbol{x}_{t-1}^{(n)}$. This is unlikely if the particle was sampled from secondary modes of the measurement distribution. However, the factor $p(\boldsymbol{x}_t^{(n)}|\boldsymbol{x}_{t-1}^{(n)})$ does not make sense in this context. It originates from the fact that in the original derivation, $\boldsymbol{x}_t^{(n)}$ and $\boldsymbol{x}_{t-1}^{(n)}$ stem from the same trajectory $\boldsymbol{x}_{0:t}^{(n)}$ and are therefore linked through the state update equation. This link is not given if the new particle comes from the $\alpha p(\boldsymbol{x}_t|\boldsymbol{y}_t)$ part of the proposal distribution - dividing a particle track into independent parts was not intended in the original particle filter model. More appropriate for particles sampled from the measurement would be a modified term $p(\boldsymbol{x}_t^{(n)}|\boldsymbol{x}_{t-1})$, where the variable that the new particle is conditioned on, is not just the one particle $\boldsymbol{x}_{t-1}^{(n)}$, but the whole previous distribution $\boldsymbol{x}_{t-1}$.

This is exactly what one obtains repeating the particle filter derivation from above without the classical approach of drawing particles from trajectories $\boldsymbol{x}_{0:t}$ and simplifying to single-time particles $\boldsymbol{x}_t$ afterwards. Given a proposal density only conditioned on the previous time step, the analog to 2.44 is

$$p(\boldsymbol{x}_t|\boldsymbol{y}_{1:t}) \approx p_N(\boldsymbol{x}_t|\boldsymbol{y}_{1:t}) = \sum_{n=1}^{N} w_t^{(n)} \delta(\boldsymbol{x}_t - \boldsymbol{x}_t^{(n)}) \quad . \tag{2.60}$$

Note that no indices $0:t$ appear for the variables $\boldsymbol{x}$ in any of these equations and that no particles appear as conditioning variables. The simplifying step to concentrate on the new time step only is thus already included here.

Assuming again a known posterior density from the previous time step (similar to 2.46)

$$p(\boldsymbol{x}_{t-1}|\boldsymbol{y}_{1:t-1}) \approx p_N(\boldsymbol{x}_{t-1}|\boldsymbol{y}_{1:t-1}) = \sum_{n=1}^{N} w_{t-1}^{(n)} \delta(\boldsymbol{x}_{t-1}, \boldsymbol{x}_{t-1}^{(n)}) \quad , \tag{2.61}$$

and particles sampled from a proposal distribution $\boldsymbol{x}_t^{(n)} \sim q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}, \boldsymbol{y}_t)$, the weight update equation corresponding to 2.51 then is:

$$\tilde{w}_t^{(n)} \propto \frac{p(\boldsymbol{x}_t^{(n)}|\boldsymbol{y}_{1:t})}{q(\boldsymbol{x}_t^{(n)}|\boldsymbol{x}_{t-1}, \boldsymbol{y}_t)} \tag{2.62}$$

$$= \frac{p(\boldsymbol{y}_t|\boldsymbol{x}_t^{(n)}) p(\boldsymbol{x}_t^{(n)}|\boldsymbol{x}_{t-1}) p(\boldsymbol{x}_{t-1}^{(n)}|\boldsymbol{y}_{t-1})}{q(\boldsymbol{x}_t^{(n)}|\boldsymbol{x}_{t-1}, \boldsymbol{y}_t)} \tag{2.63}$$

using expression 2.49 for the posterior $p(\boldsymbol{x}_t|\boldsymbol{y}_t)$.

This results in the intended difference in the recursive weight update equation

$$\tilde{w}_t^{(n)} = \frac{p(\boldsymbol{y}_t|\boldsymbol{x}_t^{(n)}) p(\boldsymbol{x}_t^{(n)}|\boldsymbol{x}_{t-1})}{q(\boldsymbol{x}_t^{(n)}|\boldsymbol{x}_{t-1}, \boldsymbol{y}_t)} w_{t-1}^{(n)} \quad , \tag{2.64}$$

which is deduced like equation 2.56 above. The term $p(\boldsymbol{x}_t^{(n)}|vx_{t-1})$ can be computed using kernel density estimation, for example (c.f. sec. 2.5).

2.7.4. Particle Degeneracy

Applying any of the particle filter methods for extended numbers of time steps leads to a degeneration of the distribution of weights to a state where are almost all weights are zero, except for very few (or a single) particles which then accumulate all the probability mass. This phenomenon is called *particle degeneracy* and is a well-studied topic in dynamic filtering (Arulampalam et al. [2002], Doucet, Arnaud and de Freitas, Nando and Gordon [2001], Crisan and Doucet [2002]). It is caused, among other reasons, by the usage of an only finite number of particles and errors introduced through simplifying assumptions in modelling. For example, measurements are usually assumed independent, while in fact they stem from the same object and are therefore correlated. This leads to a too high reliance on measurement information, which accumulates over time. Several strategies have been proposed to counter its effect.

One possible strategy is the appropriate choice of the proposal distribution. Another way of avoiding degeneracy is to resample the distribution from time to time. Resampling is the process of creating a new set of particles from an already estimated distribution by creating copies of high-weight particles and omitting low-weight particles. More precisely: the probability of creating a copy of particle $\boldsymbol{x}_t^{(n)}$ in the new set is $w_t^{(n)}$. Giving all new particles the same weight $1/N$ results in a new set of particles that approximate the same distribution.

The bootstrap filter includes a resampling step after every measurement update. Another possibility is to resample after a fixed number of iterations. A more flexible method is estimating the number of particles that still contribute to the estimate, and to resample if that number falls below a threshold. The exact formula is usually impossible to compute. An approximation of this is given by (e.g. Arulampalam et al. [2002]):

$$N_{\text{eff}} \approx \hat{N}_{\text{eff}} = \left(\sum_{n=1}^{N} \left(w_t^{(n)} \right)^2 \right)^{-1} \quad , \tag{2.65}$$

which is easily computable.

Part I.

Dynamic Scene Interpretation from a Moving Platform

CHAPTER 3

Optical Flow Subspaces for Self-Motion Estimation

One of the most important tasks for a mobile agent is navigation. Integral part of this is a notion of self-motion through a dynamic environment. A variety of sensors have been used for this, including feedback from motors, wheel odometers, single cameras, systems of several cameras, inertial measurement units (IMU), global navigation satellite systems (GNSS, e.g. GPS), radar, lidar, or time-of-flight cameras Kammel et al. [2008], Urmson et al. [2008].

Visual odometry, i.e. self-motion estimation using visual input alone, has several appealing properties. Sensors are cheap but capture data for a variety of tasks, allowing integration of existing work on semantic reasoning, for example. Motion is estimated from correspondences in successive frames, either sparse tracks between several frames or dense correspondence sets between two frames (optical flow). Both representations are influenced by at least three different sources: scene geometry, i.e., distance of the tracked point in the scene, self-motion and independent motion of other objects. These are hard to separate due to the ambiguity created when projecting the 3-dimensional scene onto a 2-dimensional image plane. Using stereo or multi-camera setups simplifies isolating contributions from these sources by providing an additional cue to depth from disparity Badino [2007], Lategahn et al. [2012]. Limits on the precision of measurements, however, limits the range of depths that can be accurately determined.

Monocular odometry requires additional constraints and therefore forces vision researchers to concentrate on finding ways to incorporate additional knowledge. Popular constraints

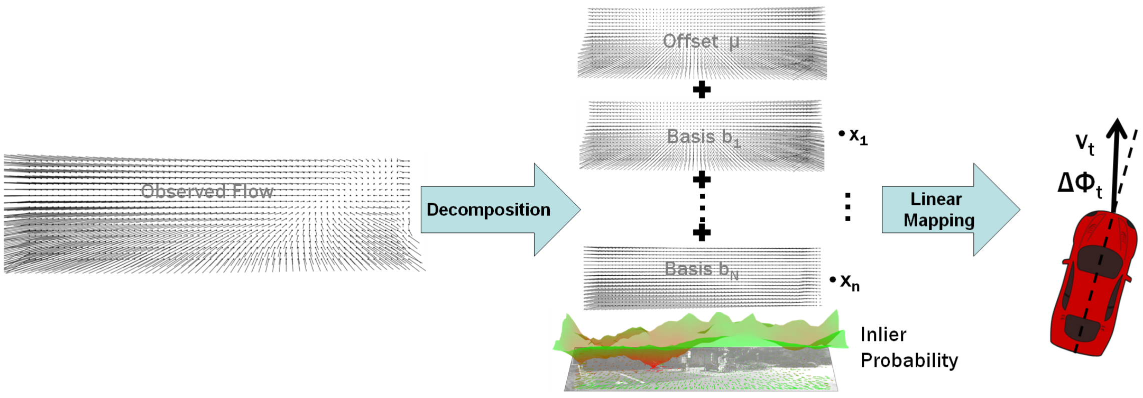


Figure 3.1.: Overview of the approach presented in this chapter. An observed optical flow field (**left**) is decomposed into a linear combination of basis flow fields plus an offset and a contribution of outlier flow (**middle**). The coefficients $x_1, \ldots, x_N$ of this decomposition are then mapped to change in object position and orientation through a linear mapping (**right**).

include a static world and a ground plane that objects are restricted to. Overviews over early approaches and the constraints they used can be found in Tian et al. [1996], Raudies and Neumann [2012]. Newer approaches mostly use tracks of sparse features over several frames and algorithms for structure from motion as well simultaneous localization and mapping (SLAM) like bundle adjustment to identify self-motion and recover static scene geometry simultaneously Nistér and Oleg NarodiBergen [2004], Campbell et al. [2005], Munguia and Grau [2007], Tardif et al. [2008], Scaramuzza et al. [2009]. This, however, is computationally expensive, making real-time application challenging.

In contrast to reasoning about scene geometry and self-motion at once using explicit geometric constraints, the approach described in this chapter learns an implicit representation of the typical geometry encountered in a certain type of scenes. This is learned in an unsupervised manner and allows reasoning about the contribution of self-motion to the optical flow field alone. Violations from the expected geometry are identified and dismissed in a natural way.

An important step in monocular odometry is the identification of feature tracks or optical flow vectors that violate the model assumption of a static world due to independent motion in the scene or erroneous image correspondences. Since Nistér and Oleg NarodiBergen [2004], random sample consensus (Ransac) is a popular scheme to identify sparse feature tracks that do not correspond to the static world or other assumptions (e.g. Scaramuzza et al. [2009]).

By forming a sparse representation of optical flow, this chapter describes an alternative way to simultaneously identify outliers and regularize observer optical flow fields in a natural and probabilistic way. Outliers caused by wrong image correspondences, missing values, moving objects or violations of the expected depth profile are modeled using a probabilistic generative model.

This model has originally been proposed by Roberts et al. [2009] but seems to not have been followed up by the authors afterwards. In practice, it replaces camera calibration with an unsupervised training step, allowing a great flexibility in camera setups: Multi-camera systems, single cameras with strong lens distortions or catadioptric imagery are treated in the same way. Inference does not require initial values and is very quick. The main drawback is the necessity for structured environments, the estimator is specialized on scenes with a depth profile similar to training data (e.g. urban street canyons). It is also not suitable for random camera motion as found in e.g. videos from hand-held cameras, since it is designed for fixed-camera setups like vehicles. A quick overview how to use the subspace model for self-motion estimation is given in Fig. 3.1. To summarize the main model properties:

- Unsupervised learning replaces camera calibration
- Usage of arbitrary cameras setups
- Implicit geometric prior replaces the need for costly explicit geometric reasoning
- Regularization and removal of outliers from various sources through latent variable model
- Easy application to self-motion estimation

This model has also been discussed and evaluated in a diploma thesis Stephan [2010] created in our group. Based on the original model from Roberts et al. [2009], this chapter describes an extension to allow for greater flexibility in flow field interpretation. The original and extended models are analyzed, applied to self-motion estimation, evaluated and compared. To summarize, new contributions of this chapter are:

- Explanation and analysis of the optical flow subspace model
- Model extension to allow for greater flexibility
- Evaluation on a new data set

In the next section, the original model by Roberts et al. is described and discussed. New extensions are presented in the following sec. 3.2. Mapping from model to motion has an interesting theoretical background that was described in Robert's paper and is summarized in sec. 3.4, followed by a short section on model usage in inference (sec. 3.5). Temporal integration and a way to avoid drift is discusses in sec. 3.6. Self-motion estimation results and parameter choices are discussed in sec. 3.8. Finally, a brief summary, conclusion and further research directions are presented in sec. 3.9.

Parts of this chapter have been published in Herdtweck and Curio [2012a] and Herdtweck and Curio [2012b].

3.1. Flow Subspace Model

In his section, the generative subspace model for optical flow representation is presented and discussed. This is one of the main contribution of Roberts et al. [2009]. It is an extension of the probabilistic PCA model (Tipping and Bishop [1999]) with an additional outlier model. It automatically classifies flow vectors violating the model assumption, as well as missing values, flow measurement errors and flow from independently moving objects as outliers ignoring them in further processing. The model is learned in an unsupervised way from observed optical flow fields.

Let $\boldsymbol{f}$ be a vector of J flow field components $\boldsymbol{f} = [f_1, ..., f_J]^T$, where the index represents an entry in the vectorized version of the flow field and thus depends on image position and whether horizontal or vertical components are measured. Each entry can stem from a linear combination of *basis flow vectors* $\boldsymbol{b}_1, ..., \boldsymbol{b}_N$ with a mean flow vector $\boldsymbol{\mu}$, or can be an outlier:

$$f_j = \begin{cases} \mu_j + \sum_{n=1}^{N} x_n b_{nj} + \epsilon_{\text{inl}} & \text{if} z_j = 1 \\ \epsilon_{\text{out}} & \text{if} z_j = 0 \end{cases} . \tag{3.1}$$

z_j is a binary random variable that determines whether a flow component is considered an outlier ($z_j = 0$) that is explained by a zero-mean Gaussian noise, or an inlier ($z_j = 1$) that is explained by a PCA-like model. For the inliers, the coefficient vector $x_i = [x_1, ..., x_N]^T$ determines how much of each basis flow vector is used to approximate the measured flow field. The random variables ϵ_{inl} and ϵ_{out} are noise terms from independent zero-mean Gaussian distributions with standard deviations σ_{inl} / σ_{out} respectively. The former allows some deviations between measured and predicted flow due to slight imprecisions in flow measurement (e.g. quantization artifacts) or model simplifications. The latter models noise as a zero-mean Gaussian variable with usually quite large variance. The space of flow fields that can be caused by observer translation and rotation is called the *flow subspace*, since it defines a subspace of all possible flow vector fields. It is approximated by the space of all linear combinations $x_1\boldsymbol{b}_1 + \ldots + x_N\boldsymbol{b}_N$ of basis flow vectors, which is isomorphic to $\mathbb{X} = \mathcal{R}^N$, the space of all flow subspace coefficient vectors $\boldsymbol{x}$.

The basis flow vectors $\boldsymbol{b}_1, ..., \boldsymbol{b}_N$, flow mean $\boldsymbol{\mu}$ and inlier variance σ_{inl} are learned from training data in an unsupervised fashion using an iterative procedure. The outlier variance is set to a fixed value in the original work, it is learned in this thesis. The exact learning procedure and derivation is detailed in sec. 3.3 below.

On a side note, the term "subspace" in the context of self-motion estimation might be associated with the work of Heeger and Jepson (e.g., Heeger and Jepson [1992]). Their subspace however, is not directly related to the subspace described here.

3.1.1. Interpretation

Since each flow vector component f_j on a stationary object in the scene does not only depend on the observer motion but also on the distance of the object to the observer, the basis flow vectors b_n are also dependent on the depth profile of the scene used in training. For a car driving through urban streets, however, most of the variance in the flow vector fields is caused by the motion and only to a lesser extent by changes in scene geometry. For structured environments like urban streets, scene geometry has many constant components like the ground and the approximate distance to the walls of houses and the left and right side of the street. Since every PCA procedure maximizes the explained variance, this leads at least for lower-dimensional flow subspaces to the implicit assumption that flow vectors at each image position stem from the same depth. This is the constant depth assumption that is implicit in the above formulation. Higher-dimensional flow subspaces should in principle capture more and more of the variance in the flow that is caused by variations in depth.

This implicit assumption limits the applicability of the described approach since flow fields from scenes with a very different geometric arrangement will probably be misinterpreted (see, for example, sec. 3.8.5). However, this limitation can also be seen as a feature. It provides an implicit regularization of flow information and allows an optimal adjustment of the system to its environment. It can also be seen as the analog to camera calibration, which in this approach is fully automatic. In any case, it renders unnecessary all explicit coding of dependencies of flow on geometric structure, intrinsic, or extrinsic camera parameters.

One possible way to visualize this implicitly learned mean depth profile is presented in Fig. 3.2. Shown there are the mean and variance of disparity values over the training set obtained from stereo information.

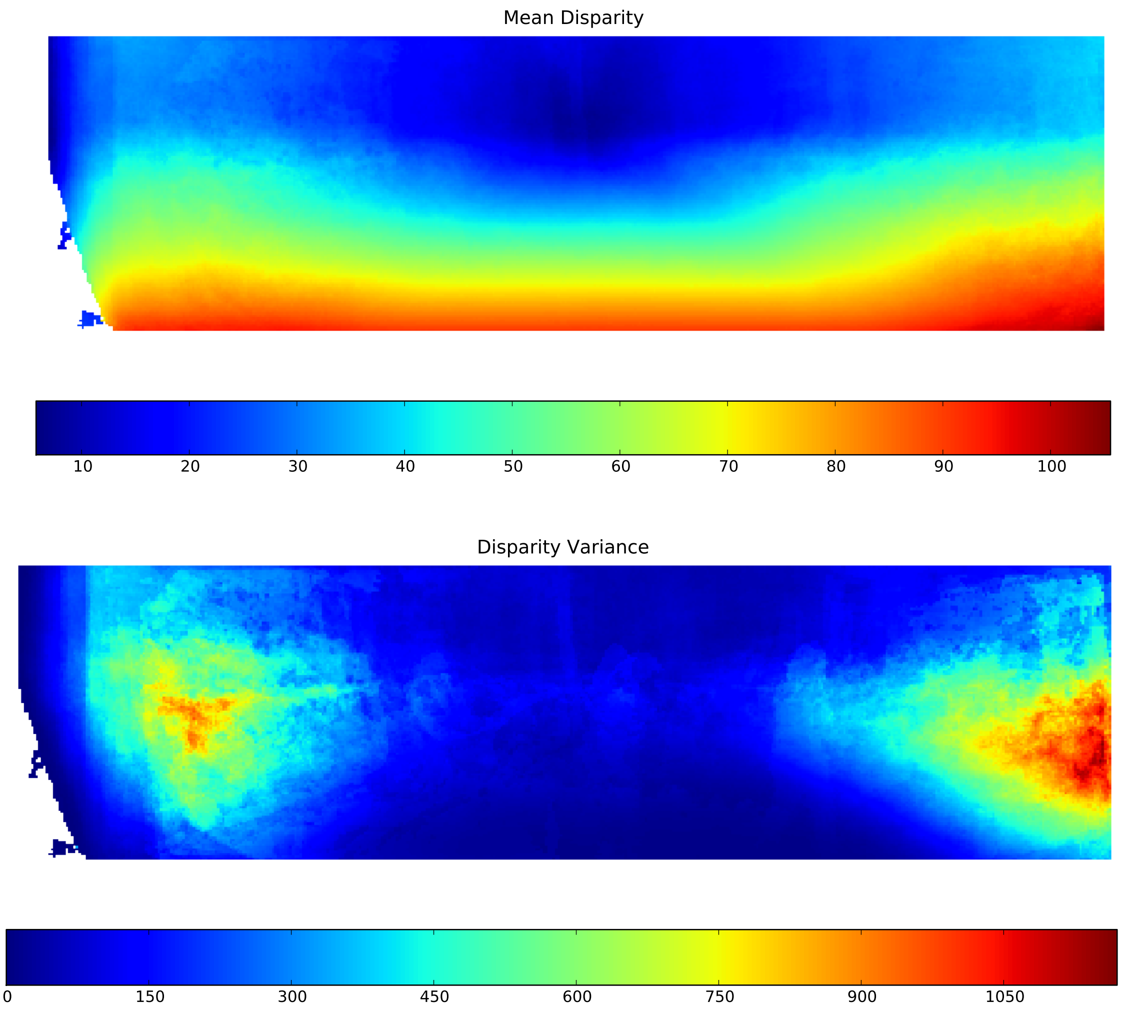


Figure 3.2.: Mean (**top**) and variance (**bottom**) of disparity values calculated from stereo for the data set used in training the estimator.

3.2. Model Extensions

The approach described above was extended in two ways, both of which mainly concern the learning procedure, which is derived in sec. 3.3.

The first extension allows learning not only the variance σ^2_{inl} of the inlier flow components, but also the variance σ^2_{out} of the noise. In the original approach, this has to be specified beforehand, but is reported to be of minor importance. Learning it requires an additional update equation in subspace learning.

We also realized in experiments, that the variance of flow varies greatly across the image. The variance of depth is far smaller on the ground plane than in those image positions where parked cars tend to appear. The presence or absence of a near-by car creates a big change in depth, resulting over all training images in a large depth variation (c.f. Fig. 3.7). Also, since flow vectors for closer objects are much longer than those for distant objects, the variance of flow vectors in image regions that contain close objects (e.g. close to the lower image border in a forward facing camera) are proportionally larger. The variance of depth profiles is visualized in Fig. 3.2 (bottom).

We therefore replaced the homogeneous variance σ^2_{inl} for inliers in the whole image by a inhomogeneous inlier variance $\sigma^2_{\mathsf{j,inl}}$ which can be different for every flow vector component $j = 1, \ldots, J$. The number of additional parameters to learn is exactly the same as an additional flow subspace basis vector b_{N+1} would cause, and can still be handled by the learning procedure. The resulting modifications in the learning procedure are also detailed in the next section.

3.3. Learning the Flow Subspace

Extending the model underlying probabilistic PCA means adding more variables to the model that also have to be estimated. In the present case, the outlier variance σ^2_{out} and variables z_{ij} have to be estimated during training. This makes the training process more complicated because more interdependent variables have to be estimated. We derive in this subsection an Expectation-Maximization (EM, Dempster et al. [1977]) algorithm for training the probabilistic PCA model with outlier term (equation 3.1). Again, we extend on the paper by Roberts and colleagues Roberts et al. [2009], which is based on the derivation of probabilistic PCA in Tipping and Bishop [1999] and uses the maximum likelihood principle. Variables are the same as introduced in equation 3.1 above.

First, one defines the vector of all model parameters: $\theta = (\boldsymbol{\mu}, \boldsymbol{B}, \boldsymbol{\sigma}_{\text{inl}}, \sigma_{\text{out}})$ where $\boldsymbol{\sigma}_{\text{inl}} = (\sigma_{\text{inl},1}, \ldots, \sigma_{\text{inl},J})$ can be constant depending on whether the inlier variance is considered homogeneous or inhomogeneous. We use here a slight inconsistency to the notation above, by denoting by $\boldsymbol{b}_1^T, \ldots, \boldsymbol{b}_J^T$ the *rows* of the basis flow matrix $\boldsymbol{B}$.

For training, one needs observed flow fields $\boldsymbol{f} = (\boldsymbol{f}_1, \ldots, \boldsymbol{f}_I)$, each consisting of J components. While inferring model parameters θ one also estimates coefficients $\boldsymbol{x} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_I)$ (using N subspace dimensions, $\boldsymbol{x}_i \in \mathcal{R}^N$) and inlier/outlier assignments $\boldsymbol{z} = (\boldsymbol{z}_1, \ldots, \boldsymbol{z}_I)$ explaining the training data given the model. Note that by construction of the model (equation 3.1) the flow components f_{ij} are mutually independent given model parameters, coefficients and assignments: $f_{ij} \perp\!\!\!\perp f_{kl} | (\theta, \boldsymbol{x}, \boldsymbol{z}) \quad \forall (i,j) \neq (k,l)$. This allows a simple treatment of outliers by setting $z_{ij} = 0$ (or $z_{ij} = 1$) if f_{ij} is missing, as will be seen in the following.

The distribution of a single measured flow component now is

$$f_{ij} | \theta, \boldsymbol{x}_i, \boldsymbol{z}_i \sim \mathcal{N}\left(f_{ij}; \hat{f}_{ij}, \sigma^2_{\text{inl},j}\right)^{z_{ij}} \times \mathcal{N}\left(f_{ij}; 0, \sigma^2_{\text{out}}\right)^{1-z_{ij}} \quad , \tag{3.2}$$

where

$$\hat{\boldsymbol{f}}_i = \boldsymbol{B}\boldsymbol{x}_i + \boldsymbol{\mu} \tag{3.3}$$

is the clean flow field expected by the model if the flow field was free of noise and outliers and all flow came from a stationary scene with the right depth profile.

One assumes that all model parameters θ are mutually independent, and independent of $\boldsymbol{x}$ and $\boldsymbol{z}$. For $\boldsymbol{x}$ one assumes a Gaussian prior around zero with unit covariance:

$$\boldsymbol{x}_i \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{1}_N) \quad \forall i = 1, \ldots, I \quad . \tag{3.4}$$

Specifying priors for other parameters is not necessary since they do not appear in the following equations. Then, one can formulate the likelihood of all unknowns $\theta, \boldsymbol{x}, \boldsymbol{z}$ using Bayes' rule (equation 2.1) and the mutual conditional independence:

$$p(\theta, \boldsymbol{x}, \boldsymbol{z} | \boldsymbol{f}) = p(\boldsymbol{f} | \theta, \boldsymbol{x}, \boldsymbol{z}) \frac{p(\theta, \boldsymbol{x}, \boldsymbol{z})}{p(\boldsymbol{f})} \tag{3.5}$$

$$= \frac{p(\theta)}{p(\boldsymbol{f})} p(\boldsymbol{x}) p(\boldsymbol{z}) p(\boldsymbol{f} | \theta, \boldsymbol{x}, \boldsymbol{z}) \tag{3.6}$$

$$= \frac{p(\theta)}{p(\boldsymbol{f})} \prod_{i=1}^{I} p(\boldsymbol{x}_i) \prod_{j=1}^{J} p(z_{ij}) p(f_{ij} | \theta, \boldsymbol{x}_i, z_{ij}) \quad . \tag{3.7}$$

One now wishes to find model parameters θ, coefficients $\boldsymbol{x}$ and assignments $\boldsymbol{z}$ that maximize this likelihood. This is equivalent to maximizing the logarithm of a multiple of the likelihood 3.7:

$$\log p(\theta, \boldsymbol{x}, \boldsymbol{z}|\boldsymbol{f}) \propto \mathcal{L}(\theta, \boldsymbol{x}, \boldsymbol{z}|\boldsymbol{f}) = \log \left(\prod_{i=1}^{I} p(\boldsymbol{x}_i) \prod_{j=1}^{J} p(\boldsymbol{z}_{ij}) p(\boldsymbol{f}_{ij}|\theta, \boldsymbol{x}_i, \boldsymbol{z}_{ij}) \right) \tag{3.8}$$

$$= \sum_{i=1}^{I} \left(\log p(\boldsymbol{x}_i) + \sum_{j=1}^{J} \log \left[p(\boldsymbol{z}_{ij}) + \log p(\boldsymbol{f}_{ij}|\theta, \boldsymbol{x}_i, \boldsymbol{z}_{ij}) \right] \right) \tag{3.9}$$

which can be specified further using equations 3.2 and 3.4:

$$\mathcal{L}(\theta, \boldsymbol{x}, \boldsymbol{z}|\boldsymbol{f}) = \sum_{i=1}^{I} \Bigg(\log \mathcal{N}(\boldsymbol{x}_i; \boldsymbol{0}, \boldsymbol{1}) + \sum_{j=1}^{J} \Bigg[\log p(\boldsymbol{z}_{ij}) + z_{ij} \log \mathcal{N}\left(f_{ij}; \hat{f}_{ij}, \sigma^2_{\mathsf{inl},j}\right) + (1 - z_{ij}) \log \mathcal{N}\left(f_{ij}; 0, \sigma^2_{\mathsf{out}}\right) \Bigg] \Bigg) \tag{3.10}$$

Finding a maximum for the log-likelihood (3.10) in closed form is only possible in rare special cases since in general this equation cannot be split further and a maximum therefore depends on all variables. One has to resort to an iterative procedure that optimizes a single variable using the currently most probable value for the other variables. We will in the following equations use the same symbols $\boldsymbol{x}_i$, $\boldsymbol{z}_i$, $\boldsymbol{\mu}$, $\boldsymbol{B}$ as before, but mean by them the mean of the distribution that describes their currently most probable value, i.e., their expectation. An exception is the flow $\boldsymbol{f}$ which is a fixed list of vectors $\boldsymbol{f}_i$, and the covariance of the distribution for $\boldsymbol{x}_i$, whose currently most probable value is denoted by $\left\langle \boldsymbol{x}_i \boldsymbol{x}_i^T \right\rangle$. This implies that the values $\boldsymbol{z}$ are no longer binary, since the expectation of a binary variable is also continuous.

Equations to update individual variables can now be obtained by setting the derivative of $\mathcal{L}$ with respect to that variable to zero. For the j-th component of the mean $\boldsymbol{\mu}$, that

$\hat{f}_{ij}$ depends on through equation 3.3, this results in:

$$0 \overset{!}{=} \frac{\partial}{\partial \mu_j} \mathcal{L}(\theta, \boldsymbol{x}, \boldsymbol{z} | \boldsymbol{f}) = \sum_{i=1}^{I} z_{ij} \frac{\partial}{\partial \mu_j} \log \mathcal{N}\left(f_{ij}; \hat{f}_{ij}, \sigma^2_{\mathsf{inl},j}\right) \tag{3.11}$$

$$= \sum_{i=1}^{I} z_{ij} \frac{\frac{\partial}{\partial \mu_j} \frac{1}{\sqrt{2\pi}\sigma_{\mathsf{inl},j}} \exp\left(\frac{-1}{2\sigma^2_{\mathsf{inl},j}}(f_{ij} - \mu_j - \boldsymbol{b}_j^T \boldsymbol{x}_i)^2\right)}{\mathcal{N}\left(f_{ij}; \hat{f}_{ij}, \sigma^2_{\mathsf{inl},j}\right)} \tag{3.12}$$

$$= \sum_{i=1}^{I} z_{ij} \frac{1}{2\sigma^2_{\mathsf{inl},j}} (f_{ij} - \mu_j - \boldsymbol{b}_j^T \boldsymbol{x}_i) \tag{3.13}$$

$$\Leftrightarrow \quad \sum_{i=1}^{I} z_{ij}(f_{ij} - \boldsymbol{b}_j^T \boldsymbol{x}_i) = \sum_{i=1}^{I} z_{ij}\mu_j \quad \Leftrightarrow \quad \mu_j = \frac{\sum_{i=1}^{I} z_{ij}(f_{ij} - \boldsymbol{b}_j^T \boldsymbol{x}_i)}{\sum_{i=1}^{I} z_{ij}} \quad , \tag{3.14}$$

Similarly for $\boldsymbol{\sigma}_{\mathsf{inl}}$:

$$0 \overset{!}{=} \frac{\partial}{\partial \sigma_{\mathsf{inl},j}} \mathcal{L}(\theta, \boldsymbol{x}, \boldsymbol{z} | \boldsymbol{f}) = \sum_{i=1}^{I} z_{ij} \frac{\partial}{\partial \sigma_{\mathsf{inl},j}} \log \mathcal{N}\left(f_{ij}; \hat{f}_{ij}, \sigma^2_{\mathsf{inl},j}\right) \tag{3.15}$$

$$= \sum_{i=1}^{I} z_{ij} \frac{\frac{\partial}{\partial \sigma_{\mathsf{inl},j}} \frac{1}{\sqrt{2\pi}\sigma_{\mathsf{inl},j}} \exp\left(\frac{-1}{2\sigma^2_{\mathsf{inl},j}}(f_{ij} - \hat{f}_{ij})^2\right)}{\mathcal{N}\left(f_{ij}; \hat{f}_{ij}, \sigma^2_{\mathsf{inl},j}\right)} \tag{3.16}$$

$$= \sum_{i=1}^{I} z_{ij} \frac{\frac{-1}{\sqrt{2\pi}\sigma^2_{\mathsf{inl},j}} \exp\left(\frac{-1}{2\sigma^2_{\mathsf{inl},j}}(f_{ij} - \hat{f}_{ij})^2\right) + \frac{1}{\sqrt{2\pi}\sigma^4_{\mathsf{inl},j}} \exp\left(\frac{-1}{2\sigma^2_{\mathsf{inl},j}}(f_{ij} - \hat{f}_{ij})^2\right)\left(f_{ij} - \hat{f}_{ij}\right)^2}{\mathcal{N}\left(f_{ij}; \hat{f}_{ij}, \sigma^2_{\mathsf{inl},j}\right)}$$

$$= \sum_{i=1}^{I} z_{ij} \frac{1}{\sigma^4_{\mathsf{inl},j}} \left(-\sigma^2_{\mathsf{inl},j} + \left(f_{ij} - \hat{f}_{ij}\right)^2\right) \tag{3.17}$$

$$\overset{\sigma_{\mathsf{inl},j}>0}{\Leftrightarrow} \quad \sum_{i=1}^{I} z_{ij}\sigma^2_{\mathsf{inl},j} = \sum_{i=1}^{I} z_{ij}\left(f_{ij} - \hat{f}_{ij}\right)^2 \quad \Leftrightarrow \quad \sigma^2_{\mathsf{inl}} = \frac{\sum_{i=1}^{I} z_{ij}\left(f_{ij} - \hat{f}_{ij}\right)^2}{\sum_{i=1}^{I}\sum_{j=1}^{J} z_{ij}} \quad . \tag{3.18}$$

When using a homogeneous inlier variance, a single scalar variance σ_{inl} is found by averaging also over all j:

$$\sigma^2_{\mathsf{inl}} = \frac{\sum_{i=1}^{I}\sum_{j=1}^{J} z_{ij}\left(f_{ij} - \hat{f}_{ij}\right)^2}{\sum_{i=1}^{I}\sum_{j=1}^{J} z_{ij}} \quad . \tag{3.19}$$

Update equations for the outlier variance can be obtained in the same way, resulting in

$$\sigma^2_{\mathsf{out}} = \frac{\sum_{i=1}^{I}\sum_{j=1}^{J}(1 - z_{ij}) f^2_{ij}}{\sum_{i=1}^{I}\sum_{j=1}^{J}(1 - z_{ij})} \quad . \tag{3.20}$$

Rows $\boldsymbol{b}_j^T$ of the basis vector fields are updated as:

$$0 \overset{!}{=} \frac{\partial}{\partial \boldsymbol{b}_j^T} \mathcal{L}(\theta, \boldsymbol{x}, \boldsymbol{z}|\boldsymbol{f}) = \sum_{i=1}^{I} z_{ij} \frac{\partial}{\partial \boldsymbol{b}_j^T} \log \mathcal{N}\left(f_{ij}; \hat{f}_{ij}, \sigma^2_{\mathsf{inl},j}\right) \tag{3.21}$$

$$= \sum_{i=1}^{I} z_{ij} \frac{\frac{\partial}{\partial \boldsymbol{b}_j^T} \frac{1}{\sqrt{2\pi}\sigma_{\mathsf{inl},j}} \exp\left(\frac{-1}{2\sigma^2_{\mathsf{inl},j}}(f_{ij} - \mu_j - \boldsymbol{b}_j^T \boldsymbol{x}_i)^2\right)}{\mathcal{N}\left(f_{ij}; \hat{f}_{ij}, \sigma^2_{\mathsf{inl},j}\right)} \tag{3.22}$$

$$= \sum_{i=1}^{I} z_{ij} \frac{\frac{1}{\sqrt{2\pi}\sigma_{\mathsf{inl},j}} \exp\left(\frac{-1}{2\sigma^2_{\mathsf{inl},j}}(f_{ij} - \mu_j - \boldsymbol{b}_j^T \boldsymbol{x}_i)^2\right) \frac{1}{\sigma^2_{\mathsf{inl},j}} \left(f_{ij} - \mu_j - \boldsymbol{x}_i^T \boldsymbol{b}_j\right) \boldsymbol{x}_i^T}{\mathcal{N}\left(f_{ij}; \hat{f}_{ij}, \sigma^2_{\mathsf{inl},j}\right)} \tag{3.23}$$

$$= \sum_{i=1}^{I} z_{ij} \frac{1}{\sigma^2_{\mathsf{inl},j}} \left(f_{ij} - \mu_j - \boldsymbol{x}_i^T \boldsymbol{b}_j\right) \boldsymbol{x}_i^T \tag{3.24}$$

$$\Leftrightarrow \quad \sum_{i=1}^{I} z_{ij} \left(f_{ij} - \mu_j\right) \boldsymbol{x}_i^T = \sum_{i=1}^{I} z_{ij} \left(\boldsymbol{b}_j^T \boldsymbol{x}_i\right) \boldsymbol{x}_i^T \tag{3.25}$$

$$\Leftrightarrow \quad \boldsymbol{b}_j^T = \sum_{i=1}^{I} z_{ij} \left(f_{ij} - \mu_j\right) \boldsymbol{x}_i^T \left(\sum_{i=1}^{I} z_{ij} \left\langle \boldsymbol{x}_i \boldsymbol{x}_i^T \right\rangle\right)^{-1} \quad . \tag{3.26}$$

For updating estimates of coefficients $\boldsymbol{x}$, one does not take a derivative of the log-likelihood (3.10) as for the equations above, but instead applies Bayes's rule (equation 2.1) and inserts the distribution (3.2) to obtain

$$p(\boldsymbol{x}_i|\theta, \boldsymbol{f}_i, \boldsymbol{z}_i) = p(\boldsymbol{f}_i|\theta, \boldsymbol{x}_i, \boldsymbol{z}_i) \frac{p(\boldsymbol{x}_i)}{p(\boldsymbol{f}_i)} \propto p(\boldsymbol{f}_i|\theta, \boldsymbol{x}_i, \boldsymbol{z}_i) p(\boldsymbol{x}_i) \tag{3.27}$$

$$= \mathcal{N}\left(\boldsymbol{f}_i; \hat{\boldsymbol{f}}_i, \sigma^2_{\mathsf{inl},j} \mathbf{1}_J\right)^{\boldsymbol{z}_i} \mathcal{N}\left(\boldsymbol{f}_i; \mathbf{0}, \sigma^2_{\mathsf{out}} \mathbf{1}_J\right)^{\mathbf{1} - \boldsymbol{z}_i} \mathcal{N}\left(\boldsymbol{x}_i; \mathbf{0}, \mathbf{1}_N\right) \quad . \tag{3.28}$$

($\mathbf{1}$ without an index here denotes a vector of 1s). Using a derivation similar to the one for probabilistic PCA (Tipping and Bishop [1999]) and abbreviating $\boldsymbol{\Lambda}_i = \sigma^{-2}_{\mathsf{inl},j} \mathbf{1}_J \boldsymbol{z}_i$ one arrives at update equations for mean and covariance of

$$\boldsymbol{x}_i = \left(\boldsymbol{B}^T \boldsymbol{\Lambda}_i \boldsymbol{B} + \mathbf{1}_N\right)^{-1} \boldsymbol{B}^T \boldsymbol{\Lambda}_i \left(\boldsymbol{f}_i - \boldsymbol{\mu}\right) \tag{3.29}$$

$$\left\langle \boldsymbol{x}_i \boldsymbol{x}_i^T \right\rangle = \left(\boldsymbol{B}^T \boldsymbol{\Lambda}_i \boldsymbol{B} + \mathbf{1}_N\right)^{-1} + \boldsymbol{x}_i \boldsymbol{x}_i^T \quad . \tag{3.30}$$

Finally, the inlier/outlier assignment $\boldsymbol{z}$ is the proportion of $\boldsymbol{f}$ being explained by the inlier distribution relative to the outlier distribution:

$$z_{ij} = \frac{\mathcal{N}\left(f_{ij}; \hat{f}_{ij}, \sigma^2_{\mathsf{inl},j}\right)}{\mathcal{N}\left(f_{ij}; \bar{f}_{ij}, \sigma^2_{\mathsf{inl},j}\right) + \mathcal{N}\left(f_{ij}; \mathbf{0}, \sigma^2_{\mathsf{out}}\right)} \tag{3.31}$$

$$\text{with} \quad \bar{\boldsymbol{f}}_i = \boldsymbol{B}\left(\boldsymbol{B}^T\boldsymbol{B}\right)^{-1}\left(\boldsymbol{B}^T\boldsymbol{B} + \mathbf{1}_N\boldsymbol{\sigma}^2_{\mathsf{inl}}\right)\boldsymbol{x}_i \quad . \tag{3.32}$$

Note that in all these equations the inlier probability z_{ij} (or $1 - z_{ij}$ for σ_{out}) serves as a weighting factor for the contribution from flow component f_{ij}. This allows a straightforward treatment of missing values: they can simply be ignored by setting the corresponding z_{ij} to 0 (or 1 for σ_{out}).

Using equations 3.14, 3.18, 3.20, 3.26, 3.29, and 3.31, one can now update all unknowns using an expectation-maximization procedure. In the expectation step the model parameters $\theta = (\boldsymbol{\mu}, \boldsymbol{B}, \boldsymbol{\sigma}_{\mathsf{inl}}, \sigma_{\mathsf{out}})$ are kept fixed, allowing an update of the variables explaining the training data ($\boldsymbol{x}$, $\left\langle \boldsymbol{x}_i\boldsymbol{x}_i^T \right\rangle$, $\boldsymbol{z}$). In the maximization step the model is updated given the current representation of the training data. Expectation and maximization steps are calculated in turns, until the model does not change any more or a maximum number of iterations is reached. In our implementation we follow the suggestion in Roberts et al. [2009] to stop iterating once the maximum change over all flow components j in $\sigma_{\mathsf{inl},j}$ falls below a threshold, which was chosen to be 10^{-6}. The necessary number of iterations mainly depends on the number N of subspace dimensions and the amount of noise in the flow $\boldsymbol{f}$. Using inhomogeneous inlier variance $\sigma_{\mathsf{inl},j}$ also prolongs computations (c.f. sec. 3.8.6).

Of mainly practical importance is the fact that we speeded up training by initializing the first four subspace dimensions with approximations to a

1. Looming flow field where all vectors face away from the image center with length proportional to their distance to the center
2. Pure sideways translation flow field, roughly corresponding to a yaw in place (i.e., without forward motion)
3. Upwards translation flow field, roughly corresponding to a pitch in place
4. Flow field of rotation around the image middle, roughly corresponding to a roll in place

In our preliminary tests on two-dimensional subspaces, these initializations resulted in the same basis flow fields than those obtained with random initialization, only fast with convergence.

Note that contrary to classical PCA the order of the basis vector fields b does not correspond to the amount of variance explained by the corresponding dimensions. There is therefore no predefined ordering in the returned basis vector matrix.

3.4. Mapping to Motion

In the study Irani [1999], the insight was published that for simple affine cameras, as well as for instantaneous motion models, the flow field caused by motion through a static scene depends linearly on this motion. By modeling the image as a set of single-pixel perspective cameras with smoothly-varying parameters, Roberts et al. [2009] have proven that this result also holds for general camera models and arbitrary motion through static scenes as long as the constant depth assumption holds. This implies, that also the relation between flow subspace coefficients $\boldsymbol{x}$ and observer motion is linear, since coefficients $\boldsymbol{x}$ are based only on those flow vector components where the constant-depth assumption holds. In theory, multiplying the flow field with a factor of two leads to the flow field observed if one were to drive the same route with twice the speed and angular velocities.

Therefore, all that remains to do for estimating self-motion based on optical flow fields, is to find this linear mapping $\boldsymbol{m}$ (c.f. Fig. 3.3). Several regression methods can be used for this (c.f. sec. 2.4). As opposed to subspace learning, however, this step of the training procedure needs ground truth motion data, which has to be acquired using for example GLNSS or IMU sensors. The mapping used in the following, which is predicted by theory to be linear, is found using iteratively re-weighted least squares optimization (Holland and Welsch [1977]).

However, while in theory the mapping from flow subspace to motion is linear, this might not be the case in practice. There are several reasons for this.

One is the distance between the center of rotation of the vehicle and the camera, which leads to small translations associated with every rotation, which are only captured by higher subspace dimensions. For the approach discussed here, the subspace basis flow fields will all be a combination of translational and rotational contributions since this coupling is already inherent in the training data. However, since this coupling is non-linear for faster rotations, this may lead to errors in flow representation for lower-dimensional subspaces and to non-linearities in the relation for higher-dimensional subspaces.

The flow fields used in experiments were calculated from rectified images of a camera with a relatively wide horizontal field of view. These rectifications may lead to non-linearities in the dependence between motion and flow in the periphery.

Another reason for learning a less constrained mapping is the possibility, that certain motions may be correlated with different geometry in the training data, leading to biases in the learned mapping. The scene geometry during a strong right-turn, for example, is associated with a slightly different geometry than a left-turn, or a drive straight ahead. This effect may be small but consistent, leading to non-linearities in the relation between flow-field and motion.

One could imagine further effects like the roll during fast turning maneuvers, or biases from limiting the flow resolution playing a role. While we have not explicitly investigated any of these effects, we tested whether using a non-linear mapping leads to better motion estimation performance than the linear fit. A general optimization procedure was used to fit multi-dimensional polynomials of degree 2,3 and 4 to flow subspace coefficients and motion training data samples. Gaussian process regression with an affine mean function and isotropic Matérn covariance was also used for training, using the publicly available toolbox Rasmussen and Nickisch [2010][1].

3.5. Inference

Once a flow subspace model θ as well as linear mapping $\boldsymbol{m}$ from flow subspace coefficients $\boldsymbol{x}$ to self-motion has been learned, self-motion can be estimated from optical flow (c.f. Fig. 3.3).

First, given a flow vector field $\boldsymbol{f}_i$, the corresponding subspace coefficients $\boldsymbol{x}_i$ and inlier/outlier assignment $\boldsymbol{z}_i$ have to be inferred. This requires a short iteration of equations 3.29 and 3.31 (the E-step in sec. 3.3) until $\boldsymbol{x}_i$ does not change significantly any more. This usually happens within 20 to 100 iterations and requires little computational power (see sec. 3.8.6). To speed up this iteration in practice, initialization with the coefficients obtained from the previous flow field cuts computation time and helps to avoid convergence in a wrong local optimum.

The resulting coefficient vector $\boldsymbol{x}_i$ is then mapped through $\boldsymbol{m}$ to a self-motion estimate $\boldsymbol{m}_i = \boldsymbol{m}(\boldsymbol{x}_i)$. The inlier/outlier assignment result $\boldsymbol{z}_i$ of observed flow components can be interpreted as a measure of how much of the input flow is explained by the model. It is thus a measure of how well the model was able to interpret the observed flow.

[1]available online: `http://www.gaussianprocess.org/gpml/`

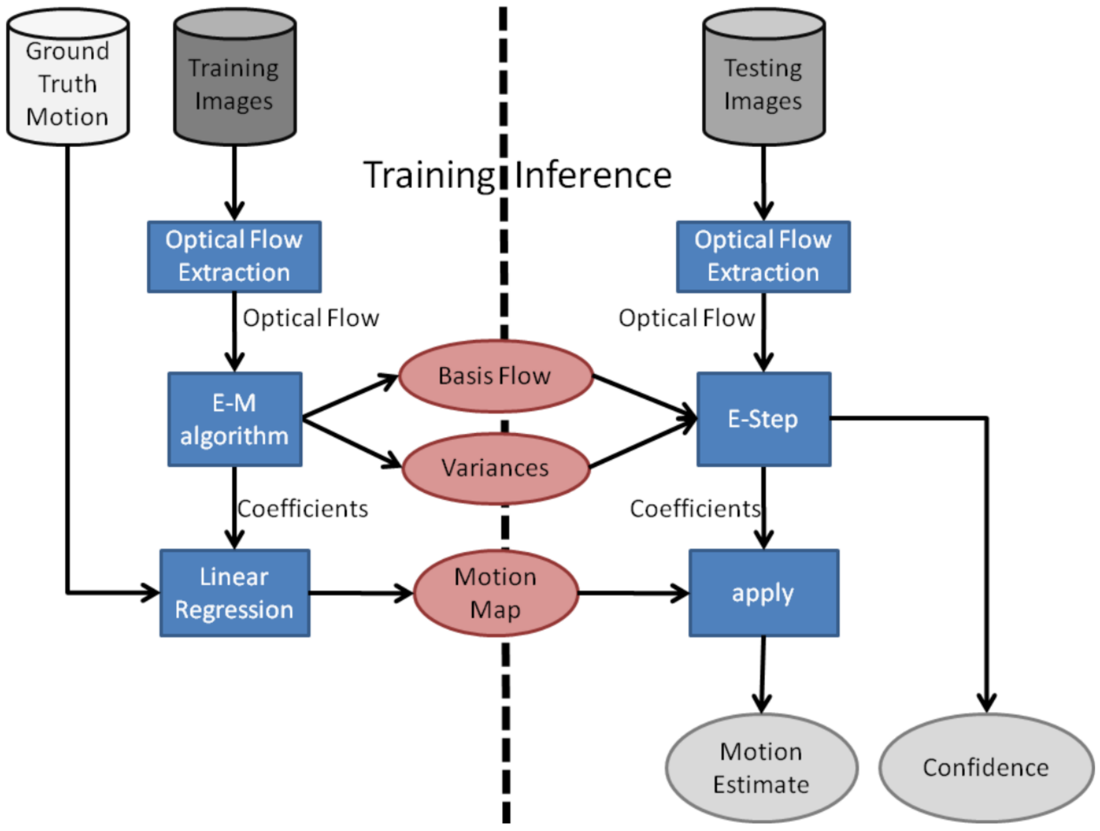


Figure 3.3.: Overview of training (**left**) and inference (**right**) of the self-motion approach based on flow subspaces.

This can be used as a confidence signal for further processing of flow representation and motion estimates.

3.6. Temporal Integration

A source of error inherent to every system that estimates incremental self-motion, is drift. Small errors accumulate over time, leading often to large errors in the integrated estimate. A slight deviation in yaw change, for example, will lead to a slightly wrong orientation, which leads to worse and worse estimates of position as the vehicle moves on. As another example, a single wrong estimate of roll change will leave the integrated roll estimate biased until the end of the drive. There are several ways to counter this effect. One is to fuse estimates with other sensors that measure the integrated state directly like GPS for position or gyroscopes that measure the direction of gravity which can stabilize pitch and roll. Alternatives are discussed in sec. 3.9.

The stabilizing effect can be tested by comparing results for integrated pitch and roll obtained with the method described so far with and without combining them with direct measurements of pitch and roll from a simulated noisy sensor. The general idea is that our estimates of pitch and roll change are usually very accurate, leading to a reliable high-frequency component in integrated pitch and roll, but estimates benefit from additional measurement with a correct low-frequency component. For mixing our high-frequency estimates with another sensor's low-frequency contribution, we propose the following filter.

Given filtered estimates $\hat{\chi}_{t-l}$, $\hat{\psi}_{t-l}$ for previous time steps ($l = 1, \ldots, L'$ and $L' = \min(L, t)$), measurements $\tilde{\chi}_{t-l}$, $\tilde{\psi}_{t-l}$ from the other sensor, and current estimates $\Delta\chi_t, \Delta\psi_t$ of pitch and roll change, the new filtered estimates of pitch and roll are calculated by

$$\hat{\chi}_t = \hat{\chi}_{t-1} + \Delta\chi_t + \alpha \left(\frac{1}{L'} \sum_{l=1}^{L'} \tilde{\chi}_{t-l} - \frac{1}{L'} \sum_{l=1}^{L'} \hat{\chi}_{t-l} \right) \tag{3.33}$$

$$\hat{\psi}_t = \hat{\psi}_{t-1} + \Delta\psi_t + \alpha \left(\frac{1}{L'} \sum_{l=1}^{L'} \tilde{\psi}_{t-l} - \frac{1}{L'} \sum_{l=1}^{L'} \hat{\psi}_{t-l} \right) \quad . \tag{3.34}$$

The gain α is assumed to be small ($0 < \alpha \ll 1$), the length $L \in \mathcal{N}$ of the running average should be long enough to suppress high-frequency errors present in the other sensor's measurements $\hat{\chi}_{t-j}, \hat{\psi}_{t-j}$. These equations are a simple integration of pitch and roll, with an additional term that corrects the estimate by the difference of the averages over the last L' frames between measurements and integrated estimates.

3.7. Data Set

For training and evaluation we chose the Karlsruhe data set by Geiger et al. [2010, 2011] since it provides realistic driving scenarios in a real-world urban environment, including varying geometry and lighting conditions, as well as moving objects. Additional object annotation labels and the availability of a stereo and optical flow algorithm by the same authors for further experiments (see sec. 4.3 in the next chapter) were additional reasons for using this data set .

The data set consists of rectified stereo sequences together with ground truth motion information from GPS and IMU for 20 drives through the city of Karlsruhe, Germany. We used only drives from the first day of capturing (September 8th, 2009) since re-calibration afterwards changed the image resolution and we did not want to risk this affecting results. The drive labeled as '2009_09_08_drive_0012' was also omitted because its second testing half consists of a waiting period at a street crossing and thus provides no interesting self-motion. The remaining five drives have a length of 100s to 150s. They were each split into a first half used for training and a second half for testing, each then consisting of 3201 frames. The frame rate for capturing is approximately 10 fps, the resolution is 1344×391 pixels. In this chapter, only the data recorded by the left camera is used.

As ground truth for change in position and yaw we used motion data from the IMU, which was smoothed with a Hanning window of size 20 since the frame rate was not perfectly constant. The few frames where the time between successive frames deviated strongly from 0.1s, the motion information was not used. A few seconds of driving backwards were detected by comparing object motion direction with object orientation. Change in position was multiplied by -1 in these frames. Pitch and roll are reported invalid on the data set homepage and indeed lack the jitter from the slight fluctuation in frame rate. In an experiment below we still used them without further pre-processing for testing the approach's ability to learn those motion dimensions as well.

3.7.1. Optical Flow

Calculating an accurate dense optical flow field from images in a fast way is still an active field of research since it was proposed as a biologically sound feature for computer vision in Gibson [1979]. Popular classical approaches have been proposed by Horn & Schunk Horn and Schunck [1981] and Lukas & Kanade Lucas and Kanade [1981]. Later approaches include Black and Anandan [1993], Bruhn et al. [2005], and Geiger et al. [2011]. We decided to use the flow calculator contained in the publicly available `libviso2` package (Geiger et al. [2011][2]) since it yields more accurate results than the implementations of the two older approaches, but still runs in near-real time. Also, it has already been applied to the data set we use since it is provided by the same authors.

We did not use the calculated optical flow fields directly, but employed a method called *bucketing*, which is a form of sub-sampling of the flow field also implemented in `libviso2`. The image is divided into a number of non-overlapping rectangular buckets, and from each bucket only one flow vector is drawn at random. The resulting flow field has three advantages compared to the original flow field: it is of smaller size which makes computations easier. It also has a much smaller ratio of missing values since only one flow vector has to be provided from each bucket. Finally, when using input with a slightly different image size, the number of buckets might still be the same, thus providing a simple way to generalize over heterogeneous input.

The drawback of bucketing is a loss of resolution, which causes a loss of precision in subsequent processing. If buckets are too large, the variation of depth within a single bucket contributes significantly to the variance of the flow observed within that bucket, which is not captured in the model. The chosen bucket size therefore has to be a trade-off between large training efforts with possibly many missing values from small buckets and high variance of flow vectors as consequence of larger buckets. We chose a bucket size of 20×20 pixels, resulting in a flow field of size 67×19 and thus in flow field vector length of $2546 = 2 \times 67 \times 19$.

3.8. Evaluation

In this section, the performance of an implementation of the described approach is evaluated when varying the number of subspace dimensions, the usage of the presented inhomogeneous inlier variance extension, and when using a non-linear mapping to motion. Results on drift reduction when integrating estimates over time and computational effort are also presented. The error measure used to create plots is the root mean squared difference between estimated and ground truth motion changes. Given, for example, estimates of yaw orientation change $\Delta\phi_1, \ldots, \Delta\phi_I$ for frames $1, \ldots, I$, and corresponding

[2]available online: `http://www.cvlibs.net/software/libviso2.html`

ground truth $\Delta\phi_{\mathsf{GT},1}, \dots, \Delta\phi_{\mathsf{GT},I}$, the root mean squared error is

$$\mathsf{RMSE} = \sqrt{\sum_{i=1}^{I} (\Delta\phi_i - \Delta\phi_{\mathsf{GT},i})^2} \quad .$$

3.8.1. Varying Subspace Dimensionality

Since the outlier variance σ_{out} is also trained in all experiments, the only free parameters in training a flow subspace model are the dimensionality N of the subspace, i.e., the complexity of the generative flow model, as well as whether to use the inhomogeneous inlier variance extension or train an estimator with homogeneous inlier variance. We varied both and report the root mean squared error of motion estimates for subspace dimensions $N \in 2, 4, 7, 10, 14$ on the test set in Fig. 3.5. As can be seen in the top left, the error in forward motion estimates drops continuously, slowly converging at ≈ 0.02m. Yaw is the most accurate of the rotational dimensions with error dropping continuously to below $0.035°$. Changes in pitch and roll for homogeneous inlier variance show a sudden drop in error at flow subspace dimensions $N = 7$ and $N = 10$, respectively. Interestingly, for estimators with inhomogeneous inlier variance, these drops in error rate are shifted towards lower dimensions, meaning that inhomogeneous variance allows an earlier representation of roll and pitch in the subspace.

In Fig. 3.6 time courses of motion estimates for the 2D and the 10D flow models are visualized for a selected range of frames of the testing sequences. For the lower-dimensional model the pitch and roll estimates are constantly 0, meaning they are not captured in the low-dimensional flow subspace. Large changes in pitch or roll instead seem to contribute to wrong representations of the flow field, leading to wrong estimates in position and yaw changes, which otherwise are rather accurate for both 2D and 10D flow subspace models.

Basis flow fields for a subspace with $N = 4$ dimensions are shown in Fig. 3.4. The first basis flow fields approximate a looming and a yaw rotation.

All these observations are consistent with the intuition that the PCA model first captures latent causes of larger flow deviations, before 'explaining' smaller variations. Flow from a driving car is dominated by contributions caused by changes in position and yaw, which can therefore be estimated already from the first 2 subspace components. Pitch and roll changes, on the other hand, are less prominent, because they are caused by bumps in the street, as well as sudden breaking, acceleration and turning maneuvers. Since the given drives are rather smooth, these effects lead to only small changes in observed flow, which are only captured at higher subspace dimensions of 7 and above.

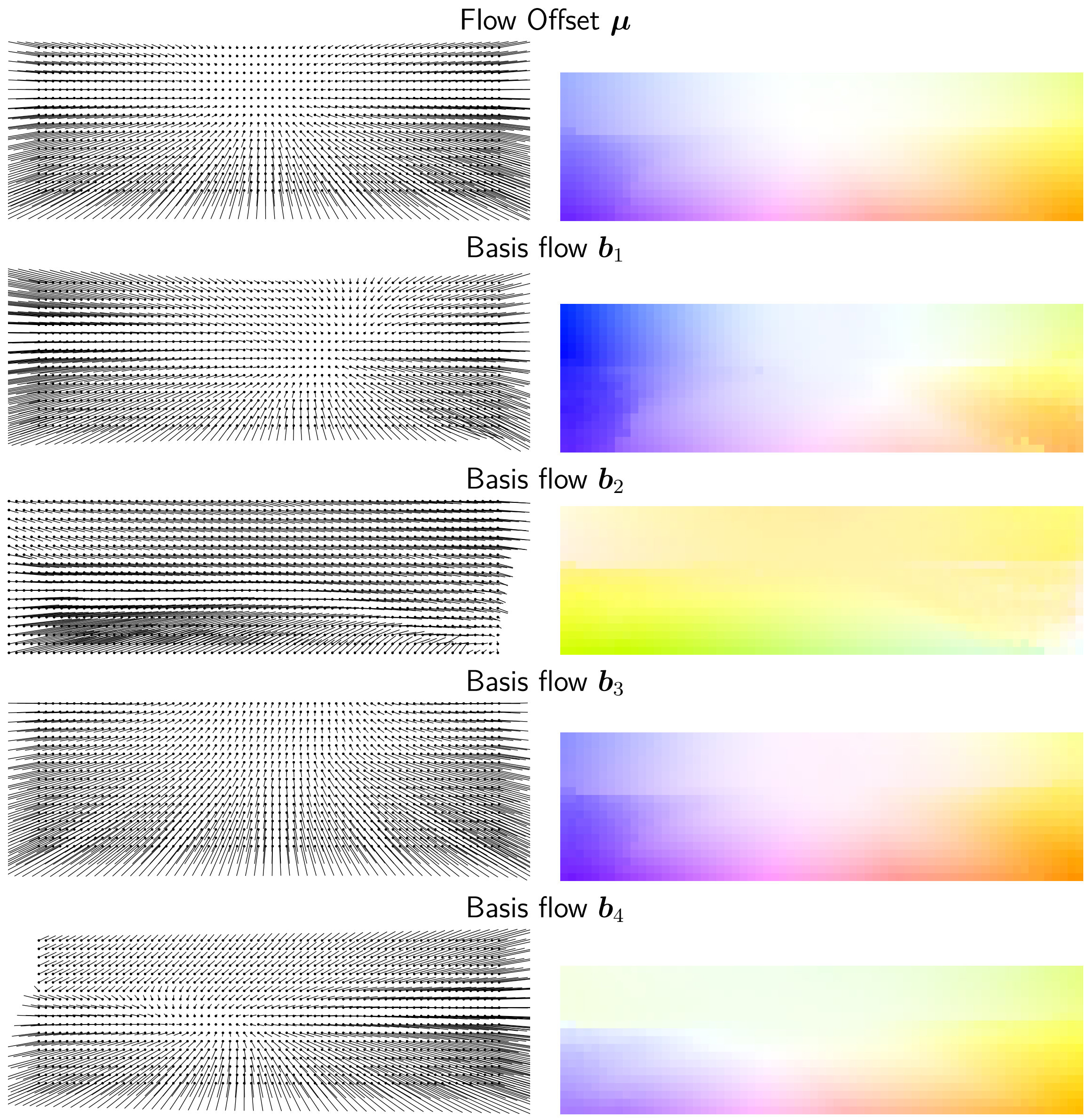


Figure 3.4.: Flow offset $\boldsymbol{\mu}$ and basis flow fields $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_N$ of a flow subspace with $N = 4$ dimensions and inhomogeneous inlier variance in two different visualizations. On the **left** flow vectors are shown as lines with dots marking their head (i.e. position in the second image), on the **right** vectors are coded as colored rectangles where hue codes orientation (red=upwards, blue=rightwards, yellow=leftwards, etc.) and saturation codes vector length (white=0). Basis flow fields are shown in the order returned by the learning algorithm when initialized with deterministic flow fields as described in sec. 3.3.

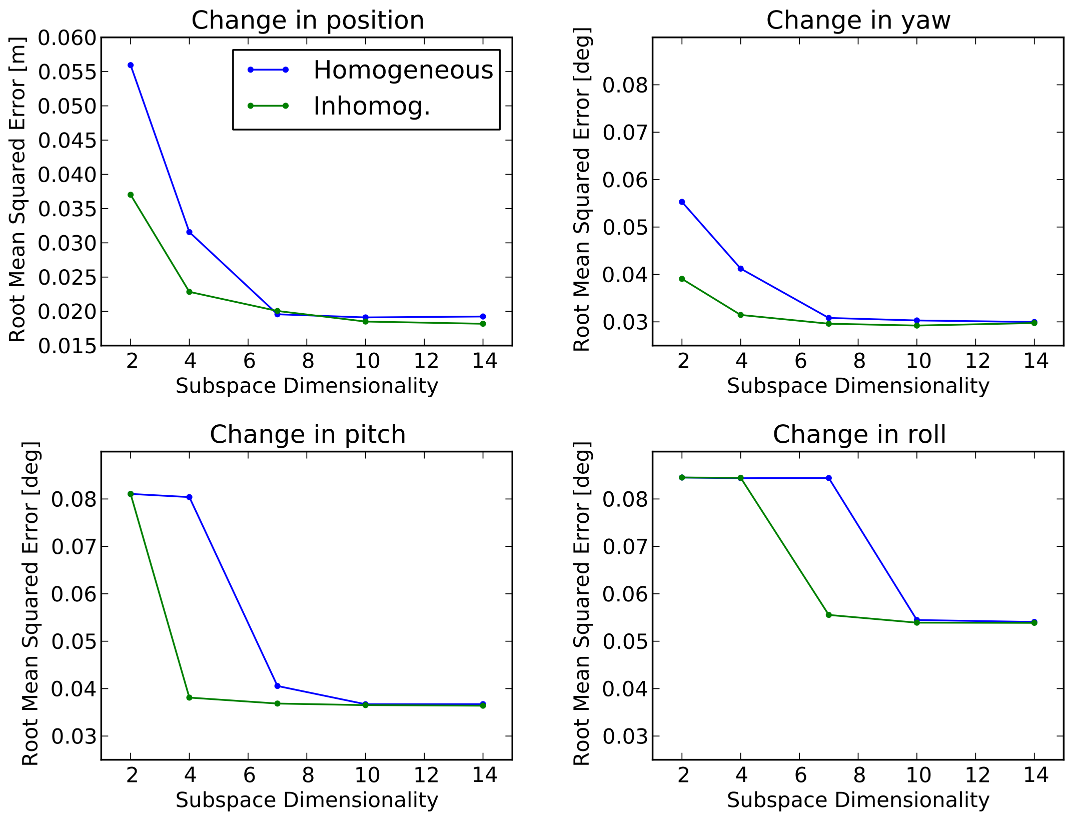


Figure 3.5.: Root mean squared error for motion estimates. Each plot shows errors of homogeneous (blue) and inhomogeneous (green) estimators for various subspace dimensionalities (shown on the horizontal axis).

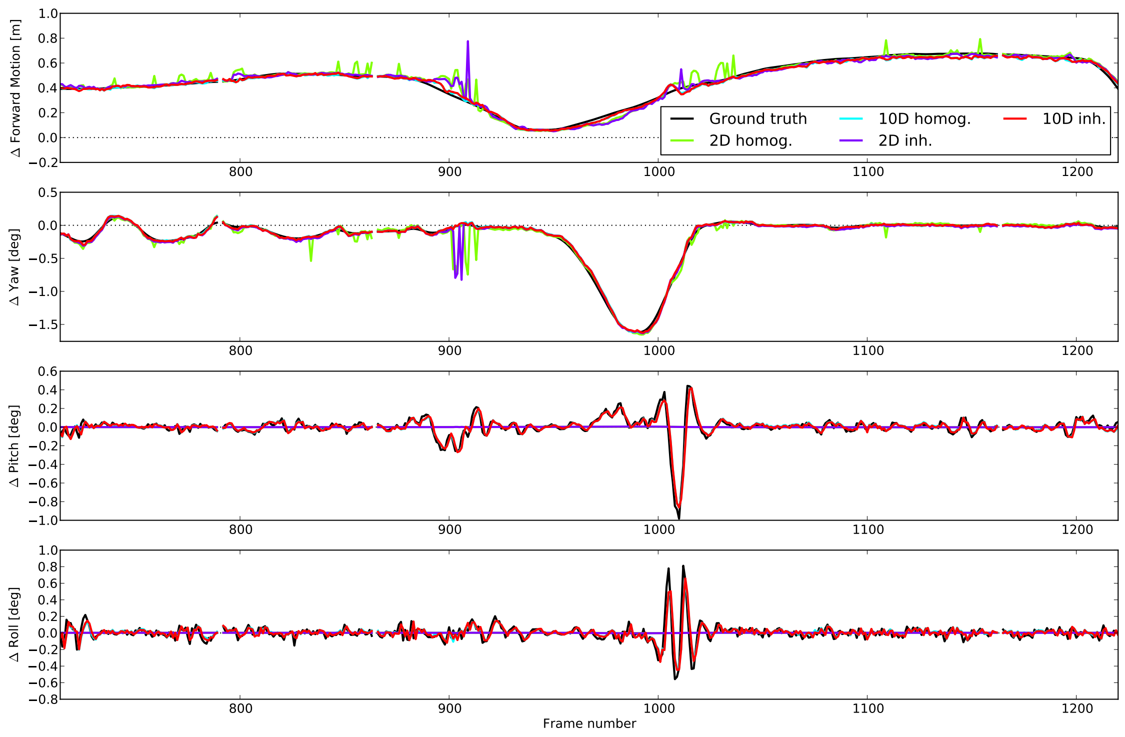


Figure 3.6.: Motion estimates for estimators using 2D and 10D flow subspaces trained with homogeneous and inhomogeneous inlier variance. Each row shows estimates for one motion dimension, from top to bottom: change in position, yaw, pitch, and roll. The ground truth is shown in black. The legend for the top plot is valid for all four plots. The green line (2D homogeneous) cannot be seen in the pitch and roll plots since it is covered by the magenta line (2D inhomogeneous).

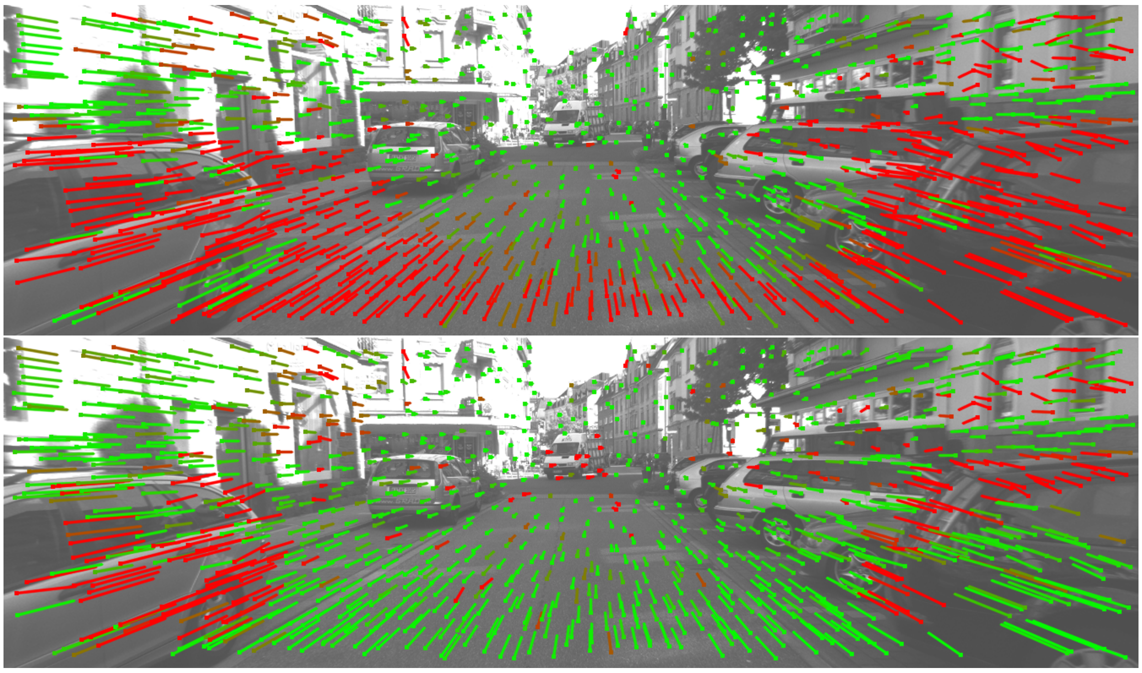


Figure 3.7.: Typical frame with flow interpreted with homogeneous (**top**) and inhomogeneous (**bottom**) inlier variance. The additional flexibility in inlier flow variance leads to better interpretations of flow fields.

Changes in geometry are the most likely cause of flow variations, that are captured by the remaining flow components.

3.8.2. Inhomogeneous Inlier Variance

Fig. 3.7 (top) shows a typical frame from the test set with optical flow field overlaid. The color of flow vectors denotes the probability z_{ij} of being an inlier, with red being a low value (i.e. probably an outlier) and green meaning a high value (i.e., probably an inlier). Since each flow vector consists of two components (horizontal and vertical), each vector color is a combination of 2 possibly different z_{ij}. We show here the minimum probability of horizontal and vertical components since often one component is correct by chance.

Note that the flow at the lower image border is red almost everywhere although the flow vectors seem to be correct. This is probably due to the effect that through bucketing of flow vectors (c.f. sec. 3.7.1) and additional noise term is inserted into the measurement process. Flow vector components in a bucket may not correspond to the expected value of that bucket although their value is correct for that pixel, because the variance of flow within that bucket is too high.

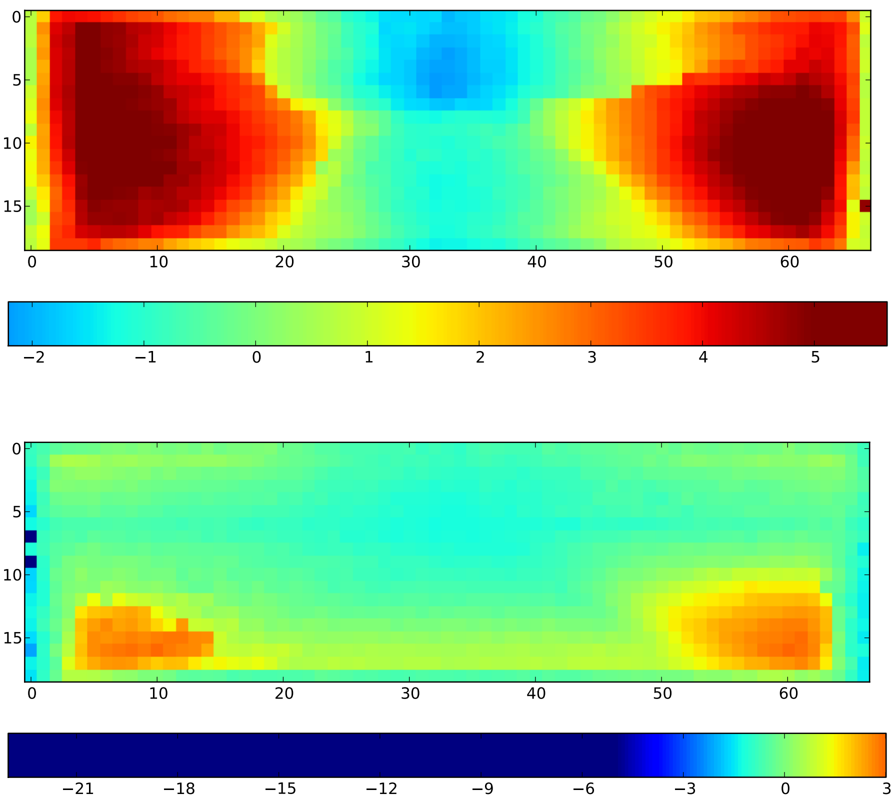


Figure 3.8.: Inlier variance for horizontal (**top**) and vertical (**bottom**) flow components. Color decodes the variance in pixels on a logarithmic scale with basis e. Color bar labels denote exponents.

Allowing a slightly higher variance for these buckets solves this problem as can be seen in Fig. 3.7 (bottom). In general allowing different variances for each pixel allows the system a greater flexibility in interpreting flow in regions where structure varies more (e.g. the lower periphery, see Fig. 3.8) while still requiring a good match in regions there geometric variations are minimal (e.g. the ground plane further away). This allows the system to find the optimal compromise between specificity and generality in each flow bucket separately.

Overall, allowing inhomogeneous inlier variance increases estimation performance, as can be seen in the results in Fig. 3.5 and Fig. 3.6. However, integrating forward motion and yaw estimates over the complete testing set into trajectories as shown in Fig. 3.5, results in better trajectories for the estimators with homogeneous inlier variance. The reasons for this unexpected result are not clear.

3.8.3. Non-Linear Motion Map

As motivated in sec. 3.4, the theoretically linear motion mapping may contain slight non-linearities in practice. Results using a linear and various non-linear mappings were therefore compared on the same training data set. The fit of the linear mapping on training data is detailed in Fig. 3.10. The linear fit (red) does show deviations from the ground truth motion (blue) in some areas.

Results for estimated motion using linear and non-linear mappings for various subspace dimensionalities, as well as homogeneous and inhomogeneous inlier variances are shown in Fig. 3.11. Interestingly, the error in estimated motion increases for non-linear motion mappings, quite drastic even for higher subspace dimensions. This could hint at over-training, which is well possible since the amount of noise in the data is quite high and a polynomial of degree 2 on a 10-dimensional subspace already has $1 + 10 + 10^2 = 111$ free parameters, with the number increasing exponentially with subspace dimension.

We therefore used the linear mapping in the rest of this thesis.

3.8.4. Temporal Integration

While estimates of pitch and roll changes are usually quite accurate for high enough subspace dimensionalities, occasional misinterpretations of the flow field can lead to errors in integrated pitch and roll that stays as bias in all subsequent estimated orientations as motivated in sec. 3.6.

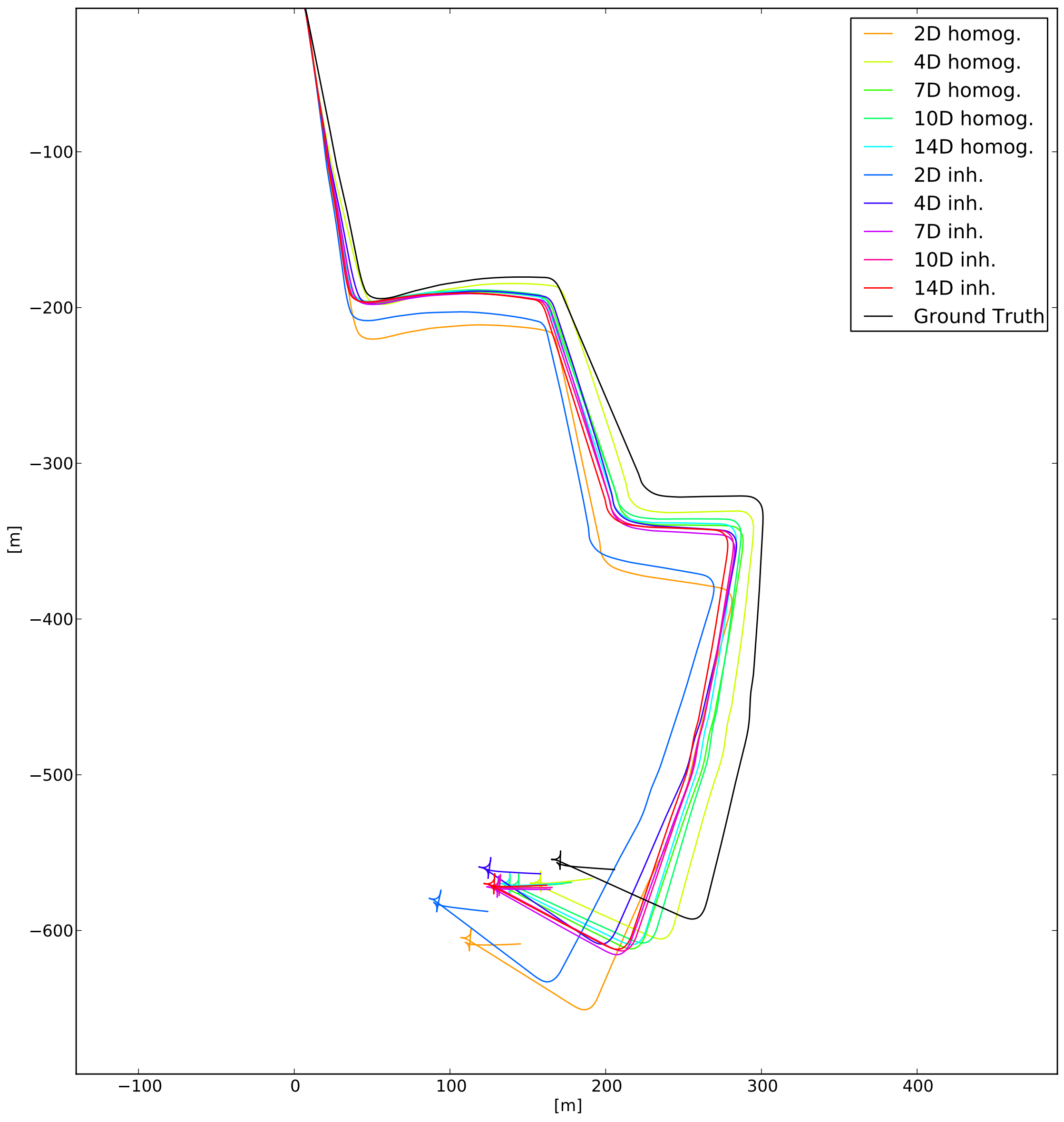


Figure 3.9.: Top-Down view of estimated trajectories obtained by integrating position and yaw change estimates over the complete test set. Displayed are the trajectories obtained from integration of ground truth motion (black), as well as estimates obtained with homogeneous (orange, green, light blue) and inhomogeneous (dark blue to red) inlier variances and various flow subspace dimensionalities.

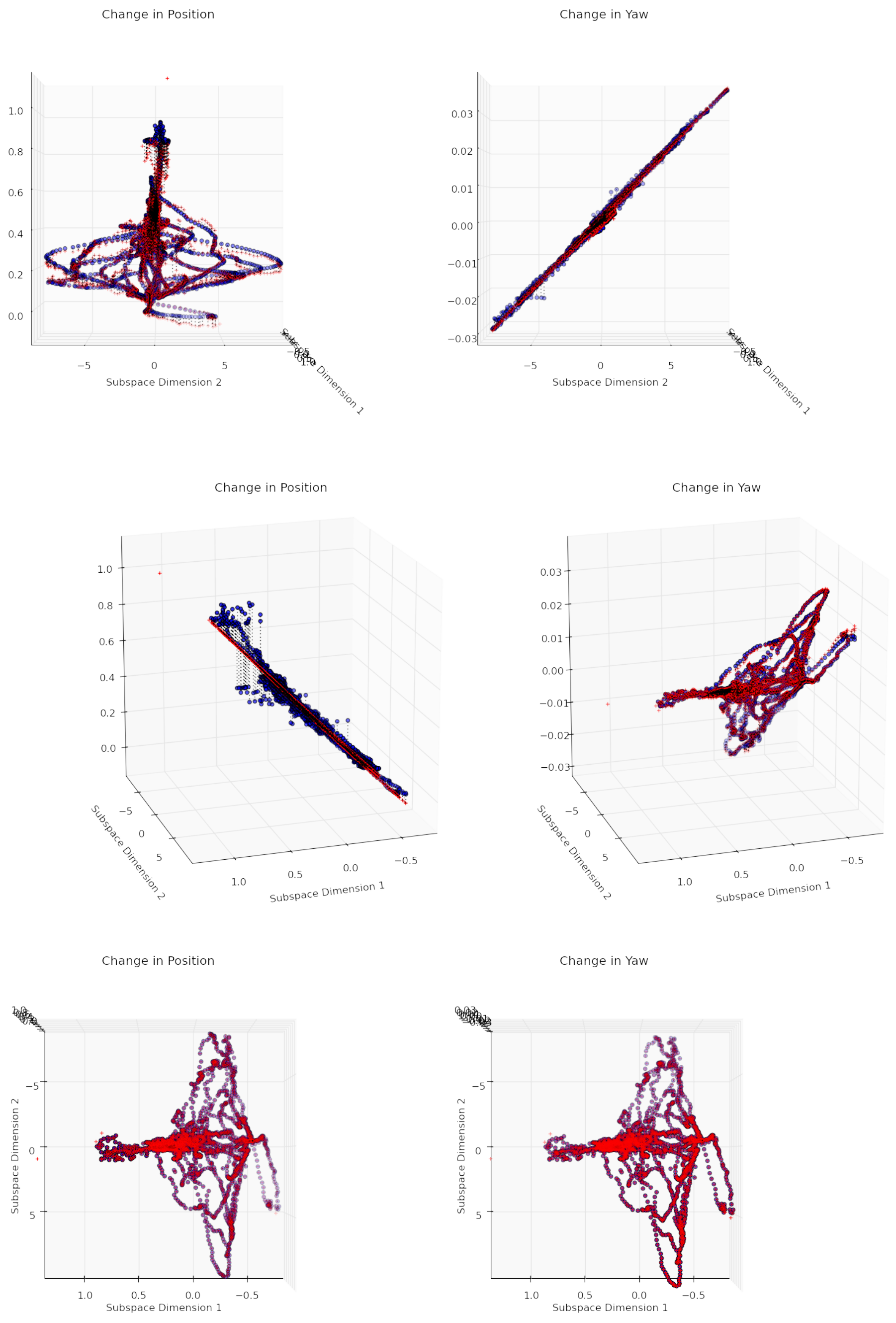


Figure 3.10.: Visualization of the mapping from flow subspace coefficients to motion. Displayed are 3D scatter plots for the mapping from a 2D flow subspace to change in position (**left**) and yaw (**right**) from different viewing angles.

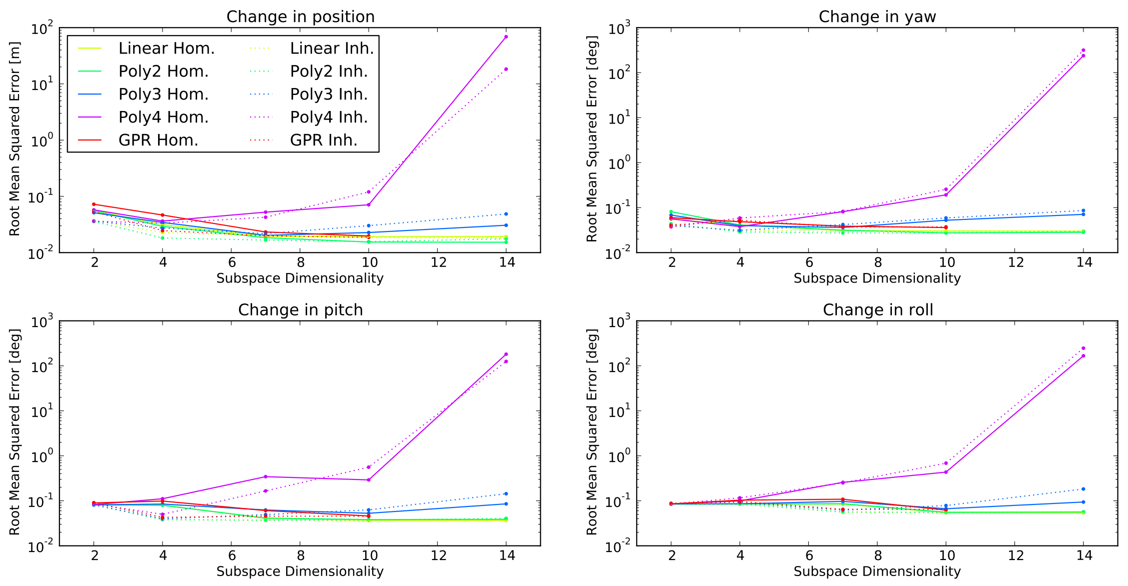


Figure 3.11.: Comparison of various mapping methods. Each plot shows the root mean squared error in a single motion dimension. Tested were a linear mapping (light green), polynomial mappings of degrees 2 (green), 3 (blue) and 4 (magenta), as well as Gaussian process regression (GPR, red) on flow subspaces trained with homogeneous (solid lines) and inhomogeneous (dashed lines) inlier variances. The decrease in performance of non-linear methods for higher subspace dimensions might indicate over-training.

To demonstrate the benefit of stabilization through direct measurements of pitch and roll from other sensors, ground truth information is used to simulate noisy orientation measurements and integrate them with estimated change in pitch and roll using the method described in sec. 3.6 above.

The gain of the filter was set to $\alpha = 0.05$, a running average length of $L = 20$ was used. The filter is simulated by creating a very smooth copy of ground truth pitch and roll (smoothed with a Hanning window of length 80 and suitable boundary conditions) and adding independent zero-mean Gaussian noise with a standard deviation of $0.5°$ to each value. Given the small range of values for pitch and roll, this is a rather large noise variance.

As can be seen in Fig. 3.12, the integrated estimates approximate the high-frequency and low-frequency orientation changes. The magenta and green line in the bottom two plots show the relative contribution of the noisy ground truth since these two lines are based on 0-estimates from the low-dimensional subspaces, and thus only show the low-frequency contribution from the filter.

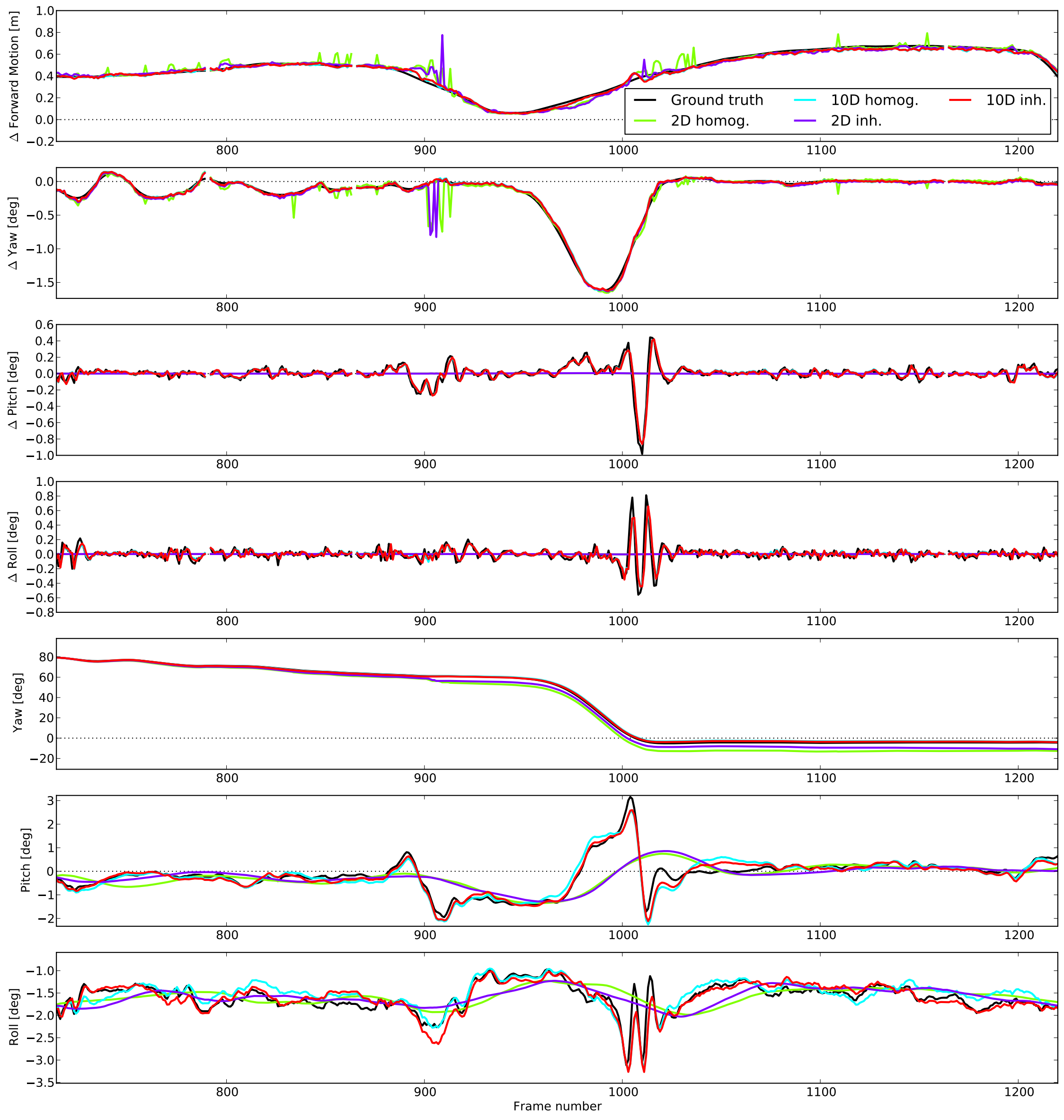


Figure 3.12.: Estimates of change in position and orientation, and integrated orientation estimates. The top four plots show estimated changes in motion, repeating Fig. 3.6. The bottom two plots show integrated orientation estimates using estimated change in orientation and noisy versions of smoothed ground truth. Integrated yaw estimates (third row from bottom) are based on estimated yaw change, only.

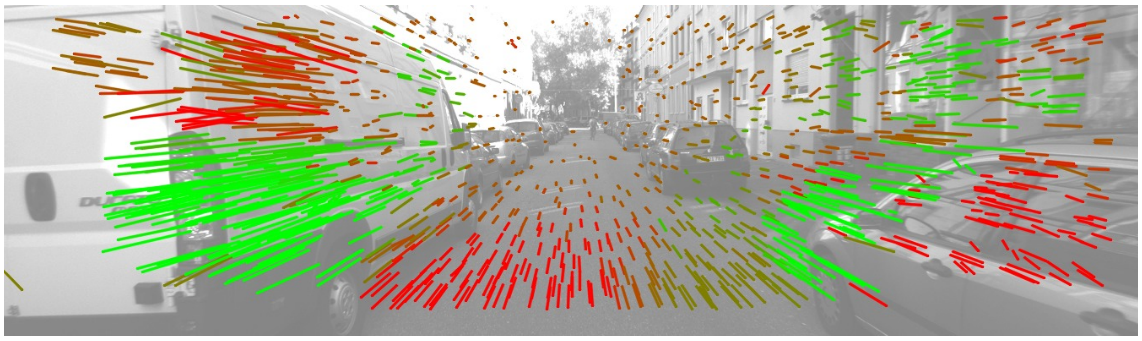


Figure 3.13.: Example of a failure case of the described approach. Color of flow vectors represents the minimum of inlier probability of the two vector components: green is a high probability (an inlier), red stands for a low probability (an outlier).

3.8.5. Model Limitations

As discussed earlier, the main limitation of the self-motion estimation approach is the constant depth assumption implicit in the flow subspace model. As long as the vehicle moves through scenes with similar geometry, the method works well as shown in this section. Deviations from the expected geometry are detected as outliers and ignored in further processing. However, if large parts of the flow field are disturbed by flow from unexpected depths, the representation of that flow field in the flow subspace will probably be incorrect, leading to a wrong motion estimate.

One such case can be observed in Fig. 3.13. The presence of close-by cars in the left and right periphery create flow vectors that violate the expected depth (visualized in Fig. 3.2). Flow vectors on the objects come from a distance much smaller to the camera than expected, and are therefore longer than they would be for the observer's current speed if the objects were not present. Since the objects are relatively big, the amount of outlier flow is higher than the number of inlier flow vectors, e.g., on the ground. This leads to a misinterpretation of the flow, making the model believe that part of the flow on the objects is actually correct and the flow on the street consists of outliers, which then leads to a wrong motion estimate.

Two ways to improve results in such situations are possible. One is a mixture of experts as described in the next chapter. Since such cases have only appeared in isolated frames so far, temporal filtering could also improve such estimates, if necessary with the help of estimated inlier probability, which gives an idea of estimation confidence.

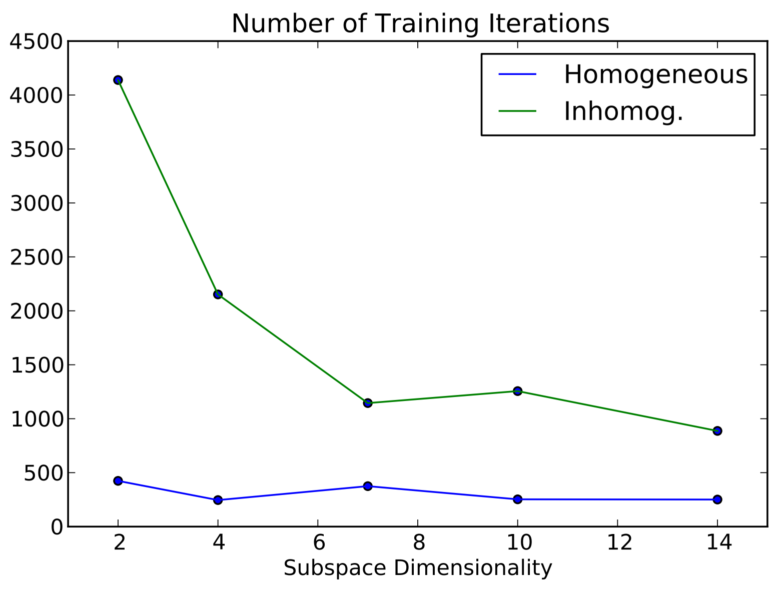


Figure 3.14.: Number of iterations for finding the flow subspace. The blue line shows the number of iterations per subspace dimensionality using homogeneous inlier variance, the green line shows the number of iterations for finding a inhomogeneous-variance flow subspace. Interestingly, the number of iterations tends to decrease in both cases although the number of parameters to fit increases.

3.8.6. Computational Effort

As a final evaluation, the number of iterations used to find a flow subspace and to infer motion using it is reported here. Figure Fig. 3.14 visualizes the number of iterations of training. The decrease in the number of iterations as the number unknowns increases, may be due to a higher number of local maxima with increasing subspace dimensionality.

As an example for computational effort during inference, the mean number of iterations used to interpret an observed flow field in terms of a given flow subspace is 21.722 ± 16.98 (std) for a 10-dimensional inhomogeneous subspace. Together with motion mapping, this took 0.077 ± 0.063 (std) seconds using a regular python session on a single core of an Intel Xeon computer (2.66GHz) and insignificant memory consumption. Re-implementation could make this approach attractive to applications with very limited computational resources.

3.9. Conclusion

This chapter has discussed and extended the generative model for optical flow interpretation from Roberts et al. [2009]. It has been shown that this subspace representation offers a viable alternative to well-known deterministic tracking and rejection schemes for self-motion estimation in structured environments. Through unsupervised learning the model incorporates knowledge about typical scene depth profiles, which affords a decoupling of geometry and self-motion contributions to the optical flow field. The model also allows a natural regularization of optical flow data and an identification of noise from independently moving objects, erroneous flow measurements, geometric irregularities and missing values. Since no explicit geometric relationships have to be formulated, the approach can be applied to data measured from arbitrary camera setups including catadioptric, multi-camera and wide-angle lens systems. Inference requires a short iteration that converges quickly, making this approach very cost-efficient.

Extending the model to inhomogeneous inlier variance allows a more exact fit to optical flow fields and thus improved self-motion estimation performance. An equation for updating the outlier variance during learning leaves only one parameter for training: the number of subspaces. This value affects the amount of variation in flow that is captured by the model. Applied to a challenging set of real-world image sequences recorded in a busy urban environment the first two components capture variation in flow caused by forward motion and yaw change. Adding more basis flow fields results in a model that can also estimate change in pitch and roll, which can be integrated over time, possibly with other sensor data to avoid drift.

During the course of this thesis, several ideas for extensions and applications were discussed. Some were implemented for preliminary testing.

In one test, self-motion estimation behavior in the case of larger numbers of missing values was analyzed. Restricting flow fields to regions with very low flow variance (mostly the ground) resulted in a drastic increase in missing values, but no noticeable decrease in estimation performance.

Since the data set used here contains stereo imagery and the flow subspace approach is able to handle arbitrary camera setups, a flow subspace from the combined flow field from left and right camera was learned using $N = 2$ flow subspace dimensions. Self-motion inference did not improve in preliminary tests, however, so this line of experiments was not continued. Cameras with less overlap are probably more helpful for this approach.

Apart from this test, the multi-camera applicability of the presented approach was not exploited at all. Using more cameras should improve motion estimates, but requires different data sets for training and testing than those available at the moment. Synthetic image sequences may be a way to study the system's behavior with input from a varying number of cameras. Optimizing sensor placement could also be an interesting topic in this context

Another extension of the subspace model (equation 3.1) was devised, using two different outlier models. Each flow vector could be either an inlier, be explained by noise, or be a consistent outlier. Outlier flow vectors from a larger object or geometric anomaly has the property of being spatially and temporally coherent, which could be used to identify outlier clusters in the flow field. Unfortunately, the two outlier sources could not be separated well enough in training. However, segmentation methods may still be able to identify clusters of coherent outlier motion in the image, which could be used to identify objects, for example.

.

It would be of practical relevance to know how much training data is actually needed by the system and whether it could be gathered on the fly. Supervision when training the mapping to motion could be reduced or removed by incorporating information from object detectors.

Two extensions to reduce drift could be the inclusion of more than two frames in either flow calculation as proposed in Irani [1999] or in motion estimation as Irani [1999], Lategahn et al. [2012] suggest. Finding the horizon in flow fields could also serve as rough measure of absolute vehicle pitch and roll, which could be integrated to avoid drift in these two dimensions.

Due to its computational efficiency, this method could be invaluable to small ground-based robots. A simple but possibly effective measure in this context could be a camera placement such that images include mostly ground. Given a frame rate that is high enough to avoid motion blur, such a camera placement would result in a flow subspace of mainly ground, so all other geometry would be identified as outliers. On such a platform, usage and placement of several cameras could be studied in real-world conditions.

Of course, this flow representation could also be used for other purposes. In particular, geometric anomalies could be identified in outlier flow since their pattern has a deterministic, smooth component. A recent method for identifying planes in flow fields has been proposed in Bouchafa and Zavidovique [2012].

CHAPTER 4

An Expert System for Self-Motion Estimation

The implicit assumption of constant depth over time limits the model described in the last chapter to well-structured environments like urban street canyons. Although the probabilistic nature, the described extension to inhomogeneous inlier variance and a higher number of subspaces increase the variety of flow fields that the model can explain, the diversity of depth profiles even in structured areas restricts its applicability. On the other hand, the amount of variation in structured environments is also limited, which can be exploited further, for example in the failure case described at the end of the evaluation in the last chapter (Fig. 3.13).

One general possibility to enhance the applicability of a computational model is to embed it in a *mixture of experts* model. Based on stacked generalizers Wolpert [1992] the term was coined by Jacobs et al. [1991] to describe a neural network that consists of several independent networks ('experts') that perform regression or classification, and a gating network that combines expert estimates based on the input Schaal and Atkeson [1995]. Several similar schemes for combining simple algorithms to more general systems, like bootstrap, cross-validation, ensembles of classifiers and boosting, have been developed, using different strategies for training, combining expert output or dividing the input space. Such systems have been applied to several standard problems like regression, classification and segmentation Pawelzik et al. [1996]. An overview centered on mixtures of experts can be found in Yuksel et al. [2012], while Brown [2010] provides a more general overview. Applications to computer vision include Enzweiler and Gavrila [2011] and Rosales and Sclaroff [2001].

In this chapter, a generalization of the flow subspace representation model from the last chapter using a mixture of experts is described. The goal is to extend the applicability of the described self-motion estimation model by softening the constraint imposed by the constant depth assumption. A mixture of self-motion estimators is suggested, each an expert for a different typical scene geometry, that cooperate to estimate self-motion in more general situations.

In the first section of this chapter, a way to define and train such a mixture of experts model is presented. Inference in this model as described in the following sec. 4.2. In sec. 4.3 results obtained using such a mixture model are compared with those in the last chapter and with other results from the literature. A short summary, discussion and possible future directions are given in sec. 4.4.

Parts of this chapter have been published in Herdtweck and Curio [2012a].

4.1. Defining Experts

The basic idea behind using a mixture of experts to extend the applicability of the self-motion estimation approach described in chapter 3, is to have several such estimators, each with its own flow subspace and thus each with its own underlying assumption on scene geometry. Each estimator is therefore an expert a particular scene geometry, making the ensemble applicable to a much wider variety of scenes than a single expert could be.

In the literature, several different methods have been developed for training expert systems, in particular incremental and unsupervised ways to partition the input space into several, possibly overlapping domains of expertise. We follow here a different approach, which has been suggested, for example, by Schaal and Atkeson [1995] and instead prepartition the input space and train experts on their own separate training set. We use depth context knowledge to define this partitioning, which is extracted using stereo disparity between the two cameras on the training image sequences. We use `libelas` (Geiger et al. [2010]) since it is freely available[1] and has been used on the same data set by the authors, who also provide the training data.

The overall training procedure is visualized in Fig. 4.2 (left). As a first step in splitting the training data, we approximate the distribution of depth profiles. We do so by sub-

[1]available online: `http://www.cvlibs.net/software/libelas.html`

sampling every 10th disparity image by a factor of 3 in u (horizontal) and v (vertical) image dimensions and regularize the resulting images by applying probabilistic PCA (Tipping and Bishop [1999], implementation by J. Verbeek[2]) with 10 principal components. We then cluster the remaining 10-dimensional condensed depth profiles using a mixture of Gaussians as described in sec. 2.6.1. The number K of experts has to be specified beforehand. The resulting depth profiles for $K = 3$ can be seen in Fig. 4.1.

Having determined approximations to the K most frequent depth profiles, every frame in the training set can be assigned to one of the clusters by sub-sampling, projection into the 10-dimensional subspace, and calculating the weighted distance to each cluster center. In formula, with a mixture of K experts

$$p(\boldsymbol{x}) = \sum_{k=1}^{K} w_k \mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$
$$\sum_{k=1}^{K} w_k = 1$$

a new projected sample $\boldsymbol{x}$ is assigned to cluster

$$\hat{k}(\boldsymbol{x}) = \operatorname{argmax}_k w_k \mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \quad . \tag{4.1}$$

Disparity profiles $\boldsymbol{x}$ that have significant contributions from several clusters (which amounted to $\approx 5\%$ of the training set) are ignored in further training.

Flow fields calculated between frames of the same cluster are then used to train one model as described above. This results in K sets of basis flow matrices $\boldsymbol{B}^k$, flow means $\boldsymbol{\mu}^k$, inlier and outlier variances $\boldsymbol{\sigma}^k_{\text{inl}}, \sigma^k_{\text{out}}$, that each define one expert for flow fields from depth profiles similar to the corresponding cluster center. Finally, for each expert $E^k = (\boldsymbol{B}^k, \boldsymbol{\mu}^k, \boldsymbol{\sigma}^k_{\text{inl}}, \sigma^k_{\text{out}}, m^k)$ a motion mapping m^k from its subspace to observer motion is trained.

4.2. Inference

Inference in an expert system proceeds without additional depth or disparity information, as visualized in Fig. 4.2 (right). Each expert E^k forms a representation $(\boldsymbol{x}^k, \boldsymbol{z}^k)$ of the observed flow field $\boldsymbol{f}$ (Note that we omit the time index i here). The system then

[2]available online: `http://lear.inrialpes.fr/~verbeek/software.php`

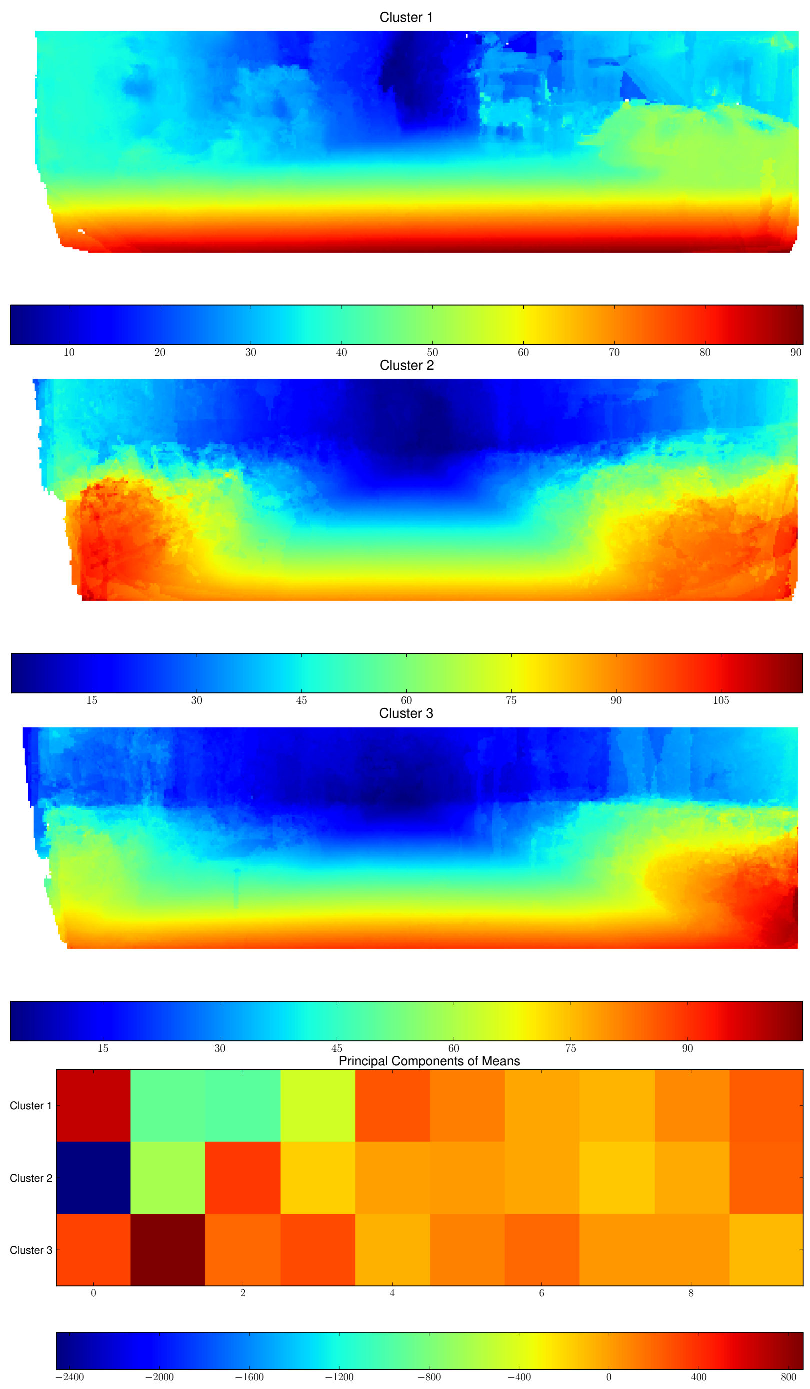


Figure 4.1.: Clustering results: Displayed are representations of the mean disparity profiles used to define a mixture of $K = 3$ experts. Each plot shows a disparity image averaged over the 10 training samples that were closest to their cluster centers in PCA space. The 3 cluster means in PCA spaced are displayed in the bottom plot. In this mixture, expert 1 is trained on images with no or very few close objects, expert 2 is trained on images with close-by objects on the left and right (e.g. narrow one-way streets), while expert 3 is trained on images with close objects only to the left (e.g. parking cars on a wide road). The representation of cluster means in PCA space suggests that 4 principal components would have been enough to form these 3 clusters.

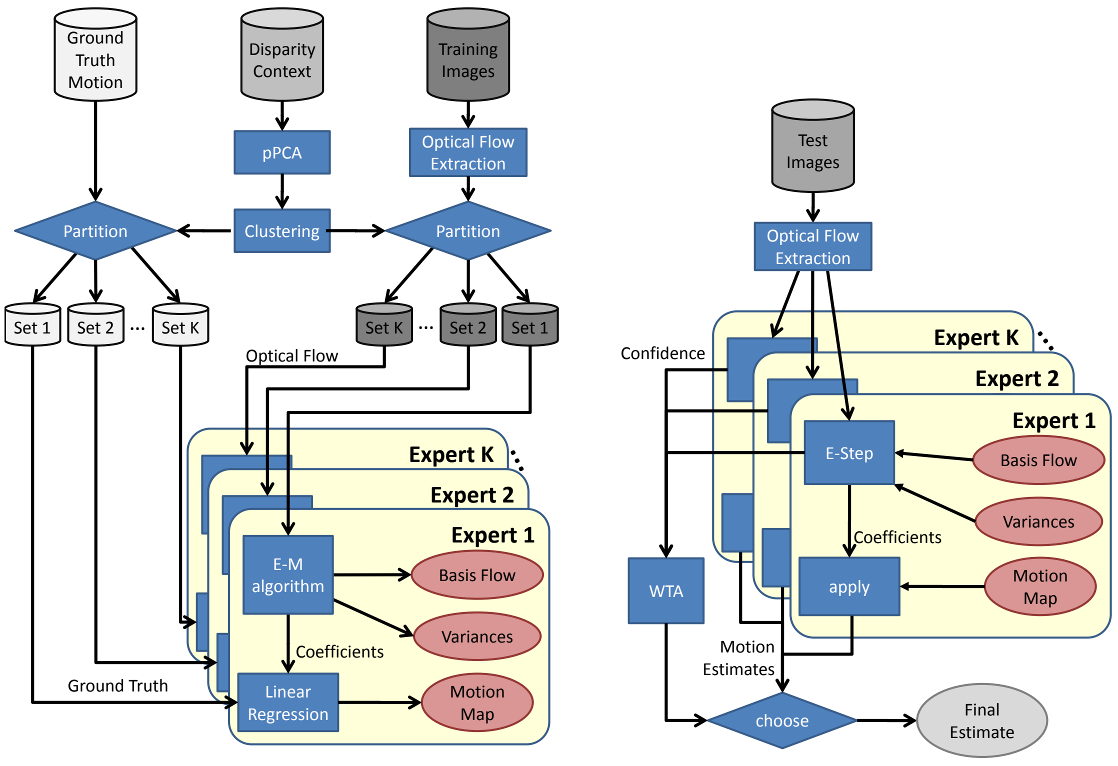


Figure 4.2.: Overview over training (**left**) of a mixture of experts and inference (**right**) in the resulting model. During training, the set of all training images, flow fields, and ground truth motion labels is split using depth context, and then used to train a single expert (yellow box). During inference, the optical flow field that has to be interpreted is sent to all experts, each responding with a motion estimate and a confidence, which is then used to select the final motion estimate.

has to decide which expert or set of experts to trust, without the use of depth context information. To this end, the confidence measure

$$c_k = \frac{1}{|J'|} \sum_{j \in J'} \boldsymbol{z}_j^k \quad , \tag{4.2}$$

is defined, where J' is the set of all observed flow vector components. The value c_k measures the average ability of subspace model (i.e., expert) k to represent the observed flow field. Intuitively, when a flow field stems from the same depth profile that the expert was trained on, the flow field should lie within that expert's flow subspace and the model should thus be able to find subspace coefficients $\boldsymbol{x}^k$ such that the observed flow $\boldsymbol{f}$ and the model prediction $\hat{\boldsymbol{f}}^k = \mu^k + \boldsymbol{B}^k \boldsymbol{x}^k$ coincide well. Since $\boldsymbol{z}^k$ approaches 1 for small differences between expected and observed flow fields, and is smaller otherwise, this will lead to c_k being close to 1. On the other hand, for an expert trained on different depth profiles, the model prediction should approximate, but never well match, all of the observed flow field, which will lead to at least some flow vector components where observed flow $\boldsymbol{f}$ and expected flow $\hat{\boldsymbol{f}}^k$ deviate, leading to lower $\boldsymbol{z}_j^k$ and thus to a smaller c_k.

Preliminary tests, where we tried mixing expert opinions weighted by c_k, were unsuccessful. We therefore use a simple winner-takes-all (WTA) scheme, which means we trust only the representation of the expert E^k with the highest c_k value for the observed flow. We then apply that expert's motion mapping m^k to the estimated coefficient vector $\boldsymbol{x}^k$ and return the resulting motion estimate.

4.3. Evaluation

The described mixture of expert system is evaluated here using different numbers of experts. First, an in-depth analysis of two failure cases of a single estimator and the improvement of a 10-expert system is presented. Next, the effect of expert number is analyzed and compared to a single estimator. After evaluating the expert choice, estimated trajectories are compared to another self-motion estimation approach in the literature.

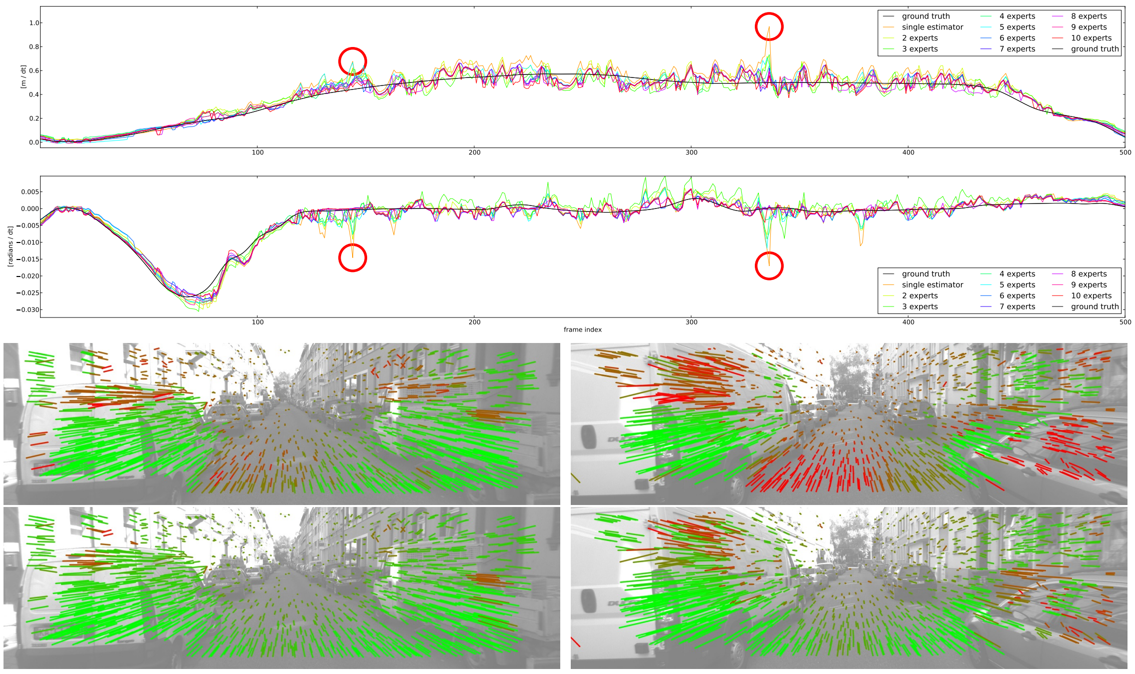


Figure 4.3.: Examples of failure cases of a single self-motion estimator and the corresponding result from a mixture of $K = 10$ experts The two top plots show the time course of ground truth (black) and estimated change in position (**top**) and yaw (**second row**) of a single estimator (orange) and mixtures of 2 to 10 experts (colors from yellow to red) on the first 500 frames of the testing set. Two wrong motion estimates of a single estimator in frames 144 and 336 are highlighted with red circles. The corresponding frames are shown in the next two lines. For both frame 144 (**left**, also described in sec. 3.8.5) and 336 (**right**) shows the inlier/outlier assignment to flow vectors for a single estimator (**third row**) and the expert with the highest confidence of a mixture of 10 experts (**bottom row**).

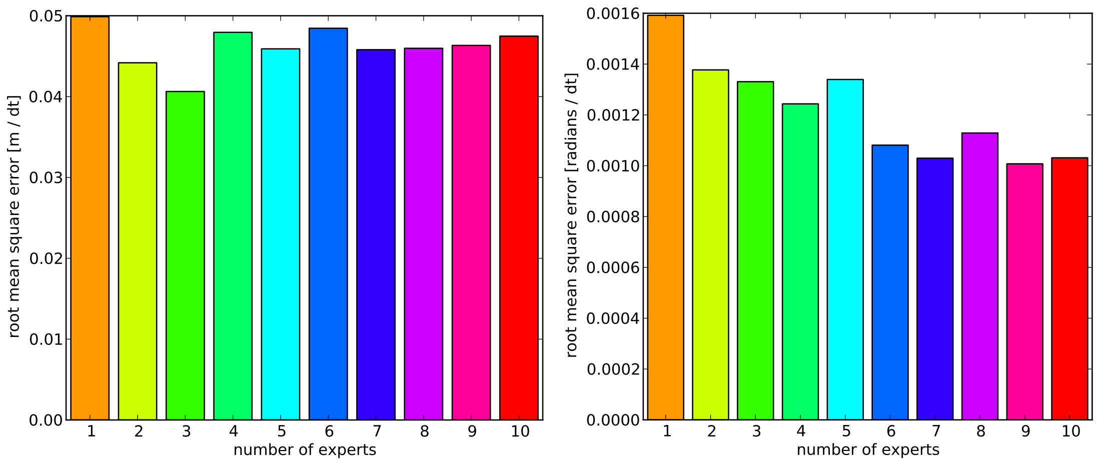


Figure 4.4.: Effect of expert number on the root mean squared error of estimates of change in position (**left**) and yaw (**right**) when using homogeneous inlier variance and $N = 2$ subspace dimensions. While the performance in position change estimates seems to stay approximately constant, the estimation of yaw changes shows a clear trend to better performance with increasing number of experts.

4.3.1. Analysis of two Examples

Fig. 4.3 shows two examples of the effect described in sec. 3.8.5: The presence of close-by cars in the left and right periphery create large numbers of flow vectors that violate the constant-depth assumption, leading to a misinterpretation of the flow in the single estimator, and thus to a wrong motion estimate of a quick left turn.

A mixture of ten experts has one expert trained on such narrow geometric configurations. This expert's flow subspace therefore contains a close match to the observed flow on both street and objects, leading to a high inlier probability of flow on both objects and street and a correct observer motion estimate. Other experts were trained on other geometric configurations, thus classifying more flow vectors as outliers, so the correct expert's estimate is chosen in the final decision stage.

4.3.2. Average Performance

This effect is not unique to these two example frames, Fig. 4.4 shows that mixtures of several experts consistently outperform the single estimator (orange bar) in estimates of changes in both position (left) and yaw orientation (right) for experts trained with homogeneous inlier variance.

Interestingly, this is not the case any more when using inhomogeneous inlier variance, as shown in Fig. 4.5. Performance increases from a single to three experts, but then quickly

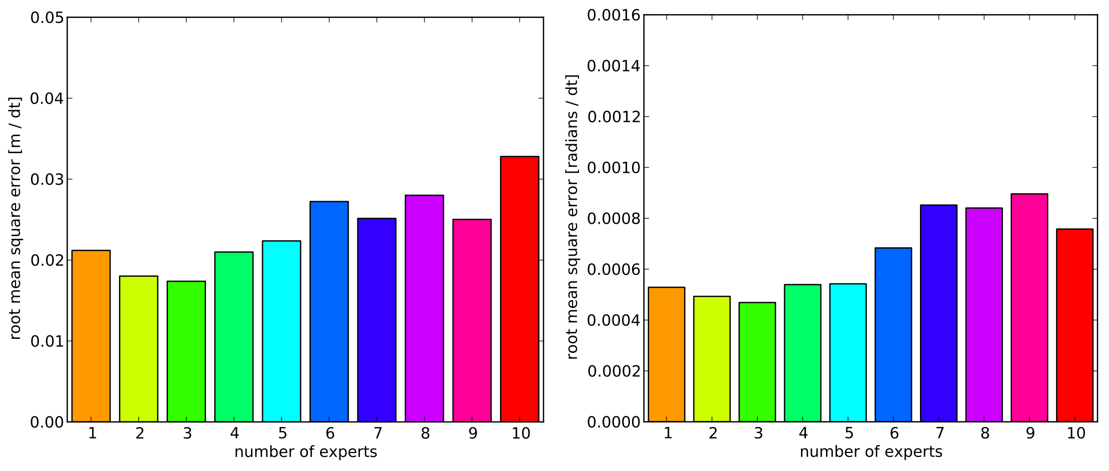


Figure 4.5.: Effect of expert number on estimates of change in position (**left**) and yaw (**right**) when using inhomogeneous inlier variance and $N = 2$ subspace dimensions. While 2 and 3 experts lead to better estimation performance for both change in position and change in yaw, adding more experts weakens the performance. See text for possible reasons.

decreases even below the level of a single estimator. One possible explanation is over-training. For self-motion estimators with $N = 2$ subspace dimensions, which were used here, inhomogeneous inlier variance introduces an increase in number of parameters to optimize by a factor of 1.5. On the other hand, the amount of training data per expert decreases with the number of experts since the overall number of training samples is constant. This could lead to several highly specialized experts that are not able to generalize correctly. This can, however, not be the only reason, since then the trend to better performance in inhomogeneous estimators should at least last until $K = 6$ experts, since there the number of free parameters is the same as for a homogeneous 9-expert system.

The absolute level of performance could also play a role, since mixtures of experts with inhomogeneous inlier variance outperform mixtures of experts with homogeneous variance for all expert numbers considered here. It is also possible, that the data set does not offer enough diversity in geometric configurations, so that 3 experts are already able to cover the whole data set and a higher number only captures irrelevant information like noise.

4.3.3. Expert Choice

To show that the confidence value c_k is a reasonable measure for choosing the right expert during testing, the choice of the expert mixture was compared with the choice

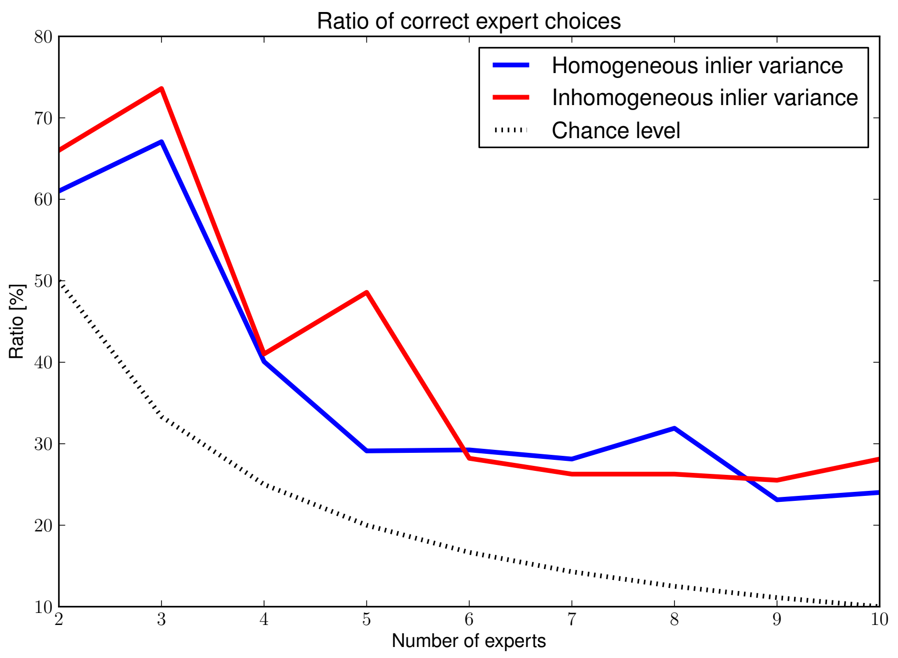


Figure 4.6.: Ratio of frames where the mixture system chose the correct expert during inference. Shown are the ratios for different numbers of experts (x axis) for mixtures of homogeneous (blue) and inhomogeneous (red) experts, as well as the chance level (dotted black line).

that would have been made by the mixture of Gaussians used in training if full disparity information had been available. As can be seen in Fig. 4.6, the ratio for experts with both homogeneous and inhomogeneous inlier variances decreases with expert numbers, which is to be expected given the increasingly hard choice between more and possibly more similar experts. Given 2 or 3 experts, the correct choice is made for more than 60% of all frames, showing that the system is mostly able to correctly 'hallucinate' the relevant aspect of depth for the given task. Interestingly, this graph also shows a larger decline between 3 and 4 experts, just like the estimation performance displayed in Fig. 4.4 (left) and Fig. 4.5 and discussed above.

4.3.4. Comparison to State of the Art

To show that the self-motion estimation approaches described in this and the previous chapter can compete at least qualitatively with state of the art self-motion estimators, trajectories produced with mixtures of 1, 3, and 7 experts were compared to results obtained using a publicly available implementation of Geiger et al. [2011], the authors of the data set used here. The real-time approach described there is based on feature

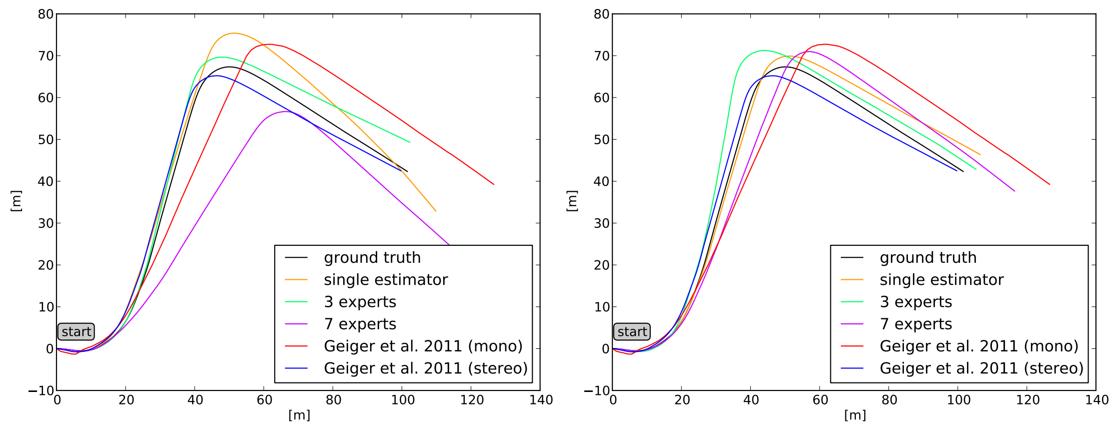


Figure 4.7.: Comparison of trajectories (top down view) obtained by homogeneous (left) and inhomogeneous (right) mixtures of experts. In each plot, the trajectory obtained by integrating ground truth data on position and yaw changes (black), results of a state-of-the-art approach by Geiger et al. [2011] using stereo (blue) and monocular (red) input are shown for comparison with trajectories obtained from integration of results of a single estimator (orange), and mixtures of 3 (green) and 7 (violet) experts.

matches between successive stereo image pairs and uses explicit geometric knowledge obtained from calibration. Example trajectories for drive '2009_09_08_drive_0016' are shown in Fig. 4.7. While the mixture of expert system does not reach the performance of the stereo approach in Geiger et al. [2011], trajectories of mixtures of experts with inhomogeneous inlier variance show qualitatively similar performance. Results from the monocular version of Geiger et al. [2011], which is labeled as experimental, are also included in Fig. 4.7 as red curve, exhibiting slightly worse performance than some of the expert results.

4.4. Conclusion

Extending the applicability of the self-motion estimation approach from the last chapter through a mixture of experts system was investigated in this chapter. By using depth information from stereo the training set was split into several clusters, each of which gives rise to a self-motion estimator tuned to a certain depth profile. Expert motion estimates are combined using the inlier probabilities assigned to flow vectors during inference.

Improved results compared to a simple self-motion estimator were shown in two examples and for increasing numbers of experts with homogeneous inlier variance. Worse performance for more than 3 experts with inhomogeneous inlier variance could be due to over-training or a limit in geometric variability in the data set. A similar trend can be observed in the ratio of correct expert choices. Until the number of experts exceeds 3 the gating uses the correct expert estimate in more than 60% of all frames, an example of hallucination of depth information.

As to future work: evaluation on a data set with constant camera setup but a much larger variation of geometric scenes (open-country roads, highways, etc.) would be an interesting endeavor, clarifying to what extend a mixture of experts removes the limitations imposed by the constant depth assumption.

Defining experts required supervision in form of context knowledge from stereo. Since the optical flow fields in training implicitly contain information on depth, an additional clustering variable in the training procedure might be able to define clusters without additional input.

An interesting investigation would also be whether a two-expert system is able to capture flow components, that a single estimator with twice the number of subspace dimensions can not. In a way, an expert system forces a structure onto the set of all free parameters, which could have positive and negative consequences. As a start, training experts with subspace dimensionalities other than $N = 2$ (possibly a mixture of experts with different subspace dimensionalities) could be tested.

Finally, a continuous combination rule of expert estimates could not only improve motion estimates, but also give rise to an approximation of the current depth profile by combining expert depth profiles in the same way.

CHAPTER 5

Application: Estimating the Focus Of Expansion

This chapter is a reformatted and slightly extended copy of the paper "Monocular Heading Estimation in Non-stationary Urban Environment" by Christian Herdtweck and Cristóbal Curio, which was presented at the IEEE International Conference on Multisensor Fusion and Information Integration (MFI 2012).

5.1. Introduction

Estimating heading information is one of the classic computer vision problems in understanding and designing autonomous systems. In this context, estimating the focus of expansion (FoE) from optical flow has been recognized early (Gibson [1979], Longuet-Higgins and Prazdny [1980]) as a fundamental vision component in spatial navigation and has become a well-researched problem. Nowadays, a "complete" motion estimate, including translation and rotation in all 3 dimensions, is often preferred, but there are applications where this simplified motion representation suffices. The FoE is still applied in several computer vision tasks such as 3D environment reconstruction or for the estimation of the time to impact, for example, in order to plan motion and avoid obstacles. Also, since its estimation is much easier than a full motion estimate, it can be accomplished on simpler hardware. Many computational methods have been suggested (Sazbon et al. [2004], Woelk and Koch [2007], Schill and Mahony [2011]), yet, general scenes still pose a technical challenge. Typical sources of errors occur in unconstrained visual environments, such as urban traffic scenes. Approaches have to deal with independently moving objects that cover a large portion of the visual field, erroneous and sparse flow measurements, costly computation of dense flow fields, over-smoothing of flow, and potentially large platform rotations.

In this chapter we suggest to employ a learning-based global model framework, adapting a statistical learning approach that has recently been proposed by Roberts *et al*. Roberts et al. [2009]. We extended this method and exploit it here for FoE estimation and demonstrate its use for pitch and roll estimation among velocity and yaw. The approach is monocular, free of camera calibration, in a classical sense, and robust to various error sources. It consists of a Latent Variable Model (LVM) that is learned from incomplete optical flow fields on a low-dimensional grid in the image. The latent low-dimensional flow subspace corresponds linearly to incremental platform motion. The LVM is learned in an unsupervised fashion using an Expectation-Maximization scheme, and is able to dissociate inlier flow (caused by the observer) and outlier flow (caused by, e.g. independently moving objects) (see Fig. 5.1). All this approach requires for training is a data set of sparse image flow and corresponding incremental platform motion. In the presence of outliers it is able to generate an error-free flow field. From this the FoE can be numerically derived with a divergence operator. Just as an intermediate step, we use these FoEs on discretized image space as baseline to further train linear regression and non-linear Gaussian-Process regression functions in order to predict the FoE directly from the current flow subspace state, thus by-passing the explicit knowledge of incremental platform motion.

This chapter is structured as follows: In sec. 5.2 we provide a brief overview of FoE estimation approaches. In sec. 5.3 we present the components of the learning-based framework for FoE estimation. We evaluate and discuss the heading estimation results on long image sequences in sec. 5.4 and conclude with sec. 5.5.

5.2. Related Work

Since Gibson Gibson [1979] popularized the usage of optical flow for explaining human navigation behavior, computer vision researchers have devoted much work to estimating the Focus of Expansion from optical flow (e.g. Prazdny [1981]). Early work can be divided into three categories, namely the continuous method employing dense optical flow fields (e.g. Longuet-Higgins and Prazdny [1980]), the discrete method using sparse correspondences between frame pairs and the direct or geometric method (Jain [1983],

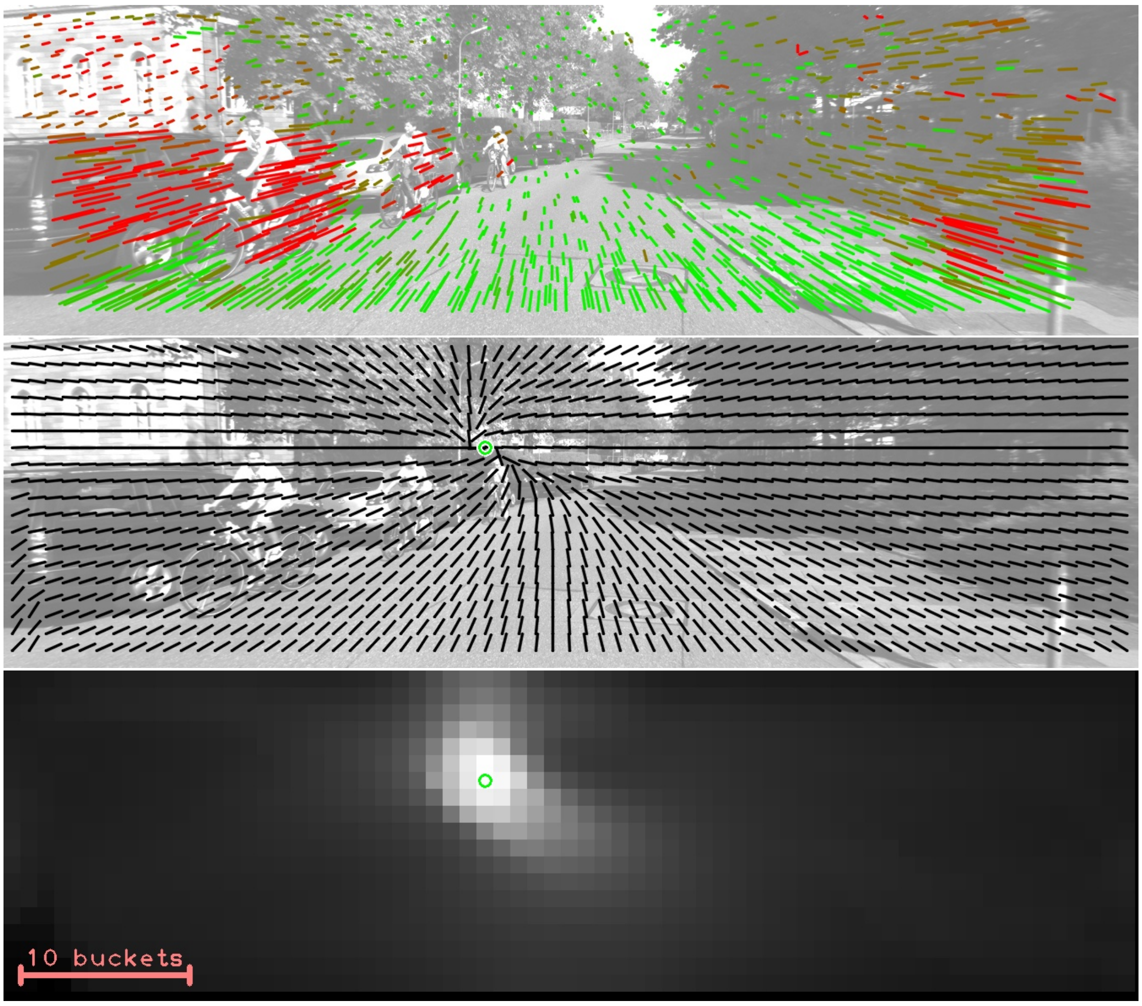


Figure 5.1.: Numerical FoE estimation for a test frame in which the car makes a slight left turn. **Top**: Observed flow after discretization (bucketing), colored according to inlier probability (ranging from inlier=green to outlier=red) as assigned by an LVM with $J = 2$ subspace dimensions. Independently moving objects, regions of unexpected depth, and false flow measurements are correctly detected. **Middle**: Synthesized flow field created by 2D LVM subspace solution. Note that this estimate is free of independent motion. The green circle marks the estimated FoE position. **Bottom**: Robust divergence calculated on synthesized flow field; white values correspond to high divergence. One bucket corresponds to 20×20 pixels.

Negahdaripour and Horn [1989], Negahdaripour [1996]) that either uses established correspondences or stationary points. One of the early algorithms has been implemented in hardware McQuirk et al. [1997]. The need for dealing with outliers was only recognized later, partly due to results on human motion perception (Hildreth et al. [1992]), which is still an active field of research (Royden [2002], Layton et al. [2012]). Since then, many different methods have been applied to FoE estimation Fejes and Davis [1999], including state-of-the-art statistical methods Woelk and Koch [2007]. FoE estimation has also been performed on panoramic input Schill and Mahony [2011].

Computationally, recent methods can be roughly categorized into local and global models. Local models for FoE estimation require strongly converging flow at one point in the image. But flow can often not be directly measured in the neighborhood of the point of interest. The correct FoE might also be occluded by an independently moving object or lie outside the visual field due to strong rotation. Global models exploit knowledge of the flow field in the whole image. Robust methods based on RANSAC have been proposed to deal with independently moving objects and sparse image flow in the visual field, while trying to find an optimal fit of projected flow into the image. Nevertheless, these approaches require a calibrated camera and are still prone, at least for the monocular case, to scale ambiguities caused by varying scene structure.

We base our algorithm on the ego-motion estimation framework proposed by Roberts *et al*. Roberts et al. [2009] that we have extended and interfaced for the task of FoE estimation.

5.3. Methods

In this section we describe two methods for estimating the FoE from sparse optical flow measurements. At the core of both methods lies a Latent Variable Model (LVM) that provides a robust way to encode flow associated with the ego-motion of a camera with a learned flow subspace Roberts et al. [2009]. We first review the LVM in sec. 5.3.1, including an unsupervised method to train it. In sec. 5.3.2 we present our first, numerical method of estimating the FoE from the trained Latent Variable Model output. For the second method we investigate supervised training of a linear regressor or alternatively a non-linear Gaussian Process with linear kernel to directly map from the latent variable space to the FoE (sec. 5.3.3). In sec. 5.3.4 the estimation of incremental platform motion is outlined.

5.3.1. Robust Flow Subspace Model

The usage of a Latent Variable Model is based on the observation that high-dimensional optical flow fields, as observed from a moving platform, reside in a low-dimensional manifold. This stems from the fact that the underlying vehicle's motion is limited to at most 6 degrees of freedom. It has been proven Irani [2002] that this motion leads to flow of at most 6 degrees of freedom, assuming stationary camera parameters, a static world, and only small depth variations at each image position over time. Despite this assumption, the model's robustness and probabilistic nature allow significant depth variation in applications. The LVM model is an extension of probabilistic PCA Tipping and Bishop [1999].

Let $\boldsymbol{f}_t$ be a vector comprising all measured flow vector components at a discrete time index $t \in \{1, 2, \dots\}$. These may include a considerable amount of missing values, e.g. in areas with low image contrast. Each component $\boldsymbol{f}_{ti}, i = 1, \dots, I$ of a flow vector $\boldsymbol{f}_t$ is modeled as either missing or noise or a linear combination of a mean $\boldsymbol{\mu}$ and *basis flow fields* $\boldsymbol{b}_1, \dots, \boldsymbol{b}_J$ plus noise. Assembling the basis flow vectors to a basis flow matrix $\boldsymbol{B}$, we can write each observed flow vector component as:

$$\boldsymbol{f}_{ti} = \begin{cases} (\boldsymbol{B}\boldsymbol{x}_t)_i + \mu_i + \epsilon_{ti}^{\mathsf{in}} & \text{if } z_{ti} = 1 \\ \epsilon_{ti}^{\mathsf{out}} & \text{if } z_{ti} = 0 \quad . \end{cases} \tag{5.1}$$

The latent variables are the following: the continuous variable $\boldsymbol{x}_t = (x_{t1}, \dots, x_{tJ})$ captures the contribution of each basis flow field, while the binary variables $\boldsymbol{z}_t = (z_{t1}, \dots, z_{tJ})$ classify each measured flow field component as an *inlier*, i.e. a match between linear combination of basis flows and observed flow if $z_{ti} = 1$, or an *outlier* explained by Gaussian noise $\epsilon_{ti}^{\mathsf{out}}$ if $z_{ti} = 0$. The inlier flow components are allowed a deviation $\epsilon_{ti}^{\mathsf{in}}$, which, like the outlier noise, is modeled by a zero-mean Gaussian $\epsilon_{ti}^{\mathsf{in}} \sim \mathcal{N}(0, \sigma_{\mathsf{in}}^2)$, $\epsilon_{ti}^{\mathsf{out}} \sim \mathcal{N}(0, \sigma_{\mathsf{out}}^2)$. Missing values are treated as outliers.

For training this model one must only provide a set of flow fields $\boldsymbol{f}_1, \dots, \boldsymbol{f}_T$ and specify the number of dimensions J the model should capture. For the original purpose of simple estimation of the vehicle's speed and change in yaw, a LVM dimensionality of $J = 2$ suffices Herdtweck and Curio [2012a]. Larger values are needed to capture more flow variability caused by different depth and further degrees of freedom of the vehicle. An expectation-maximization (EM, Dempster et al. [1977]) scheme is then used to minimize the negative log likelihood of the observed flow given the model by simultaneously estimating the interdependent basis flow vectors $\boldsymbol{b}_j$, robust mean of the flow, $\boldsymbol{\mu}$, variances of inlier and outlier flow components, σ_{in} and σ_{out}, parameter vectors

$\boldsymbol{x}_t$ and inlier/outlier assignments $\boldsymbol{z}_t$. For the detailed formulation of the likelihood and EM equations see Roberts et al. [2009]. The original paper required a priori selection of the variance of the outlier noise distribution. We extended the EM procedure by an update equation for this term:

$$\sigma^2_{\text{out}} = \frac{\sum_t \sum_{i=1}^{I} (1 - \mathrm{E}(z_{ti}))\boldsymbol{f}^2_{ti}}{\sum_t \sum_{i=1}^{I} (1 - \mathrm{E}(z_{ti}))}. \tag{5.2}$$

For inference on a test data set we keep the basis flow fields, flow mean and inlier/outlier variances fixed. We then estimate for each observed flow field $\boldsymbol{f}_t$ the optimal subspace coefficients $\boldsymbol{x}_t$ and the inlier/outlier flow assignment $\boldsymbol{z}_t$. For this we only need to run the E-step of the EM-algorithm.

5.3.2. Numerical Solution to FoE Estimation

Being able to synthesize a robust flow field $\hat{\boldsymbol{f}}_t = \boldsymbol{B}\boldsymbol{x}_t + \boldsymbol{\mu}$ with our Latent Variable Model, we expect to find the optimal FoE corresponding to the actual heading of the observing moving platform. We first re-construct a flow field from the components in the vector $\hat{\boldsymbol{f}}_t$ whose flow vectors we then normalize to unit length. We denote the resulting normalized flow field by $\bar{\boldsymbol{f}}_t$. We interpret FoE points as singularities, so called shock points, in the flow field $\bar{\boldsymbol{f}}_t$. We detect singularities of $\bar{\boldsymbol{f}}_t$ by determining the divergence of $\bar{\boldsymbol{f}}_t$ that we approximate by a ring integral for each flow field location resulting in the flux flow $\mathcal{F}$ at each flow vector position (at grid cells resulting from bucketing of sparse flow, see fig Fig. 5.1)

$$\left(\operatorname{div} \bar{\boldsymbol{f}}_t\right)(p) \approx \mathcal{F}\left(\bar{\boldsymbol{f}}_t\right)(p) = \frac{\oint \left\langle \bar{\boldsymbol{f}}_t, \boldsymbol{\nu} \right\rangle ds}{\text{Ring Area}}, \tag{5.3}$$

where $\boldsymbol{\nu}$ denotes the normals on a ring around grid position p with diameter in our case of 7 grid cell points through which the flux flow $\mathcal{F}$ is computed. The computation of this ring integral is implemented efficiently as a convolution of the components of $\bar{\boldsymbol{f}}_t$ with a precomputed kernel containing the normal vectors of the ring. This procedure has also been used in Curio [2004], Engel and Curio [2008]. In the resulting field $\mathcal{F}$ high values indicate locations of strong singularities.

Using $\mathcal{F}$ we sample locations using a non-maximum suppression scheme which selects the local maxima of a height map. We keep FoE hypotheses that are above a set flux flow threshold θ, which controls the number of FoE hypotheses. Note that for a backward

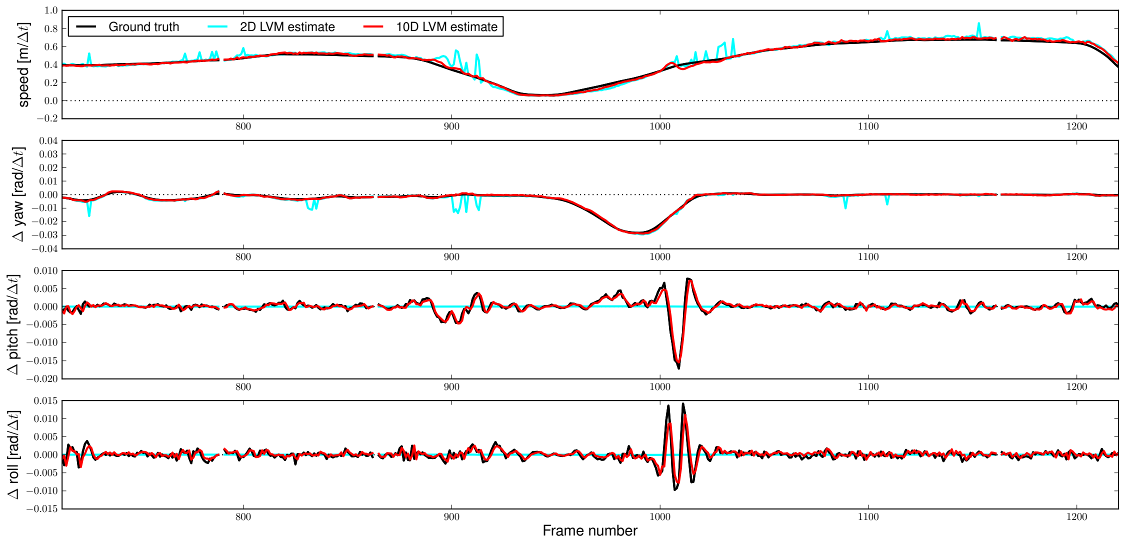


Figure 5.2.: Time courses of estimated platform speed (**top**), yaw (**2nd line**), pitch (**3rd line**) and roll (**bottom**) using an LVM of dimension $J = 2, 10$ and linear regression on a test set. Estimates are shown in cyan $(J = 2)$ and red $(J = 10)$, the ground truth in black. Note the sensitivity of the higher-dimensional estimate to changes in pitch and roll during breaking and acceleration while driving out of a curve around frame 1000.

moving platform the sign of the flux flow would switch, which can easily be detected and handled.

We can roughly categorize three main classes of singularities that can be identified by their flux flow helping us to determine the thresholds of valid FoE hypotheses. In our implementation $\mathcal{F}\left(\bar{\boldsymbol{f}}_t\right)(p)$ yields the highest flux value for concentric flow which could be produced by a moving platform pointing straight ahead with pure forward translation. The more the platform rotates the more the maximum flux flow deviate from the center, the less flux flow we expect, but still enough to detect the FoE. For strong rotation the optimal FoE would be out of the viewing field, thus the maximum would reside on a flux flow ridge, possibly with an unstable extremum residing on the edge of the image. Setting the threshold parameter θ will control what types of interest points are obtained by the algorithm.

5.3.3. Direct FoE Prediction from LVM

The second method for estimating the FoE based on the model described in sec. 5.3.1 takes a more direct approach than the first one: we by-pass synthesizing a flow field $\bar{\boldsymbol{f}}_t$ and calculating its divergence by learning a direct mapping from subspace coefficients

$\boldsymbol{x}$ to the position of the FoE in the image. We first train a linear mapping using the iteratively re-weighted least squares algorithm with a bi-square weighting function to obtain a matrix $\boldsymbol{M}$ and offset $\boldsymbol{m}$ which define the mapping $\boldsymbol{o}(\boldsymbol{x}) = \boldsymbol{M}\boldsymbol{x} + \boldsymbol{m}$ from subspace coefficients on the training set $\boldsymbol{x}_1, \cdots, \boldsymbol{x}_T$ to ground truth FoEs $\hat{\boldsymbol{o}}_1, \cdots, \hat{\boldsymbol{o}}_T$. We train this mapping on the same training set that is used for unsupervised learning of the LVM with ground truth FoEs obtained as described in sec. 5.4.1. During inference we simply apply the learned mapping to the estimated subspace coefficients: $\boldsymbol{o}_t = \boldsymbol{o}(\boldsymbol{x}_t)$.

We also estimate a non-linear mapping from subspace coefficients to FoEs using Gaussian-Process regression since this offers a prediction of the variance of the FoE estimation, that can be exploited for judging the reliability of the estimate for further high-level heading information integration in other tasks. We use the publicly available GPML toolbox[1] implementation of an affine mean function, a Matérn covariance function with isotropic distance measure and employ a Gaussian likelihood function to allow exact inference. All further parameters are determined from training data using cross validation.

5.3.4. Mapping to Motion

As mentioned earlier, this subspace method was initially designed as a basis for self-motion estimation. We also trained a mapping, similar to the one for the FoE, to incremental platform motion, as done in Herdtweck and Curio [2012a], but we provide first results here for pitch and roll, that are usually difficult to estimate in open environment. A similar mapping for $J = 2$ is used in the ground truth, described in sec. 5.4.1 below.

5.4. Evaluation

In this section we provide results obtained by the methods described in the previous section. We train each method on different LVM sizes ($J = 2, 3, 4, 5, 7, 10$) on the training set and then perform inference on the test set as described in sec. 5.4.1. For training our model and evaluating our estimates we only consider frames for which the ground truth FoE estimate is within the field of view and the platform motion is not close to zero. Both these cases would lead to divergence fields with only unstable and weak flux minima and thus unreliable ground truth information. Our system can detect such cases and discard the corresponding frames automatically. In the estimated time

[1]available online: `http://www.gaussianprocess.org/gpml`

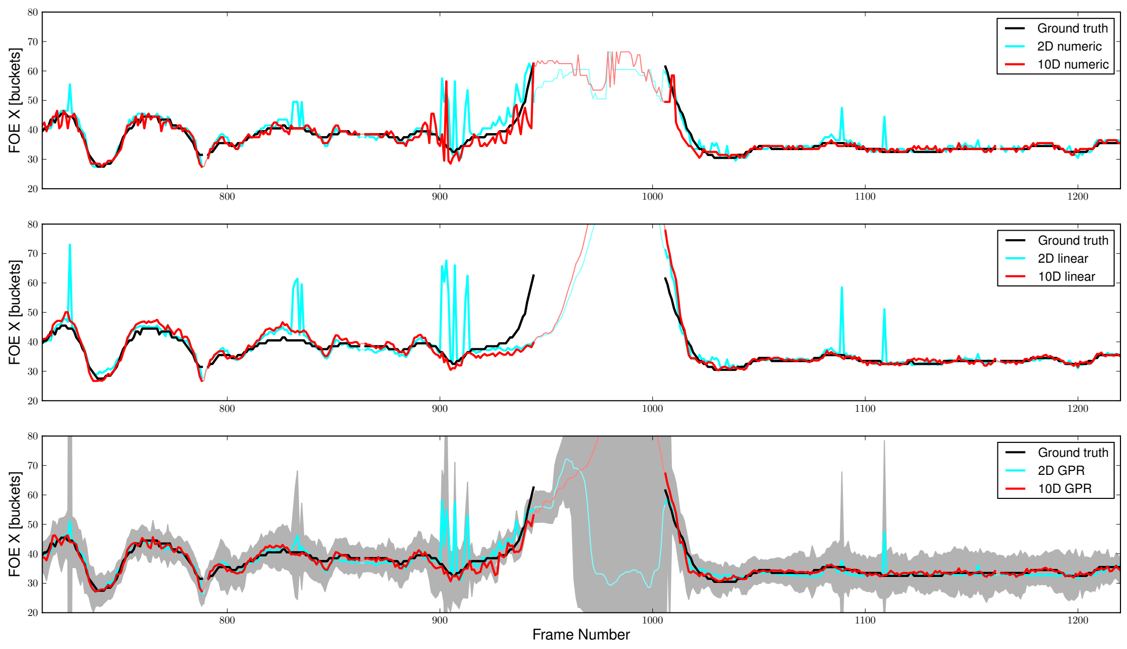


Figure 5.3.: Time courses of estimated lateral FoE coordinates using the divergence operator (**top**), linear mapping (**middle**) and non-linear mapping (**bottom**) on frames $713 - 1220$ of the test set. FoE estimates with LVM dimensionality $J = 2$ are shown in cyan, with $J = 10$ in red, the ground truth FoEs in black. For the non-linear mapping, the gray region indicates the estimated uncertainty. At time points where the lines are lighter and thinner the ground truth FoE can not be uniquely determined since it is out of the field of view of the sensor.

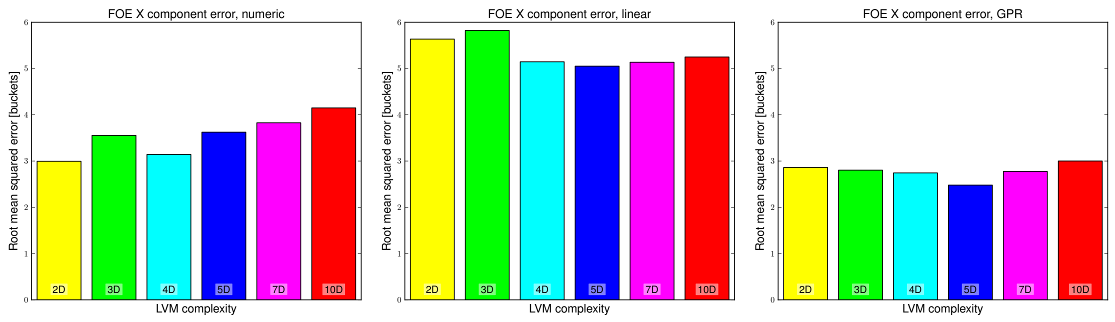


Figure 5.4.: Root mean squared error (RMSE) for different LVM model complexities (J) for lateral FoE estimation using divergence (**left**), linear (**middle**) and non-linear regression (**right**) on the complete testing data.

course plots (Fig. 5.3), the remaining “invalid” frames are still rendered as lighter and thinner lines.

We report details about the data set (sec. 5.4.1) and the subspace model (sec. 5.4.2), as well as results on FoE estimation (sec. 5.4.3) and incremental platform motion (sec. 5.4.4). We finish with remarks on the computational effort for calculations (sec. 5.4.5).

5.4.1. Data Set

We perform training and inference on the publicly available data set by Geiger and colleagues (Geiger et al. [2010], Kitt et al. [2010]), which is quite challenging for the task of FoE estimation due to the high degree of clutter and independent motion, that even occludes the FoE in some examples. It consists of rectified stereo images captured from a car in an urban environment, and corresponding ground truth motion from an on-board IMU and GPS. We use the left camera’s frames of six image sequences (numbers 10, 12, 15, 16, 19, 21) from the first recording day to ensure a constant camera setup. We divide each sequence into a first half used for training and a second half for evaluation, thus arriving at 3207 frames for training and testing, respectively. The rectified images have a resolution of 1344×391 pixels and were captured at ≈ 10 frames per second.

Optical Flow. For calculating optical flow we use *libviso2* Geiger et al. [2011]. It includes a bucketing procedure that reduces the size of the calculated flow fields. In particular, each frame is divided into a grid of non-overlapping buckets of 20×20 pixels, from which one flow vector is chosen at random. The bucket size is chosen to trade off the number of missing values and to provide a reasonable size of flow feature vector that is amenable to statistical learning.

Ground Truth FoE. Since the described data set has no labeled FoE, we have to resort to estimating it based on the LVM and the ground truth information of incremental platform motion provided with the images. To obtain ground truth FoEs we trained a 2-dimensional LVM on our training set and learned a linear mapping from this subspace to the ground truth platform motion on the same set of training data. By inverting this mapping and applying it to the ground truth motion we obtained ground truth subspace coefficients in this particular flow subspace, that only capture the vehicles motion. These were then used together with the basis flow fields to synthesize clean flow fields, on which we calculated the FoE using the numerical divergence method described above.

The initial choice of a 2-dimensional subspace was based on earlier experiments which revealed that the first two flow subspace dimensions of the LVM mainly capture speed and yaw, which were two reliable degrees of freedom of the data set. Adding more

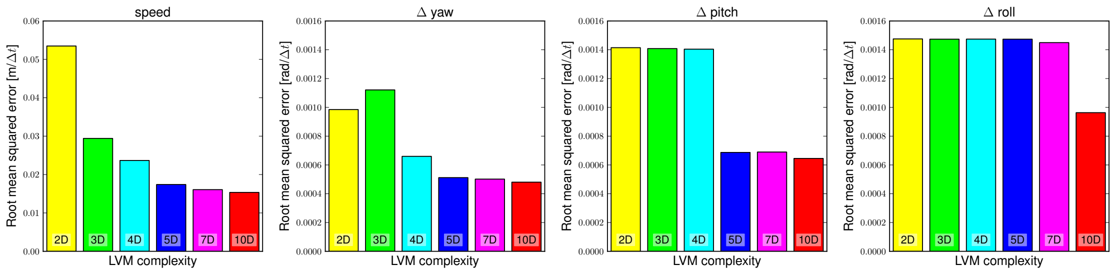


Figure 5.5.: Root mean squared error (RMSE) for different LVM model complexities (J) for incremental platform motion estimates using a linear mapping for the complete testing data. From left to right: speed, change in yaw, pitch, and roll.

dimensions adds flow variations caused by different scene depths as well as changes of vehicle pitch and roll to the representation, which would lead to noisier synthesized flow fields.

Pitch. The data set we analyzed contains only some variation in camera pitch with unreliable ground truth (drift over time). We therefore report in the following only results on the lateral (X) component of the estimated FoE. Note, however, that for large enough subspace dimensionalities our system can be trained to accurately estimate changes in pitch and even roll (c.f. Fig. 5.2).

5.4.2. Robust Flow Subspace Model

Since we have no ground truth for correctly trained subspace basis flows or inlier assignment, we can not directly evaluate the quality of fit of the trained LVM. However, detailed visual inspection of the results in Fig. 5.1 reveals that the 2-dimensional LVM is able to correctly discard flow from independently moving objects (passing bikes), false flow measurements in the trees and on the ground, as well as, for example, the very far structure on the right.

5.4.3. FoE Estimates

A more rigorous evaluation is possible for the results of the full FoE estimation pipeline. We compared the lateral coordinates estimated by all methods described above with our obtained ground truth FoE values. Fig. 5.3 shows estimates and ground truth FoEs for the divergence-based FoE estimation, as well as the FoE estimation using a robust linear and non-linear mapping from subspace coefficients to the lateral FoE position. Displayed are estimates obtained using a LVM model of $J = 2$ and $J = 10$ subspace dimensions as input to the regression estimators.

For all three methods there is a visible effect of dimensionality: the number of spikes in the estimates based on the lower-dimensional subspace is clearly reduced using the higher-dimensional LVM. This effect is probably due to the first step of the FoE estimation process, the interpretation of the seen flow field in terms of basis flow fields. The limited flexibility of the $2D$-estimator leads to misinterpretations of flow fields stemming from scenes with a strong deviation from the expected depth profile. The higher-dimensional estimator has captured such variations in scene depth during training and is therefore able to correctly interpret the flow fields, leading in the second step (numerical estimation or mapping) to a correct FoE estimate.

The linear estimator (middle plot) shows slightly higher peaks and more deviation from the ground truth for 2 and 10 dimensional estimators on the displayed part of the test set, confirming the overall trend summarized in the next Fig. 5.4. The Gaussian-Process regression does not only outperform both other methods, but also accurately assesses the quality of its estimate, e.g. in cases where it needs to interpolate or extrapolate. In cases of wrong estimates as well as for the middle region, where the FoE is outside the image, the gray uncertainty range grows drastically, which could be used for further information integration in other vision tasks, such as FoE tracking.

Inspection of the left plot of Fig. 5.4 shows that the divergence-based estimation of the FoE seems to get worse for more degrees of freedom in the underlying LVM. This likely may be due to more complex patterns emerging from combinations of more basis flow components, resulting in unstable extrema in the obtained divergence fields.

The linear mapping shows slight improvement for subspace dimensionalities $J \geq 4$. This might be attributed to the larger LVM subspace, which seems to capture more of the variance observed in real-world flow, leading to a more powerful representation and thus improved interpretation of observed flow fields.

Finally, we observe that non-linear Gaussian-Process regression outperforms the other two for all LVM complexities, thus providing further empirical evidence that the relationship between motion and FoE is non-linear.

5.4.4. Observer Motion Estimation

Fig. 5.5 shows the RMSE of incremental platform motion estimates. While speed and change in yaw overall improve with LVM complexity, pitch and roll show a clear drop in error from four to five subspace dimensions (Δ pitch) and from seven to ten dimensions (Δ roll). It appears that only higher subspace dimensions can capture flow variations caused by pitch and roll because they are less salient than the dominant flow field changes caused by speed and yaw. For estimation results over time see Fig. 5.2.

5.4.5. Computational Effort

To show the real-time applicability of the described methods, we report here timing results of FoE estimation on an Intel Xeon CPU with 2.66 GHz and sufficient RAM. We use implementations in python 2.6 and Matlab, all not parallelized. Calculating the divergence took 0.02μs in Matlab. The estimation time of the flow subspace coefficients by the LVM ranges from 30ms for $J = 2$ to 53ms for $J = 10$. Mapping from subspace coefficients to FoE depends on the implementation of the mapping. For the simple linear regression, for example, we measured 0.032ms for $J = 2$ and 90ms for $J = 10$.

5.5. Conclusion and Outlook

We have studied and extended a new global approach for the robust estimation of heading information. It is learning-based and does not require classical camera calibration. It consists of a robust continuous Latent Variable Model that represents a flow subspace as a means to suppress different sources of errors occurring during self-motion induced optical flow perception in urban environments. Our goal was thus to estimate and generate the optimal optical flow field as it would have been perceived by a moving observer, overriding flow from independently moving objects, measuring optimal image correspondences, and neglecting ambiguous flow from scene structure deviating from a mean depth. For on-line application our approach does not require information about incremental platform motion, only during training. It is able to predict from the current LVM state directly the FoE in the image with a non-linear Gaussian-Process regression, bypassing less stable numerical local detection with a divergence operator on the optimized flow field. Besides outperforming robust linear regression, non-linear Gaussian-Process regression provides in addition a variance estimate of the FoE prediction that can be of use for further integration into other vision tasks. Further, in this paper we also have presented estimation results for pitch and roll of the camera for challenging image footage taken from urban environments. Ground truth for the FoE's Y-component could not yet be provided through our generic approach, but would be straight forward to learn with Gaussian-Process regression, since larger LVM dimensions capture the relevant features to predict the corresponding platform motion.

CHAPTER 6

Estimation and Filtering of Object Orientation

An important goal of dynamic visual perception is the estimation and prediction of positions of other moving objects. In the automotive context, for example, it is crucial to estimate the motion of all objects in the vicinity in order to avoid collisions. Object orientation is an important cue since most vehicles are restricted in their motion by their orientation (a car cannot drive leftwards, for example).

In dynamic tasks, orientation is usually inferred from consecutive estimates of position. Direct estimation of this variable can, however, have benefits like an increase in precision, faster filter initialization, and allows potential motion directions even for static objects.

One problem that is often encountered when estimating orientation from images, is ambiguity. Objects can look very similar from very different viewpoints, as documented, for example, by many reports in the literature (Andriluka et al. [2010], Savarese and Fei-Fei [2010], see below) of confusions of rear and front views of cars. In this chapter, a learning based approach for the estimation of object orientation from static images is described, whose output is able to represent such ambiguous knowledge. Orientation is modeled as a set of circular continuous values, which are then interpreted using kernel density estimation. This allows a very general and multi-modal probabilistic representation of orientation knowledge.

Most research on viewpoint estimation is motivated by work on object detection. Since object appearance, which object detectors build on, is strongly dependent on viewpoint, an implicit or explicit representation of viewpoint is contained in many object detectors (c.f. sec. 6.1.1). In this chapter, this relationship is exploited in the other direction: representations built by object detectors are used as input to a learning-based regression framework that has as goal an estimate of viewpoint.

Since in object detectors viewpoint variables often only identify one of several viewpoint ranges, it is often represented as a discrete variable. Changing its role to output of a regression, that further probabilistic reasoning is applied to, requires a richer representation. It is therefore modeled in this and the next chapter as a continuous circular variable.

Requiring circular output limits the number of frameworks available for regression. While one could regress onto a 2-dimensional linear variable and then only use the angle as regression output, a more elegant solution is an adaptation of the internal workings of a regression approach to circular variables. Random regression forests are a very flexible method that allows such an adaptation and additionally offer a good generalization performance as well as fast run time. They have been used for viewpoint regression before Fanelli et al. [2011b,a], but the limitation of viewpoint ranges in those studies allowed keeping a linear representation of viewpoint.

Finally, a filtering framework is presented to disambiguate orientation estimates using temporal context. Also adjusted to usage with circular continuous variables, two particle filter methods are applied to static viewpoint estimates. Given the general form of viewpoint regression output, measurement update equations based on target-tracking in clutter were applied. Related inference methods on partly circular spaces are mentioned in sec. 6.1.4.

To summarize, the following contributions are made in this chapter:

- Using the representation formed by a part-based object detector for viewpoint estimation
- Representing viewpoint as a continuous circular variable using circular statistics
- Adaptation of random regression forests to multi-modal circular output
- Dynamic filtering of viewpoint estimates using two different particle filter methods with measurement update based on target-tracking in clutter

This chapter is structured as follows: after discussing related work, the part-based object detector is described that provides the input features for the viewpoint regression forests. Next, modifications to the standard random regression forest approach are described. After a first evaluation of results obtained with regression on static images, a circular particle filter for providing temporal context is described and evaluated.

Parts of this chapter have been submitted for publication as Herdtweck and Curio [2013].

6.1. Related Work

6.1.1. Multi-View Object Detection

As mentioned above, estimating object orientation is most often found in the literature as a means to improve object detection. This is partly motivated by results on biological vision systems, showing viewpoint-dependent coding in humans (Bülthoff and Edelman [1992]). Since object appearance can change drastically with varying viewpoint, several different explicit and implicit representations of viewpoint information have been presented in the object detection literature. To mention just a few more recent approaches, Thomas et al. [2006] enhances codebook features with tracks over continuously changing viewpoints, thus creating an implicit viewpoint structure. In Savarese and Fei-Fei [2007] and in a later improved version of the same approach by Su et al. [2009], viewpoint orientation is encoded in the spatial arrangement of features in a more explicit sense using homographies. This way, new views can be approximated by warping known exemplar images. Özuysal et al. [2009] uses refined bounding boxes and their aspect ratios to select one of 16 view-tuned detectors. The approach by Felzenszwalb et al. [2010] is a collection of deformable part based detector components specialized on certain viewpoint ranges. Andriluka et al. [2010] follow a similar approach training a bank of 8 object detectors, one for each viewpoint class. López-Sastre et al. [2011] extend the learning component of Felzenszwalb et al. [2010] to viewpoint-labeled ground truth samples.

Using 3-dimensional object models simplifies estimation and usage of viewpoint information. The detection approach by Glasner et al. [2011] uses image patches of discriminative object parts to find a matching orientation of a 3-dimensional object model. Pepik et al Pepik et al. [2012] also use a 3-dimensional object model to extend Felzenszwalb et al. [2010] by including an explicit discrete viewpoint variable and adding 3-dimensional bounding box information to parts. Non-photo-realistic renderings from 8 different viewpoints of 3-dimensional object models and their parts are compared to image shape context in Stark et al. [2010]. This is extended in Zia et al. [2011] removing the need for discrete-viewpoint representation.

Results for viewpoint estimation are seldom specified in work on object detection. Exceptions report mean accuracy when classifying object viewpoints into one of several (usually 8) classes. Savarese and Fei-Fei [2007], Su et al. [2009], report an average ratio of correct viewpoint classification on fully visible household objects of 57.2% and $\approx 67\%$ (averaged over 3 different elevation levels), respectively. The study Özuysal et al. [2009]

uses 16 car detectors but reports errors only in graphical form. Again classifying viewpoint into 1 of 8 categories, but in real-world images of people Andriluka et al. [2010] report classification performance between 31.1% and 42.2% . Using 3-dimensional object models Stark et al. [2010] and Zia et al. [2011] report 81% and 84% classification accuracy, respectively, for 8 car viewpoint classes. Using $10°$ intervals, performance is at 67.4% and 73.3%, respectively. Within the correctly classified samples, Zia et al. [2011] report a mean angular error in azimuth an elevation of $\approx 4°$. Glasner et al. [2011] correctly classified car viewpoint into 1 of 8 classes in 85.3% and median angular errors of $24.03°$ and $10.4°$. In López-Sastre et al. [2011], a more detailed analysis of viewpoint estimation results can be found. The correct classification ratios for 16, 8 and 4 viewpoint classes of cars are 66%, 73.75%, and 75.75%, respectively. For the household object data set, an average correct classification ratio of 79.2% and 69.5% are reported for their approach and Felzenszwalb et al. [2010], respectively, using 8 viewpoint classes.

Apart from the work in object detection, some studies focus on viewpoint estimation and tracking without detection. Using stereo measurements and a weak cuboid car model, Barrois et al. [2009] estimate the full object pose. In Gosch et al. [2008] the full object pose is estimated from object outlines and represented on a sphere for tracking.

Finally, here is a quick overview over popular data sets. The Pascal visual object challenges Everingham et al. [2012] contain viewpoint labels for some of the classes, specifying 8 viewpoint ranges. Özuysal et al. [2009] have presented a data set captured at a car exhibition with images of slowly turning cars. Savarese and Fei-Fei [2007] have assembled a data set for 8 object classes captured from 8 different azimuth and 3 elevation angles in 3 different distances. Glasner et al. [2011] captured car images while walking around them. They created 3D object models from the resulting images using structure from motion, which are available for download. ICARO (López-Sastre et al. [2010]), like Pascal, contains text viewpoint labels together with objects. Additional annotation data gathered for some of the above-mentioned studies, is available for download, including 3D object models for the data set by Savarese and Fei-Fei [2007] and stereo sequences used in Barrois et al. [2009].

6.1.2. Temporal Integration of Object Orientation and Viewpoint

The last two sections of this chapter are concerned with temporal integration of viewpoint estimates. We mention here a few other studies that also integrate object orientation estimates over time. Methods for obtaining and filtering orientation estimates and application of filtered estimates vary greatly.

Object orientation is used in various object tracking approaches, where it is inferred from consecutive object position estimates (e.g. Koller et al. [1993]).

In Gosch et al. [2008] a subspace of image contours is learned, that can be mapped to the view sphere of azimuth and elevation angles. Estimates obtained using this method are then tracked on the view sphere using a deterministic physical motion model that involves a potential field and friction. Predicted positions are used as priors for further image contour extraction.

The aforementioned study Andriluka et al. [2010] not only estimates viewpoints, but in a second stage also gathers viewpoint estimates for person tracklets using a simple HMM with Gaussian transition probabilities between the 8 viewpoint classes used. In a third stage, full body articulation is estimated from the results of earlier stages. Unfortunately, neither of these two studies reports filtered error rates.

In a line of work, Schairer and colleagues (e.g. Schairer et al. [2010]) use the spherical Fourier transform on the 3-dimensional space of Euler angles to identify and track orientations of an omnidirectional camera.

Finally, there is a large body of literature on estimation of observer orientation relative to the world, incorporating data from other sensors like GNSS or inertial measurement units. In low-cost augmented reality systems, for example, orientation is mainly estimated using inertial sensors, but cameras are often included to counter drift.

6.1.3. Random Regression Forests

Usage of random regression forests is quite wide-spread in the literature. Proposed in Breiman [2001], random forests are a very general tool for regression and classification. For regression, they are used to predict customer behavior Lariviere and Vandenpoel [2005], interpret chemical analyses Svetnik et al. [2003], and medical data Criminisi et al. [2011], and of course in computer vision. In Shotton et al. [2008]) random decision forests are used to classify image regions. In Gall and Lempitsky [2009] they are used as a generalized Hough transform for object detection, combining classification and regression in a single forest. The authors introduce a deterministic contribution to the purely random selection of training data based on the estimation performance of already trained trees, similarly to Bernard et al. [2012]. They also propose to not create node tests at random, but to select tests that minimize the class label entropy and regression uncertainty on a randomly created set of tests. This technique is also used in Criminisi et al. [2011], Shotton et al. [2011] and Fanelli et al. [2011b]. This latter study also estimates orientation, but only in a limited range, so circular methods are not used. This approach is developed further in Dantone et al. [2012], introducing conditional regression forests, that learn a regression function conditioned on additional parameters like rough orientation. The study by Shotton et al. [2011] includes parameter studies for training random regression forests for body part classification from a large amount of depth data. Biau [2012] is a new theoretical analysis of random regression forests.

6.1.4. Circular Statistics in Computer Vision

Using circular statistics in computer vision is quite uncommon since most investigated variables are (approximately) linear. A few studies using circular methods in computer vision and related applications are summarized here. Further work on general inference using circular statistics are mentioned in sec. 2.3.

Sudderth [2006] discussed the usage of von Mises and wrapped Gaussian distributions in a generative model for for hand tracking. In Agiomyrgiannakis and Stylianou [2007, 2009], mixtures of wrapped Gaussian distributions are used to model the phase of speech data. In tracking of white matter fibers, a circular particle filter and sampling from a von Mises-Fisher distribution is proposed in Zhang et al. [2009]. A mixed space of three linear and two circular (spherical) dimensions is proposed in Detry and Piater [2010] as a model for object pose. A distribution over this space is also approximated using kernel density estimation. The study Pons-Moll et al. [2011] uses von Mises-Fisher distributions to model noise when combining visual and inertial measurements of rotation, requiring sampling of circular data. In Da Costa et al. [2012] image texture orientations are modeled using von Mises distributions, which are used for segmentation.

6.2. The Deformable Part Object Detector

Estimating the orientation of an object from an image requires as first step a suitable representation of object appearance and context. Describing object context is discussed later, this section is concerned with a representation of the appearance of an object.

Describing object appearance has been investigated by computer vision researchers for many years, leading to diverse representations (see sec. 1.1). In order to allow detection in general cases, all these representations have to include some information on object orientation, since object appearance strongly depends on object viewpoint.

The proposed approach for estimating object orientation makes use of this prior work on object appearance representations by using parts of the representation formed by an object detection algorithm as feature input. The 2D-part-based approach by Felzenszwalb et al. [2010][1] is particularly well suited for three reasons. First, the arrangement of object parts relative to the object bounding box is highly dependent on viewpoint, more so than color or texture information, for example. Secondly, the detector output already includes a very rough but explicit representation of viewpoint. Finally, the detector shows good performance on standard test sets and is publicly available. This detector is therefore described in more detail in this section.

A deformable part object detector consists of several detector components, each in itself a partly independent object detector that is specialized on objects of certain viewpoints. The car detector model used in this thesis is visualized in Fig. 6.1. The detector represents objects and parts as patterns of orientation histograms. If applied to an image, it therefore first calculates such histograms at various positions and scales, that are then used by all detector components. Each detector component contains a pattern of orientation histograms that represent one possible appearance of the object (usually from a certain viewpoint range). This detector component pattern (the *root filter*) is compared to image orientations in a sliding window fashion on various scales.

In addition to the root filter, each component contains an additional set of part filters, which are orientation histogram patterns at a finer scale, together with a prior where these patterns should appear relative to the root filter. This prior has an associated cost function that allows and penalizes deviations from the expected position. These patterns are also searched for in a sliding window fashion at various scales.

In a next stage, an overall matching score for each detector component at various image positions and scales is computed by combining scores from responses for the root filter and each part filter, taking into account the position prior of each part filter. A greedy non-maximum suppression is performed on the scores from all detector components, resulting in a set of distinct object candidates. For each candidate, a bounding box is created by combining votes from part and root filters. The score leading to each object candidate is also returned as a confidence measure. The proposed re-weighting of object candidates using detections for other object types was not used in this approach.

To better understand the connection between detector components and viewpoint, their formation during training is informative. An early step in training is a sorting of all training bounding boxes by their aspect ratio. This already introduces an order of viewpoint, since for most objects aspect ratio varies continuously with viewpoint. Given

[1]available online: `http://people.cs.uchicago.edu/~rbg/latent/`

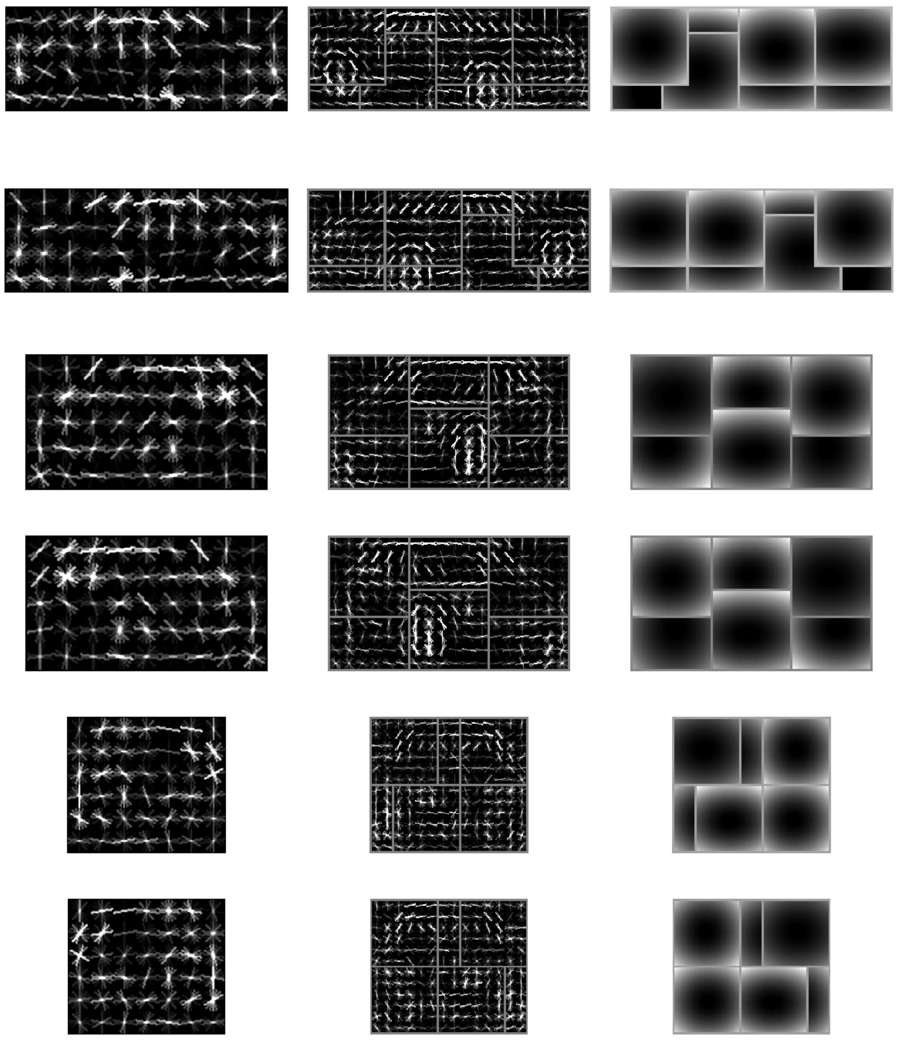


Figure 6.1.: Part-based car model used in this thesis. Model and code for visualization used for this figure is contained in the software package from Felzenszwalb et al. [2010]. Each row visualizes one detector component, with the overall object representation (the root filter) on the left, part filters overlaid in the middle and part position distribution on the right. Note that the aspect ratio changes continuously from top to bottom, except that every second row has the same aspect ratio as the previous.

a predefined number C of components, the set of sorted boxes is split into C equally-sized subsets. Each of these C subsets is then doubled in size by creating mirrored versions of all objects. A heuristic clustering procedure on this doubled subset then creates two equally-sized clusters that maximize within-cluster appearance similarity and between-cluster dissimilarity. This also makes sense in terms of object viewpoint, since an object and its mirrored version have the same aspect ratio but different viewpoints. This mirroring and clustering procedure dissociates objects of one viewpoint from objects of the viewpoint of the mirrored object. The result are $2C$ equally sized subsets of training samples, resulting in $2C$ detector components.

Fig. 6.2 (right) shows the ground truth orientations in our symmetrized training data set that each object detector component responds to maximally. The viewpoint selectivity of components is easy to see, mirror components are also easily identifiable.

To return to the task of viewpoint estimation: the input feature vector for the proposed approach consists of the positions, sizes and aspect ratios of the object parts relative to the middle and size of the object bounding box. The index $c \in [0, \ldots, 2C]$ is also included, specifying which component created the candidate. As image context features, the absolute size, aspect ratio and position of the detection bounding box within the image is added to the feature vector. Since the used object components had 9 parts, this resulted in an overall feature vector of length $51 = 9 \times 5 + 1 + 5$.

6.3. Regression Trees for Orientation Estimation

Two factors were relevant for our choice of a mapping from object appearance and context to orientation. First it must be able to create multi-modal estimates of orientations since the literature has shown that this mapping can be ambiguous. Secondly, the output variable must be circular and continuous. As a result, the choice of regression approaches for this task is quite limited. Also, the nature of the mapping (linear/quadratic/exponential) is not known a priori, so a regression method is needed that does not make strong assumptions, and can handle high-dimensional input features. Random regression forests have been shown in the literature (see sec. 6.1) to be suitable for the task of orientation estimation and offer a great flexibility. This allows an easy adaptation not only to circular output, but to a possibly multi-modal representation of results. Further favorable features are a quick runtime in inference, the availability of various training procedures from the literature (e.g. Gall and Lempitsky [2009]) and the resistance to noise in training input.

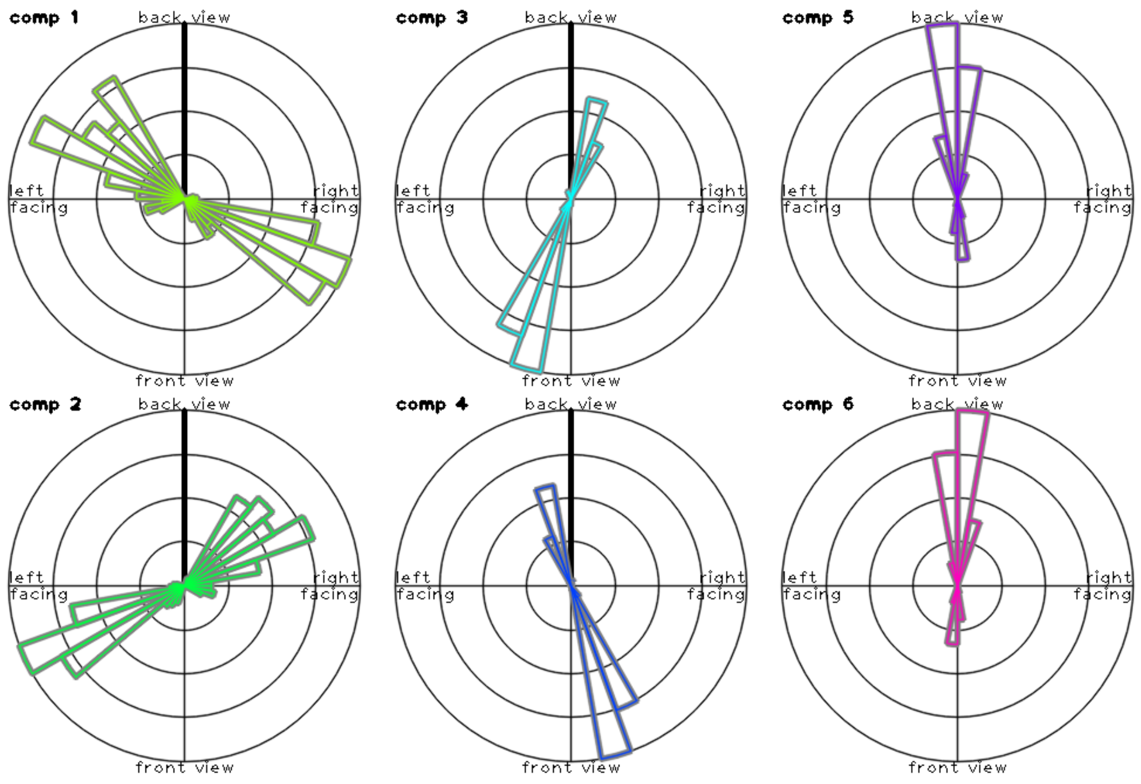


Figure 6.2.: Viewpoint angles that the 6 object detector components for type car respond to. Each plot shows a circular histogram visualizing how often the corresponding detector component has responded to car objects with the given ground truth viewpoint. The the radius towards the top shows the frequency for viewpoints of $0°$ (corresponding to back views of cars), the radius towards the left shows frequency for $90°$ (left views), etc. As can be seen from the similarity of distributions when mirrored at the vertical axis, detector component 1 (top left) was trained on images that were mirrored versions of training images for component 2 (bottom left). The same is true for components 3 and 4 (middle column) and 5 and 6 (right column). One can also already see that the same component that responds to cars oriented with viewpoint φ responds to cars oriented $\varphi + 180°$. Note that histogram bar radius, and not area, corresponds to frequency.

Assuming some general knowledge about random regression forests (see sec. 2.4.2) we describe in this section the adjustment of the general approach to the output of possibly multi-modal distributions of circular orientations and the training procedure used.

6.3.1. Selecting Training Data

The first step in training a regression forest is the selection of a subset of training samples to train each tree. Providing disjoint training sample sets for each tree encourages differences in trees, leading to a better generalization performance. On the other hand, limited numbers of training samples often mean that training sample sets must overlap to ensure sufficient specificity. Given the statistics of the training set a suitable compromise had to be found, resulting in two different ways of creating training subsets for each tree. These are discussed in sec. 6.5.2 after the training set has been described, and evaluated separately.

6.3.2. Defining Nodes

For defining a node n of a random regression tree, a set of binary tests b is created at random, and evaluated on the training data H_n that defines that node. The test that creates the highest information gain when splitting this training data is then used to define that node. Tests must be quick to evaluate and easy to generate. Making use of the high dimensionality of input features, tests for viewpoint estimation are defined as 2-tuples $b = (a, t)$ where a is a feature index, selecting one of the 51 feature channels (size, position, aspect ratio of object or one of its parts or detector component index) and t is a threshold, against which values of the selected feature channel are compared. Denoting by $\boldsymbol{g}_i(a)$ the value of the a-th index of feature vector $\boldsymbol{g}$, the test is defined by

$$b = 0 \Leftrightarrow \boldsymbol{g}_i(a) < t \quad . \tag{6.1}$$

The criterion for selecting (a, t) from a set of randomly created tests is inspired by the entropy measure for linear variables, applied to circular variables. When splitting the training set H_n we chose

$$(\hat{a}, \hat{t}) = \operatorname{argmax}_{a,t} \log \mathrm{C}(\phi_i : \boldsymbol{g_i}(a) < t) + \log \mathrm{C}(\phi_i : \boldsymbol{g_i}(a) \geq t) \tag{6.2}$$

as node test, where C is the circular concentration in 1D defined in sec. 2.3.1.

6.3.3. Defining Leaves

Regression tree leaves determine the output of the regression procedure. They are created once the diversity of features and/or ground truth values is too small to split again, the number of training samples falls below a threshold or a maximum tree depth is reached. The remaining set of ground truth values is often represented by a mean and covariance matrix, that is combined with results from other tree through a weighted sum.

This is not the case in the proposed system, since the distribution of samples forming a leaf is often multi-modal due to the described similarity of objects. Forming a single mean between samples sets of $\approx 0^\circ$ and $\approx 180^\circ$ of different size could lead to any possible angular mean (c.f. sec. 2.3.1). A representation of orientation values as a mixture of two wrapped Gaussians or von Mises distributions would have been a viable option for most cases, but instead we chose to not fix the number of modes at all and simply save all the viewpoint samples that reach the leaf during training. This way, no information is lost through potentially false model assumptions, and combination of results from different trees can be performed by simple list concatenation. These can then be interpreted as samples from an unknown distribution, which can then be approximated using kernel density estimation.

6.4. Data Set

One additional motivation for developing this viewpoint estimation approach was the availability of the new Kitti data set Geiger et al. [2012] which contains a large number of human labels for 3D object position and orientation, as well as ground truth for observer motion. There exist other data sets for viewpoint estimation, (see sec. 6.1), but the Kitti data covers the widest variety of data found in real-world driving scenarios. Since a fusion approach with self-motion was also a goal of this work (see next chapter), and ground truth for self-motion was thus also required in the data set, the Kitti data set was used in this chapter.

The data set has some properties similar to the Karlsruhe data set used for evaluation in the last two chapters. The same authors report to have captured 6 hours of data while driving through the city of Karlsruhe, Germany, recording observer motion with GPS and IMU and stereo imagery at 10 fps, which is available as rectified image pairs. As opposed to the data set used before (Geiger et al. [2011, 2010]), additional depth information is available, that was recorded by a velodyne laser scanner. Also, an additional pair of stereo cameras captured color images. Main feature for our purposes are human labels for object position and yaw orientation in the 3D point clouds. They are specified as object tracks with a constant height, length and width and changing 3D position and yaw orientation. Available are also occlusion and truncation labels. Human annotators labeled every car, van, truck, tram, pedestrian, cyclist and sitting person close enough to be captured by the laser scanner in every few frames with precomputed interpolation in-between. Given projection matrices allow transferring this ground truth information into the image or into another coordinate system centered at the IMU.

An appealing property of the Kitti data set is the wide variety of captured traffic scenarios, ranging from quiet cross-country roads and suburban residential areas to urban street crossings with many independently moving objects and slow drives through the busy pedestrian zone of Karlsruhe. While the weather conditions were generally favorable and drives were all taken during day-time, reflections and the limitations of low dynamic range cameras provide realistic challenges with respect to image quality.

For evaluation we used all car annotations from the first day of capturing (26/09/2011) that are available in the raw section of the database. The car class was chosen because it is the predominant object class in the data set. The drives were split into disjoint training, validation and test sets by considering their index when ordered by capturing time modulo 4: all drives with index 0 or 1 modulo 4 were used for training, those with 2 modulo 4 are for validation and the remaining for testing. This apparently random ordering resulted, as was seen later during evaluation, in a concentration of tracks with considerable change of object height in the validation set.

Of all the object classes contained in the data set, cars are by far the one with most exemplars. This class is therefore used to demonstrate the system in this chapter. The described approach, however, is in general also applicable to other object types like pedestrians.

To enlarge the training set, every image and corresponding ground truth label was mirrored. The object detector was then run on every image and all detections, irrespective of detection score, were compared to annotated object bounding boxes. For each of the 33914 ground truth labels, the detection box with the biggest overlap was associated with the object if the corresponding area of intersection was more than 80% of the area of union of ground truth an detection bounding boxes. The required overlap was only achieved for 7953 objects, which is partly due to the fact that many annotated objects are relatively small and therefore below the minimum size used by the detector. Requiring a smaller overlap resulted in a much higher ratio of detections from a wrong detector component, and thus more noise in the training data. Of the remaining objects 3002

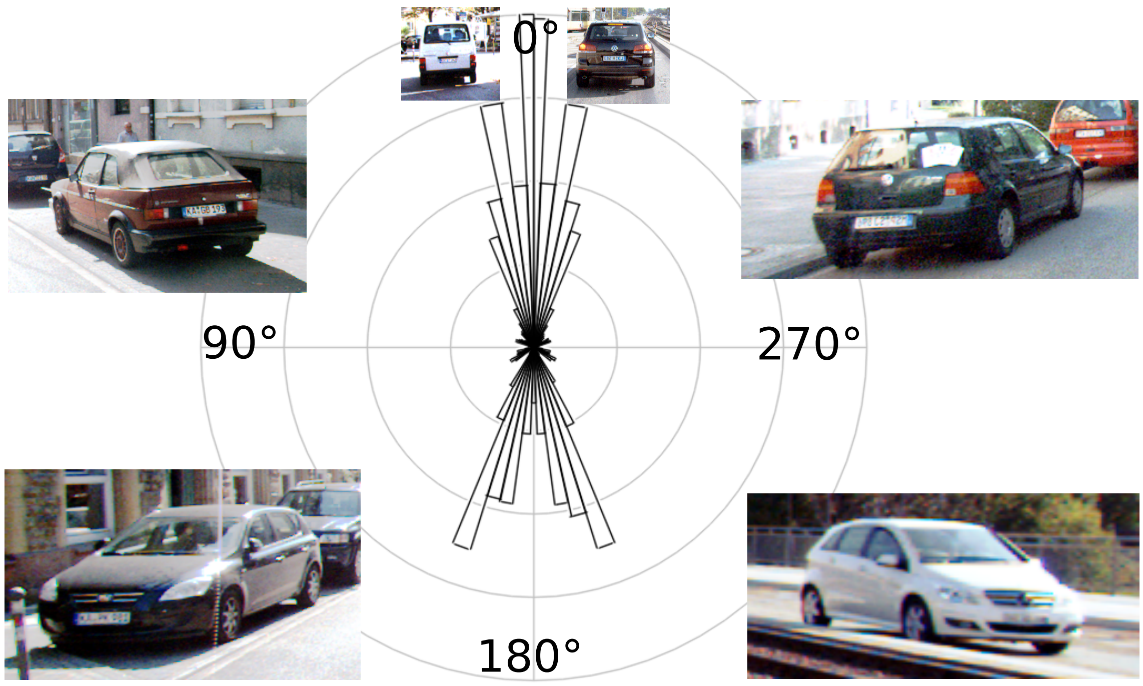


Figure 6.3.: Statistics over training data and and a few examples of objects from the dominant viewpoint ranges.

were labeled as partially or fully occluded or truncated and were therefore dismissed as well, resulting in a final training set of 4951 samples. The same procedure was repeated on the validation and test sets, resulting in 3351 car samples in the test set.

To obtain a bigger and noisier data set for comparison, a similar procedure was repeated on validation and test set. For the 'high noise' test set, an overlap of only 60% between ground truth and detector bounding box was required. Partially occluded and partially truncated objects were also allowed.

Since the images have a very wide field of view, object viewpoint has to be calculated from a combination of object orientation and position. For example, a car in the right periphery that is parking parallel to the street has a orientation label of $\approx 0°$, corresponding to a yaw orientation parallel to the observer (see Fig. 6.3 top left). Since it is in the right periphery, however, the line of sight between the observer and the object, meets the object not in its rear but in its left side. This makes the car appear like a car would that was in front of the observer and facing slightly left. In fact, all example images in the top row of Fig. 6.3 have world orientation of $\approx 0°$, the two cars in the bottom both have an orientation of $\approx 180°$, but all appear under very different viewpoints due to their different position in the image.

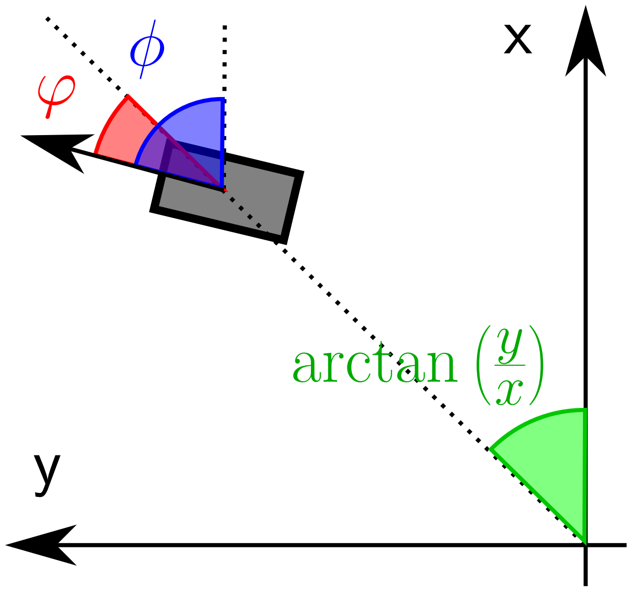


Figure 6.4.: Relation between object world orientation ϕ relative to the observer's optical axis, and the viewpoint φ relative to the line of sight from the observer to the object.

For training viewpoint regression forests, the object appearance has to match the apparent viewpoint, rather than the world yaw orientation label. Input to the regression must therefore be compensated for this effect by calculating viewpoints φ (orientation relative to the line of sight from observer to object center) from yaw orientation labels ϕ (relative to the observer's optical axis). As visualized in Fig. 6.4, this can be accomplished using the formula

$$\varphi = \phi - \arctan\left(\frac{y}{x}\right) \tag{6.3}$$

where x and y are the positions of the object on the 3D ground plane. x is the direction of the optical axis of the observer and y points left.

The resulting training data set of car objects has a very heterogeneous distribution of ground truth viewpoints, which is plotted in Fig. 6.3. This leads to complications in creating training data sets for regression tree training, as described in sec. 6.5.2.

6.5. Static Evaluation

Before continuing in the next section with temporal integration of viewpoint estimates, estimation performance of the described regression approach on single images is described in this section. After discussing parameter choices for training, the resulting forests are analyzed and evaluated on a set of test images.

6.5.1. Feature Space Analysis

Two different methods were used to analyze the features space. The first is multi-dimensional scaling (MDS), a technique that analyzes the local similarity structure of features. To apply it, the Euclidean distance of each feature vector to each other feature vector is calculated and assembled in a large distance matrix. From these distances alone, the MDS algorithm[2] creates a low-dimensional space of feature points, such that features that are close together in the original features space, are also close in MDS space, and vice versa. The first 3 dimensions of this MDS space capture more than 99% of the variance in the distance matrices and therefore approximate the similarity structure of the feature space well. Visualizations of these dimensions are shown in Fig. 6.5. Each dot represents one training sample, colored by its ground truth value. Dots of similar color are usually close together, showing that similar orientation leads to similar feature space representations; orientation information therefore must be contained in the feature space. However, clusters of different orientations are not perfectly separated. There are overlaps of clusters of different colors, hinting at ambiguities in the mapping from features to orientations as expected.

Another method to analyze the feature space for the task of self-motion estimation is a statistic over features channels used in all nodes in the trained random regression forest. This shows how relevant each feature dimension is for the regression forest, hinting at the information content of that feature dimension.

Results of this analysis are shown in Fig. 6.2. Context features and relative image position of parts play the most important role for regression forests. Size and aspect ratios of most parts are rarely used. Interestingly, the index of which detector component detected the object is even less important, although component index should give a first rough categorization into viewpoint ranges. Instead, the regression forest uses for the first rough categorization of features the horizontal image positions of parts 2 and 5, and

[2]Matlab implementation `CMDSCALE` of algorithm from Seber [1984]

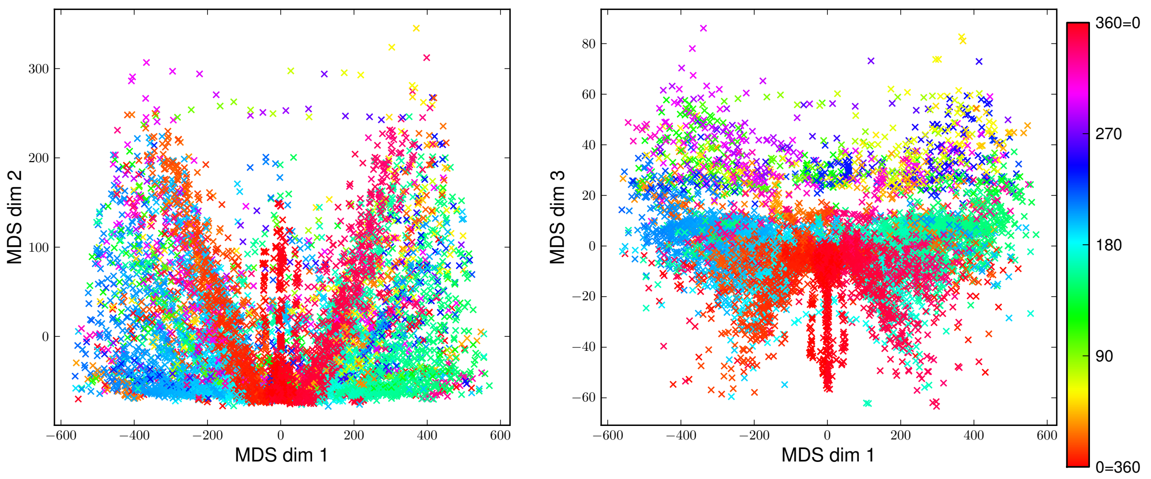


Figure 6.5.: Visualization of similarity structure of the input feature space using multi-dimensional scaling (MDS). Displayed are scatter plots for the three most prominent dimensions found. Each dot corresponds to a training sample, color represents ground truth viewpoint. The local smoothness of the feature space is reflected by the proximity of dots with same colors, confusions are indicated by clusters of different colors intersecting each other.

the vertical image position of part 7 and the whole object box, all of which only play medium roles afterwards.

6.5.2. Parameter Selection

Given the flexibility of regression forests, the number of parameters to specify for their training is relatively high. As already hinted at above, the creation of training subsets for each tree is non-trivial due to the heterogeneity of ground truth viewpoint labels in the training data set (see Fig. 6.3). This could cause biases in regression output, for the following reason: Imagine, for example, a tree whose training set consists of 50% back views (i.e. $\varphi \approx 0°$), but only 5% front views ($\varphi \approx 180°$). Since back views and front views are easily confused, there will be hardly any leaves containing more front than back views, resulting in a very low probability of ever getting a front view as result.

To avoid such cases, training sets were balanced in ground truth orientations in two different ways. First, all training samples were classified into one of 12 equally-sized orientation bins. The smallest number s of samples per bin determined the number of training samples for each tree as follows: for each tree, a training set was assembled by choosing s samples at random from each bin. This results in nearly disjoint sample sets from bins with many samples and equal sample sets for each tree from the smallest bin.

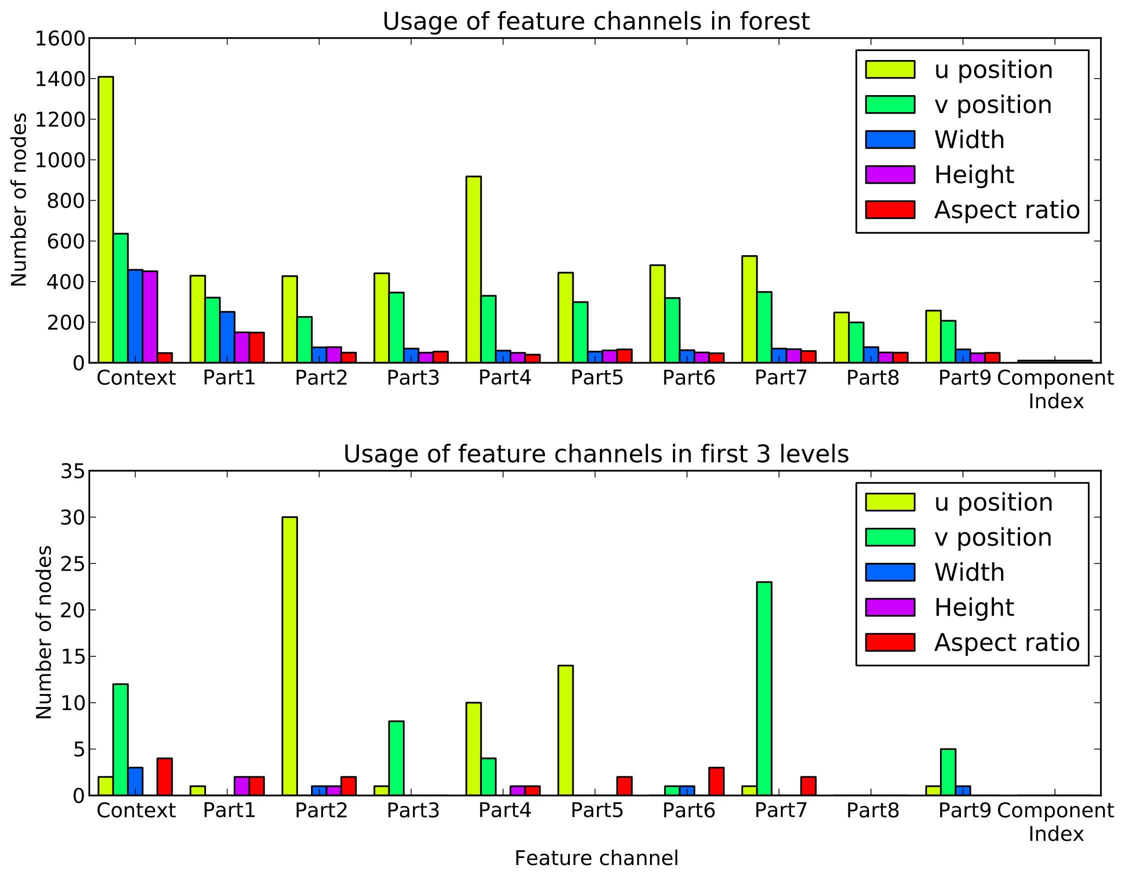
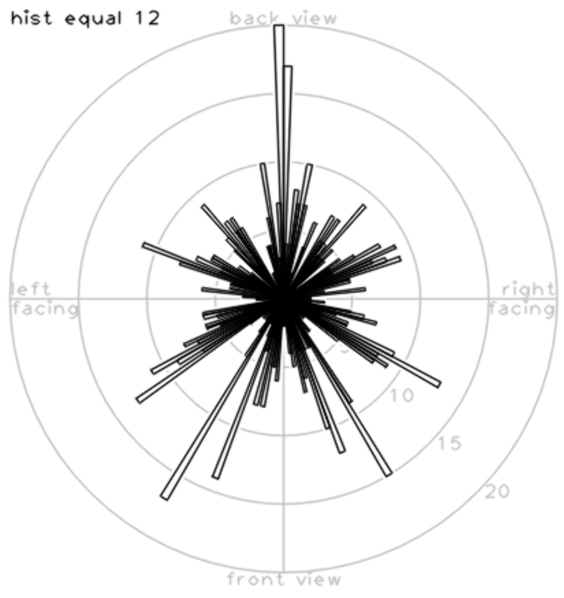


Figure 6.6.: Usage of feature channels by the complete viewpoint regression forests (**top**) and the first 3 levels (**bottom**). Each bar represents one features channel, its height is the number of nodes in the trained regression forest that uses that feature channel in its associated test.



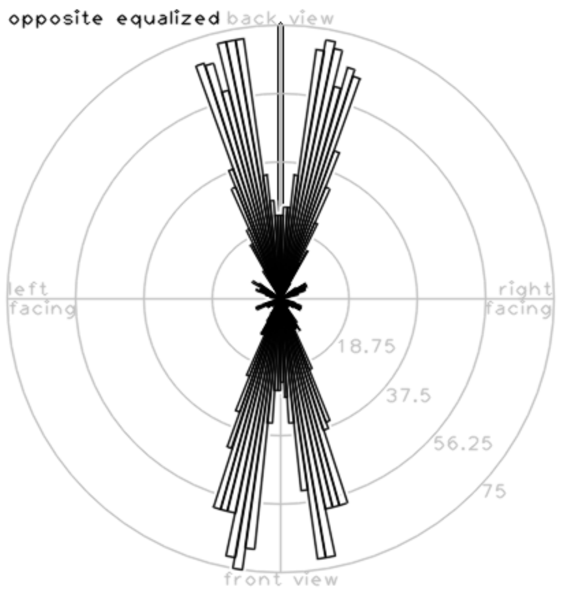


Figure 6.7.: Distribution of all ground truth viewpoint labels when using the two proposed methods to counter training data heterogeneity, shown as histograms with 180 bins. Histogram equalization (**left**) ensures the same number of training samples in each of 12 equally-sized bins. Equalizing opposite bins of a 36-bin histogram results in very different distribution of ground truth labels (**right**).

A similar procedure was also tested, using 180 bins, ensuring same contributions from bins of opposite orientation only, not from all bins at the same time. Since the data set consists of detections in images and their mirrored counterparts, this leads to a 4-way symmetric data set: frequency at angle $\pm\varphi$ is the same as at $180° \pm \varphi$. The distribution of viewpoint labels available for training, resulting from both splitting methods, are shown in Fig. 6.7. In addition, usage of the complete training data set for each tree was tested, as well as choosing a fixed ratio of training data at random for each tree.

Other parameters to chose for training are the number T of trees, the number of random tests to generate for selection in each node, as well as the criteria to form a leaf node: the maximum depth in a tree to allow, the minimum number of samples to split, as well as the minimum variance allowed for splitting. The choice of training data split for each tree and the number of random tests to generate are the two variables that ensure diversity between trees: a high overlap between training sets and a number of randomly generated tests result in node splits that are close to optimal, which for similar training data statistics will lead to similar trees. A low overlap in tree training data and a low number of random tests to chose from, on the other hand, leads to more randomness in node tests, and thus to more diverse trees. Using a higher noise level in training data, and thus more training samples, was also tested.

Exhaustive testing of each combination of training parameters was performed, which meant, training a regression forest and evaluating it on validation data for each parameter combination. The results showed no clear preference for any specific parameter, except that a limitation of the maximum tree depth to 10 levels leads to bad results, probably restricting the specificity of trees to much. Further details of the analysis are therefore not reported here.

In the end, two forests were used for evaluation, that differ mainly in the data split method. One uses the 12-bin histogram equalization (c.f. Fig. 6.7 left) and a concentration threshold of 200 for creating a leaf, the other employed the 180-bin opposite equalization (c.f. Fig. 6.7 right) and no constraint on the concentration during training, so leaves were only created when the number of training samples for a node fell below 10. The number of trees to train was fixed $T = 20$, 2000 random tests are created to chose from for each node. The maximum depth of a tree was not limited.

6.5.3. Viewpoint Estimation Results

Evaluated on the test set, the regression forest returned on average 150.36 (± 34.42 std) samples for each input feature vector. Using kernel density estimation, these are interpreted as samples from a distribution with density function p, whose local maxima $\varphi_{\mathsf{locMax}}$ are used for evaluation. Kernel density estimation and maximization for circular variables is described in sec. 2.5 in the methods chapter, a bandwidth of $b^2 = 0.3$ radians was used. The value $p(\varphi_{\mathsf{locMax}})$ of the kernel density at each local maximum is compared to the density $p(\varphi_{\mathsf{max}})$ at the global density maximum, and only those local maxima with

$$p(\varphi_{\mathsf{locMax}}) > 0.1\, p(\varphi_{\mathsf{max}}) \tag{6.4}$$

are retained. This ensures that only prominent local maxima are used for evaluation. The average number of local maxima (or modes) found this way is 1.9 (± 0.27 std), showing that at least some of the ambiguity in features space can still be found in regression results.

Comparing such a multi-modal density to a single ground truth value is possible in several different ways. Four measures were used: The first is the angular difference between the global density maximum φ_{max} and the ground truth orientation. This measure ignores potential other modes with slightly lower density value, making a hard decision for the strongest mode. To honor the possible presence of significant local maxima close to the

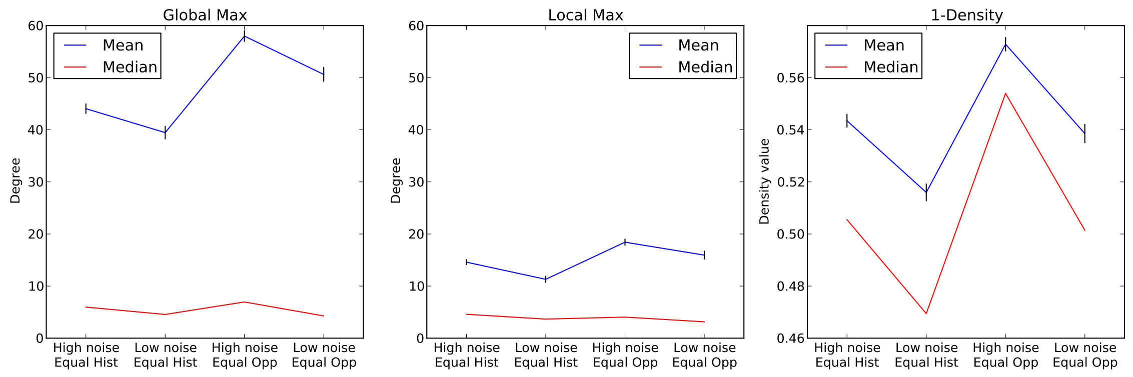


Figure 6.8.: Mean (blue) and median (red) performance for three different evaluation measures (one plot each), on low-noise and high-noise testing data and for two different ways to split training data. Black lines denote the standard error of the mean (SEM).

ground truth, the third proposed measure is the angular distance from the best local maximum $\varphi_{\mathsf{locMax}}$ to the ground truth viewpoint. Finally, an evaluation of the density itself at the position of the ground truth viewpoint $p(\varphi_{\mathsf{GT}})$ measures a combination of both, the presence of a mode near the ground truth and its strength. Since for both other measures high values are worse than low values, $1 - p(\varphi_{\mathsf{GT}})$ is reported in the following.

Mean and median values for all these measures are summarized in Fig. 6.8. Results include those for the low-noise condition that was used during training (80% overlap between detection and ground truth, full visibility), and the high-noise condition (60% overlap, partial occlusion and truncation allowed).

In Fig. 6.9, results for the low-noise condition are split with respect to ground truth viewpoint, giving an idea which orientations are more problematic to the system than others. Comparing these plots with the distribution of training data in Fig. 6.3 shows that errors are rather small for viewpoint angles where training data is available, except for the back views ($\approx 0°$) when using equal contributions from opposite histogram bins (two left columns).

6.6. Temporal Integration

As seen in the previous section (sec. 6.5.3), results from viewpoint regression are often multi-modal, leading to ambiguous estimates. The correct mode is often the dominant, but results suggest that sometimes a wrong mode has more support. Temporal context could disambiguate estimates and correct individual cases of wrong estimates.

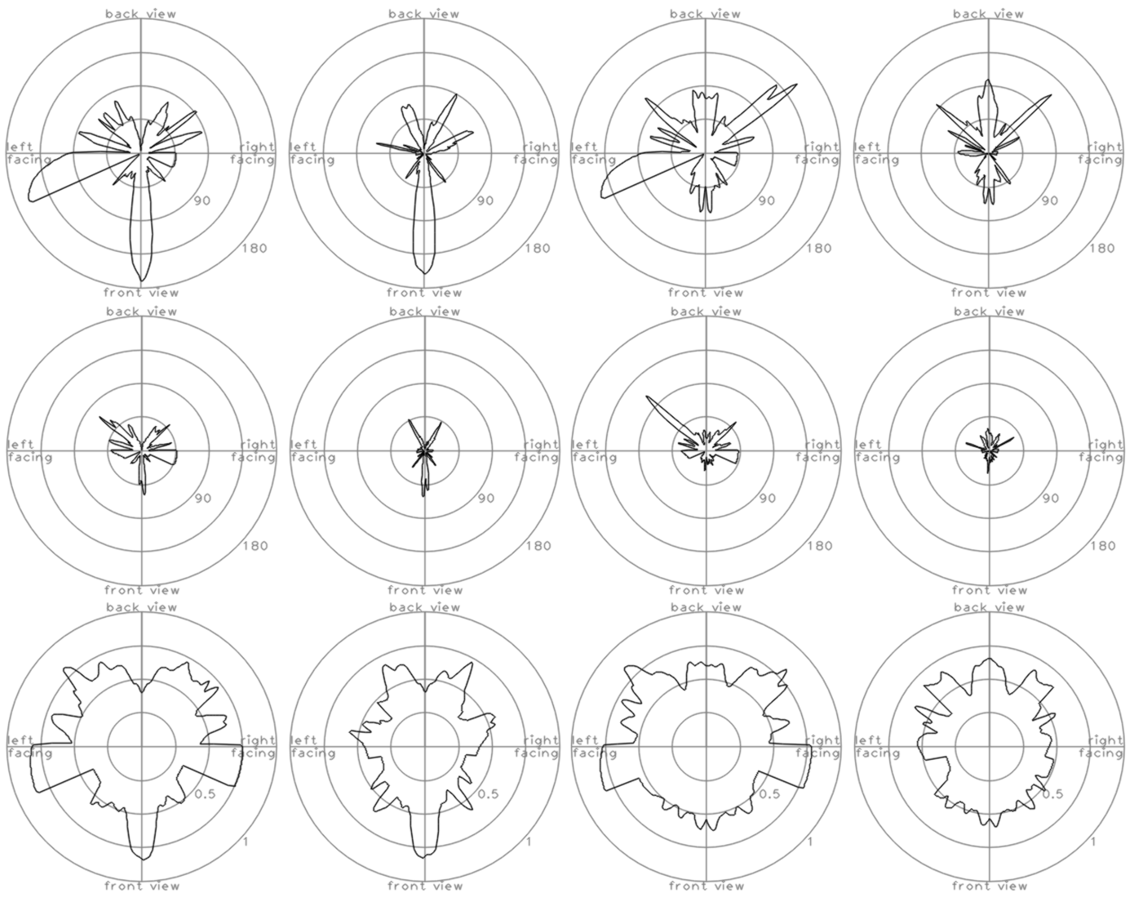


Figure 6.9.: Viewpoint estimation performance per ground truth viewpoint. **Top row** shows angular distance from ground truth viewpoint φ_{GT} to global density maximum φ_{max}, and to the closest local maximum $\varphi_{\mathsf{locMax}}$ (**middle row**). **Bottom row** visualizes $1 - p(\varphi_{\mathsf{GT}})$, i.e., the density of the estimates at the ground truth position. Each column shows results for different training data split and test noise level: histogram equalization with high and low noise (**left two columns**) and equalization of contributions to the training set of opposite bins (**right two columns**). Each plot shows for all ground truth viewpoint angles φ_{GT} the slightly smoothed (using kernel density estimation with bandwidth 0.001) mean error as radius. Trends in each column are similar, and differ mainly in scaling. The error at 180° (frontal views) is much more prominent when using histogram equalization than when equalizing opposite histogram bins, probably due to the small bin width. Low and high noise conditions (**columns 1&3 versus 2&4**) differ in peaks at around $\pm 50^\circ$ (rear left/right views) and 110° (front left view).

We test this hypothesis in this and the next section by implementing two particle filters (described in general in sec. 2.7) and adapting them to use with a circular filtering density. Particle filtering was chosen over other filtering approaches since it is capable in principle to maintain a multi-modal filtering density and makes no further assumptions that are incompatible with circular state or measurement variables. The first variant used for evaluation is the bootstrap filter - a particle filter variant that uses the prior distribution as proposal distribution and a resampling step after every measurement update - is used since it is easy to implement and fast in inference. The second variant employs a more complex proposal distribution in order to draw particles from the measurement distribution. This simplifies maintaining several modes at the expense of a higher computational cost. This approach is derived in sec. 2.7.3.

Since the filtering density is represented as independent particles, adapting the filter's update step to a circular variable consists mainly of a modulo operation after every update to keep particle positions in $[0, 2\pi)$. Formulation of the measurement update for sets of samples from regression forests, instead of the classical single measurement with Gaussian noise, is more demanding.

6.6.1. Choice of Filter

Arguably the simplest particle filter is the bootstrap filter. Results obtained using this filter are compared in the following to a more involved filter model. This second model motivated by the goal to support the maintenance of several modes in the filtering distribution, since measurements for viewpoint are — as shown in sec. 6.5.3 — usually not unimodal. If the filter 'decided' to track the wrong mode at any time, the tracking process would fail consistently. We therefore follow the approach outlined in sec. 2.7.3 and use importance sampling to replace in every step a few particles with particles drawn from the measurement distribution to encourage a number of particles in secondary modes. The proposal distribution used is

$$q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}, \boldsymbol{y_t}) = \alpha p(\boldsymbol{x}_t|\boldsymbol{y}_t) + (1-\alpha)p(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}) \tag{6.5}$$

where the parameter $\alpha \in [0, 1]$ determines the ratio of particles drawn from the measurement (c.f. 2.59). For $\alpha = 0$ this is equivalent to the bootstrap filter.

In practice this means that all particles are updated as described above with additive prediction noise, but a few are replaced by new particles created from measurements. To this end we sample orientation density estimates φ obtained from regression tree outputs and replace arbitrary particles with them. In order to maintain a correct representation of the posterior density, the importance weights

$$\tilde{w}_t^{(n)} \propto \frac{p(\boldsymbol{x}_t^{(n)}|\boldsymbol{x}_{t-1})}{q(\boldsymbol{x}_t^{(n)}|\boldsymbol{x}_{t-1}, \boldsymbol{y}_t)} \tag{6.6}$$

are calculated up to proportionality.

6.6.2. Measurement Update

Given a set of estimates $\boldsymbol{y}_t = (\varphi_{t,1}, \dots, \varphi_{t,M})$ from the regression forest, a suitable expression for the likelihood $p(\boldsymbol{y}_t|x_t^{(n)})$ must be formulated. Dealing with multiple measurements for a single true object is a frequent issue when tracking a target in clutter. The solution can also be applied here for the estimates returned from the regression forest. However, one assumption for this procedure is that measurements are independent. This is not the case for the relatively large number (≈ 150 in our experiments) of samples, that usually cluster around a few modes. We therefore first obtain a few independent modes from the set of samples using kernel density maximization (c.f. sec. 2.5.2). We then estimate a weight for each mode as the number of samples assigned to that mode divided by the overall number of samples. We also calculate the variance of the samples that were assigned to each mode. The resulting mode positions $\hat{\varphi}_{t,1}, \dots \hat{\varphi}_{t,L}$, weights $\hat{w}_{t,1}, \dots, \hat{w}_{t,L}$ and variances $\hat{\sigma}^2_{t,1}, \dots, \hat{\sigma}^2_{t,L}$ represent the original sample set similarly to a mixture of Gaussians.

Having obtained the modes which are now assumed to be independent measurements, we formulate the mutually exclusive events:

$$\begin{aligned} E_{t,l} &= \{\text{'mode l contains the measurement caused by the true state at time t'}\} \\ &\quad \forall l = 1, \dots, L \\ E_{t,0} &= \{\text{'all modes are caused by clutter, none contains the true state at time t'}\} \end{aligned}$$

If we denote by $\alpha \in [0, 1]$ the probability that all modes are caused by clutter $\alpha = p(E_{t,0})$, then $p(E_{t,l}) = \hat{w}_{t,l}(1-\alpha)$ approximates the probability that $E_{t,l}$ is true. Given that $E_{t,l}$ is true, the true state is contained in the mode centered at $\hat{\varphi}_{t,l}$, then the probability of $\hat{\varphi}_{t,l}$ given the particle $\varphi_t^{(n)}$ is modeled as a wrapped Gaussian:

$$p(\hat{\varphi}_{t,l}|\varphi_t^{(n)}, E_{t,l}) = \frac{1}{\sqrt{2\pi}\hat{\sigma}_l} \sum_{k \in \mathcal{Z}} \exp\left(\frac{\left(\hat{\varphi}_{t,l} - \varphi_t^{(n)} - 2\pi k\right)^2}{-2\hat{\sigma}^2_{t,l}}\right) \quad . \tag{6.7}$$

Summing over all events results in

$$p(\boldsymbol{y}_t|\varphi_t^{(n)}) = p(\{\text{“number of measured modes is L”}\}) \times \sum_{l=0}^{L} p(\hat{\varphi}_{t,l}|\varphi_t^{(n)}, E_{t,l})p(E_{t,l}) \tag{6.8}$$

$$\propto \alpha p(\hat{\varphi}_1, \dots, \hat{\varphi}_L|E_{0,l}) + \frac{1-\alpha}{\sqrt{2\pi}} \sum_{l=1}^{L} \frac{\hat{w}_{t,l}}{\hat{\sigma}_{t,l}} \sum_{k\in\mathcal{Z}} \exp\left(\frac{\left(\hat{\varphi}_{t,l} - \varphi_t^{(n)}\right)^2}{-2\hat{\sigma}_{t,l}^2}\right) \times p(\hat{\varphi}_{-l}|E_{t,l}) \quad , \tag{6.9}$$

where $\hat{\varphi}_{t,-l} = \hat{\varphi}_{t,1}, \dots, \hat{\varphi}_{t,l-1}, \hat{\varphi}_{t,l+1}, \dots, \hat{\varphi}_{t,L}$, i.e. the set of all modes except the l-th.

Assuming that modes caused by clutter stem from a uniform distribution on $[0, 2\pi]$ results in $p(\hat{\varphi}_1, \dots, \hat{\varphi}_L|E_{0,l}) = (2\pi)^{-L}$ and $p(\hat{\varphi}_{-l}|E_{t,l}) = (2\pi)^{-(L-1)}$, and the implemented formula

$$p(\boldsymbol{y}_t|\varphi_t^{(n)}) \propto \frac{\alpha}{2\pi} + \frac{1-\alpha}{\sqrt{2\pi}} \sum_{l=1}^{L} \frac{\hat{w}_{t,l}}{\hat{\sigma}_{t,l}} \sum_{k\in\mathcal{Z}} \exp\left(\frac{\left(\hat{\varphi}_{t,l} - \varphi_t^{(n)}\right)^2}{-2\hat{\sigma}_{t,l}^2}\right) \quad . \tag{6.10}$$

6.7. Dynamic Evaluation

The filtering approach for viewpoint estimates is evaluated using an artificial example composed of all detected cars in the test set, ordered by their ground truth orientation label. This results in a car that changes size and color in every frame and rotates slowly in anti-clockwise direction.

Two particle filter variants described in sec. 2.7 are tested for their ability to disambiguate multi-modal viewpoint estimates created by random forest regression. To simplify filter initialization, particles were created around the correct mode in the beginning ($0°$) with a large standard deviation of $90°$.

6.7.1. Parameter Optimization

Fixing the number of particles to $N = 1000$ and using the clutter model for measurement updates, there remain 3 parameters to chose: the ratio $\alpha \in [0, 1]$ of clutter, the system uncertainty $\epsilon \in \mathcal{R}$ and the bandwidth $b \in \mathcal{R}$ used for kernel density maximization. The system uncertainty ϵ and bandwidth b were set to the error in prediction and measurement on the training data tracks, but the resulting parameters proved inadequate when testing them on the validation set. A search for optimal factors for these two parameters, as well as a clutter ratio α was therefore conducted on the validation set, choosing samples according to the noisier parameters (overlap 60%, partial occlusion and truncation allowed). The mean error results are shown in Fig. 6.10 for both filters

For the bootstrap filter (Fig. 6.10 left), several regions of low errors can be identified, but performance can also decrease drastically for slight changes of parameters. This was also seen when applying filters with the few best parameter sets on the test set. A somewhat larger gap at $\approx 100°$ in the test set caused the bootstrap filter to track the wrong modes (e.g. $\approx 0°$ instead of $\approx 180°$ between frames 2000 and 4500). It could only bridge that gap for rather large values for the prediction noise, results for $\alpha = 0.003, \epsilon = 3.6614°, b = 0.0037°$ are shown in Fig. 6.11.

The filter that draws samples from the measurement distribution, on the other hand (Fig. 6.10 right), showed a much smoother and predictable behavior since it can track the usually two predominant modes and therefore is not affected as much by faster changes in orientation. In particular, the system uncertainty can be set to a much lower value than for the bootstrap filter, since measurements can more easily correct prediction errors. Parameters used in testing were $\alpha = 0.1, \epsilon = 0.12°, b = 14.20°$.

The ratio of particles drawn from measurements for the more involved filter variant was set to 0.1.

6.7.2. Results

The resulting modes of the filtering density after every time step are plotted as dark blue dots in Fig. 6.12 and Fig. 6.12 together with light blue dots representing modes of the corresponding regression results, which were used as measurement input. A reduction in signal to noise ratio is obvious already from both figures, both filters re also able to correctly track the distribution in the area around $180°$, which often gives rise to ambiguities with $0°$.

Main difference is the filtering distribution, which is shown in light gray in both plots. For the bootstrap filter (Fig. 6.12) it is unimodal, which causes its dependency on correct parameter choices and correct behavior of the observed object. The somewhat sudden jump of the viewpoint in this test from below $90°$ to over $135°$ was challenging to overcome, leading to failure of filters with different parameters. If, however, parameters are correctly adjusted to viewpoint change behavior , the bootstrap filter exhibits very good performance in terms of accuracy and computations time. The plotted time course resulted in a mean angular error of $5.0°$, and a median angular error of only $2.88°$.

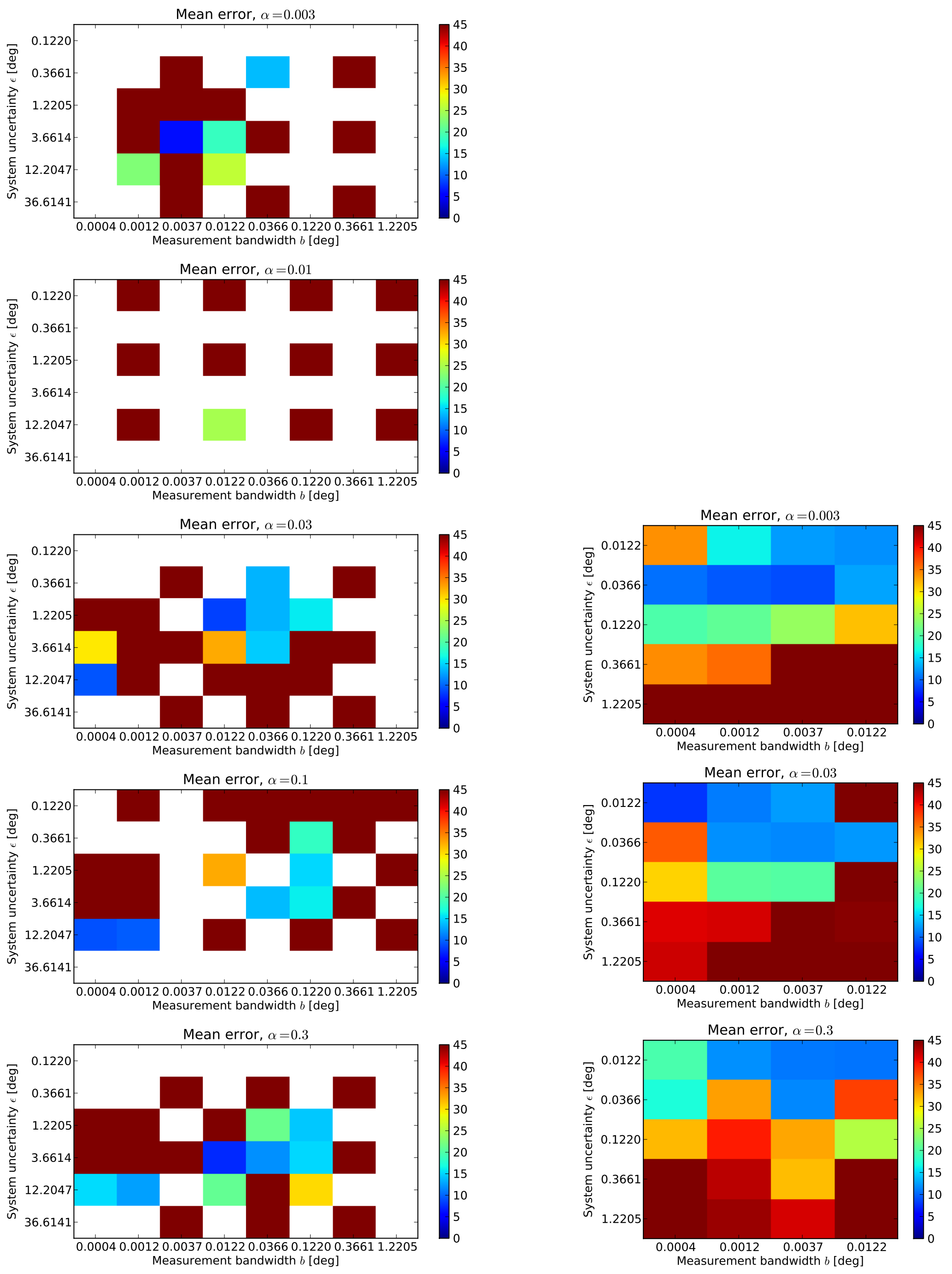


Figure 6.10.: Mean error for tracking on the validation set for various parameter combinations for clutter ratio α, system uncertainty ϵ and bandwidth b for measurement update. Each row shows results for a different clutter ratio, the color of each box in the images encodes the error from $0°$ (blue) to $\geq 45°$ (red) for a single combination of system uncertainty and measurement bandwidth. White color means that the given parameter combination has not been tested. The left column shows results for a simple bootstrap filter, the right column results for the filter with particles created from measurement distributions.

For the filter drawing samples from the measurement distribution (Fig. 6.12), two modes are maintained throughout most of the test, resulting in a much more robust behavior. This robustness, however, comes at the expense of slightly worse performance. The mean angular error for the plotted filtering results is 11.2°, the median angular error 2.15°

6.8. Conclusion

A new method for object viewpoint estimation is presented in this chapter. It takes into account that objects often have similar appearance for very different viewpoints by creating multi-modal viewpoint estimates. This ambiguity has been reported in the literature and is also found in a feature space analysis. Input features for the approach stem from representations formed by a part-based multi-view object detector, but in contrast to detection research, viewpoint is represented as a continuous circular variable. Random regression forests are adapted to multi-modal circular output which can be further processed using kernel density estimation on a circular space. The resulting multi-modal estimates are disambiguated using temporal context. Two particle filter methods for this task were described and tested on a sequence that contained all test samples. To incorporate multi-modal measurements, an approach from target tracking in clutter was adapted.

The random regression forests are trained on car detection output on a part of the Kitti data set, which contains ground truth information for observer motion and semantic object labels. Evaluation on another part of the data set shows competitive performance of this approach compared to results reported in the literature.

Dynamic filtering results showed that ambiguities in regression output should be maintained in filtering in order to track the correct mode when the viewpoint changes abruptly.

One limitation of the work presented in this chapter is the detector performance. More training samples and a better overlap of detected objects with ground truth annotations might be obtained by re-training the detector or using more detector components. The study López-Sastre et al. [2011] suggests to use ground truth viewpoint labels already in detector training to improve its applicability to viewpoint estimation.

Using more training samples, one could create more balanced training sets including frontal views in a similar quantity than back views. The imbalance between these two viewpoints is the most probable reason for the worse estimation results for frontal views of cars obtained with the original car detector model used here. These training samples

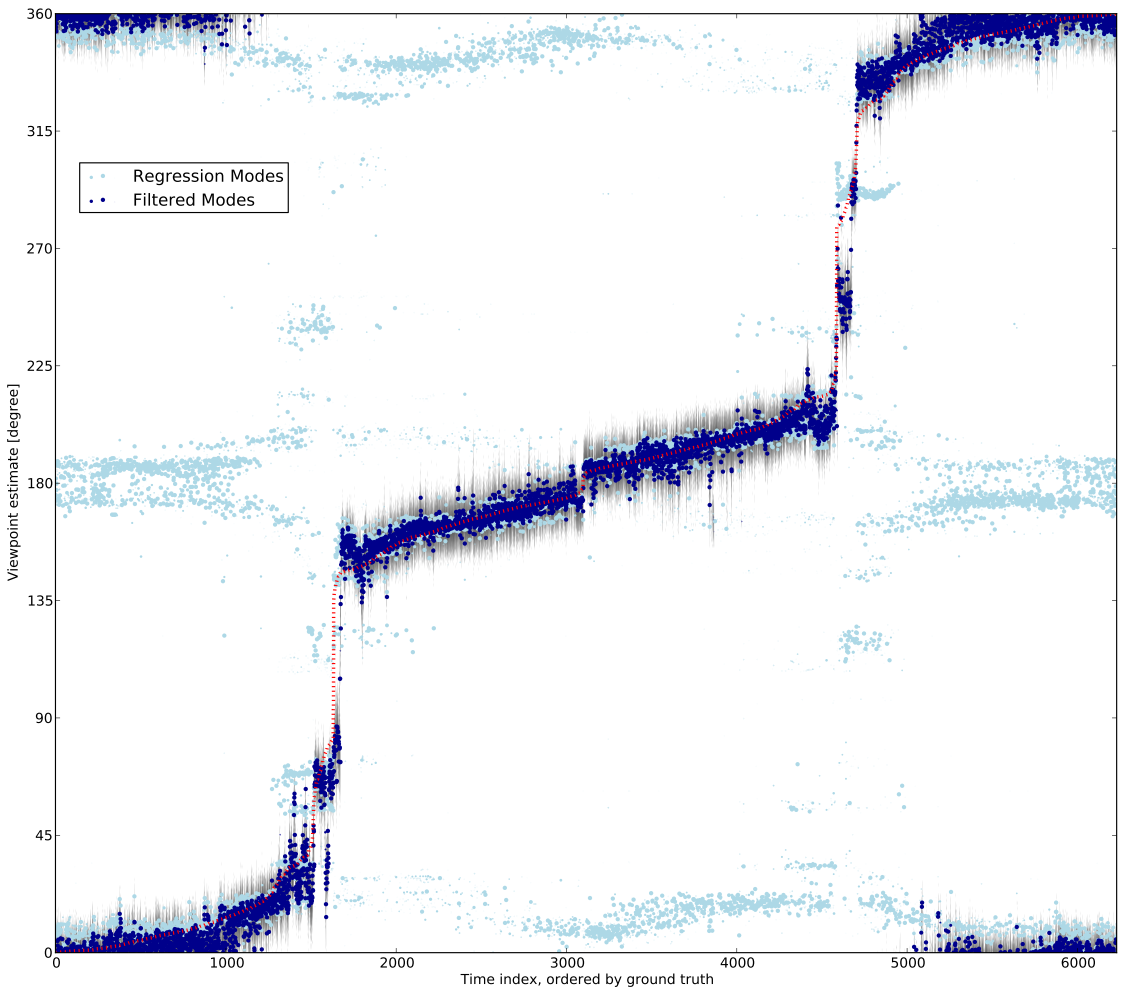


Figure 6.11.: Results of viewpoint filtering for a slowly rotating car with varying identity. The bootstrap filter applied to low-noise data was used to create this plot. Modes of estimates from individual frames, which are used as measurement input for the filter, are shown as light blue dots, dark blue dots show modes of the filtering distribution. The filtering distribution is shown in light gray in the background, dot size denotes mode weight, the ground truth is overlaid as red line. Time and ground truth orientations evolve on the horizontal axis, the vertical position of dots shows viewpoint angle φ.

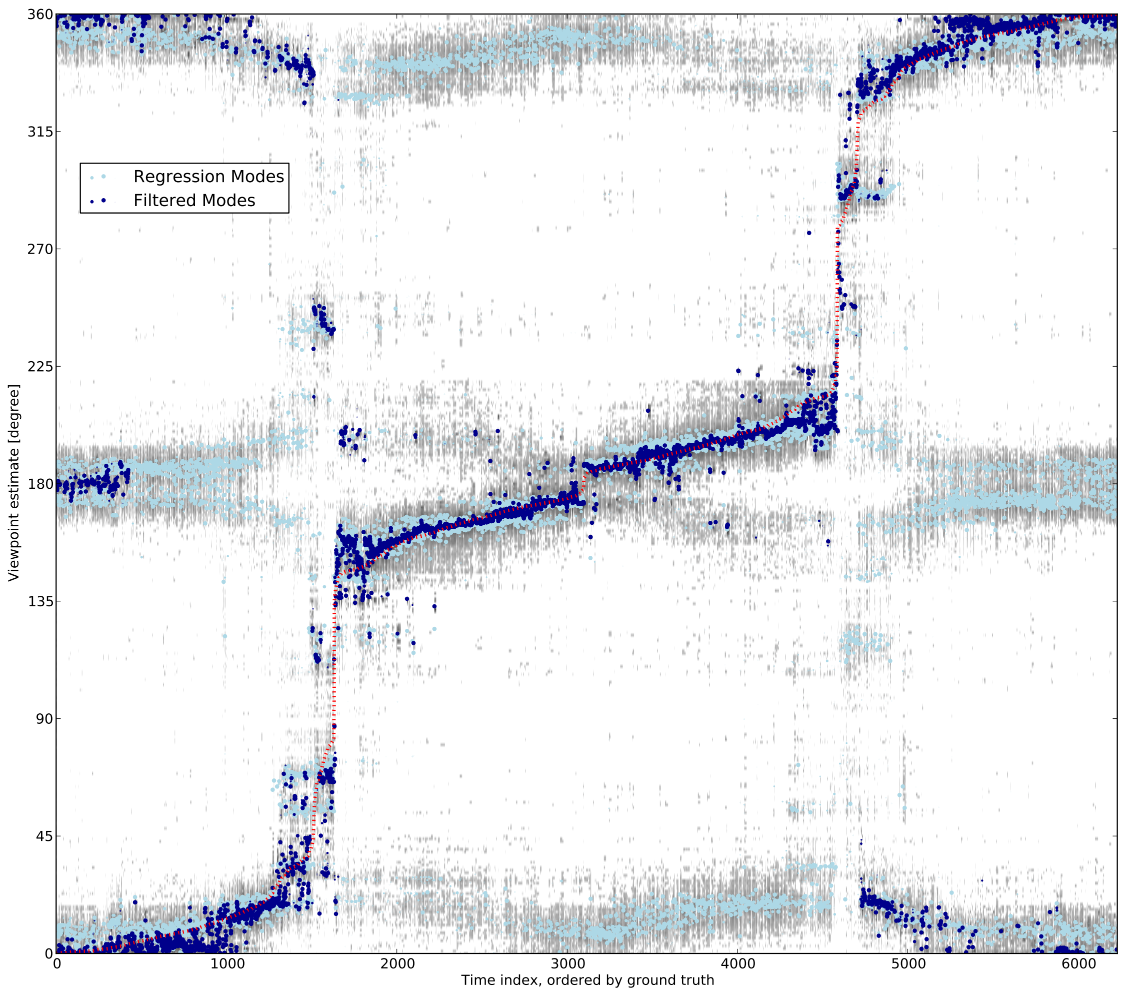


Figure 6.12.: Results of viewpoint filtering for a slowly rotation car with varying identity. This plot is the analog of Fig. 6.11, but evaluated using a measurement-dependent proposal distribution.

could be created by augmenting the existing image sequences with computer graphics using ground truth information, possibly also from the 3-dimensional point clouds.

An extension to articulated objects like pedestrians could include an enhanced mapping to viewpoint and gait phase, which would result in a two-dimensional circular target space.

Besides the application described in the next chapter, viewpoint estimates could also be used to determine observer motion relative to observed static objects by tracking the change in viewpoint and triangulation.

CHAPTER 7

Object Position and Orientation Filtering under Self-Motion

In this chapter, the object orientation estimates obtained from individual frames of an image sequence are integrated with position and self-motion estimates to a dynamic object state filtering framework.

As motivated in the last chapter, an estimate of the state of objects close to a moving observer is of critical importance for fully and partly autonomous navigation.

While object orientation is estimated by several object detectors, usage of orientation estimates in object position filtering is not commonly found in the literature. The most relevant study we found is Andriluka et al. [2010], where viewpoint is estimated from object detector output and fused over tracks of pedestrians to help in determining the full body pose. Contrary to the work presented in this chapter, however, viewpoint is represented as discrete variable and is only filtered in a second stage, after tracks have already been formed. Similarly in Ozturk et al. [2009], head orientation is included into a filtering framework for position and orientation of top views of pedestrians. Here, also, orientation is not used to predict future positions since body orientation, and not head orientation, are more predictive. Barth et al. [2009] use stereo measurements for tracking position and orientation of a vehicle, predicting its motion even in highly dynamic turn maneuvers.

Another novelty presented in this chapter is the usage of random regression forests for estimating the image position corresponding to the ground center point of the 3-dimensional object bounding box. Also based on the deformable part object detector output, the regression forest has to capture the dependency of the image ground center point on orientation and position of the object.

Finally, we present a particle filter that combines object orientation and ground point location estimates with monocular estimates of observer motion in a dynamic object state estimation framework. This requires system and measurement update equations for a mixed circular and linear state. Measurement equations, as in the chapter before, reflect the general form of measurement distributions from regression forests.

This combination is new to the best of our knowledge. Studies filtering orientation only are listed in sec. 6.6. Object position tracking, deducing orientation from successive position estimates, is a well-studied topic (e.g. Koller et al. [1993], Dellaert and Thorpe [1997]). An overview over object tracking approaches can be found in Yilmaz et al. [2006]. Newer methods using visual odometry to compensate for observer motion include Gammeter et al. [2008], Leibe et al. [2007]. Related in terms of methods is Detry and Piater [2010], where kernel density estimation is also used to integrate estimates of position and spherical orientation over different positions (as opposed to time as is the case here).

To summarize, the main contributions in this chapter are:

- Inclusion of measurements of orientation and location into object state filtering
- Estimation of the image object center ground point from object detector representations
- Particle filter for a mixed circular and linear state
- Information fusion with monocular odometry

This chapter is structured as follows: the object state and a way to predict it while compensating for self motion is described in the first section. In sec. 7.2 the regression-based approach to ground center point estimation is presented. Two different ways to incorporate orientation and ground point position distributions as measurements into the filtering process are described in sec. 7.3. The filtering framework is evaluated in sec. 7.5. A short summary, discussion and possible future directions are given in sec. 7.6.

Parts of this chapter have been submitted for publication in Herdtweck and Curio [2013].

7.1. Object State and Prediction

Including orientation as object state variable could be beneficial for several tasks. This is demonstrated here for the task of object state filtering under self-motion. Basing the object state vector on a common object state representation, the following 6-dimensional state of an object is tracked:

- Object position (x, y, z) in the world, with origin centered in the observer's IMU, the x axis pointing forward (along the observer's optical axis), y axis pointing left, and z axis pointing up. Position is defined here as the point in the center of the lower plane of the 3d object bounding box, which is the middle between the four lower bounding box vertices. It is referred to in the following as world ground center point or simple object position
- Object yaw orientation ϕ relative to the x-axis, i.e., observer optical axis
- Object speed (or to be more precise: engine throttle) v that drives the object
- The angle ω that the front wheels deviate from the forward direction

It is also assumed that the height of the coordinate system origin over the ground (denoted by h) is known and constant over time. Since the model origin is moving with the observer, world positions must be updated every time step compensating for observer motion, i.e., change of the world coordinate origin.

Although the height of observed objects can deviate from the ground plane, for the self-motion compensation a simple model of the observer driving over a constant ground plane is assumed. This means that only change in yaw and forward motion are needed to update particle positions. Estimates of change in yaw, pitch and roll are used in the measurement process detailed below. All these variables are predicted using the ego-motion estimation approach described in chapter 3 above. The state of the observer vehicle itself is not tracked since motion estimates are available whenever flow fields can be measured. Motion estimates are, as seen in the results (sec. 3.8) above, sufficiently smooth over time.

Assuming an ego-motion estimate of forward motion $\Delta x_{\text{ego},t}$, change in yaw $\Delta\phi_{\text{ego},t}$, and a particle $\boldsymbol{x}_{t-1}^{(n)} = (x_{t-1}, y_{t-1}, \phi_{t-1}, v_{t-1}, \omega_{t-1}, z_{t-1})$ from the previous time step, the positions that this particle has relative to the new observer positions is calculated by

$$\begin{aligned}\begin{pmatrix}\tilde{x}\\ \tilde{y}\end{pmatrix} &= \boldsymbol{R}(-\Delta\phi_{\text{ego},t})\begin{pmatrix}x_{t-1} - \Delta x_{\text{ego},t}\\ y_{t-1}\end{pmatrix}\\ \tilde{\phi} &= \phi - \Delta\phi_{\text{ego},t} \quad ,\end{aligned} \tag{7.1}$$

where $\boldsymbol{R}$ maps an angle θ to the 2-dimensioanl rotation matrix

$$\boldsymbol{R}(\theta) = \begin{pmatrix}\cos(\theta) & -\sin(\theta)\\ \sin(\theta) & \cos(\theta)\end{pmatrix} \quad .$$

Speed and front wheel angle of the observed object are are assumed constant. For an update of the height z_t of the particle in the next step, the distance

$$d_{t-1} = x_{t-1}^2 + y_{t-1}^2 + z_{t-1}^2 \tag{7.2}$$

from the observer to the observed object is calculated even before correcting for observer self-motion.

Given an ego-motion compensated particle, its new location can be predicted through an Ackerman model with constant speed and front wheel angle. Since this model does not account for slip, and speed and wheel angle may slowly change, a zero-mean Gaussian system uncertainty ϵ with constant variance is added to each value:

$$x_t = \tilde{x} + \cos(\tilde{\phi})\, v_{t-1}\, \Delta t + \epsilon_x \tag{7.3}$$
$$y_t = \tilde{y} + \sin(\tilde{\phi})\, v_{t-1}\, \Delta t + \epsilon_y \tag{7.4}$$
$$\phi_t = \tilde{\phi} + \omega_t\, v_{t-1} \Delta t + \epsilon_\phi \tag{7.5}$$
$$v_t = v_{t-1} + \epsilon_v \tag{7.6}$$
$$\omega_t = \omega_{t-1} + \epsilon_\omega \quad . \tag{7.7}$$

Δt is the time difference between the discrete time points $t-1$ and t. The height z_t of the observed object is updated assuming a ground plane of constant slant. The observer height object must approach the observer's height h over the ground plane as the observed object comes closer, and vice versa. Therefore, the object height is updated by

$$\begin{aligned} d_t &= x_t^2 + y_t^2 + z_t^2 \\ z_t &= h + (z_{t-1} - h)\frac{d_t}{d_{t-1}} + \epsilon_z \quad . \end{aligned}$$

A simple bootstrap filter is used for tracking, using a measurement model that is described in the next two sections.

7.2. Ground Point Regression Forests

Measuring orientation can be done using regression forests as done in the previous chapter and is quickly repeated in the next section. Measuring the world position (x, y, z) of an object is more involved, and needs an additional step that is detailed in this section.

The object world ground center point is related to the position and size of a bounding box returned for that object from the object detector through two non-linear relations. First, the world position must be projected into the image through a many-to-one mapping

$$(u, v) = \mathsf{img}(x, y, z) \quad , \tag{7.8}$$

which is usually non-linear and cannot be inverted without additional knowledge. The projection function img can be obtained from camera calibration and is assumed to be known in the following. The second transformation relates a bounding box in the image to a single point that corresponds to the image position of the world center ground point. This image center ground point is usually located in the lower half of the image bounding box, often close to the middle between the left and right edges. Its exact position, however depends on the exact position and orientation of the object.

For finding the image ground center point given an object detection box, we propose using a regression forest, which is referred to in the following as ground point regression forests. Since the exact position of the image ground center point is closely related to object orientation and position, which are input and output of viewpoint regression, respectively, the same input features are used. The main difference between viewpoint and ground point regression forests is the space of the output variable. For viewpoint regression, it is a one-dimensional circular space, while ground points are two-dimensional linear variables. However, thanks to the generality of regression forests, the same code can be used for both problems, wit the exception that the circular concentration operator has to be replaced with the linear equivalent for calculating the information gain obtained from a split of the training data. The determinant of the linear covariance operator is used for this task in ground point regression as described in sec. 2.4.2.

As ground truth for training, the true object center position of training samples was projected into the image. To simplify generalization, the resulting image coordinates (u, v) were then expressed as relative to the middle and size of the best detection for the object. Denoting by $(u_{\min}, u_{\max}, v_{\min}, v_{\max})$, the left, right, top, and bottom coordinates of the bounding box of a detection of the object, the relative image position of the ground point is

$$\begin{aligned} u' &= \left(u - \frac{u_{\max} + u_{\min}}{2}\right) / (u_{\max} - u_{\min}) \\ v' &= \left(v - \frac{v_{\max} + v_{\min}}{2}\right) / (v_{\max} - v_{\min}) \quad . \end{aligned} \tag{7.9}$$

As in training of object orientation forests, we only trained on detections which had an overlap with the ground truth bounding box of at least 0.8. Other parameters of the regression forests were the same as for viewpoint regression: 20 trees, 2000 tests to chose from in training each node. The minimum ground truth variance required for a split was not restricted. The training subset for each tree consisted of 30% of the complete training set, sampled at random for each tree.

7.3. Measurement Model

The described state space was designed such that prediction is easily computable. However, it has the drawback of containing dimensions that are hard to measure. Using the viewpoint regression trees described in the last chapter (chapter 6) and ground point regression forests just described, 3D position (x, y, z) and yaw orientation ϕ of a tracked object can be compared to relative image positions (u', v') and viewpoint φ of objects detected in the image. The relation between orientation and viewpoint is given by 6.3 (see also Fig. 6.4). Projecting the world ground center point into the image through 7.8 and expressing the resulting image coordinates relative to the detection bounding box using 7.9 enables a direct comparison of particle position with ground point regression output.

For this projection, an additional step compensates for observer pitch and roll, which can strongly affect projections if objects are far away or in the periphery. This procedure has also been suggested for pitch alone in Dellaert and Thorpe [1997]. Since this requires estimates of observer pitch and roll, estimates of change in observer orientation are integrated. To counter the effect of drift in these estimates, measurements from an orientation sensor are simulated by smoothing ground truth pitch and roll information over 80 frames and adding zero-mean Gaussian noise with standard deviation of $0.5°$. These simulated sensor measurements are fused with orientation change estimates as described in sec. 3.6 and using the same parameters as in the corresponding evaluation (sec. 3.8.4).

For deriving a formalization of the likelihood $p(\boldsymbol{y}_t|\boldsymbol{x}_t^{(n)})$, the same procedure as described in sec. 6.6 is used. Local maxima of regression forest outputs, which are needed for this procedure, are obtained using kernel density maximization (c.f. sec. 2.5.2). Application of the measurement model to the ground point regression output requires the following changes: For a two-dimensional linear position variable $\boldsymbol{x}$ with measurements

$$\boldsymbol{y}_t = (\boldsymbol{y}_{t,1}, \dots, \boldsymbol{y}_{t,M}) = \left(\begin{pmatrix} u'_{t,1} \\ v'_{t,1} \end{pmatrix}, \dots, \begin{pmatrix} u'_{t,M} \\ v'_{t,M} \end{pmatrix} \right) \tag{7.10}$$

one has to replace the wrapped Gaussian with a regular two-dimensional Gaussian and perform clustering in a two-dimensional linear space instead of a one-dimensional circular space.Assuming that the clustering modes are returned with covariance matrices $\hat{\Sigma}_{t,1}, \ldots \hat{\Sigma}_{t,L}$, and assuming as clutter distrribution independent detections from a uniform distribution in the lower half of an image with P pixels, leads to the position likelihood (c.f. equation 6.10):

$$p(\boldsymbol{y}_t|\boldsymbol{x}_t^{(n)}) \propto \frac{\alpha}{P/2} + \frac{1-\alpha}{\sqrt{2\pi}} \sum_{l=1}^{L} \frac{\hat{w}_{t,l}}{\sqrt{|\hat{\Sigma}_{t,l}|}} \exp\left(-\frac{1}{2}\left(\boldsymbol{x}_t^{(n)} - \boldsymbol{y}_{t,m}\right)^T \boldsymbol{B}^{-1}\left(\boldsymbol{x}_t^{(n)} - \boldsymbol{y}_{t,m}\right)\right) \quad . \tag{7.11}$$

For combining orientation and position measurements ($\boldsymbol{y}_t = (\boldsymbol{y}_{t,\mathsf{pos}}, \boldsymbol{y}_{t,\mathsf{ori}})$) that were obtained independently through different regression forests, the likelihood factorizes into separate terms for orientation and position:

$$p(\boldsymbol{y}_t|\boldsymbol{x}_t^{(n)}) = p(\boldsymbol{y}_{t,\mathsf{pos}}|\boldsymbol{x}_t^{(n)}) \times p(\boldsymbol{y}_{t,\mathsf{ori}}|\boldsymbol{x}_t^{(n)}) \quad . \tag{7.12}$$

7.4. Data Set

The Kitti data set described in the last chapter (sec. 6.4) was used for evaluation also in this chapter. Detections with an overlap of at least 60% with corresponding ground truth annotations, which were allowed to be partially occluded or truncated, were first ordered by ground truth object track. The resulting detection tracks were split into segments that have a length of least 5 detections, with no more than 2 frames between each detection. For each such segment, the filter was initialized with particles drawn from a noisy version of the ground truth position, speed and front wheel angles. Since the latter two variables are not contained in the ground truth labels, they are estimated by compensating the ground truth position and orientation labels with ground truth self-motion, smoothing the resulting world-centered positions and orientations and calculating derivatives.

7.5. Evaluation

A separate evaluation of ground point regression performance was not conducted. The good results describe here, which require ground point estimates, suggests at least adequate performance.

As initialization for particle orientations, noise with a very high standard deviation of $\frac{\pi}{2}$ (corresponding to $90°$) was used, the other variables were initialized using Gaussian noise with 10 times the standard deviation of the system noise ϵ.

Initially, the system uncertainty was estimated by comparing filtered results on the training set with ground truth values. However, tests on the validation set showed, that due to the large number of stationary vehicles in the data set the learned noise parameters were not appropriate. They were therefore manually set to:

$$[\epsilon_x, \epsilon_y, \epsilon_\phi, \epsilon_v, \epsilon_z]^T = \left[0.1\text{m}, 0.1\text{m}, 1°, 0.2\frac{\text{m}}{\text{s}}, 0.5°, 0.02\text{m}\right]^T \quad . \tag{7.13}$$

For the measurement update, the bandwidth of the kernel density maximization for ground point regression was set to 10 times the standard deviation of regression errors on the test set, resulting in values of ≈ 0.5 for both u and v image directions. The bandwidth for the circular kernel density maximization of yaw estimates was set to $0.3 \approx 17°$as used before.

A visualization of the particle density evolution for an example track on the test set using 10000 particles and estimated observer motion is presented in Fig. 7.1. The filter accurately tracks all particle dimensions, with density variances depending on whether the variable can be measured or inferred or neither. The front wheel angle, for example, is very uncertain for a few frames where the tracked object is very slow, because it cannot be inferred from consecutive positions any more if the vehicle speed is low. A slight lag in speed v can also be observed since acceleration and deceleration cannot be predicted by the system model. This leads to a minor error in estimates of object height z. The very tight fit of the image coordinates u and v (top two rows) is supported by the usage of ground truth information in choosing the detection boxes that are used for filtering. For evaluation, modes of the filtering density are obtained using mean shift clustering (c.f. sec. 2.6.2).

To summarize filtering performance, the mean and median errors of estimated viewpoints within each track were calculated and further unified to a single mean over track means and a median over track medians. The resulting numbers are shown in Fig. 7.2. Each plot compares performance of static regression with the filtering described above using viewpoint measurements or inferring object orientation from consecutive object positions. For filtered viewpoint estimates, the distance of the maximum a posteriori (MAP) estimate to the ground truth orientation. The large difference between mean and median in regression results are caused by the multi-modality of the regression output.

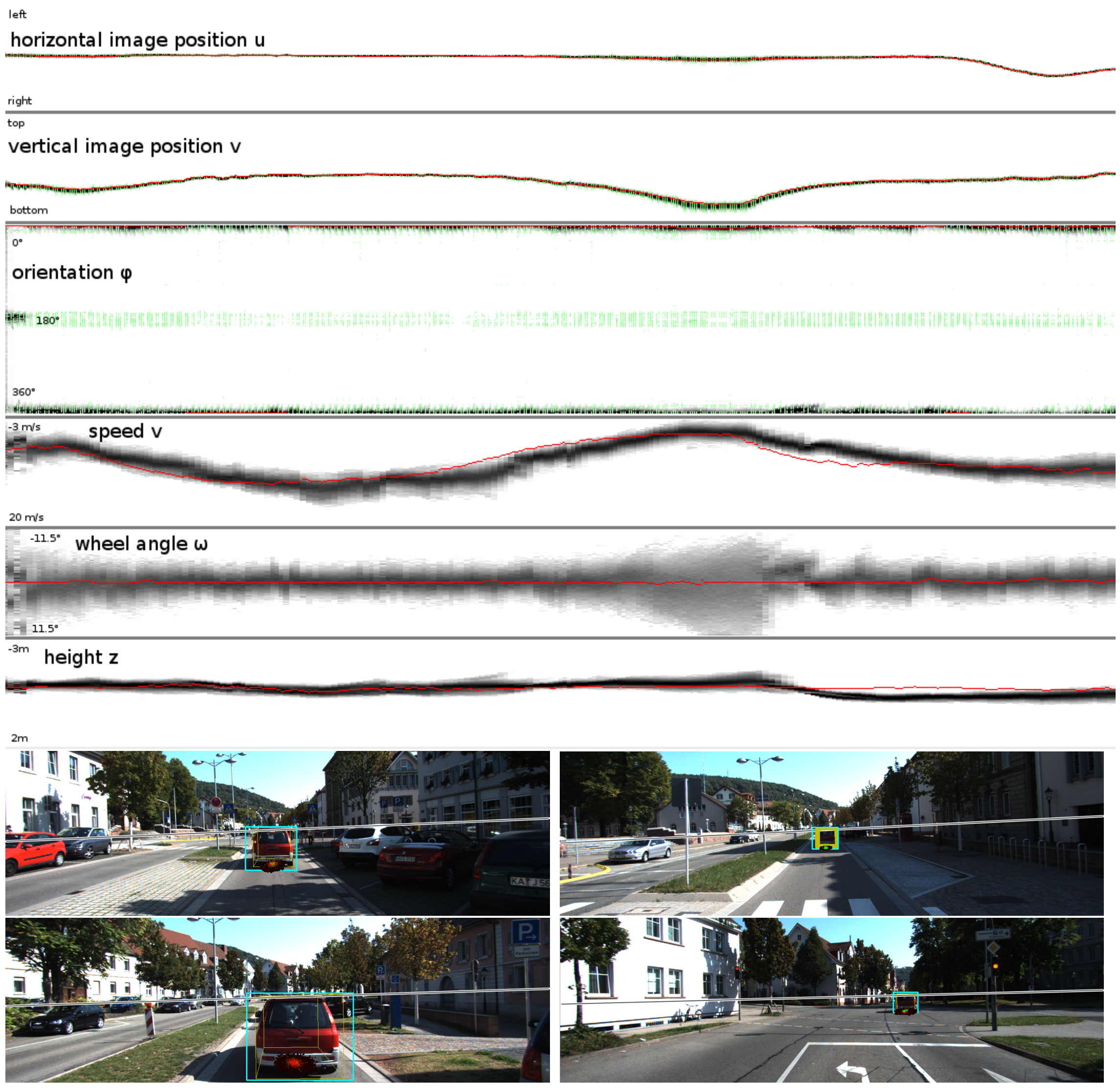


Figure 7.1.: Filtering density for the longest track in the test sequence, together with a few visualizations of the particle distribution projected into corresponding images. In the top plot, each row visualizes the density of particles for one dimension. For easier comparison with position measurements, image position u and v are visualized instead of x and y position. In each plot, columns visualize the weighted particle density (white=0, black=0.2), time progresses from left to right. The ground truth is overlaid in red, measurements for u, v and φ are inserted as green distributions. In the visualizations of individual frames, the ground truth object box is overlaid in yellow, the detection box in cyan. The roughly horizontal black-on-white line visualizes the horizon, which depends on the current observer pitch and roll that are used for measurement updates. Projections of particle positions into the image are colored with values from black (weight=0) to red (current maximum weight). Green dots indicate position estimates from the ground point regression forest.

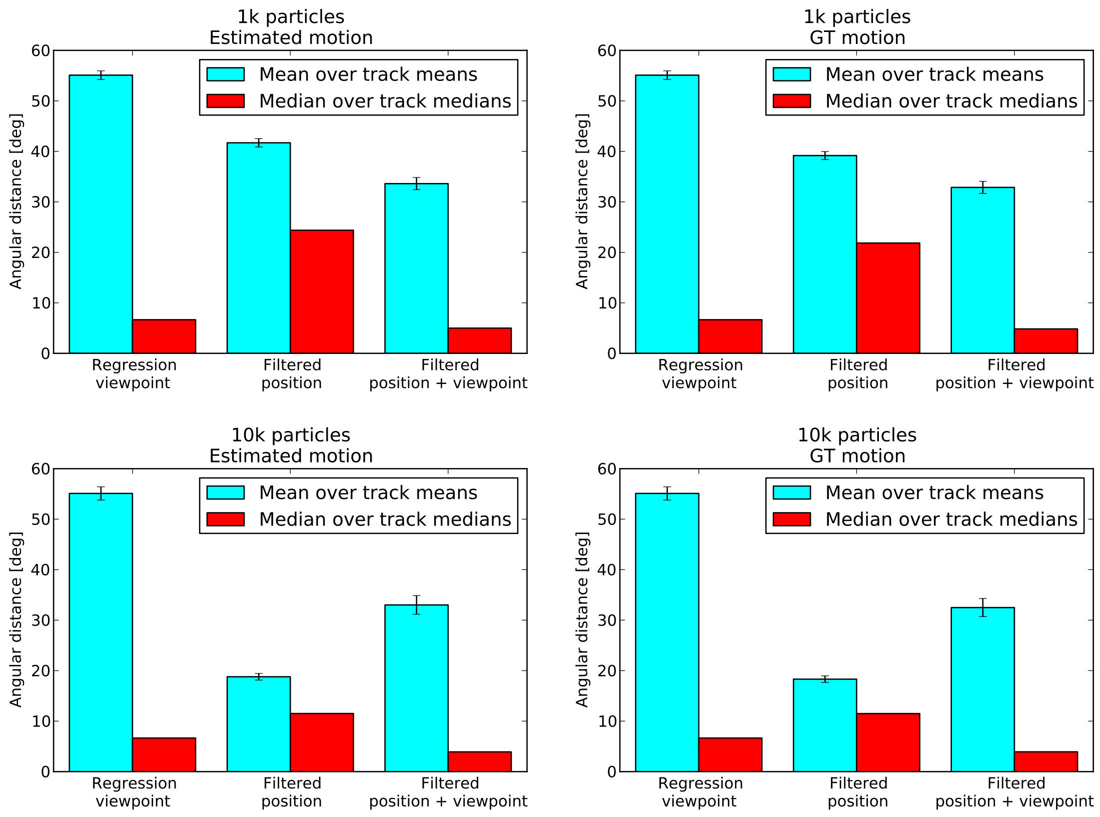


Figure 7.2.: Viewpoint estimation performance comparison between static regression, filtering with measurements of object position only, and filtering with measurements of object position and viewpoint. Plots in the **top** row show errors when filtering with 1000 particles, plots in the **bottom** row when using 10000 particles. Plots in the **left** column were created with estimated observer motion, plots in the **right** column with with ground truth observer motion. In each plot, bars show the mean over tracks of mean errors within tracks (cyan) and the median over tracks of median errors within tracks (red). The error for filter results is the angular distance between ground truth and maximum a posteriori (MAP) estimate of the yaw filtering density. For viewpoint regression, the corresponding measure is the angular distance between the ground truth and the maximum of the output density. To account for the randomness in filtering results, each track is filtered 6 times (top row) or 3 times (bottom row) with new initializations.

Filtering position and viewpoint estimates leads to the lowest median in all four plots, the lowest mean error is obtained for filtering with viewpoint and orientation for 1000 particles and for position filtering when using 10000 particles. The difference can be explained intuitively: if orientation is not measured, the filter as to divide its resources (particles) to all orientations at once until object motion has shown which orientation is the correct one. The risk of missing the correct orientation is much bigger with a higher number of particles. On the other hand, when orientation is also measured, the filter can right away concentrate its resources in that respect. Using orientation measurements can thus be used as a means to save computational effort in filtering since its performance does not depend as much on the particle number.

According to this reasoning, the mean results for the orientation-enhance filter should always be better than for the filter without orientation measurements. This is, however, not the case for the larger particle number. A possible reason could be, that with a rather vague initialization of orientation as used in these tests, the presence of a wrong mode in orientation measurements could have harmed the filter's orientation estimation from successive positions. Using the filter proposed in sec. 2.7.3 and used in the last chapter could improve performance in this case, since it maintains multiple modes in the tracking distribution.

7.6. Conclusion

This chapter presented the combination of work from earlier chapters in a single filtering framework. Monocular self-motion estimates and multi-modal circular object viewpoint estimates were combined in a joined object state filtering framework. To this end, object detection bounding boxes were compared with image projections of the 3-dimensional object ground center point. This was accomplished by training a random regression forest to predict the exact position of the object ground center point in the image using as features the part-based object detector output as in object viewpoint estimation earlier on. Projections into the image were corrected for observer pitch and roll by integrating orientation change estimates from the self-motion estimator and a simulated IMU. The particle filter methods and measurement update equations described in the previous chapter were adapted to a mixed circular and linear space and applied to object sequences from the Kitti data set. Results show that incorporating orientation measurements into an object position filtering approach leads to better viewpoint estimates than both object position filtering without viewpoint estimates, and viewpoint estimation alone.

The filtering framework described here forms a robust basis for a full-fledged object tracking framework, which includes track initialization and termination, as well as an assignment of detections to tracks. Numerous approaches for these tasks are available, including the classical joint probabilistic data association filter (JPDAF, Bar-Shalom and Fortmann [1988]) or the more recent probability hypothesis density (PHD) filter (Sidenbladh [2003]). Color information could be a helpful cue to assign detections to tracks.

Predicting object motion for more than one frame could result in a system for collision detection and avoidance, which is of great interest in applications.

As a further line of thought, optical flow on objects also contains information about object motion. Flow patterns on moving objects could for example be mapped to either speed and angular velocity or a prediction for the next image position of that object. This would require either a removal of the optical flow caused by observer motion or inclusion of observer motion estimates as context variables.

Last but not least, we made preliminary attempts at 'closing the loop' between self-motion estimation and semantic inference. After self-motion estimates have enabled a prediction of object position, surely this knowledge can be used to help self-motion estimation as well. In a preliminary experiment, flow vectors were removed from all image positions that are close to objects according to ground truth labels. This should result in a cleaner flow field that should yield better self-motion estimates. To our surprise this had no positive effect, although earlier tests (see sec. 3.9) had shown that the subspace self-motion estimation approach can deal with large quantities of missing values. Further tests could clarify reasons for this and make an improvement of self-motion through semantic information possible.

Knowing the presence and height of static objects could also help in introducing an absolute scale for unsupervised learning of the mapping from subspace to motion, and serve as additional cue in self-motion inference.

Part II.

Biologically Motivated Processing of Static Images

CHAPTER 8

Mimicking the Gist of a Scene

The gist of a scene is a concept from human visual perception research (Potter [1976], Potter et al. [2004], Fei-Fei et al. [2007], Oliva [2005], Greene and Oliva [2009b]). It describes the first rough impression that is formed in the early visual processing stages within a few hundred milliseconds. Having seen an image for such a short time (e.g., when browsing through a list of images) humans can still give a rough description like "a man with a big brown dog walking along the beach" (Fei-Fei et al. [2007]). This implies that a few prominent objects (man and dog), the general image category (beach scene) and a rough geometrical layout ("along the beach") have been identified in a consistent manner even before the eyes have made their first saccade. This representation is considered a starting point for further visual processing, allowing the focusing of attention to specific aspects of the image.

Having a similar representation in the computer could help solving various problems in computer vision. It could serve as starting value for more detailed and task-specific visual processing of the image, or disambiguate between different possible interpretations of the image. In dynamic tasks, it could be created only for the first image and then in regular intervals, assuming that the interpretation can be tracked in-between.

In this chapter, we suggest a way to form such a representation through holistic inference on several aspects of static images. To get as much information out of a single image, results from several computer vision algorithms are combined with prior knowledge such that the result is a consistent representation of image content. While image analysis was also seen as a holistic task in the very early days of computer vision (e.g. Hanson and Riseman [1978], Marr [1982]), research quickly split into mostly disjunct areas, each concerned with a subset of specific tasks. After significant progress has been made in many areas, researchers have started to tackle combinations of different problems again, using results from different areas to improve the results in others. Examples range from combining two closely related tasks like segmentation and object detection (Leibe et al. [2006]) and usage of viewpoint information to constrain object detection (Leibe et al. [2007], Held et al. [2012]), to approaches like the one by Heitz et al. [2008], which is similar in spirit to the presented approach. It combines results of multi-class image segmentation, object detection and monocular depth estimation, resulting in improved object detection and segmentation results.

Similar to Heitz et al. [2008] the proposed algorithm uses results from a scene type classifier, object detector, surface type classifier with viewpoint reasoning to build a joint representation containing a consistent combination of all of these aspects. Avoiding tighter integration of contributing algorithms has the advantage that individual algorithms can easily be replaced and allows to concentrate on how to combine results.

Simple statistics like the probability of finding a car in a street scene and the mean height of the camera over the ground, as well as heuristics like the horizon ratio relation and the fact that objects stand on the ground were used to combine knowledge from different algorithms in an iterative procedure.

8.1. Related Work

This coarse scene interpretation has been the focus of perceptual research for quite some time: in Potter [1976] participants could easily detect a cued scene within a string of rapidly presented images. Similarly, the study by Thorpe et al. [1996] demonstrated that decisions on whether a scene contained an animal or not could be made within roughly 130ms. The authors also showed that this is roughly the time it takes for the signal to travel from the retina to the decision/motor areas in the brain, leaving almost no time for top-down processing. In Biederman et al. [1982] it was shown that objects that appear in unexpected places (and therefore violating the assumed layout of the scene) take surprisingly long to spot. Another series of studies (summarized in Oliva [2005], Greene and Oliva [2009a]) has focused on modeling the holistic aspects of scene gist. The authors argued that since a scene is laid out in three-dimensional space, fast categorization should be based on low-level image properties that are diagnostic of spatial relationships. In their recent experiments Greene and Oliva [2009a], it was shown that global properties of the scene (such as concealment, naturalness, navigability) were judged on average after 40ms presentation time - even faster than scene categories (such as mountain, lake, field) at 60ms presentation time. In summary, evidence from perceptual studies shows that scene interpretation is an extremely robust and fast ability of the human brain. Given a few hundred milliseconds time, the interpretation of a scene will contain the scene type, its rough geometric layout including the vantage point, as well as a few prominent objects (Potter [1976], Potter et al. [2004], Fei-Fei et al. [2007], Oliva [2005], Greene and Oliva [2009b]).

Most similar in spirit to the presented approach are the approaches by Hoiem et al. [2006] and Heitz et al. [2008]. Hoiem et al. [2006] builds on the seminal study by Hoiem et al. [2005], who presented an approach for monocular depth estimation based on image statistics. For this, each superpixel in an image is first classified as belonging to the ground surface, a roughly vertical surface (which is sub-categorized further), or the sky, which suffices for a monocular 3D reconstruction. In further studies, this classification was used together with a viewpoint estimation to improve object detection (Hoiem et al. [2006]) as well as occlusion estimation (Hoiem et al. [2008]).

The second approach Heitz et al. [2008], as mentioned above introduces cascaded classification models to combine results of individual algorithms to a combined estimate. The approach builds on the monocular depth estimation presented in Saxena et al. [2007].

8.2. Data Set

Training and testing are performed of a subset of the LabelMe data set Russell et al. [2007b]. It contains over 50,000 annotated natural images in a huge variety of scene types, ranging from beach scenes over forests and country roads to urban areas and highways, and from wide panoramic views to portraits. Images can be uploaded and labeled by anybody using an online tool. Annotations are available for regions and objects in free-form test. An example image with annotations is shown in Fig. 8.1.

To simplify the task of creating a gist representation, the proposed system is restricted to outdoor images. A subset of the complete LabelMe data set is therefore used, that consists of 170 manually selected folders which at the time of extraction (2008) resulted in roughly 76,000 images, 20,000 of which were annotated. These were partitioned into roughly equal-sized training and testing sets.

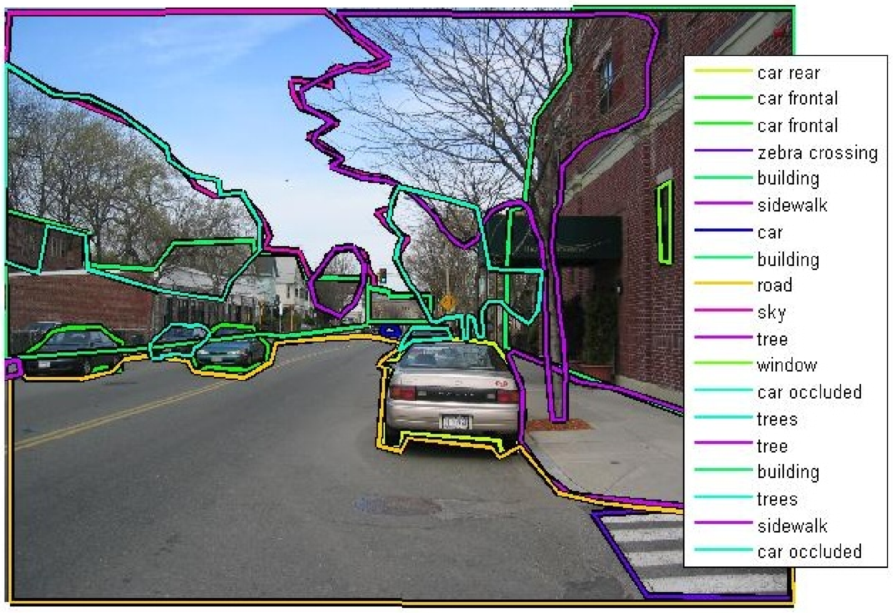


Figure 8.1.: Example image and annotations from LabelMe (left)

8.3. Algorithms

The algorithms, whose output forms the input for the proposed gist representation, are the following:

8.3.1. Scene Type Classification

The spatial envelope Oliva and Torralba [2001] uses local orientation filters on different scales for each color channel and classifies the scene using a support vector machine (SVM) on the filter outputs. Results below are reported using the original models for the eight scene types (tall buildings, inside city, street, highway, coast, open country, mountain, forest). Later, we attempted to capture a larger variety of scene classes in the data set by defining 19 categories that formed a compromise between capturing as many images as possible but still creating homogeneous classes. Re-training the SVMs on manually labeled training images did not lead to improvements in gist formation.

8.3.2. Object Detection

For detecting objects, several algorithms were investigated. The first one (Serre et al. [2005]) is a object classifier, modeled very closely on physiological and psychophysical results. Using it in a sliding window fashion proved inefficient and application on salient regions obtained from Itti and Koch [2001] proved useless. Next, the extension from Lienhart et al. [2002] of the cascade of boosted classifiers (Viola and Jones [2001]) approach, as well as the detector using histograms of oriented gradients (Dalal and Triggs [2005b]) were used. For both were applied the models available with the code, and also re-trained them on training data. Both were configured to even return detections with a low probability, which required minor modification in the detector post-processing code.

Re-training detectors on LabelMe annotations proved challenging, since object labels are given as free-form text. As proposed in Russell et al. [2007b] the object labels were enhanced with synonyms and super-ordinate categories of labels using WordNet (Miller et al. [1990][1]). Objects whose labels contained words like 'occluded' or 'partial' were ignored, as well as annotations in plural ('trees') or annotations with meanings unknown to WordNet ('boat ramp'). A further improvement of annotations was achieved by correcting the most common spelling errors ('folage','paht','buiding',etc.), adding meaning to unknown terms like 'Mona Lisa', 'Venus de Milo', and using WordNet meanings to dissociate for example 'arms' (meaning 'rifles' or 'tree branches') from the body part.

The resulting clean set of objects was converted to bounding boxes and used to re-train object detectors for some of the most common basic-level object classes using the respective publicly available code.

These object boxes were also used to gather statistics on object position and size, making use of the fact that all images were taken by humans, that tend to create images with a certain configuration of objects. The resulting statistics are visualized in Fig. 8.2.

8.3.3. Surface Orientation

The system proposed in Hoiem et al. [2005][2] classifies image regions by their orientation, allowing a simple form of geometric reasoning from a single image. After creating several over-segmentations using the approach from Felzenszwalb and Huttenlocher [2003], each resulting superpixel, is assigned probabilities for belonging to one of the surface type classes 'ground', or 'sky' or one of the vertical classes 'facing left', 'facing the camera', 'facing right', 'porous' (e.g. leaves), or 'solid'. Each superpixel is then assigned to the class that assigns the highest probability to it. An example for the image shown in Fig. 8.1 is presented in Fig. 8.3. While this yields a much coarser reconstruction than,

[1]available online: `http://wordnet.princeton.edu`
[2]available online: `http://www.cs.uiuc.edu/homes/dhoiem/projects/software.html`

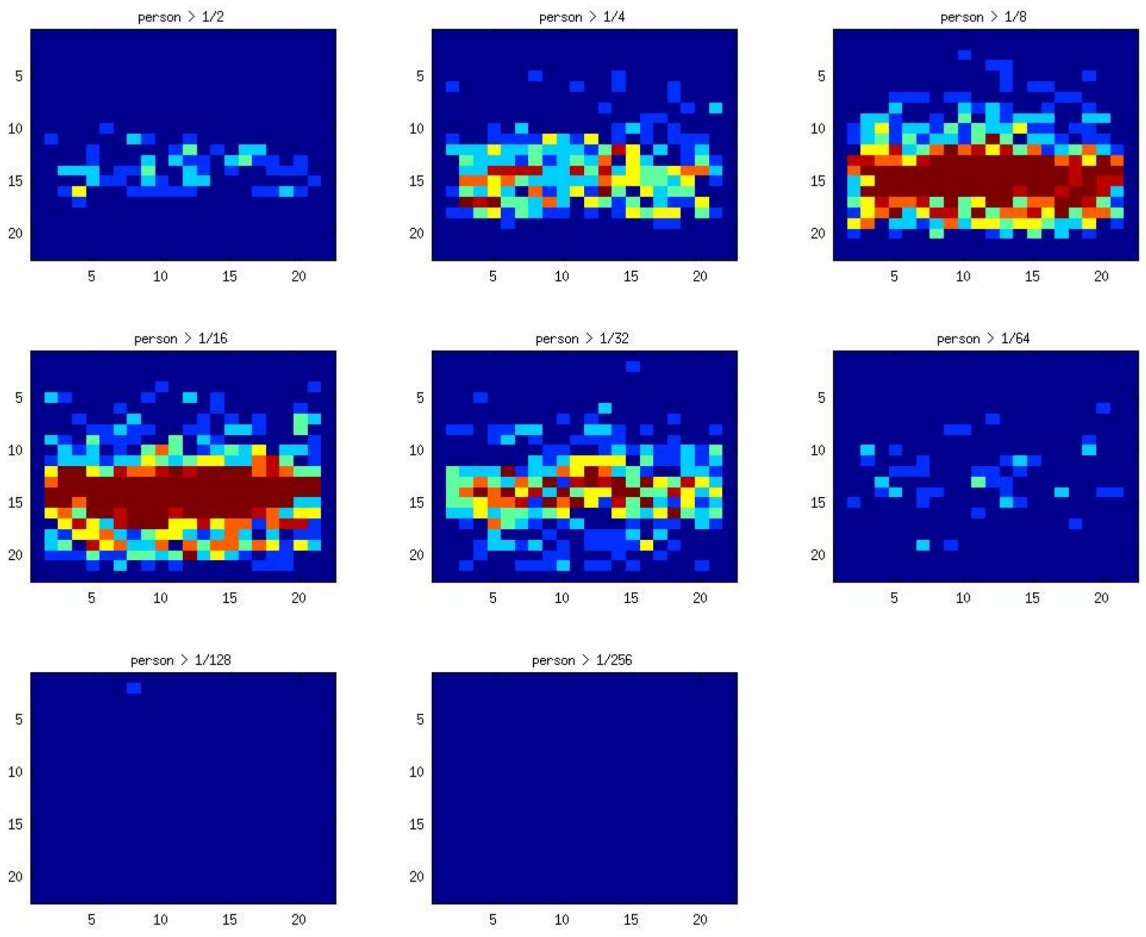


Figure 8.2.: Prior for size and position of people in images. Each plot shows the distribution for a different set of heights (relative to the image height)

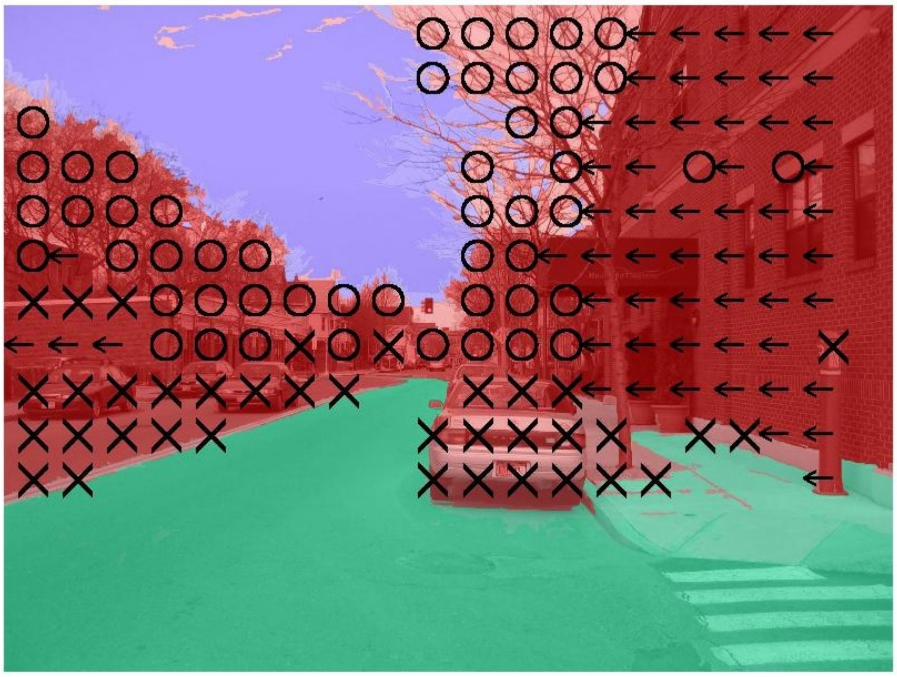


Figure 8.3.: Estimated surface types for the example image from Fig. 8.1. Colors and symbols denote assigned surface types: green is ground, blue is sky, red is one of the vertical classes 'left-facing' (←), 'porous' (o), or 'solid' (x). This figure was created using the publicly available code from Hoiem et al. [2005]

for example, the approachSaxena et al. [2007], it was found to be robust enough for the rough geometrical inference required in this project. This approach was therefore used without any re-training.

8.3.4. Viewpoint

Given two or more objects of known size, the height of the camera over the ground and the horizon in the image can be estimated using the horizon ratio relation already described in Gibson [1979], analyzed in Rogers [1996] and also used in Hoiem et al. [2006]. It states that for objects on a horizontal ground plane, the distance from the top-most pixel of the horizon divided by the distance from the bottom-most pixel of the object is constant, irrespective of the distance the object has from the viewer (see also sec. 9.2). This requires a simple perspective camera model with now skew, no camera roll and roughly horizontal ground plane that all objects are located on. Assuming these requirements are fulfilled and given objects of known height, one can create an estimate of camera height over the ground and of the horizon position in the image. These two variables are referred to as 'viewpoint' in this chapter.

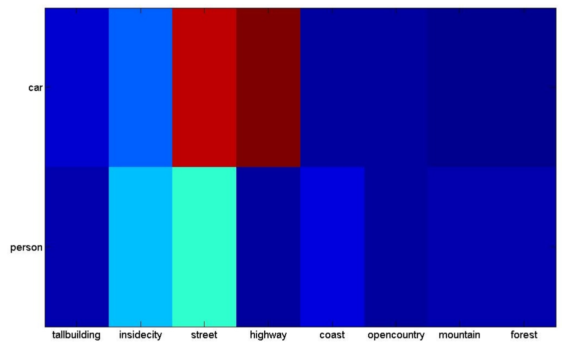


Figure 8.4.: Learned statistic for number of people and cars in different scene types.

The horizon in the image is also constrained by surface orientation estimates, since it almost always is below all sky pixels and above all ground pixels.

There is also a strong tendency in the data set for camera heights of ≈ 1.6m (eye height) and horizons in the middle third of the image. This was exploited by forming a weak prior over typical camera heights and horizon positions. Having manually labeled the horizon in a set of training images, the horizon ratio relation and the known height of cars and people was used to estimate mean and standard deviation of the height of the camera over the ground and the horizon position in the image.

8.3.5. Further Statistics

In order to enable an update of object probability given scene type and vice versa, statistics were gathered how often which object type appears in which scene type. The mean number of objects of the two most common object classes in the eight scene type classes are shown in Fig. 8.4. The dependency of scene type on surface orientation patterns and of viewpoint on scene type were also studied.

8.4. Combination

Having learned priors from training data and created estimates from algorithms for a new image, the proposed approach tries to find a consistent subset of confident estimates. To this end, each scene type and object detection, as well as the scene-independent viewpoint prior is considered a candidate, whose consistency with the gist is in doubt. All candidates are assigned a confidence value $\mathrm{conf} \in [0, 1]$. For each scene type, this is the normalized SVM classifier output, for objects it is the normalized score calculated by the detector.

Using the prior for object position and size, the confidence of object candidates is adjusted. This step already reduces the number of false positives in the object candidate list. Surface orientation estimates were found to be reliable enough to be trusted without adjustment through the system. Since the viewpoint is only estimated based on other estimates, there is only one viewpoint candidate which represents the average view in all images of the training set.

Based on these candidates, the following steps are performed by the algorithm:

First of all the most confident candidate of all is selected in a winner-takes-all fashion, removed from the set of candidates and considered a gist part.

Based on the current set of gist parts (in the beginning only one) all the candidates' confidences are modified to ensure that in future iterations, only candidates enter the gist that are consistent with the current gist estimate, i.e. set of gist parts. In detail, the following updates are made:

scene $\rightarrow$ obj using a correlation matrix, confidences of objects that are unlikely to occur in the currently estimated scene type (e.g. flowers in a street) are decreased while objects of more probable classes, like a building in a city, are increased

obj $\rightarrow$ scene using the same correlation matrix as in "scene $\rightarrow$ obj" the type of objects found in an image modifies confidences in scene type estimates

obj $\rightarrow$ view given two or more object boxes with known real-world heights, one can estimate the height of the horizon and the height of the camera above the ground using the horizon ratio relation. Every combination of two objects in the gist therefore creates a new viewpoint candidate

geom $\rightarrow$ obj as also used in Hoiem et al. [2006], many false positives can be removed by requiring vertical surface types within every object bounding box and ground pixels below it. Objects that do not match these criteria get their confidences lowered

geom $\rightarrow$ view assuming that the horizon is above the ground and below the sky, one can calculate for every pixel row in the image a confidence of the horizon being in that row.

view $\rightarrow$ scene uses the same inference as "scene $\rightarrow$ view" to change scene type confidences

view $\rightarrow$ obj uses the same inference as "obj $\rightarrow$ view" to change object confidences

The combination of existing, a-priori confidences $\operatorname{conf}(x_i)$ and the a-posteriori confidences $\operatorname{conf}(x_i|y_1, \ldots, y_{\#y}$ deduced from other estimates $y_1, \ldots, y_{\#y}$, where x and y are one of $\{o, s, v\}$, is formed using a simple weighted average. Weights for this process are hand-tuned to represent reliability of queues. The same confidence adjustment is performed for each gist part given the other gist parts. If through this adjustment, a gist part's confidence falls below certain threshold, it is considered inconsistent and moved back to the list of candidates again. For example, even the most confident of all candidates and therefore first gist part, which biases the choice of all other candidates to enter the gist, could be removed after a few iterations if the other gist parts favour a different gist configuration. To summarize the gist formation steps:

1. Find the most confident candidate and move it into the gist
2. Change candidates confidences using the above functions to make sure that only those candidates can enter the gist that are either very good a-priori estimates or are consistent with the existing gist hypothesis
3. Check gist-internal consistency by changing all the gist parts' confidences based on the other gist parts as described above
4. Look for gist parts that have become inconsistent with the rest and remove them from the gist, adding them to the list of candidates again
5. Repeat

This iteration is continued until the gist has converged to a stable state which then forms a consistent gist estimate including the best objects, scene type and view point estimates, as well as a matching geometry. An overview over this procedure is visualized in Fig. 8.5

8.5. Results

Despite parameter optimization to correctly chose the weights and thresholds within the combination framework, improving the training set, using different object detectors and re-training of object detectors and scene classes were used, the described system did not converge in a stable gist representation for many images. Two stages of the estimation process for an example image are shown in Fig. 8.6. Two final results of a later version of the described algorithm are shown in Fig. 8.7. Although the approach did identify many false positives and showed remarkably good results for some images, it failed to converge or converged to wrong estimates for many cases. Restriction to easier cases like only updating objects based on fixed other cues was also not successful.

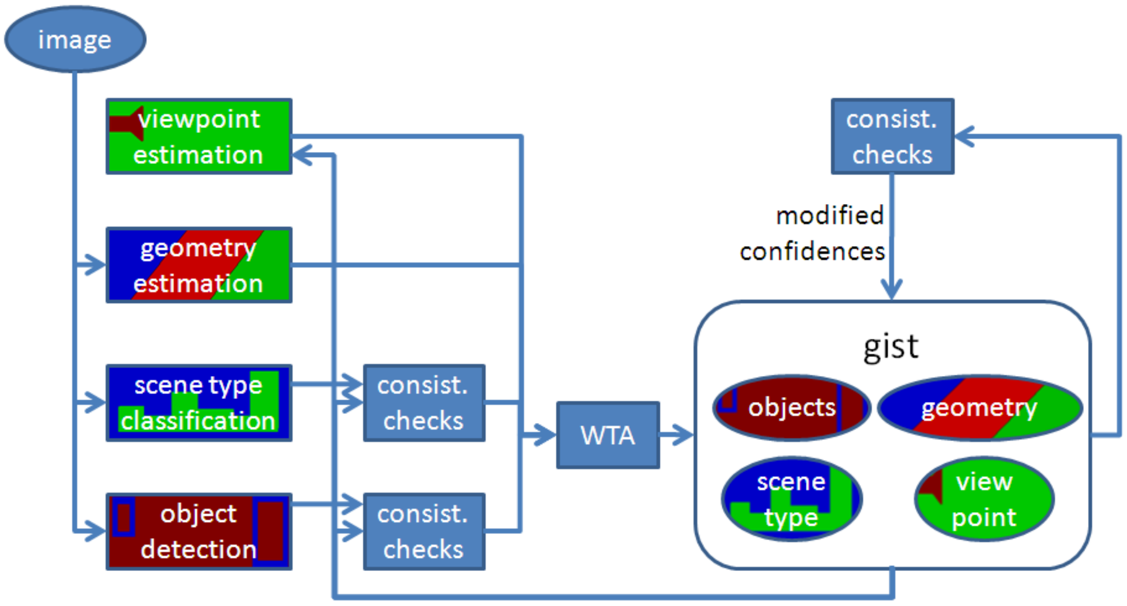


Figure 8.5.: Gist formation system overview. Given an image, algorithms calculate surface orientations ('geometry'), scene type class probabilities and detect objects. Results are considered candidates for an inclusion into the gist, which is the set of confident and consistent estimates. First, the most confident estimate is considered reliable and moved into the gist, meaning it is considered true for the moment. Confidences of all candidates are influenced by estimates in the gist, but not the other way round. Confidences of gist estimates influence each other, as well. If two or more objects are contained in the gist, viewpoint estimates are created and also considered candidates, together with the view prior. If the confidence of an estimate within the gist falls below a threshold it is removed from the gist and considered a candidate again.

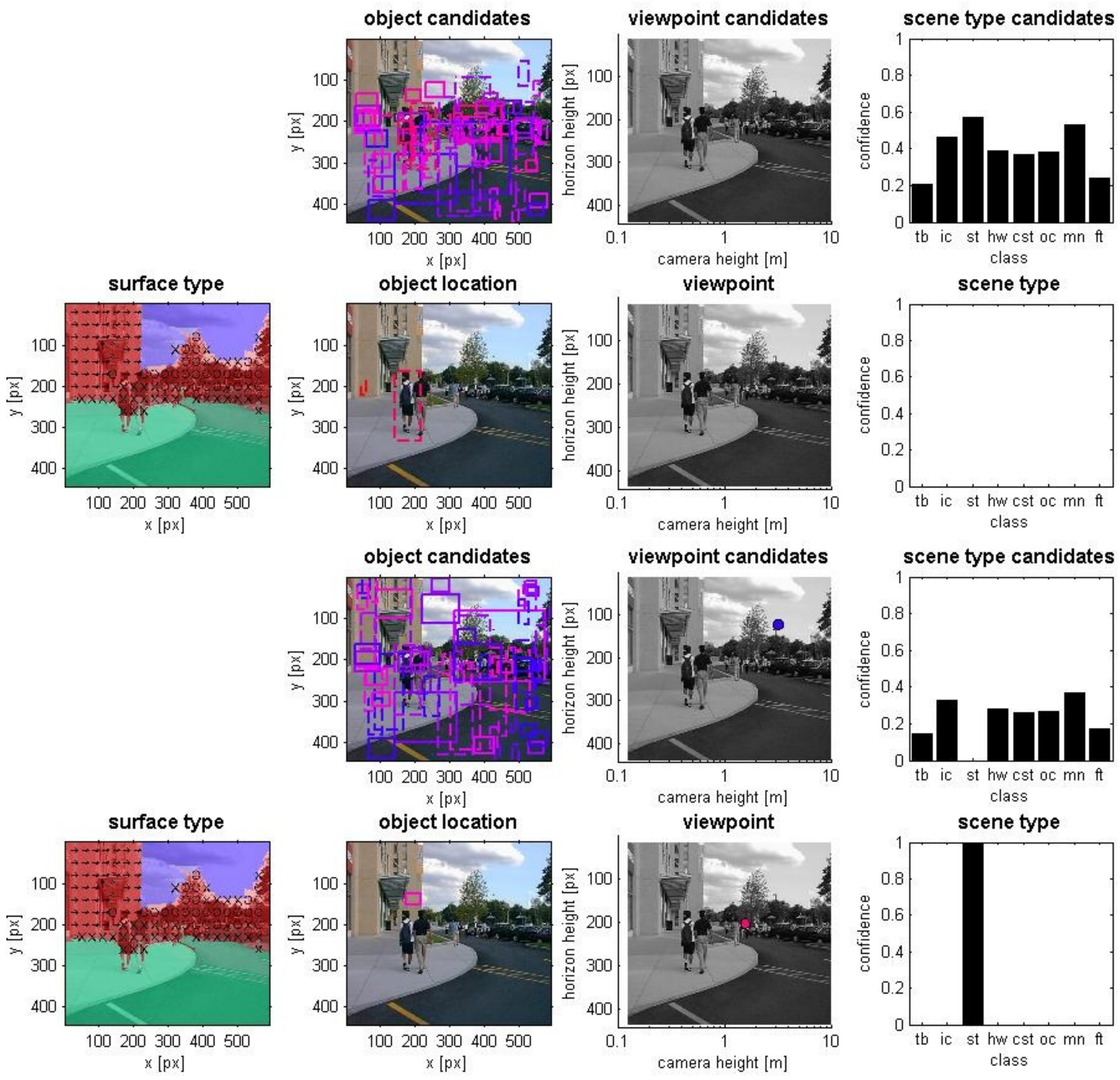


Figure 8.6.: Example results obtained with the proposed algorithm. The top two rows represent an early state of gist formation, the bottom image a later one. The first and third row show the candidates that have not been included in the gist, while the second and fourth row list the candidates included in the gist. In this example, gist formation starts off with a correct and a wrong person detection, which are however consistent with the viewpoint. Unfortunately, the correct detection is removed again from the gist since it loses confidence from other sources. What remains is a correct viewpoint and scene type estimate with a wrong car detection.

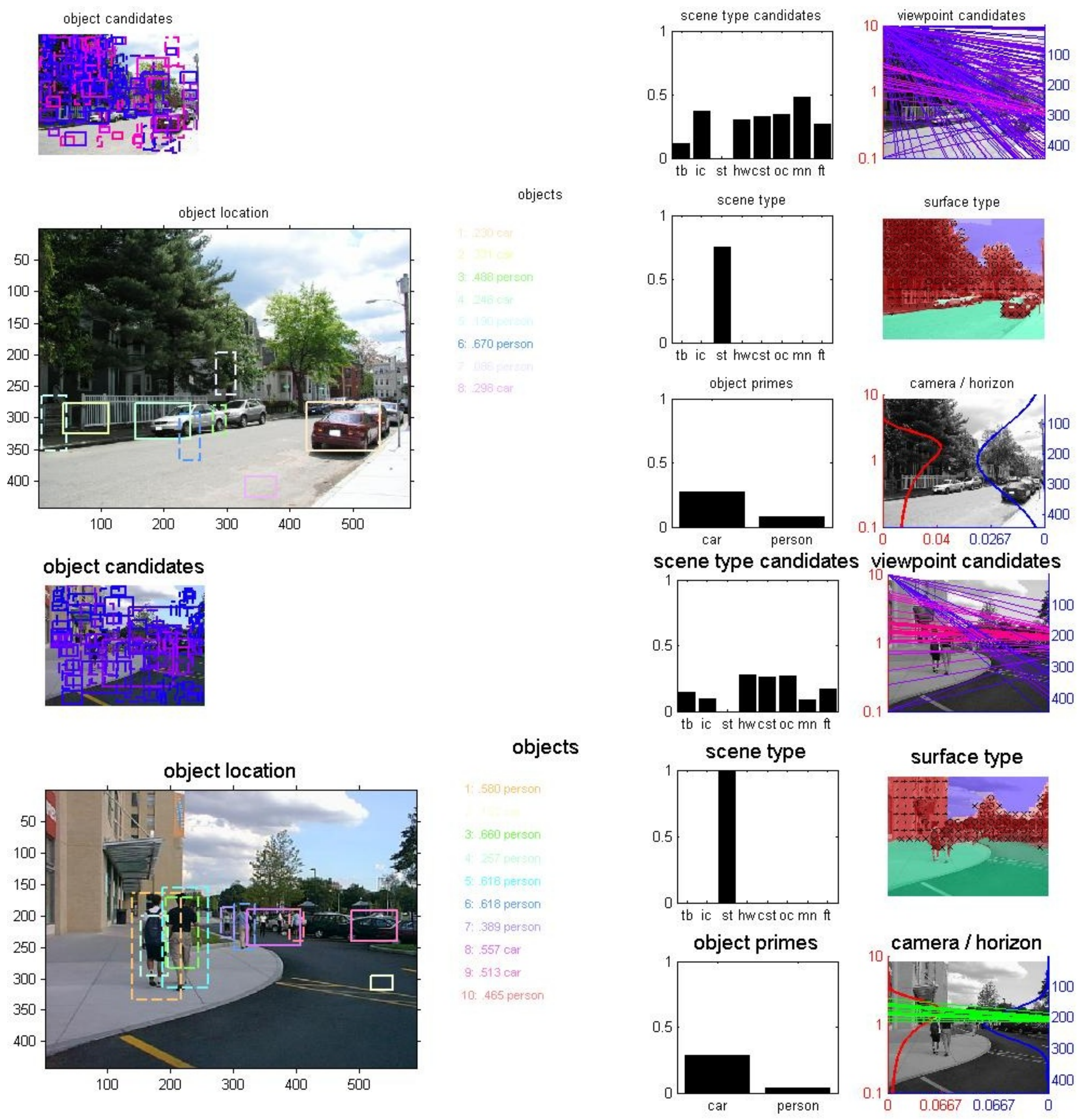


Figure 8.7.: Some more final results from a newer version of the algorithm. In this visualization, object candidates are shown in the top left, gist parts in the big lower left image with a list of confidences for the color-coded boxes next to it. In the top row of the right part, candidates for scene type and viewpoint are shown. Viewpoint candidates are visualized as lines connecting a camera height (in meters) on the left with a horizon height on the right. Color codes confidence (red=1, blue=0). The second row on the right shows the estimated scene type of the gist and the surface orientation estimates which are not changed. The bottom row on the right shows the probability for object types given the scene type and the prior distribution for viewpoints as two distributions for camera height (red) and horizon position (blue). Viewpoint parts accepted into the gist are overlaid as green lines.

8.6. Conclusion

The reasons for the disappointing results were not analyzed thoroughly. Although our basic idea of combining different algorithms has also been implemented in other approaches (e.g. Heitz et al. [2008]) the gist implementation proposed here did not improve upon the results of the best single estimator much. One reason could be that the input from algorithms was too noisy. The surface type estimation and scene type classification performed remarkably well, but the detected objects often were inadequate. Re-training on a data set as noisy as LabelMe may also have been the wrong approach. Current object detectors and new hand-labeled data sets like Kitti (Geiger et al. [2012], c.f. sec. 6.4) may help solving this task as well.

Other research (e.g. Hays and Efros [2008] has shown that some tasks can be handled simply by using a much larger data base. In Li et al. [2010], scene classification is achieved simply by using a large number of object detectors and pooling their output. Maybe a much larger number of object detectors or scene classes could help creating a gist representation.

Other feature cues like surface types could complement object detection (c.f. Heitz and Koller [2008]).

Finally, another reason could lie in the particular combination and update framework. Finding and maintaining a stable state in such a large, complex system is hard to engineer or analyze. Other inference tools like conditional random fields or more general probabilistic graph models that have become popular recently would most likely lead to more stable results given enough training data.

CHAPTER 9

Horizon Estimation by Human and Machine

This chapter is based on a manuscript by Christian Herdtweck and Christian Wallraven, which in a revised version has been published in 2013 in the Public Library of Science (PLoS) ONE (Herdtweck and Wallraven [2013]). Parts of this chapter have been published in Herdtweck and Wallraven [2010a] and were presented in form of a poster (Herdtweck and Wallraven [2010b]).

9.1. Introduction

An important aspect of the gist of a scene, which has been discussed in the last chapter (chapter 8), is the viewpoint. While there is a large body of work in both perception and computer vision literature dedicated to the other aspects of gist (object recognition, scene classification, space perception, geometry estimation), little is known about how humans estimate the viewpoint. Still, knowledge about the viewpoint is often assumed in perceptual work and related cues are used in computer vision applications.

One aspect of viewpoint, that is mentioned most frequently, is the *horizon* of an image. Since this is also a quite intuitive measure of viewpoint, this is the main focus here.

In this chapter, three experiments on horizon estimation are presented, two psychophysical and one computational.

In the first experiment ground truth data on the 'estimatability' of the horizon in the stimulus set is obtained. Participants are asked to estimate the horizon heights in images, given enough time for careful decisions. The resulting distributions of horizon heights are then used as ground truth measure for experiments 2 and 3.

In Experiment 2 the theoretical considerations in the literature (Foulsham et al. [2008]) is validated, that the gist of an image might contain information about the viewpoint. Such considerations seem reasonable since object presence, geometric arrangement and even scene type of an imaged scene are closely related to the viewpoint of the camera. Since the gist is extracted by early visual processes before the first saccade, a horizon estimate or related viewpoint information must be available after 300ms. To support this, participants were asked to perform the same task as in experiment 1, but this time with only one very short (153ms) presentation of each stimulus. This time was chosen to make a thorough image analysis impossible and to limit participants' ability to memorize the image, so they had to rely on their first impression of the image for horizon estimation. The influence of several image manipulations on estimation performance was also measured.

In the last experiment, the role of different cues to horizon estimation were investigated. Several simple computer vision algorithms were created that estimate the horizon using different single cues. This could not only help in mimicking horizon estimation (and possibly gist) in an artificial vision system. Comparing the behavior of our algorithms with human results for the same image manipulations and same data set, might also give hints to what cues the human visual system might be using.

9.2. Viewpoint and Horizon

Since "horizon" is a term widely used in different meanings, but often only explained by "where the sky meets the ground", the term is defined in more detail here. Participants in our experiments received similar explanations, together with visualizations that are shown in Fig. 9.1 and examples in Fig. 9.2.

The astronomical horizon (in the rest of this chapter simply referred to as the horizon) is defined by the 'horizon plane', a plane that is perpendicular to gravity and located at the same height as the viewer's eyes/camera. In any image taken by this camera / viewed through these eyes, the plane is only visible as a horizontal line, which is the horizon line referred to in the following. It is not dependent on the slant of the ground surface nor on the presence of occluders (c.f. Fig. 9.1) — it is always straight ahead with respect to gravity.

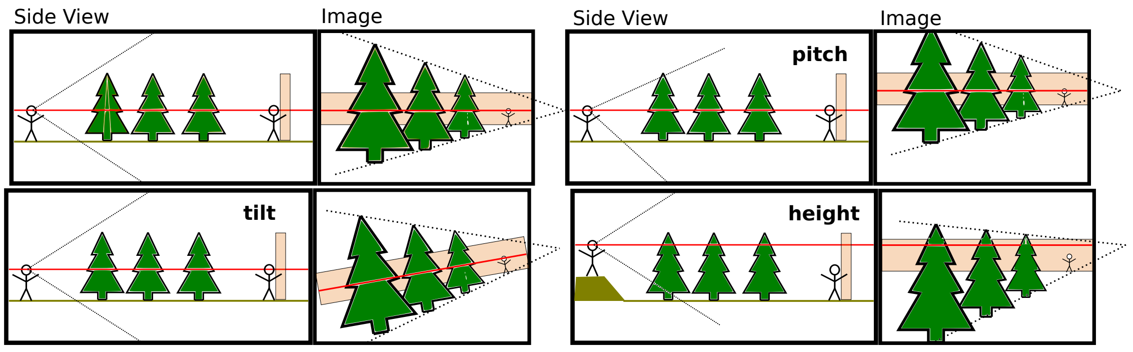


Figure 9.1.: Visualization of changes in viewpoint parameters and their effect on the horizon in the image. For each plot the left part shows a side view of a simple scene while the right part shows the scene through the left person's eyes. The red line in the side view indicates eye height, the red line in the image is the horizon. The black lines in the left part indicate the viewing frustrum of the viewer, the black dotted lines on the right indicate parallel lines in the real world that converge on the horizon. **Top left**: Reference scene with viewer looking straight ahead with zero tilt, zero pitch and from 'normal' eye height. **Top right**: A change in head pitch of the left person results in a vertical movement of all the image including the horizon. **Bottom left**: Tilting the viewer's head will tilt the whole image including the horizon **Bottom right**: If the left person views the scene from a higher position, the position of objects with respect to the horizon changes, the further away the object the smaller the change.

The *visible* horizon is the (usually not straight) boundary line above which only sky can be seen and below which there is no sky (except, of course, reflections or smaller patches). This is usually above the (astronomical) horizon, except if the observer is standing on a highly elevated place. It can also slightly differ from the *true* horizon, which takes into account the fact that the earth is not flat.

Further horizons have been defined in the psychophysics literature. The *terrestrial* (sometimes also referred to as *truncated*) horizon for example refers to the image height of the point on the surface plane that is furthest away from the viewer (which for finite surfaces without slant is slightly below the horizon). On can also extend the definition of horizon to the *horizon of an arbitrary plane* which is the line in the image where all parallel lines within this plane meet. This kind of horizon only depends on the plane's slant and coincides with the horizon used here for horizontal planes. There is also the *morphological* horizon which is the straight-ahead direction with respect to one's own body, so it is independent of the visual stimulus and only depends on the viewer's body orientation.

The most general rule to find the astronomical horizon in an image is, that everything that is above the viewer's eye / camera (and consequently above the horizon plane) is above the horizon line in the image, and the same holds true for things below the viewer / camera. The description of horizon as 'where sky and ground meet' is true if the ground is sufficiently flat and there are no occluders, as is the case for many coastal scenes (e.g., Fig. 9.2 left). However, usually there will be buildings, plants or mountains occluding that line, or the ground surface is slanted, so it must be estimated. One cue in that case is to estimate where on the occluder the viewer's eye height would be. In case of other people of similar height on a flat ground this is easy, the horizon must be close to their eyes as well. For other occluders one must estimate their distance and size to find the viewer's eye height there (see Fig. 9.2 middle left). If the occluder is very far away, then its ground contact point (i.e., the terrestrial horizon) is a good approximation to the horizon because the own eye height is negligible in the distance (imagine the height of people on the island in the left image of Fig. 9.2). Another clue to the horizon height is *perspective*. Horizontal lines that are parallel in the real world will meet in the vanishing point which is on the horizon. This can be used in structured environments like cities or parks where often many parallel lines are present (as is the case in the middle right image of Fig. 9.2). This has also been referred to in the literature as the *parallel lines* and the *horizon rules* (Wu et al. [2007]). All these clues (except the 'above stays above'-rule) fail, however, if the ground is not flat. In that case one can try to account for the effect by estimating the amount of ground slant or rely on other senses, gut feeling and heuristics (see for example Fig. 9.2, right).

The horizon is influenced by most other aspects of the viewpoint as illustrated in Fig. 9.1 for pitch and tilt of the camera as well as change of viewer height.

Although the proper definition of the horizon sounds very theoretic, it is a measure that is quite intuitive once participants were familiarized with the practical application of horizon estimation as described above. It is also a measure that strongly depends on many viewpoint parameters as can be seen with camera pitch and tilt as well as height above ground in Fig. 9.1.

In real-life situations, additional sources of information can be used to estimate the horizon: these include dynamic visual cues such as optical flow, proprioceptive, and vestibular cues. Final horizon estimates are likely formed by combining estimates from all these input sources. The focus of this chapter is horizon estimation from static pictures of outdoor scenes in the context of the aforementioned prior work on scene gist

9.3. Related Work

The horizon is a factor that is used quite extensively in the literature on space perception and self-orientation as well as computer vision. However, very little work has been done

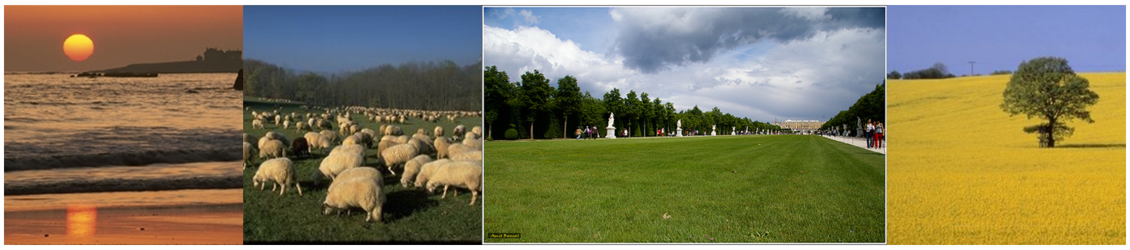


Figure 9.2.: Images to explain horizon cues. **Left**: In the left half of the image one can see where sky and ocean meet, so the horizon must be at that height. **Middle left**: assuming the the ground has no slant towards the forest in the back, the horizon can be estimated by imagining a person right in front of the trees and pointing at that person's eye height. **Middle right**: a case where perspective gives a clear horizon estimate (parallel lines are the borders between grass and path or the tree tops). Note that this is not at other people's eye height, presumably the picture was taken from a sitting/kneeling position. **Right**: an example where the surface slant is hard to estimate, so the horizon could be anywhere below the sky. The tree might be seen as standing slightly higher than the viewer, so can give a lower upper bound.

on how humans actually estimate it. Therefore, the literature in the mentioned fields is summarized here with a bias towards horizon estimation. Pointers to literature on early visual perception and computational methods used for viewpoint estimation are also given.

9.3.1. Self-Orientation

In order to identify the horizon in the world one needs to determine one's body and eye orientation with respect to gravity. A long line of research has been devoted to the estimation of self-orientation, early examples being the observation in Aubert [1861] that in a dark room the apparent orientation of a line changes with body tilt, and the famous experiment (Wertheimer [1912], Asch and Witkin [1948]) showing that a tilted room affects the perceived upright direction. Gibson, who later emphasized the importance of the horizon for visual perception (Gibson [1978]), also contributed in his earlier work to this line of research by describing the effect of gravity and strong motion on the otherwise very stable perception of the horizontal and vertical (Gibson and Mowrer [1938]). Newer studies on self-orientation perception, relevant for horizon estimation, are concerned with estimation of body pitch. Cohen and Larson [1974] strapped participants to a bed that could be pitched by a motor and asked them to adjust their pitch to certain orientations which produced angular errors up to 15°. In the same study participants had to estimate the horizontal component of the straight-ahead direction relative to their body (the morphological horizon) which they could do with a precision of around 10°. Matin et al.

[1982] determine the importance of extra-retinal eye information for estimation of the horizontal in darkness by paralyzing the eyes with curare. The higher precision of an estimate of the horizontal given visual stimulation as compared to darkness was shown in many studies, for example in Stoper and Cohen [1986], where participants had to adjust a chair's height until a illuminated target appeared at eye level. A long line of research was performed by Matin and Li on estimating the visually perceived eye level (VPEL) i.e., the straight-ahead direction with respect to true gravity given a pitch of the observer, a room of various pitch angles and different lighting conditions (darkness, whole room, degraded line stimuli), which they summarized in the great circle model (Matin and Li [1992]). Other studies analyzed the effect of gravity (Di Zio et al. [1997], Cohen et al. [2001]), gymnastic expertise (Bringoux et al. [2000]), gender (Tremblay et al. [2004]) or response strategies (Bringoux et al. [2004]) on the estimation of the body-referenced or gravity-referenced straight-ahead.

9.3.2. Space Perception

The orientation of one's body affects and is affected by the percept of the visual surrounding, in particular the visual horizon. There is therefore quite some literature in space perception that assumes an estimate of the visual horizon (often in form of the horizon of a ground plane) or the horizontal direction and uses it to explain our perception of distance and height of objects and slant of surfaces. An early overview over this field was given in Sedgwick [1986]. More recent work includes the study Rogers [1996] on the horizon-ratio relation (Gibson [1978], Sedgwick [1980], the horizon line in an image intersects objects standing on the ground at eye height which can be used to estimate their height). From simple line displays participants estimated the height of objects with high precision if the horizon line was close to the middle of the display. Otherwise performance dropped, probably because participants perceived the line not as the horizon but as an edge of an object. Even without any hint on a visual horizon participants made strong assumption on the perspective in the image which shows the presence of a strong bias on perspective in images.Ozkan and Braunstein [2010] conducted a similar study and found high agreement between relative distance estimates of two ellipses in front of a simple line drawing with distance estimates of cylinders rendered on top of photographs with clearly visible horizons. They also examined the influence of the ground and ceiling surface and an explicit vs. an implicit horizon, finding that both the implied vanishing points of ground and ceiling surfaces as well as the height where the ground surface terminates influence horizon estimates. The effect of ceiling vs. ground plane was also a topic of the study Thompson et al. [2007] with the result that the accuracy of blind-walking to targets on the ceiling and ground plane were surprisingly similar but are affected by modifications of the horizon height.

The measure that is most directly linked to the horizon is angular declination or elevation of an object, which is the angular difference between the straight-ahead direction (i.e., to the horizon) and a line from the observer's eye to an object's ground contact point. In the study Philbeck and Loomis [1997] angular declination was the most informative cue for distance estimation. Ooi, Wu and He found in several studies that angular declination is estimated quite accurately (Ooi et al. [2006]), that a change of angular declination changes estimated distances (Ooi et al. [2001]), that these estimates depend on a scanning of the ground surface (Wu et al. [2004]), and that for judging the straight-ahead direction the ground surface parallel lines are used (Wu et al. [2007]). The latter study also showed that for judging the straight-ahead direction information from the ground plane is more important than that from the ceiling. This might be due to a higher importance of regions below the horizon for estimating the horizon position, which will also be tested in experiment 2. Studies in virtual reality (Messing and Durgin [2005]) and with degraded viewing conditions (Rand et al. [2011]) further underline the importance of horizon for distance estimation.

There are many more cues in literature for estimating the horizon including motion information (Wertheimer [1912], Sedgwick [1986]). The strong regularity of the horizon has also influenced animal and human physiology (Land and Fernald [1992], Cooper et al. [2011]) and biases our saccade direction and more generally the way we take and look at photographs (Foulsham et al. [2008]).

9.3.3. Computer Vision and Graphics

In computer vision the horizon is often used as a hidden variable that is determined through indirect measures and affects the image interpretation. In the work Hoiem et al. [2006, 2008] the horizon height in the image is estimated using converging lines, Gaussian priors, and known real-world heights of objects to estimate the height of detected objects. Converging lines have also been used in Kosecka and Zhang [2002] to estimate the camera orientation with respect to a roughly planar world. In Coughlan and Yuille [2003] as well as Deutscher et al. [2002] the authors follow a similar goal, replacing traditional edge detection with probabilistic models that elegantly pool edge evidence over the image.

Finally, Torralba and Sinha [2001] have proposed to use the GIST descriptor (Oliva and Torralba [2001]) of an image as feature for horizon estimation. They trained a mixture of linear regressors on the GIST feature to guide object detection. This is also used in Lalonde et al. [2007], Sivic et al. [2008] to create image compositions with correct perspective. We use this descriptor in our computational experiment.

9.4. Methods

9.4.1. Ground Truth

In order to evaluate the performance of human participants and algorithms stimuli are required for which the true horizon position in the image is known. There are several ways to obtain such data. Some new camera models have integrated sensors that measure the camera pitch. Unfortunately that information is only visualized in the camera display, but usually not saved in electronic ways. Another possibility is to used rendered images of artificially created 3D scenes as stimulus set. This would allow for very controlled manipulations of the images at the expense of naturalness of the stimuli. Both these methods would provide the true, physical position of the horizon in the scene. We are, however, interested in a purely *visual* estimate of the horizon which might be very uncertain or even different from the physically true position. Given for example a surface with a slight slant and few other cues to hint at the slant, the true horizon is of course still at a well-defined single height in the image which can easily be estimated if one were in the scene. The horizon in the image, however, is not a single well-defined point because it depends on the viewer's estimate of the surface slant. In this case, a slight deviation in the horizon position should be penalized less than in images with many cues to the true horizon position. We therefore decided to use existing, natural images and gather estimates from several participants to approximate a distribution of visual horizon 'estimatability' over the image height in experiment 1, and use this as a reference for the evaluation of participant and algorithm performance in experiments 2 and 3.

9.4.2. Stimuli

We restrict the stimulus set to landscape images of outdoor scenes, since the ecological value for horizon estimates in closer-up shots like portraits and indoor scenes is doubtful, and since horizon estimation in these cases might be hard or impossible. Images were selected from two sources: the very large LabelMe image data base (Russell et al. [2007b][1]) and — since LabelMe has a bias towards man-made environments —

[1]`http://labelme.csail.mit.edu`

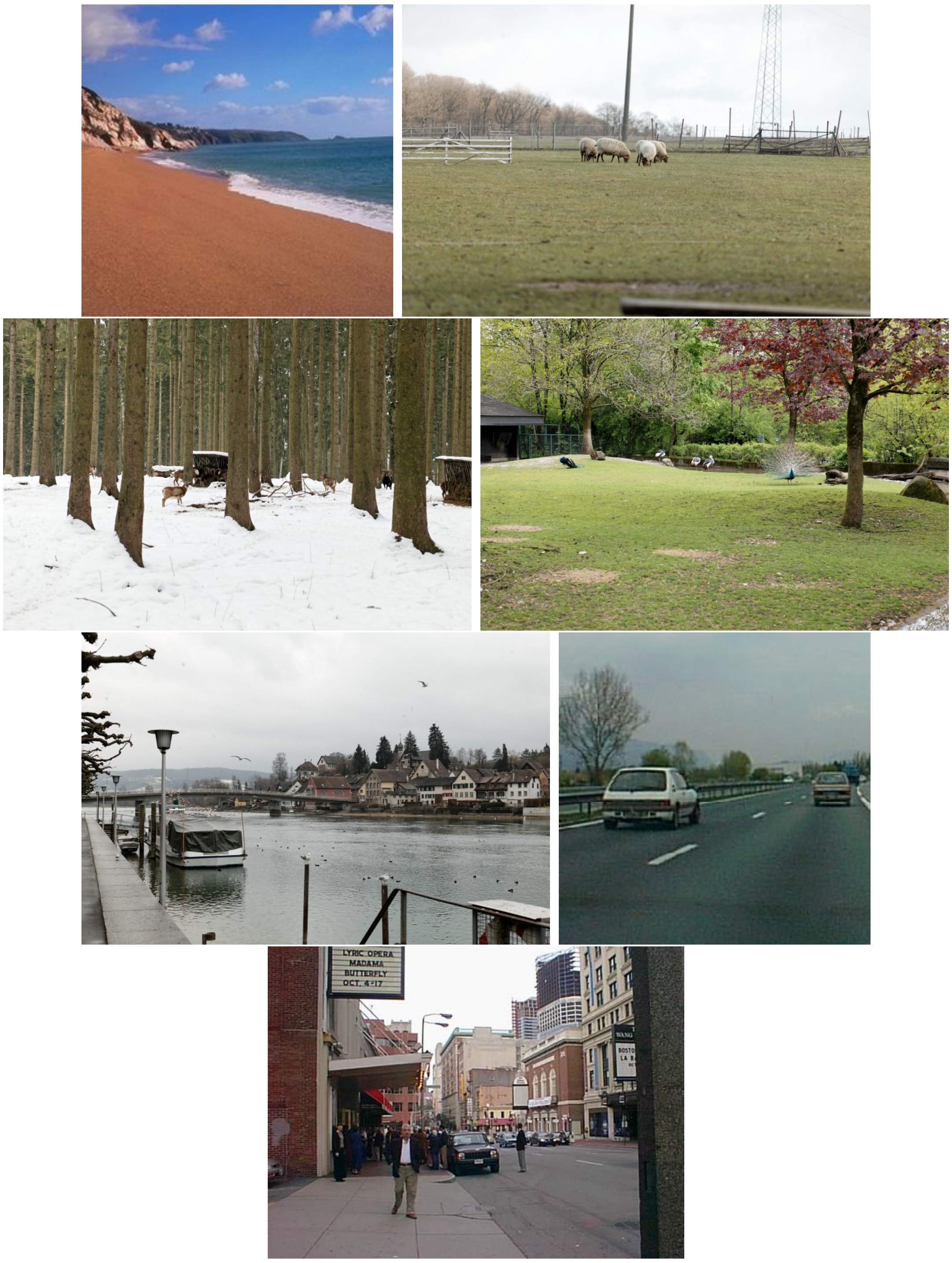


Figure 9.3.: Example stimuli from every scene class. From left to right (roughly from natural to man-made): coast, open country, forest, enclosed natural scene, other, non-urban street, city.

a database of 3645 images from German zoos and wildlife parks[2]. From LabelMe all image folders were considered that predominantly contained images from outdoor scenes (171 in 2008). To avoid further biases in the image selection process, random images from these two sets were chosen with the following restrictions: they had to be free of noticeable camera tilt and provide enough visual queues for horizon estimation. In this subset we labeled and verified the horizon position in random order until 300 images were present with a roughly homogeneous distribution of our (twice verified) horizon labels in the middle third of the image. These are referred to as 'expert' labels later on, they were only used for stimulus selection. The reason for choosing a distribution in the middle third is two-fold. Firstly to avoid biases participants might have caused by avoiding to click close to the upper or lower image border. The second reason is that some of the tested image manipulations would have rendered our expert horizon outside the displayed part of the image in some cases. The stimulus set had to be restricted to 300 images because more images would have made the experiments too long.

To also study the influence of scene type on horizon estimation, one of seven scene types was assigned to each of these 300 images. These scene types were (roughly from natural to man-made): coast (32 scenes, mostly beach), open country (29 wide views of natural landscapes), forest (18 scenes dominated by trees which are quite close), enclosed natural scenes (46 images of open natural regions which are enclosed by forest and/or walls), non-urban street (16 views of streets in mainly natural landscapes), city (47 pictures taken in urban regions), and other (12 images that could not be assigned to any other class). Examples for each of these scene types are shown in Fig. 9.3. For experiment 2 13 images were removed from the stimulus set which were very similar to other images to reduce the number of trials.

To show how important different cues are for human horizon estimation, the images were manipulated corresponding to the following six conditions:

norm the original image as baseline

inverted the image flipped (mirrored at the horizontal axis); this affects the holistic processing of the image leaving the structure itself and local features intact

blurred the image is convolved with a Gaussian filter with stand deviation $\sigma \approx 0.015\times$ image size (for most images this resulted in $\sigma = 10$ pixels); this removes low-frequency information leaving the holistic impression unaffected

lower the lower two thirds of the image are cropped, to study the importance of information from the lower image region for horizon estimation. To avoid participants

[2]`http://images.kyb.tuebingen.mpg.de`

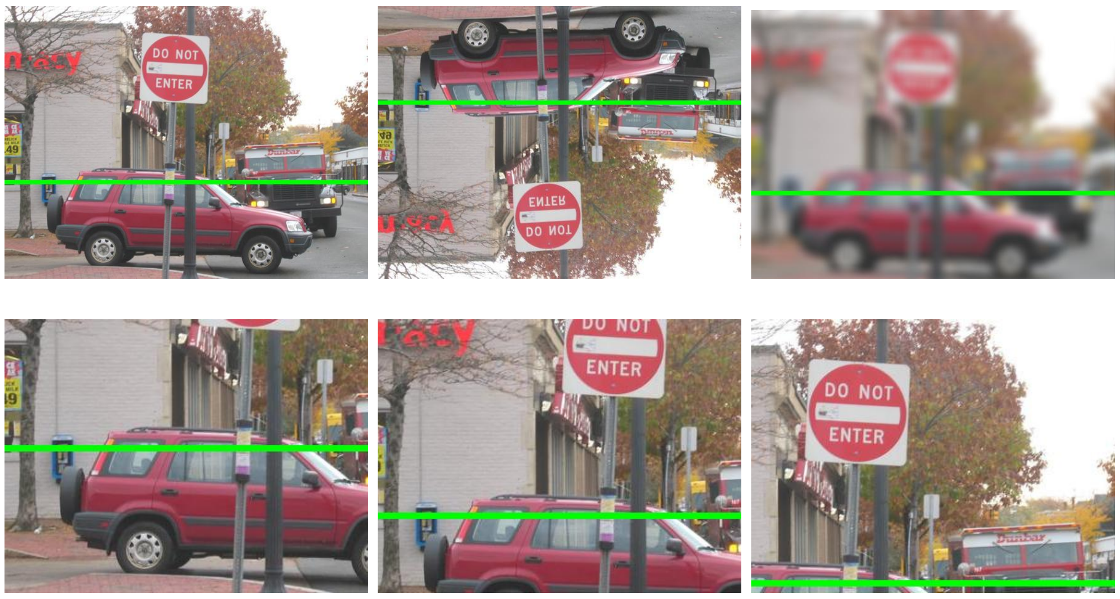


Figure 9.4.: Examples of stimuli in the different conditions. From left to right: normal, inverted, blurred, lower subwindows, middle subwindow, upper subwindow; green bar is the expert estimate.

noticing this manipulation through the change in image size or aspect ratio we removed image rows randomly from the left and right image border until the aspect ratio was the same as the original. The remaining image was shown with a bigger magnification to preserve occupied screen space

middle same as the lower condition except that one sixth of the image from the top and bottom were removed, leaving the middle two thirds of the image

upper the same procedure as for the lower condition, removing the lower third of the image to study the importance of the remaining upper two thirds of the image.

Examples for the six conditions for a single image are shown in Fig. 9.4.

One reason for including the three 'subwindow conditions' (lower, middle, upper) was to diminish the photographer's bias that might favor certain objects and viewpoint compositions and that might give additional cues to horizon estimation. The only image region that is present in both, upper and lower subwindow conditions, is the middle third of the image. This is the second reason mentioned above why stimuli were chosen such that the expert horizon estimate was in the middle third. In experiment 1 participants were only presented with the original images.

9.4.3. Procedure for Psychophysical Experiments

Twelve paid participants (6 male, 6 female, mean age 31.33 years, age std 6.61 years) took part in experiment 1, twenty participants (12 female, 8 male, mean age 28.05 years, std 8.0) took part in the second experiment. They were informed that they would view the appropriate number of images and that they would be asked to estimate the horizon in those.

For experiment 1, participants were placed in front of a laptop at an otherwise empty desk in a regular, empty office. Experiment 2 called for a more controlled setup since the very short presentation time required a high level of concentration. To avoid external distractions or influences of low-level factors like viewing angle on the results we placed participants in a dark room at a fixed distance of 63cm from a flat CRT screen (screen width 39cm, viewing angle $34°$, resolution 1280×1024 pixels, 85Hz refresh rate), with their heads resting on a chin rest fixed in front of the screen roughly at the middle of the image.

Each participant of each experiment was carefully instructed about the horizon definition and cues to its estimation much as described in the Introduction above. Following the theoretical explanations participants were familiarized with the estimation user interface and performed test trials with increasingly difficult images that did not appear in the experiment. The experimenter gave feed-back until participants felt comfortable with task and interface. The experiment was started when participants had understood the task and estimated the horizon with reasonable precision in 10 images without instruction. The user interface used in the experiments was written in Matlab[3] using PsychToolbox 3.0 Kleiner et al. [2007]. Images were presented in the middle of the screen, filling half its height with a 50%-gray background. Each participant was shown all stimuli in randomized order that was different for each participant.

Each trial was started with a 50%-gray screen on which a 'cursor' in form of a horizontal blue line spanning the whole screen width was visible. As a 'start button' a dark gray bar spanning the screen width was shown at a pseudo-random height in the middle third of vertical region of the screen that contained the images. Participants started the trial by moving the blue line, which was controlled by the mouse, into the dark region and clicking left. The blue cursor then disappeared and instead a fixation cross was shown for 400ms, followed by the stimulus. In experiment 1 this was displayed until participants had estimated the horizon position in (or outside of) the visible image region by placing the blue line and clicking the left or right mouse button. In experiment 2 the image

[3]The Mathworks, Inc., Natick, USA

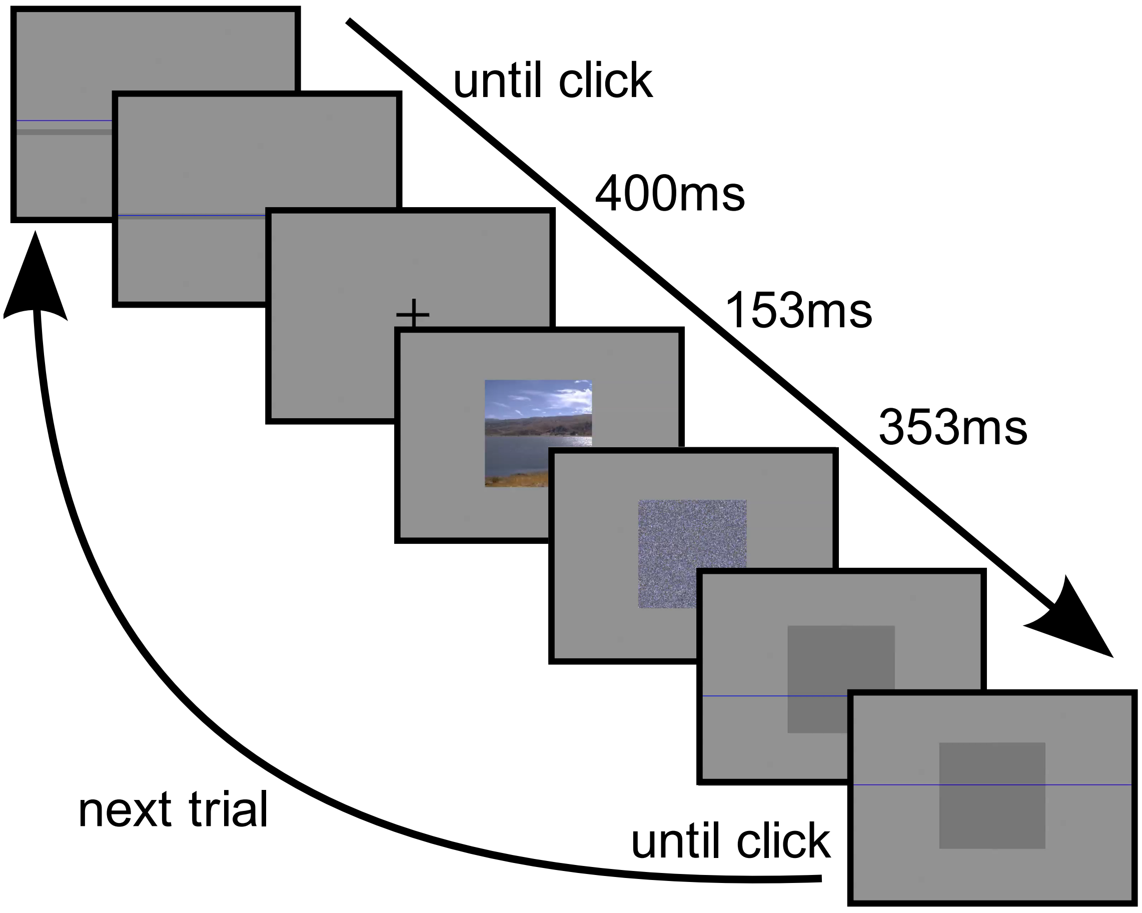


Figure 9.5.: Chain of events in a typical trial. A dark gray bar appears, the participant moves the blue line into it and clicks; then a fixation cross appears for 400ms, followed by the stimulus which stays on for 153ms, and is immediately masked for 353ms. Finally, the mask is replaced by a dark gray rectangle and the blue line re-appears which the participant moves to the position of the estimated horizon and clicks.

was only shown for 153ms and then immediately masked by a pixel-scrambled version of the image for 353ms. Only then the blue cursor line re-appeared allowing participants to estimate the horizon on a dark gray rectangle of the same size and position as the stimulus. After participants clicked the left or right mouse button the dark gray 'start button' bar for the next trial appeared. The vertical position of this bar was chosen to be at a minimum distance of 16 pixels from the estimated position to avoid the feeling of feed-back. We also emphasized that fact in the briefing because in a preliminary experiment some participants had interpreted the bar as feed-back in some cases. The procedure for experiment 2 is visualized in Fig. 9.5.

The short presentation time in experiment 2 made most cognitive strategies to horizon estimation impossible, forcing participants to rely on their first impression or 'gut feeling'. To further enforce this participants were instructed not to try to be too precise but rather emphasize on speed. Participants in experiment 1 were instead instructed to respond as quickly and accurately as possible, with emphasis on precision rather than speed. For both experiments we told participants that if they could not estimate the horizon in the image they should place the horizon line at an arbitrary position and click the right instead of the left mouse button, indicating that this trial should not be included in the evaluation.

We recorded the time from image presentation start until click in seconds, the position where participants clicked and whether participants clicked with the left or right mouse button. Estimated positions were normalized for height of the full upright image, so 0.0 indicates the uppermost image row in the unmodified image, 0.5 the middle of the image and 1.0 the bottom of the original image, already accounting for the condition to make estimates comparable between conditions. Results from trials where participants responded with a right click were discarded from the analysis of positions, but not from the analysis of reaction times.

One other major difference between experiments 1 and 2 was the number of trials. For experiment 1 each participant estimated the horizon in each image of the original 300 images. For experiment 2 each participant estimated the horizon of every image in the slightly reduced stimulus set, for every condition, resulting in 287×6 trials. To avoid fatigue we divided these into four blocks of 430 trials each and encouraged participants to pause for at least five minutes between blocks. Care was taken that there were at least 4 other images presented between the presentation of the same image in different conditions. Participants were also told that they could make minor breaks between (not within!) trials if they noticed a decrease in concentration.

As a preprocessing step for experiment 1 we removed outlier positions (not reaction times) from the analysis to avoid noise from manual errors in our ground truth. For this we calculated for each image the standard deviation of participant estimates and removed all estimates that were more than three standard deviations away from the participant mean of the corresponding image. There were 16 such cases.

For experiment 2, we did not exclude outliers. However, early analyses showed that two participants had not understood our instructions about the inverted condition. Their results were inconsistent with estimates in the inverted image. Instead their results for the inverted condition showed a remarkably good performance if they were applied to the upright images, suggesting that those two participants had performed a mental rotation or flipping of the image and clicked at the position where in the then upright image the horizon would be. Since we cannot predict what other influences this process might have had we excluded these two participants' results completely from the analysis.

Participants of both experiments filled in a questionnaire after the experiment.

9.4.4. Computational Experiments

We designed several algorithms that exploit different cues to horizon estimation in images, including those mentioned in the perception literature (e.g., Sedgwick [1986]). Each of these algorithms gets a single image channel (e.g., the luminance) as input and returns for every image row a confidence value for that row containing the horizon. We chose a simple winner-takes-all framework to deduce from these confidences the algorithm's estimate. The only exception from this is the algorithm **[gst]** which does not give a confidence for each image line but directly returns the estimated horizon position. Each algorithm's parameters were optimized using excessive random search on a separate training set.

The algorithms are the following:

[div] is a very simple algorithm that uses a mixture of global and local features. It tests for each line how well it divides the image into a light region (i.e., containing high values) above and a dark region (i.e., containing lower values) below it by calculating the concentration of high values above and of low values below the given line. This is inspired by the fact that at day time the sky is lighter than the ground, so if applied to the luminance channel of an image this should yield the best separation between sky and ground. To allow for a more precise localization a weighted average of the (global) lightness concentration difference with the local vertical gradient at that line is calculated (after light smoothing of the image). A linear weighting of global and local features then yields an estimate for each line. Interestingly, parameter optimization resulted in a weighting that strongly biased the algorithm towards the local information.

[-div] is the same algorithm as **[div]** except that light and dark are exchanged. The assumption that the region above the horizon is lighter than below might not be true for images taken at night time and especially might only hold for the luminance of an image, not for color channels.

[lin] works only on a local scale. It looks for horizontal lines in the image by calculating the vertical gradient in the image and summing its absolute value over image lines, assuming that around the horizon the differences between neighboring lines is greatest.

[gab] hinges on the same idea as **[lin]**: that around the horizon there should be the most notable vertical gradient. This algorithm, however, takes a more biologically motivated approach to finding these gradients, namely by convolving the image with a horizontal gabor filter which has been proposed as a good model for line-sensitive cells in L1 of the visual cortex. The gabor filter used is aligned horizontally with optimized wavelength, bandwidth and selectivity. After filtering in Matlab using code by Peter Kovesi[4], it sums for every line the magnitude of the response per pixel.

[van] uses the perspective cue most frequently proposed in the perception literature. It tries to estimate the vanishing point by finding points where many lines converge. In order to do this the Canny edge detection filter is applied to the image, whose output is Hough transformed to identify the corresponding orientations and 'offsets' (distance of a line from the lower left image corner) associated with these. All resulting lines are plotted into a new, empty image, resulting in an image with high values at point where many lines meet. To emphasize a single strong peak in a line as opposed to several minor ones, we do not sum over bin counts x, but instead over values $\exp(x)$ per line, resulting in the returned confidences.

[gst] is the only algorithm we could find in the computer vision literature, that was explicitly designed to estimate the horizon of an image. It has been proposed by Torralba and colleagues in their work on scene understanding (Torralba and Sinha [2001]). It uses the *spatial envelope* as feature, the response of a set of filters tuned to different orientations and scales, which has been proposed as a psychophysically-motivated feature for scene classification. The algorithm does not calculate a confidence for each image line but instead returns a single estimate from a mixture of linear regressors which are trained on example images and corresponding horizon labels using an expectation-maximiation procedure. We re-trained these regressors using our stimulus set and mean estimates in a cross-validation procedure. We also extended the algorithm to use all color channels and not just luminance. Therefore this algorithm is the only one applied to not just one color channel like the other algorithms but uses information from all channels simultaneously.

[dum1] and **[dum2]** are algorithms included to give a lower bound on algorithm performance. They 'estimate' the horizon by random guessing, ignoring the image information completely. **[dum1]** draws estimates from a uniform distribution over an interval $[L, R]$ with $0 \leq L < R \leq 1$ while **[dum2]** estimates from a normal distribution with mean and standard deviation $\mu, \sigma \in [0, 1]$.

[exp2] is included in evaluation plots to compare computational results on test images with those obtained from human subjects. Note that results may vary from those reported of experiment 2 because of the choice of images in the test set (see below).

[4]function `spatialgabor`, available at `http://www.csse.uwa.edu.au/pk`

Algorithms **[gab]**, **[van]** and **[gst]** were implemented in Matlab, all others in python. Evaluation and experiments were done using ipython[5]), using the packages numpy and scipy[6], as well as mlabwrap[7], matplotlib[8] and OpenCV[9]. All images in all conditions were transformed into the CIE L*a*b* color space (Wyszecki and Stiles [2000]) and algorithms were applied to each of the resulting channels independently.

All algorithms have parameters that need to be optimized. To make the best use of the limited set of results for 300 images, we adopted a 10-fold cross validation procedure for training and testing our algorithms. Images are divided into 10 sets of 30 images, trying to keep roughly equal ratios of scene classes and mean ground truth densities in all subsets. For each of the 8 algorithms, 3 color channels and each of the 10 image subsets S_i parameters are searched that best reproduce the human results on the 9 image sets $S_j, j \neq i$ (their 'training sets') in normal condition. This results in 10 optimal parameter sets for each algorithm and color channel. As criterion for reproduction of human results we chose the value of the ground truth density derived from experiment 1 at the image line with the highest horizon confidence. The parameter search was done using first an extensive random search in parameter space followed by a refinement around selected maxima using the Nelder-Mead simplex algorithm (`scipy.optimize.fmin`).

As exceptions to this procedure, algorithm **[gst]** was not traing on individual color channels but on the full L*a*b* images. Since evaluation of **[gst]**'s performance on a given image set for certain parameter involves training a regressors, a 3-fold cross validation for each parameter evaluation of **[gst]** had to be performed during the parameter search.

The performance of the final regressor — trained on all 9 training subsets with optimal parameters found on them — was averaged over all 10 folds and is reported in the following as training performance. Algorithms **[dum1]** and **[dum2]** were not trained on individual color channels since they do not 'look' at the image at all. Instead confidences were averaged over 10 runs of **[dum1]** and **[dum2]** to account for the randomness.

To finally evaluate the algorithms' performance, we report in the Results section the performance of algorithms with optimal parameters on the left-out subsets which together make up all the original 300 images. For example, results on S_1 are computed with parameters searched using $S_2, \ldots, S_{10}$. We also ran algorithms with optimal parameters on their respective training sets in other conditions.

[5]ipyton 0.9.1 (Pérez and Granger), python 2.6.2
[6]numpy 1.5.1, scipy 0.9.1, `http://scipy.org`
[7]mlabwrap version 1.1, `http://mlabwrap.sourceforge.net`
[8]matplotlib version 1.0, Hunter [2007]
[9]OpenCV version 2.1,`http://opencv.willowgarage.com/`

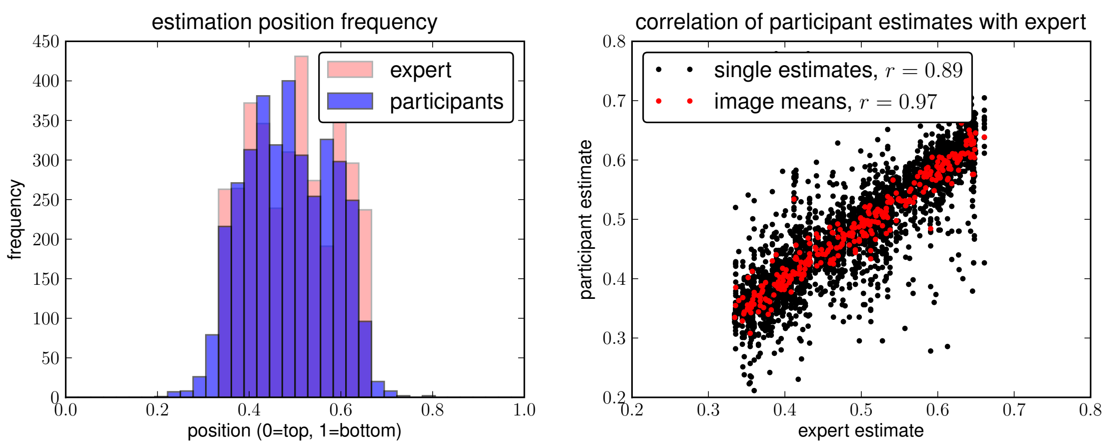


Figure 9.6.: Agreement of results for experiment 1 with expert labels. **Left**: Histogram of relative image positions participants clicked at compared to the distribution of expert estimates by the authors. **Right**: correlation of indivual subjects' estimates (black) and their mean (red) with the expert estimate. Both individual estimates and estimate mean show a very high correlations with the expert labels.

9.5. Results

9.5.1. Ground Truth Experiment

The distribution of estimates from experiment 1 agreed well (correlation $r = 0.89$, c.f. Fig. 9.6 left) with our expert estimates that were used to select the stimulus set. Estimates show a roughly homogeneous distribution of horizons in the middle third as can be seen from the histogram in Fig. 9.6 (right).

Participants were told to click with the right instead of the left mouse button in case they could not estimate the horizon in the image. This happened only twice in all of the first experiment, which is within the expected noise caused by manual errors.

The agreement with our expert estimates and the low number of right-click results leads us to believe that the task was sufficiently clear and intuitive.

Estimates agreed not only with our expert estimates but also with another. The standard deviation between participants for each image was 3.44% image height on average (3.66 before removing outliers as described in the Methods section). This deviation between participants varied greatly with scene type. Fig. 9.7 (left) shows that agreement on open coastal images is generally much higher (std of 2% image height) than on more closed scene types like open country, forest or closed nature (all close to 4% image height). Street scenes within cities (class city) and outside urban areas (class non-urban street) resulted in standard deviations of approximately 3% of the images' heights. We conclude

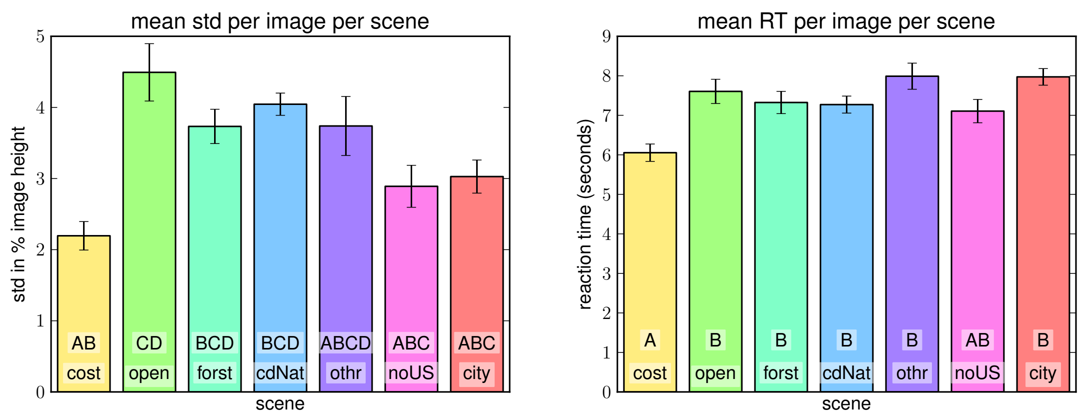


Figure 9.7.: Reaction time and consistency of experiment 1 results with respect to scene type. **Left**: Standard deviation of participant results per image in the ground truth experiment, shown as average per scene type. Results of pairwise comparisons with the Scheffé criterion are indicated by letters A–D: first row denotes group, second bar shows letter of other groups, to which this group is not significantly different; "other" is not significantly different from any group. **Right**: mean reaction time of participants per image in the ground truth experiment, shown as average per scene type. Error bars denote standard error. Bars with letters A–B are significantly different from another. The class "non-urban streets" (noUS) is not significantly different from any other class.

that humans are able to estimate the horizon from a monocular image with reasonable accuracy, at least if given enough time.

The reaction time results show a much more homogeneous distribution with respect to scene type (Fig. 9.7 right). Participants in experiment 1 took 7.32s on average to estimate the horizon. The most notable difference is again caused by the relatively easy images from the coast class (c.f. Fig. 9.2).

9.5.1.1. Density Estimates

The 12 estimates for each image (11 estimates for 2 cases of right-click images) are used in experiments 2 and 3 as ground truth distributions for the visual horizon. We chose this measure because it captures not only the position of the horizon but also the variability in estimates caused by the absence of non-visual cues. To measure how well other estimates agree with these estimates, we fitted a mixture of Gaussians to these 'ground truth estimates' and report the value of the resulting probability density function at the position of the estimate to evaluate. The method for fitting a distribution to these estimates was `scipy.stats.gaussian_kde`. For a given image $i \in 1, 2, \ldots, 300$ it approximates the unknown distribution from which samples (estimates) E_i are drawn, by replacing each estimate $e_{i,j} \in E_i$ by a Gaussian distribution centered at $e_{i,j}$ with a deviation given by the covariance of E_i. This yields a distribution with density function (Härdle et al. [2004])

$$p_i(x) = \frac{1}{\text{const}_i} \sum_{e_{i,j} \in E_i} \exp\left(-\frac{1}{2}(x - e_{i,j})^T \left(f_i^2 \operatorname{cov}(E_i)\right)^{-1} (x - e_{i,j})\right) \quad ,$$

$$\text{where} \qquad f_i = |E_i|^{-1/(d+4)}$$

is Scott's factor (Scott [1992]), const$_i$ is a normalization constant, $|E_i|$ the number of estimates in E_i (typically 12), and $d = 1$ is the dimensionality of estimates. We call an evaluation of p_i at a point $x_0 \in [0, 1]$ a *confidence* and the density p_i from which it was taken a *ground truth density* in the following. Since the ground truth densities have an integral of 1 but are close to 0 over most of the interval $[0, 1]$ on which they are defined, point evaluations $p_i(x_0)$ will often give values much larger than 1. Examples of such density functions are plotted in black on the left side of example stimuli in Fig. 9.8.

9.5.2. Psychophysical Experiment

Although 13 similar images were removed from the experiment and despite very quick reaction times, the experiment took 2 hours 6 minutes on average (std 28.6 minutes) not counting time for breaks.

In this subsection results for experiment 2 are reported: Reaction times, confidence based on ground truth estimates from experiment 1, inter-rater correlation and mean position per condition.

The main measure of performance is the confidence as defined above: we evaluated the ground truth density obtained from experiment 1 estimates at the estimated horizon positions in experiment 2, resulting in a confidence which measures how well an estimate can be explained by the ground truth estimates from experiment 1.

The mean reaction time per trial was 1.74s (std 1.56s). This is significantly faster than replies in experiment 1 (7.32s), showing that participants followed a more intuitive and less cognitive strategy. This faster reaction time is not entirely explainable by time required for purely low-level visual processing, motion planning and execution. Although higher-level visual processing was severely limited by the masking paradigm, participants seem to still have processed the visual information to make their estimate.

Memory might have played a role in two ways in this task. First, participants had to retain in memory either the horizon estimate they had already formed or some visual

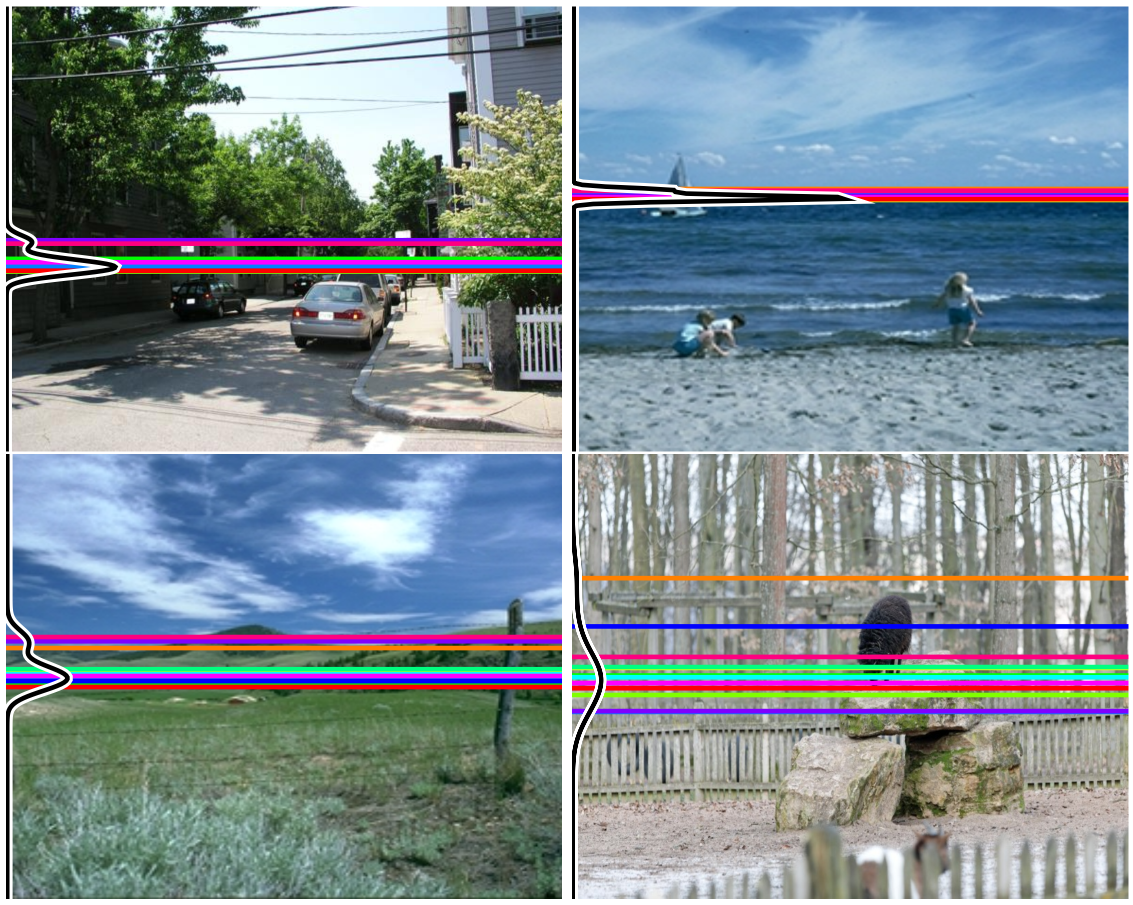


Figure 9.8.: Examples of stimuli and results from experiment 1. Individual participant estimates are shown as colored lines, the black curves on the left of each image show the estimated estimate density function. Note how the variance of estimates varies across images.

information of the image to base an estimate on. We can not completely rule out that enough higher-level visual information has been retained to employ more cognitive strategies to the task. On a longer time scale, a few participants reported having realized that they had seen images repeatedly in different conditions. In those cases, recognition and retrieval of memory of earlier estimates could have had an influence on reaction time and estimated position.

Correlating the distance the mouse was moved with reaction times showed that the time for mouse motion was not an important factor in reaction times.

9.5.2.1. Subjective Confidence

The first question we were interested in was, whether participants felt they could perform the experiment i.e., whether it is feasible to estimate the horizon from such a short presentation time. This seems to be the case as only 2.8% of all replies were a 'do-not-know' indicated by using the right instead of the left mouse button to reply. Another indicator is the reply participants gave to our questions in the debriefing questionnaire. We asked them how certain they were about their estimates on a scale from 1 (very uncertain) to 10 (very certain), which resulted in a mean response of 6.92 with a std of 1.36. We also asked how difficult the experiment was (1 easy, 10 hard), for which we got 5.33 as mean reply (std 2.27).

9.5.2.2. Reaction Time per Condition

As shown in Fig. 9.9 (top left), the reaction time per condition shows an interesting effect: One would expect the fastest reaction time for the normal condition but participants replied significantly quicker for the blurred version of our images (Scheffé test with $\alpha = 0.01$). This may be explained by the lack of detail in those images that made attempts at careful and exact and therefore slower placement of the cursor impossible. Participants may therefore have felt less need to invest time for more precise cursor placement. This might have led to worse estimation results as shown in Fig. 9.9 (top right). Leaving out image information from top and/or bottom of the image, as well as image inversion, also slowed down participants significantly.

9.5.2.3. Confidence per Condition

Regarding confidence per condition (c.f. Fig. 9.9 top right), the worst performance for this measure is obtained from inverted and blurred images, which affect holistic processing in

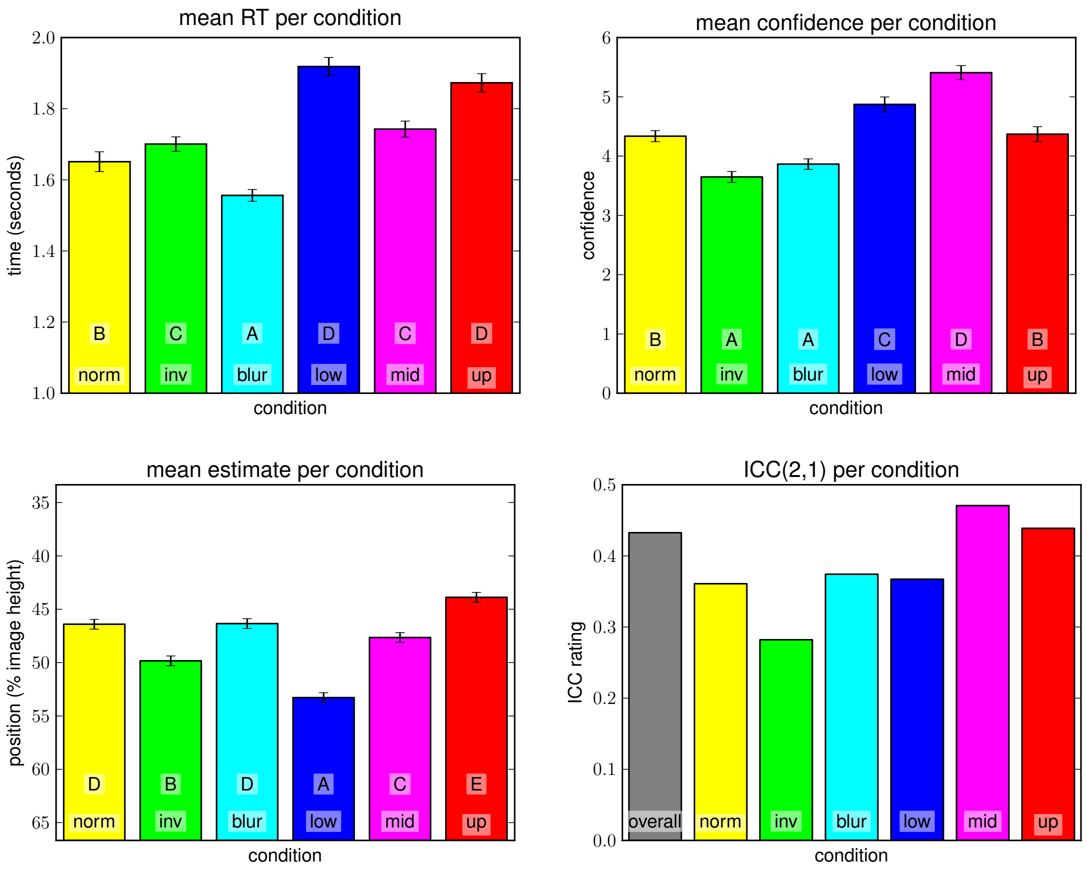


Figure 9.9.: Results of experiment 2 per condition. **Top left**: Mean reaction time per condition. **Top right**: Confidence assigned to estimates per condition. **Bottom left**: Average position of estimates per condition, normalized to 0=image top, 1=image bottom. Although images were the same, positions vary significantly. **Bottom right**: Agreement of subject ratings as intraclass correlations ICC(2,1) per condition. In all plots, error bars denote standard errors. Bars with different letters A–E are significantly different according to a post-hoc comparison using the Scheffé criterion.

the first and high-frequency detail in the second case. Both these sources of information seem to be relevant to the task of visual horizon estimation. Again, the normal condition is not the one with the best performance. The lower and especially middle subwindow condition led to estimates that agree better with results from experiment 1. One possible explanation for this concerns all three subwindow conditions: participants saw less of the image, so there were less possible regions to falsely estimate the horizon at. The region around the horizon, however, filled a larger range of the displayed stimulus, thus offering more detail of the region around the horizon. A middle bias of participants would explain the performance increase in the middle condition but should also result in a smaller performance increase in lower and upper subwindow conditions, rendering these two the same level between the normal and the middle subwindow conditions. This is not the case, so this can not be the only explanation. We would rather suggest that information near the horizon, especially just below it, is most important for estimation. The fact, that the performance in the lower subwindow condition is much better than in the upper subwindow condition might hint at the ground dominance which has also been reported in the literature (Wu et al. [2007]).

9.5.2.4. Position per Condition

Although every image was shown in every condition and although participant responses were normalized to positions in the full upright image, tests showed a significant difference between mean estimation position over different conditions (see Fig. 9.9, bottom left). Participants estimated the horizon in a lower position for the lower subwindows condition than in the middle subwindow condition, and even higher in the upper subwindow condition. There are at least two possible explanations for this behavior: Participants may have had a bias toward the middle of the visible part of the stimulus. In the lower subwindow condition, the middle of the visible part of the image is not at height 0.5 but at 0.667 of the full image. Also, the horizon estimation process could include a rivalry between a horizon estimated from the upper part of the image (a 'sky-based' estimate) and another estimated based on the lower image part (a 'ground-based' estimate), e.g., the lowest visible sky-pixel-position versus the highest visible ground-pixel-location. Removing information from the upper image part may result in a less confident estimate of the horizon from the upper image part and therefore to an estimate that tends more towards a 'ground-based' estimate. The overall tendency in the experiment to a horizon estimate slightly above the middle of the image might be due to the stimulus set. However, there might also be a tendency of participants to click a bit too far up in the images, which could partly explain why the mean estimate in the inverted condition is lower. However, if this was the only influencing factor, the inverted condition would have to be exactly as much below 0.5, as normal and blur are above 0.5.

9.5.2.5. ICC per Condition

The final measure to evaluate agreement of participants uses the intraclass correlation coefficient `ICC(2,1)` (Shrout and Fleiss [1979]), which measures the reliability of the ratings of several judges that all judge the same set of items. To compute these quantities we had to omit images in conditions for which at least one of the participants replied by a left click. If trials from all conditions are treated as independent items to rate, the overall reliability of participants is ICC$(2,1) = 0.432605$ (based on 1231 trials). The overall reliability of the resulting ratings is ICC$(2,k) = 0.932083$. Reliability of raters per conditions are shown in Fig. 9.9. Reliability for the inverted condition is much smaller than for the rest, while the middle subwindow condition show the highest reliability. Interestingly, here the upper subwindow conditions caused better participant performance than the lower subwindow condition.

9.5.2.6. Results per Scene Type

As expected, participants responded quickest and most accurate for coastal images (c.f. Fig. 9.10). The next easiest class seems to be non-urban street scenes (labeled 'noUS' in the figure). Images of this scene type often offer wide views as well as perspective queues from the streets themselves or objects like cars or small houses. City and open country images result in the highest reaction times but not the worst confidences. This might be a hint at cognitive strategies being used more in these cases. Closed natural scene like forest and closed nature yield the worst confidences with respect to scene type despite relatively long processing.

Interestingly, the pattern of reaction time per scene in experiments 1 and 2 are very similar. Some degree of anti-correlation between standard deviation of estimates from experiment 1 and confidence of estimates in experiment 2 is to be expected: if the standard deviation in experiment 1 is high, then the fitted distribution will attain smaller values distributed over a bigger region of the image. An estimate in experiment 2 that completely agrees with an estimate of experiment 1 will therefore be assigned a lower confidence than would be the case if the standard deviation in experiment 1 had been low. A relatively bad estimate, on the other hand, has a better chance to get some confidence in more diverse images, but in these cases the confidence is low enough to not make much difference in the mean. An interesting case in this respect is the open

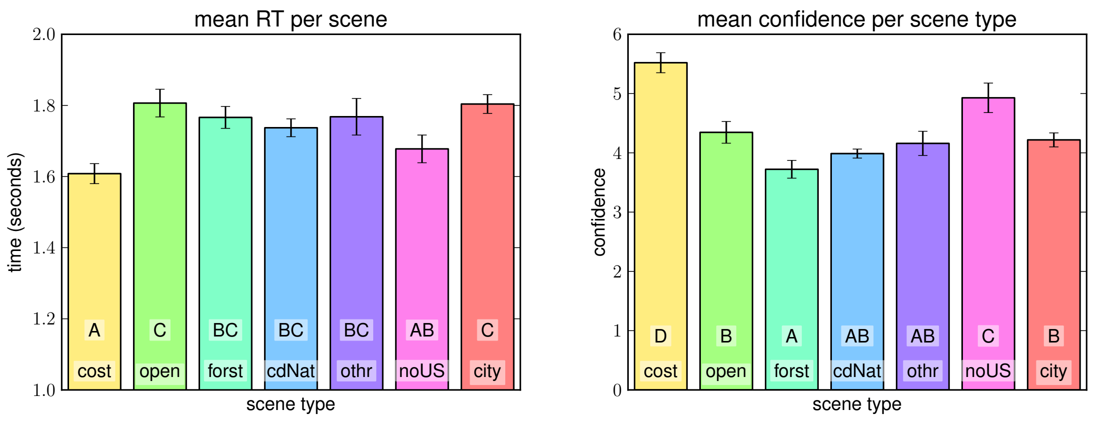


Figure 9.10.: Results from experiment 2 per scene type. **Left**: Mean reaction time per scene type. **Right**: Mean confidence per scene type. In all plots error bars denote standard error; bars containing letters A–D are significantly different from all bars that contain none of the same.

country class. According to experiment 1 it is the scene class for which fixing a ground truth horizon is hardest. Still, in terms of confidence in experiment 2 it outranks the classes forest, closed nature and other.

Overall we conclude from the subjective participant confidence, the overall high agreement of estimates with the ground truth estimates from experiment 1 and the high agreement between participants, that *horizon estimation is possible from simple visual processing* and that the horizon or related information could therefore well be part of the gist.

9.5.3. Computational Experiments

In this subsection, some of the computational results are described that have been obtained by applying the simple horizon estimation algorithms described above to luminance, a* or b* color channels of the stimulus set. Evaluation proceeds first like in the psychophysical experiment by evaluating the ground truth density from experiment 1 at the estimated horizon position. This allows a comparison with human results (shown in black). To test whether algorithms 'behave' like the human participants, correlation coefficients of algorithm results with respect to condition and scene type are also computed as described below

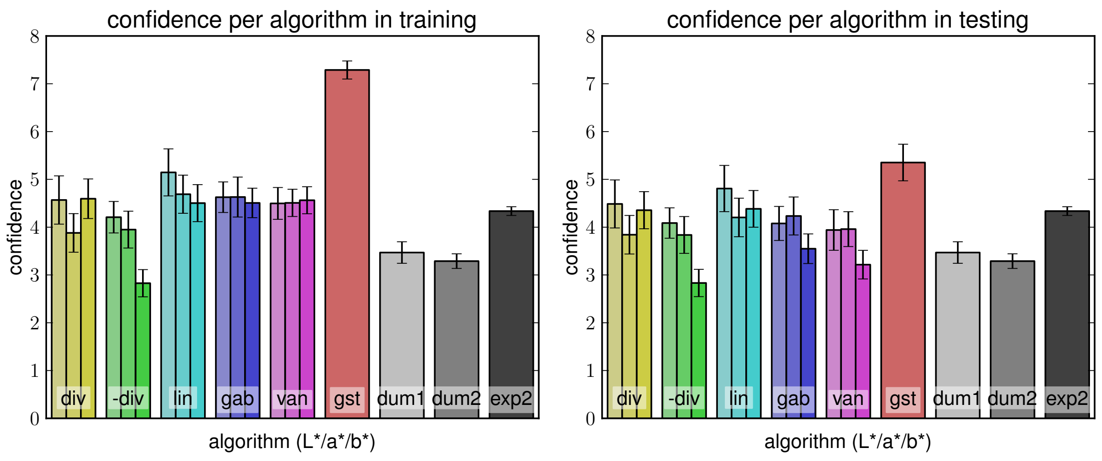


Figure 9.11.: Algorithm confidence in training and testing. Mean confidence of algorithm results for normal condition with respect to densities from ground truth experiment in training (left) and testing (right); error bars denote standard error.

9.5.3.1. Confidence

The first measure for evaluating the algorithms' performance is the same as used in experiment 2: confidence assigned by calculating the agreement of the estimate with those from experiment 1. We first tested whether algorithms were able to capture the information contained in the training set during parameter optimization. When evaluated on the training set (left plot in Fig. 9.11) one can see that most algorithms managed to achieve the human performance from experiment 2. The **[gst]** algorithm clearly stands out which might be due to the high number of parameters that allow a good fit to data. This bears, however, the risk of over-fitting. All algorithms reach human performance on the training set except for **[-div]**, for which one can therefore not expect good performance on the test set, either.

Testing algorithms in the normal condition (right plot of Fig. 9.11), the advantage of **[gst]** diminishes, but still leaves it the best-performing algorithm considered here. However, it its agreement with the ground-truth-densities from experiment 1 is nearly matched by **[lin]** and **[div]** for the luminance channel. In general, the luminance information seems to be more informative for horizon estimation algorithms than color information. When compared to human results the luminance channel results reach similar performance. Results for the b channel often only work as well as intelligent guessing.

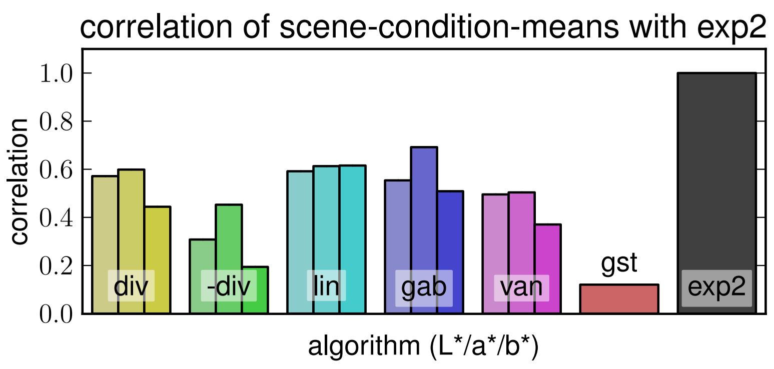


Figure 9.12.: Correlation of algorithms' 'behavior' with human results. Bar height shows the correlation of pattern of confidence means per condition and scene type with human results from experiment 2.

9.5.3.2. Correlation of Behavior

Finally, the algorithms' 'behavior' i.e., pattern of mean confidences per condition and scene type, was correlated with human results. Since the ground truth experiment did not provide results per condition the mean confidence of algorithm was correlated with the mean confidence assigned to the human estimates in experiment 2 (i.e. with the corresponding results in **[exp2]**). The results are shown in Fig. 9.12. **[gst]** shows remarkably bad correlation with results of participants in the quick experiment, although its performance when compared to experiment 1 ratings was highest. It seems to 'behave' more like a carefully thinking human than a person that has seen the scene only very quickly. **[lin]** shows not only very good confidence but also the best correlation here, while **[-div]** produces low values in both tests. Over all algorithms, the a* channels slightly outperform L* and b* information, with a* of **[gab]** exhibiting the best correlation in this test.

9.6. Discussion

Estimation of the viewer's orientation with respect to a viewed scene is an important task in human visual perception as well as computer vision. Although many studies in both fields assume some viewpoint knowledge in the processing system, it has not been explicitly tested how and how fast this knowledge is acquired. Literature on early vision suggests that simple viewpoint information like the horizon may be part of the gist, the representation of a visual stimulus that forms in the viewers mind within a few hundred milliseconds.

In this chapter, three experiments were presented that address the ability to estimate the horizon in human and machine. Humans are well able to estimate the horizon from purely visual input in a consistent manner. Performance depends on the type of image displayed, with coastal scenes being the easiest to interpret. When forced to rely on only the first visual impression from a masked presentation of only 153ms, performance drops slightly but still shows remarkable agreement with estimates from longer viewing times. We conclude therefore that viewpoint information is part of the first visual impression of an image.

By manipulating possible feature channels in the shown stimuli, and by comparing human estimation results with those created with simple, single-feature image processing algorithms, we addressed the question what information is used to estimate the horizon. Results from the psychophysical experiments suggest that high-frequency information plays no crucial role in estimation and that image information close to the horizon is more important than that in higher or lower image regions. Estimates based on information below the horizon show significantly higher agreement with estimates based on longer viewing times which is consistent with the prediction in the literature (Wu et al. [2004]). However, agreement between participants is higher for estimates based on the upper two thirds of the image, than those based on the lower two thirds. It seems, therefore, that both global and local processing are necessary for horizon estimation.

Computational studies show a similar trend: the **[gst]** algorithm, which estimates the horizon from a holistic representation of the image, exhibits the best agreement with human estimates from longer viewing times. However, its pattern of 'behavior' with respect to changes in scene type and stimulus condition is least correlated with human results from short presentation times. Simple horizontal gradients perform relatively well in both evaluation criteria, consistent with simple local visual processes.

CHAPTER 10

Summary

In this final chapter, the key aspects of this thesis are summarized and discussed. Furthermore, promising directions for future work are highlighted.

The aim of this work was the analysis of new data-driven representations for computer vision applications. Three new representations were presented, together with learning-based methods to extract them from visual input. Applications for self-motion estimation, viewpoint estimation, and dynamic object state filtering show that with machine learning and appropriate data sets, new representations can be formed that can further help in solving various computer vision tasks.

Optical Flow Subspace

The first representation is a decomposition of optical flow fields into a linear combination of basis flow fields and an offset, based on probabilistic PCA. An additional noise term in the model provides a natural way to deal with missing values and outliers due to object motion, flow measurement errors or violations of model assumptions. The result is a dense, regularized representation of the flow field, consisting of a field of inlier/outlier probabilities and a few subspace coefficients that define the contributions of each basis flow field to the overall approximation of the observed field. An algorithm for learning in an unsupervised manner the basis flow fields, flow offset, and all variances involved has been suggested with this model in Roberts et al. [2009].

In this work, this representation has been thoroughly analyzed and extended in several ways. Update equations for the noise variance and a different variance term for each flow vector component were derived to extend the robustness of the model and allow for a better fit to observed flow fields. A further extension is the combination of several such models in a mixture of experts system, which further extends the applicability of this method.

This flow subspace representation has been applied to the task of monocular self-motion estimation in Roberts et al. [2009], extending the insight from Irani [1999] that observer motion is linearly related to the observed flow field. Its extended version was used in this thesis in the same way, applied on a much larger data set captured by a vehicle driving through a busy urban area. A large amount of independent motion from cars and pedestrians, varying lighting conditions and traffic scenarios make monocular self-motion estimation on this data set challenging. Nevertheless, the approach exhibits good estimation results that approach the performance of a state-of-the-art stereo odometry approach.

Compared to monocular visual odometry approaches that mostly use longer tracks of sparse features and Ransac-like outlier rejection schemes to identify outliers, this approach has the advantage of being much more computational efficient, calibration-free and applicable to arbitrary constant camera setups. The probabilistic nature of the approach not only enables the natural dismissal of outliers; it also offers a confidence measure of the resulting estimate.

Further analysis has shown that changes in yaw and forward motion are mostly captured in the first two subspace dimensions, but pitch and roll can also be accurately estimated when using more subspace dimensions. The proposed extension with inhomogeneous inlier variance further improves results, while non-linear motion mappings have not proven beneficial. A greater number of experts only leads to improvement if their subspaces are trained with homogeneous inlier variance, possibly due to a higher basis level of performance or insufficient diversity in geometrical configurations in the data set.

Multi-Modal Circular Viewpoint

The second representation that was investigated in this thesis, is a multi-modal distribution of object viewpoint. Used mostly as helper for multi-view object detection, viewpoint is often represented in an implicit way or as a discrete variable, selecting one of a few viewpoint classes like 'left frontal' or 'back view'. Quite a few reports of confusions of e.g. frontal and rear views of cars can be found in the literature, giving rise to ambiguous viewpoint estimates, and were also found in a feature space analysis in this work. Using flexible methods like random regression forests and kernel density estimation on the 1-dimensional circular space, such ambiguities can be properly represented.

This is the basis of a later disambiguation using temporal context of either object viewpoint alone or object viewpoint and position. To this end, not only random regression forests and kernel density estimation, but also mean shift clustering and particle filtering have been extended to mixed linear and circular spaces. While this is not the first report of application of these methods to circular spaces, all derivations were repeated independently from the literature and combined in a unique way.

For estimation of object viewpoint, we build on previous work in multi-view object detection. The input to the random regression forest is the representation built by a popular object detector, that has been used for various multi-view detection studies.

Evaluation was performed on the new Kitti data set, which provides even more image data and ground truth information for a larger variety of scenarios than the data set used before. Slightly impaired by the inhomogeneity of the training data, the approach still delivers competitive performance when compared to other image-based viewpoint estimation approaches.

Fusing object viewpoint estimates with monocular self-motion estimates in a particle filtering framework for self-motion compensated dynamic object state estimation shows that viewpoint estimates are beneficial for predicting the future state of other objects.

General-Purpose Representation of Static Images

Motivated by research on the early stages in human visual processing, the third representation proposed in this thesis is a holistic general-purpose interpretation of a single, static image. This 'gist' of a scene forms within a few hundred milliseconds in human visual perception, a consistent combination of prominent objects, scene layout, viewpoint and scene type. Following the trend to combine algorithms for related vision tasks, an iterative algorithm was proposed that uses prior knowledge and results from an object detector, surface type classifier, and scene type estimator. From individual algorithms' results it takes those that are consistent with other estimates, and seeks to arrive at a stable configuration of consistent estimates. This artificial equivalent of the human 'gist' could have been used as an improvement of individual algorithms, e.g., by lowering the false positive rate of an object detector, but also as a initialization for more involved vision tasks.

Unfortunately, the proposed approach to build this representation proved to be inadequate. The suggested framework of picking the most consistent single candidate estimate and then re-evaluating the confidence for all other candidates often failed to converge to a correct interpretation. After several promising attempts but no stable results, this project was therefore discontinued.

More rewarding was a study that aimed at clarifying the presence and formation of horizon in the human gist. Participants were able to adequately estimate the horizon position after presentation times of only 150ms, proving the presence of horizon information in these early representations. Varying the presence of several cues to horizon estimation and analyzing results per scene type showed that low-frequency information is more important than high-frequency image features, although an algorithm using only local horizontal gradients showed remarkable performance and correlation with human behavior.

Future Work

Many ideas for future work have been discussed during this thesis, most of which have been mentioned in the final sections of the corresponding chapters. In the following, some of the most interesting and promising avenues are summarized.

No measures were taken to avoid drift in yaw and speed estimates. Three possibilities to avoid drift in integration of motion estimates were listed in sec. 3.9. By considering more than two frames for a motion estimate this approach will probably allow for more accurate trajectories. Incorporating estimates of absolute pitch and roll from flow fields, vehicle orientation could be kept stable without fusion with other sensors, which is necessary for e.g. calculating correct projections.

The computational simplicity and flexibility in the choice of cheap optical sensors make this self-motion estimation approach a promising avenue for small robotics projects. An application in this area could be promising.

In this context, less supervision in training could also help. Closing the loop with semantic inference could remove the need for any ground truth motion in training.

The orientation estimation framework can probably produce even better results when trained on a more balanced data set. This could be achieved by re-training the object detector on the available data or by introducing renderings of artificial objects into the data set. Introducing orientation estimates into a dynamic object state filtering approach has proven beneficial, by adding a track initialization and termination stage and automatically assigning detections to filters will almost certainly result in a successful full object tracking system. This could also inspire other object tracking approaches to include orientation estimates.

As to the gist representation: literature has shown that similar ideas can be successfully implemented, so using established inference mechanisms based on e.g. graphical models or conditional random fields could succeed in creating a consistent artificial gist.

Acknowledgements

This thesis would not have been possible without the help of many people.

I am very grateful for the help and support from my supervisors. Christian Wallraven has provided me with an exciting start into the PhD, left me great freedom for exploring new ideas while still restraining my originally very naive approach. Even after leaving to Korea he still supervised me as long as possible and taught me to connect computer vision and human perception.

Cristóbal Curio has not only supervised me but invested an incredible amount of work into this PhD student he inherited so unexpectedly. With hardly any time to prepare he came up with the dynamic vision project that proved very rewarding. He seems to have had our project on his mind day and night, coming up with new innovations that could have occupied me for another three years. This way, we had the luxury to discuss and pick from a plethora of possible avenues for our project.

Thanks also to Andreas Schilling who always had an open door to discuss my research and provided me with helpful comments and insights. Heinrich Bülthoff has funded my PhD for longer than intended, thank you for that.

This thesis would not have been possible without the friendly working environment in the Human Perception, Cognition and Action group of the Max Planck Institute for Biological Cybernetics. Especially Björn Browatzki, Janina Esins and Mirko Thiesen have made working here a great pleasure. Daniela Diessel has also helped me with corrections of an early version of this thesis, which certainly needed great patience.

My flat mates Helen Hermann, Christine Schurr, Caroline Schneider and Liesa Schnee have helped me during these last months with countless announcements like "I have cooked, you want something, too?" and also before that by providing a great place to come home after work.

A big thank you goes to my family, especially to my parents Angelika and Joachim Herdtweck, who have supported my scientific interest from the very beginning.

Finally, and most of all, I am grateful to Julia Budde, who has supported me in countless ways for so many years. Her help and encouragement were invaluable.

Bibliography

M. Abramowitz and I. A. Stegun. *Handbook of Mathematical Functions*. National Bureau of Standards, New York, 1970. URL `http://people.math.sfu.ca/~cbm/aands`.

Y. Agiomyrgiannakis and Y. Stylianou. Stochastic Modeling and Quantization of Harmonic Phase in Speech using Wrapped Gaussian Mixture Models. In *International Conference on Acoustics, Speech and Signal Processing*, volume 4, pages 1121–1124. IEEE, 2007.

Y. Agiomyrgiannakis and Y. Stylianou. Wrapped Gaussian Mixture Models for Modeling and High-Rate Quantization of Phase Data of Speech. *Transactions on Audio, Speech, and Language Processing*, 17(4):775–786, 2009.

Mykhaylo Andriluka, Stefan Roth, and Bernt Schiele. Monocular 3D pose estimation and tracking by detection. In *International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 623–630. IEEE, 2010.

M Sanjeev Arulampalam, Simon Maskell, Neil Gordon, and Tim Clapp. A Tutorial on Particle Filters for Online Nonlinear / Non-Gaussian Bayesian Tracking. *Transactions on Signal Processing*, 50(2):174–188, 2002.

S. E. Asch and H. A. Witkin. Studies in space orientation: 1. Perception of the upright with displaced visual fields. *Journal of Experimental Psychology*, 38:325–337, 1948.

Hermann Aubert. Eine scheinbare bedeutende Drehung von Objecten bei Neigung des Kopfes nach rechts oder links. *Virchows Archiv*, 20(3-4):381–393, 1861.

Hernán Badino. A Robust Approach for Ego-Motion Estimation Using a Mobile Stereo Platform. In Bernd Jähne, Rudolf Mester, Erhardt Barth, and Hanno Scharr, editors, *Complex Motion*, volume 3417 of *Lecture Notes in Computer Science*, pages 198–208. Springer, 2007. ISBN 978-3-540-69864-7.

Hernán Badino, Uwe Franke, and David Pfeiffer. The Stixel World - A Compact Medium Level Representation of the 3D-World. In Joachim Denzler, Gunther Notni, and Herbert Süße, editors, *Pattern Recognition*, volume 5748 of *Lecture Notes in Computer Science*, pages 51–60. Springer, 2009. ISBN 978-3-642-03797-9.

Arindam Banerjee, Inderjit S. Dhillon, Joydeep Ghosh, and Suvrit Sra. Clustering on the Unit Hypersphere using von Mises-Fisher Distributions. *Journal of Machine Learning Research*, 6:1345–1382, 2005.

Yaakov Bar-Shalom and Thomas E. Fortmann. *Tracking and Data Association*. Academic Press, 1988.

Björn Barrois, Stela Hristova, Christian Wöhler, Franz Kummert, and Christoph Hermes. 3D Pose Estimation of Vehicles Using a Stereo Camera. In *Intelligent Vehicles Symposium (IV)*, pages 262–272. IEEE, 2009.

Alexander Barth, Jan Siegemund, Uwe Franke, and Wolfgang Förstner. Simultaneous Estimation of Pose and Motion at Highly Dynamic Turn Maneuvers. In J. Denzler, G. Notni, and H. Süß e, editors, *DAGM*, pages 262–271, 2009.

Simon Bernard, Sébastien Adam, and Laurent Heutte. Dynamic Random Forests. *Pattern Recognition Letters*, 33(12):1580–1586, 2012.

Gérard Biau. Analysis of a Random Forests Model. *Journal of Machine Learning Research*, 13:1063–1095, 2012.

Irving Biederman. Recognition-by-Components: A Theory of Human Image Understanding. *Psychological Review*, 94(2):115–147, 1987.

Irving Biederman, R J Mezzanotte, and J C Rabinowitz. Scene perception: detecting and judging objects undergoing relational violations. *Cognitive Psychology*, 14(2): 143–177, 1982.

Christopher M. Bishop. *Pattern recognition and machine learning*, volume 4 of *Information science and statistics*. Springer, 2006. ISBN 9780387310732.

Michael J Black and P. Anandan. A Framework for the Robust Estimation of Optical Flow. In *International Conference on Computer Vision (ICCV)*, pages 231–236, 1993.

Samia Bouchafa and Bertrand Zavidovique. c-Velocity: A Flow-Cumulating Uncalibrated Approach for 3D Plane Detection. *International Journal of Computer Vision*, 97(2): 148–166, 2012.

Leo Breiman. Random Forests. *Machine Learning*, 45:5–32, 2001.

L. Bringoux, K. Tamura, M. Faldon, M. A. Gresty, and A. M. Bronstein. Influence of whole-body pitch tilt and kinesthetic cues on the perceived gravity-referenced eye level. *Experimental Brain Research*, 155(3):385–392, 2004.

Lionel Bringoux, Ludovic Marin, Vincent Nougier, Pierre-Alain Barraud, and Christian Raphel. Effects of gymnastics expertise on the perception of body orientation in the pitch dimension. *Journal of Vestibular Research*, 10(6):251–258, 2000.

Gavin Brown. Ensemble learning. In C. Sammut and G. Webb, editors, *Encyclopedia of Machine Learning*, chapter 400. Springer, first edition, 2010.

Andrés Bruhn, Joachim Weickert, and Christoph Schnörr. Lucas/Kanade meets Horn/Schunck: Combining local and global optic flow methods. *International Journal of Computer Vision*, 61(3):211–231, 2005.

Heinrich H Bülthoff and Shimon Edelman. Psychophysical support for a two-dimensional view interpolation theory of object recognition. *Proceedings of the National Academy of Sciences*, 89(1):60, 1992.

Jason Campbell, Rahul Sukthankar, Illah Nourbakhsh, and Aroon Pahwa. A Robust Visual Odometry and Precipice Detection System Using Consumer-grade Monocular Vision. In *International Conference on Robotics and Automation (ICRA)*, pages 3421–3427, 2005.

Liangliang Cao and Li Fei-Fei. Spatially coherent latent topic model for concurrent object segmentation and classification. In *International Conference on Computer Vision (ICCV)*, 2007.

M. M. Cohen and C. A. Larson. Human spatial orientation in the pitch dimension. *Perception and Psychophysics*, 16:509–512, 1974.

M. M. Cohen, A. E. Stoper, R. B. Welch, and C. W. DeRoshia. Effects of gravitational and optical stimulation on the perception of target elevation. *Attention, Perception and Psychophysics*, 63(1):29–35, 2001.

Dorin Comaniciu and Peter Meer. Mean shift analysis and applications. In *International Conference on Computer Vision (ICCV)*, pages 1197–1203. IEEE, 1999.

Dorin Comaniciu and Peter Meer. Mean Shift : A Robust Approach Toward Feature Space Analysis. *Transactions on Pattern Analysis and Machine Intelligence*, 24(5): 603–619, 2002.

Dorin Comaniciu, Visvanathan Ramesh, and Peter Meer. The variable bandwidth mean shift and data-driven scale selection. In *International Conference on Computer Vision (ICCV)*, volume 1, pages 438–445. IEEE, 2001.

Dorin Comaniciu, Visvanathan Ramesh, and Peter Meer. Kernel-Based Object Tracking. *Transactions on Pattern Analysis and Machine Intelligence*, 25(5):564–577, 2003.

Emily A Cooper, Johannes Burge, and Martin S Banks. The vertical horopter is not adaptable, but it may be adaptive. *Journal of Vision*, 11(3):20, 2011.

James Coughlan and Alan Yuille. Manhattan World: Orientation and Outlier Detection by Bayesian Inference. *Neural Computation*, 15(5):1063–1088, 2003.

Antonio Criminisi, Jamie Shotton, Duncan Robertson, and Ender Konukoglu. Regression Forests for Efficient Anatomy Detection and Localization in CT Studies. In Bjoern Menze, Georg Langs, Zhuowen Tu, and Antonio Criminisi, editors, *Medical Computer Vision. Recognition Techniques and Applications in Medical Imaging*, volume 6533 of *Lecture Notes in Computer Science*, pages 106–117. Springer, 2011.

D. Crisan and A. Doucet. A survey of convergence results on particle filtering methods for practitioners. *Transactions on Signal Processing*, 50(3):736–746, 2002.

Cristóbal Curio. *A learning-based computer vision approach for the inference of articulated motion*. PhD thesis, Fakultät für Elektrotechnik und Informationstechnik, Ruhr-Universität Bochum, 2004.

Jean-Pierre Da Costa, Frédéric Galland, Antoine Roueff, and Christian Germain. Unsupervised segmentation based on Von Mises circular distributions for orientation estimation in textured images. *Journal of Electronic Imaging*, 21(2):021102, 2012.

Navneet Dalal and William Triggs. Histograms of Oriented Gradients for Human Detection. In *International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 886–893. IEEE, 2005a.

Navneet Dalal and William Triggs. Histograms of Oriented Gradients for Human Detection. In *International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 886–893. IEEE, 2005b.

M. Dantone, J. Gall, G. Fanelli, and L. Van Gool. Real-time facial feature detection using conditional regression forests. In *International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2578–2585. IEEE, 2012.

Frank Dellaert and Chuck Thorpe. Robust car tracking using Kalman ltering and Bayesian templates. In *Conference on Intelligent Transportation Systems*, volume 1, 1997.

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society, Series B (Methodological)*, 39(1):1–38, 1977.

Renaud Detry and Justus Piater. Continuous Surface-point Distributions for 3D Object Pose Estimation and Recognition. In *Asian Conference on Computer Vision (ACCV)*, 2010.

J. Deutscher, M. Isard, and J. MacCormick. Automatic Camera Calibration from a Single Manhattan Image. In *European Conference on Computer Vision (ECCV)*. Springer, 2002.

Marco Di Marzio, Agnese Panzera, and Charles C. Taylor. Kernel density estimation on the torus. *Journal of Statistical Planning and Inference*, 141(6):2156–2173, 2011.

Paul Di Zio, Wenxun Li, James Lackner, and Leonard Matin. Combined influences of gravitoinertial force level and visual field pitch on visually perceived eye level. *Journal of Vestibular Research*, 7(5):381–392, 1997.

Neil Doucet, Arnaud and de Freitas, Nando and Gordon, editor. *Sequential Monte Carlo Methods in Practice*. Springer, 2001. ISBN 978-0-387-95146-1.

David Engel and Cristóbal Curio. Scale-invariant medial features based on gradient vector flow fields. In *International Conference on Pattern Recognition (ICPR)*, pages 1–4. IEEE, 2008.

Markus Enzweiler and Dariu M Gavrila. A Multi-Level Mixture-of-Experts Framework for Pedestrian Classification. *Transactions on Image Processing*, 20(10):2967–2979, 2011.

M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2012 Results, 2012. URL `http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html`.

Gabriele Fanelli, Jürgen Gall, and Luc Van Gool. Real time head pose estimation with random regression forests. In *International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 617–624. IEEE, 2011a.

Gabriele Fanelli, Thibaut Weise, Jürgen Gall, and Luc Van Gool. Real Time Head Pose Estimation from Consumer Depth Cameras. In *Annual Symposium of the German Association for Pattern Recognition (DAGM)*, pages 101–110. Springer, 2011b.

Li Fei-Fei, Asha Iyer, Christof Koch, and Pietro Perona. What do we perceive in a glance of a real-world scene? *Journal of Vision*, 7(1):1–29, 2007.

Sandor Fejes and Larry S Davis. Detection of independent motion using directional motion estimation. *Computer Vision and Image Understanding*, 74(2):101–120, 1999.

Pedro F. Felzenszwalb and Daniel P. Huttenlocher. Efficient Graph-Based Image Segmentation. *International Journal of Computer Vision*, 59(2):167–181, 2003.

Pedro F Felzenszwalb, Ross B Girshick, David McAllester, and Deva Ramanan. Object detection with discriminatively trained part-based models. *Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1627–45, 2010.

N. I. Fisher. *Statistical analysis of circular data*. Cambridge University Press, 1995.

Tom Foulsham, Alan Kingstone, and Geoffrey Underwood. Turning the world around: Patterns in saccade direction vary with picture orientation. *Vision Research*, 48(17): 1777–1790, 2008.

Keinosuke Fukunaga and Larry Hostetler. The estimation of the gradient of a density function, with applications in pattern recognition. *Transactions on Information Theory*, 21(1):32–40, 1975.

Jürgen Gall and Victor Lempitsky. Class-specific hough forests for object detection. In *International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1022–1029. IEEE, 2009.

S Gammeter, A Ess, T. Jäggli, K. Schindler, B. Leibe, and L. VanGool. Articulated Multi-body Tracking Under Egomotion. In *European Conference on Computer Vision (ECCV)*, pages 816–830. Springer, 2008.

Andreas Geiger, Martin Roser, and Raquel Urtasun. Efficient Large-Scale Stereo Matching. In *Asian Conference on Computer Vision (ACCV)*, 2010.

Andreas Geiger, Julius Ziegler, and Christoph Stiller. StereoScan: Dense 3d Reconstruction in Real-time. In *Intelligent Vehicles Symposium (IV)*. IEEE, 2011.

Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite. In *International Conference on Computer Vision and Pattern Recognition (CVPR)*, Providence, USA, 2012. IEEE.

James J. Gibson. The Ecological Approach to the Visual Perception of Pictures. *Leonardo*, 11(3):227–235, 1978.

James J. Gibson. *The Ecological Approach to Visual Perception*. Houghton Mifflin, Boston, 1979.

James J. Gibson and O. H. Mowrer. Determinants of the perceived vertical and horizontal. *Psychological Review*, 45(4):300–323, 1938.

Daniel Glasner, Meirav Galun, Sharon Alpert, Ronen Basri, and Gregory Shakhnarovich. Viewpoint-aware object detection and pose estimation. In *International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1275–1282. IEEE, 2011.

Christian Gosch, Ketut Fundana, Anders Heyden, Christoph Schnörr, and Christoph Schön. View point tracking of rigid objects based on shape sub-manifolds. In *European Conference on Computer Vision (ECCV)*, pages 1–12. Springer, 2008.

M R Greene and A Oliva. Recognition of Natural Scenes from Global Properties: Seeing the Forest Without Representing the Trees. *Cognitive Psychology*, 58(2):137–179, 2009a.

M R Greene and A Oliva. The briefest of glances: The time course of natural scene understanding. *Psychological Science*, 20(4):464–472, 2009b.

G. Griffin, A. Holub, and P. Perona. Caltech-256 Object Category Dataset. Technical report, California Institute of Technology, 2007. URL `http://authors.library.caltech.edu/7694`.

A. R. Hanson and E. M. Riseman. VISIONS: A computer system for interpreting scenes. *Computer Vision Systems*, pages 303–333, 1978.

Wolfgang Härdle, Merlene Müller, Stefan Sperlich, and Axel Werwatz. *Chapter 3: Nonparametric Density Estimation*, page 39ff. Springer Series in Statistics. Springer, 2004.

Nils Hasler, Bodo Rosenhahn, Thorsten Thormählen, Michael Wand, Jürgen Gall, and Hans-Peter Seidel. Markerless Motion Capture with Unsynchronized Moving Cameras. In *International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 224–231. IEEE, 2009.

James Hays and Alexei A Efros. im2gps: estimating geographic information from a single image. In *International Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2008.

D. J. Heeger and A. D. Jepson. Subspace methods for recovering rigid motion I: Algorithm and implementation. *International Journal of Computer Vision*, 7(2):95–117, 1992.

Geremy Heitz and Daphne Koller. Learning Spatial Context: Using Stuff to Find Things. In *European Conference on Computer Vision (ECCV)*, page 30. IEEE, Springer, 2008.

Geremy Heitz, Stephen Gould, Ashutosh Saxena, and Daphne Koller. Cascaded Classification Models: Combining Models for Holistic Scene Understanding. In *Neural Information Processing Systems (NIPS)*, 2008.

David Held, Jesse Levinson, and Sebastian Thrun. A probabilistic framework for car detection in images using context and scale. In *International Conference on Robotics and Automation (ICRA)*, pages 1628–1634. IEEE, 2012.

C. Herdtweck and C. Wallraven. Estimation of the horizon in photographed outdoor scenes by human and machine. *PLoS ONE*, 8(12):e81462, 2013. doi: 10.1371/journal.pone.0081462. URL `http://dx.plos.org/10.1371/journal.pone.0081462`.

Christian Herdtweck and Cristóbal Curio. Experts of probabilistic flow subspaces for robust monocular odometry in urban areas. In *Intelligent Vehicles Symposium (IV)*, pages 661–667. IEEE, 2012a.

Christian Herdtweck and Cristóbal Curio. Monocular Heading Estimation in Non-stationary Urban Environment. In *Conference on Multisensor Fusion and Information Integration (MFI)*, 2012b.

Christian Herdtweck and Cristóbal Curio. Monocular Car Viewpoint Estimation with Circular Regression Forests. In *Intelligent Vehicles Symposium (IV, submitted)*. IEEE, 2013.

Christian Herdtweck and Christian Wallraven. Horizon estimation: perceptual and computational experiments. In *Applied Perception in Graphics and Visualization (APGV)*, pages 49–56. ACM, 2010a.

Christian Herdtweck and Christian Wallraven. Horizon estimation: Perceptual and computational experiments. In *Perception 39 ECVP Abstract Supplement*, page 106, 2010b.

E. C. Hildreth, H. B. Barlow, and H. C. Longuet-Higgins. Recovering Heading for Visually Guided Navigation in the Presence of Self-Moving Objects. *Philosophical Transactions: Biological Sciences*, 337(1281):305–313, 1992.

D. Hoiem, A. Efros, and M. Hebert. Putting objects in perspective. In *Proceedings IEEE Computer Vision and Pattern Recognition (CVPR)*, 2006.

Derek Hoiem, Alexei A. Efros, and Martial Hebert. Geometric Context from a Single Image. In *International Conference of Computer Vision (ICCV)*, volume 1, pages 654–661. IEEE, 2005.

Derek Hoiem, Alexei Efros, and Martial Hebert. Closing the Loop on Scene Interpretation. In *International Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2008.

Paul W. Holland and Roy E. Welsch. Robust regression using iteratively reweighted least-squares. *Communications in Statistics: Theory and Methods*, 6(9):813–827, 1977.

Berthold K. P. Horn and Brian G. Schunck. Determining Optical Flow. *Artificial intelligence*, 17(1):185–203, 1981.

John D Hunter. Matplotlib: A 2D graphics environment. *Computing In Science & Engineering*, 9(3):90–95, 2007.

Michal Irani. Multi-frame optical flow estimation using subspace constraints. In *International Conference on Computer Vision (ICCV)*, volume 1, pages 626–633. IEEE, 1999.

Michal Irani. Multi-Frame Correspondence Estimation Using Subspace Constraints. *International Journal of Computer Vision*, 48(3):173–194, 2002.

Michael Isard and Andrew Blake. CONDENSATION - Conditional Density Propagation for Visual Tracking. *International Journal of Computer Vision*, 29(1):5–28, 1998.

L. Itti and C. Koch. Computational Modelling of Visual Attention. *Nature Review of Neuroscience*, 2(3):194–203, 2001.

Robert A. Jacobs, Michael I. Jordan, Steven J. Nowlan, and Geoffrey E. Hinton. Adaptive Mixtures of Local Experts. *Neural Computation*, 3(1):79–87, 1991.

Ramesh Jain. Direct Computation of the Focus of Expansion. *Transactions on Pattern Analysis and Machine Intelligence*, 5(1):58–64, 1983.

Hervé Jégou, Matthijs Douze, and Cordelia Schmid. Hamming embedding and weak geometric consistency for large scale image search. In *European Conference on Computer Vision*, volume I, pages 304–317. Springer, 2008.

R. Kalman. New approach to filtering. *Transaction of the ASME - Journal of Basic Engineering*, pages 35–45, 1960.

Sören Kammel, Julius Ziegler, Benjamin Pitzer, Moritz Werling, Tobias Gindele, Daniel Jagzent, Joachim Schröder, Michael Thuy, Matthias Goebl, and Felix von Hundelshausen. Team AnnieWAY's Autonomous System for the 2007 DARPA Urban Challenge. *Journal of Field Robotics*, 25(9):615–639, 2008.

Bernd Kitt, Andreas Geiger, and Henning Lategahn. Visual odometry based on stereo image sequences with RANSAC-based outlier rejection scheme. *Intelligent Vehicles Symposium (IV)*, pages 486–492, 2010.

M Kleiner, D Brainard, D Pelli, A Ingling, R Murray, and C Broussard. What's new in Psychtoolbox-3? *Perception (ECVP Abstract Supplement)*, 14, 2007.

D. Koller, K. Daniilidis, and H.-H. Nagel. Model-Based Object Tracking in Monocular Image Sequences of Road Traffic Scenes. *International Journal of Computer Vision*, 10(3):257–281, 1993.

J Kosecka and W Zhang. Video Compass. In *European Conference on Computer Vision (ECCV)*, pages 476–490. Springer, 2002.

Walter Kropatsch. History of CV: A personal Perspective. In *The 3rd NFN Cognitive Vision Workshop with the topic "History of Computer Vision"*, 2008. URL `http://www.icg.tugraz.at/News/historyOfCV`.

Jean-Francois Lalonde, Derek Hoiem, Alexei A. Efros, Carsten Rother, John Winn, and Antonio Criminisi. Photo Clip Art. *ACM Transactions on Graphics*, 26(3), 2007.

Michael F. Land and Russell D. Fernald. The evolution of eyes. *Annual Reviews in Neuroscience*, 15:1–29, 1992.

B. Lariviere and D. Vandenpoel. Predicting customer retention and profitability by using random forests and regression forests techniques. *Expert Systems with Applications*, 29(2):472–484, 2005.

Henning Lategahn, Andreas Geiger, Bernd Kitt, and Christoph Stiller. Motion-without-Structure: Real-time Multipose Optimization for Accurate Visual Odometry. In *Intelligent Vehicles Symposium (IV)*. IEEE, 2012.

Oliver W. Layton, Ennio Mingolla, and N. Andrew Browning. A motion pooling model of visually guided navigation explains human behavior in the presence of independently moving objects. *Journal of Vision*, 12(1):1–19, 2012.

B Leibe, Aleš Leonardis, and B Schiele. *An Implicit Shape Model for Combined Object Categorization and Segmentation*, pages 508–524. Springer Berline, 2006.

Bastian Leibe, Nico Cornelis, Kurt Cornelis, and Luc Van Gool. Dynamic 3D Scene Analysis from a Moving Vehicle. In *International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8. IEEE, 2007.

Li-Jia Li, Hao Su, Eric P Xing, and Li Fei-Fei. Object Bank: A High-Level Image Representation for Scene Classification & Semantic Feature Sparsification. In *Neural Information Processing Systems (NIPS)*, number 24, 2010.

Stan Z. Li and Anil K. Jain, editors. *Handbook of face recognition*. Springer, 2011.

Rainer Lienhart, Alexander Kuranov, and Vadim Pisarevsky. Empirical Analysis of Detection Cascades of Boosted Classifiers for Rapid Object Detection. Technical report, Microprocessor Research Lab, Intel Labs, Intel Corporation, Santa Clara, CA 95052, USA, 2002.

H.C. Longuet-Higgins and K. Prazdny. The Interpretation of a Moving Retinal Image. *Proceedings of the Royal Society of London, Series B: Biological Sciences*, 208(1173): 385–397, 1980.

R. J. López-Sastre, C. Redondo-Cabrera, P. Gil-Jimenez, and S. Maldonado-Bascon. ICARO: Image Collection of An- notated Real-world Objects. ,, 2010.

Roberto J. López-Sastre, Tinne Tuytelaars, and Silvio Savarese. Deformable part models revisited: A performance evaluation for object category pose estimation. In *ICCV Workshops*, pages 1052–1059. IEEE, 2011.

D. Lowe. Object Recognition from Local Scale-Invariant Features. In *International Conference on Computer Vision (ICCV)*, volume 2, pages 1150–1157, 1999.

B. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Seventh International Joint Conference on Artificial Intelligence*, pages 674–679, 1981.

David Marr. *Vision - A Computational Investigation into the Human Representation and Processing of Visual Information*. W. H. Freeman and Company, San Francisco, 1982.

L. Matin and W. Li. Mislocalizations of Visual Elevation and Visual Vertical Induced by Visual Pitch: the Great Circle Model. *Annals of the New York Academy of Sciences*, 656 (Sensing and Controlling Motion: Vestibular and Sensorimotor Function)(1):242–265, 1992.

L. Matin, E. Picoult, J. K. Stevens, M. W. Jr Edwards, D. Young, and R. MacArthur. Oculoparalytic illusion: visual-field dependent spatial mislocalizations by humans partially paralyzed with curare. 216(4548):198–201, 1982.

I. S. McQuirk, Hae-Seung Lee, and B. K. P. Horn. An analog VLSI chip for estimating the focus of expansion. In *IEEE International Conference on Solid-State Circuits Conference (ISSCC)*, volume 2, pages 40–42, 1997.

Ross Messing and Frank H. Durgin. Distance Perception and the Visual Horizon in Head-Mounted Displays. *Transactions on Applied Perception*, 2(3):234–250, 2005.

George A. Miller, Richard Beckwith, Christiane Fellbaum, Derek Gross, and Katherine J. Miller. Introduction to WordNet: An On-line Lexical Database. *International Journal of Lexicography*, 3(4):235–244, 1990.

Rodrigo Munguia and Antoni Grau. Visual Slam for Monocular Odometry. In *International Symposium on Intelligent Signal Processing*, pages 1–6. IEEE, 2007.

S. Negahdaripour and B. K. P. Horn. A direct method for locating the focus of expansion. *Computer Vision, Graphics, and Image Processing*, 46(3):303–326, 1989.

Shahriar Negahdaripour. Direct Computation of the FOE with Confidence Measures. *Computer Vision and Image Understanding*, 64(3):323–350, 1996.

David Nistér and James Oleg NarodiBergen. Visual Odometry. In *International Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, page 652. IEEE, 2004.

A Oliva and A Torralba. Modeling the Shape of the Scene: A Holistic Representation of the Spatial Envelope. *International Journal of Computer Vision*, 42(3):145–175, 2001.

Aude Oliva. Gist of a Scene. In Laurent Itti, Geraint Rees, and John K. Tsotsos, editors, *Neurobiology of Attention*, pages 251–258. Academic Press, 2005.

Teng Leng Ooi, Bing Wu, and Zijian J. He. Distance determined by the angular declination below the horizon. *Nature*, 414:197–200, 2001.

Teng Leng Ooi, Bing Wu, and Zijian J He. Perceptual space in the dark affected by the intrinsic bias of the visual system. *Perception*, 35:605–624, 2006.

Kerem Ozkan and Myron L Braunstein. Background surface and horizon effects in the perception of relative size and distance. *Visual Cognition*, 18(2):229–254, 2010.

Ovgu Ozturk, Toshihiko Yamasaki, and Kiyoharu Aizawa. Tracking of humans and estimation of body/head orientation from top-view single camera for visual focus of attention analysis. In *ICCV (Workshop)*, pages 1020–1027. IEEE, 2009.

Mustafa Özuysal, Vincent Lepetit, and Pascal Fua. Pose Estimation for Category Specific Multiview Object Localization. In *International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 778–785, 2009.

Klaus Pawelzik, Jens Kohlmorgen, and Klaus-Robert Müller. Annealed Competition of Experts for a Segmentation and Classification of Switching Dynamics. *Neural Computation*, 8:340–356, 1996.

Bojan Pepik, Michael Stark, Peter Gehler, and Bernt Schiele. Teaching 3D Geometry to Deformable Part Models. In *International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3362–3369, Rhode Island, 2012. IEEE.

Fernando Pérez and Brian E Granger. Ipython: a system for interactive scientific computing. *Computing in Science & Engineering*, (3):21–29.

J W Philbeck and J M Loomis. Comparison of two indicators of visually perceived egocentric distance under full-cue and reduced-cue conditions. *Journal of Experimental Psychology: Human Perception and Performance*, 23:72–85, 1997.

Gerard Pons-Moll, Andreas Baak, Jürgen Gall, Laura Leal-Taixé, Meinard Müller, Hans-Peter Seidel, and Bodo Rosenhahn. Outdoor Human Motion Capture using Inverse Kinematics and von Mises-Fisher Sampling. In *International Conference on Computer Vision (ICCV)*, pages 1243–1250, 2011.

M. C. Potter. Short-term conceptual memory for pictures. *Journal of Experimental Psychology: Human Learning and Memory*, 2(5):509–522, 1976.

M. C. Potter, A. Staub, and D. H. O'Connor. Pictorial and conceptual representation of glimpse pictures. *Journal of Experimental Psychology: Human Perception and Performance*, 30:478–489, 2004.

K. Prazdny. Determining the instantaneous direction of motion from optical flow generated by a curvilinearly moving observer. *Computer Graphics and Image Processing*, 17(3):238–248, 1981.

K. M. Rand, M. R. Tarampi, S. H. Creem-Regehr, and W. B. Thompson. The importance of a visual horizon for distance judgments under severely degraded vision. *Perception*, 40(2):143–154, 2011.

Carl Edward Rasmussen and Hannes Nickisch. Gaussian Processes for Machine Learning (GPML) Toolbox. *Journal of Machine Learning Research*, 11:3011–3015, 2010.

Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

Florian Raudies and Heiko Neumann. A review and evaluation of methods estimating ego-motion. *Computer Vision and Image Understanding*, 116(5):606–633, 2012.

Narges Razavian, Hetunandan Kamisetty, and Christopher James Langmead. The von Mises Graphical Model: Regularized Structure and Parameter Learning. Technical Report September, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, 2011.

R. Roberts, C. Potthast, and F. Dellaert. Learning general optical flow subspaces for egomotion estimation and detection of motion anomalies. In *International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 57–64. IEEE, 2009.

S. Rogers. The horizon-ratio relation as information for relative size in pictures. *Attention, Perception and Psychophysics*, 58(1):142–152, 1996.

Rómer Rosales and Stan Sclaroff. Learning Body Pose via Specialized Maps. In *Neural Information Processing Systems (NIPS)*, pages 1263–1270, 2001.

Constance S. Royden. Computing heading in the presence of moving objects: a model that uses motion-opponent operators. *Vision research*, 42(28):3043–58, 2002.

B. Russell, A. Torralba, K. Murphy, and W. T. Freeman. LabelMe: a database and web-based tool for image annotation. *International Journal of Computer Vision*, 2007a.

Bryan Russell, Antonio Torralba, Kevin Murphy, and William Freeman. LabelMe: a database and web-based tool for image annotation. *International Journal of Computer Vision*, (1):157–173, 2007b.

Silvio Savarese and Li Fei-Fei. 3D generic object categorization, localization and pose estimation. In *International Conference on Computer Vision (ICCV)*, pages 1–8. IEEE, 2007.

Silvio Savarese and Li Fei-Fei. Multi-view Object Categorization and Pose Estimation. In R. Cipolla, S. Attiato, and G.M. Farinella, editors, *Computer Vision*, pages 205–231. Springer, 2010.

Ashutosh Saxena, Sung H Chung, and Andrew Y Ng. 3-D Depth Reconstruction from a Single Still Image. *International Journal of Computer Vision*, 76(1):53–69, 2007.

D. Sazbon, H. Rotstein, and E. Rivlin. Finding the focus of expansion and estimating range using optical flow images and a matched filter. *Machine Vision and Applications*, 15:229–236, 2004.

Davide Scaramuzza, Friedrich Fraundorfer, and Roland Siegwart. Real-Time Monocular Visual Odometry for On-Road Vehicles with 1-Point RANSAC. In *International Conference on Robotics and Automation (ICRA)*, pages 4293–4299, 2009.

Stefan Schaal and Christopher C. Atkeson. From Isolation to Cooperation : An Alternative View of a System of Experts. *Advances in neural information processing systems*, pages 605–611, 1995.

Timo Schairer, Benjamin Huhle, and Wolfgang Straßer. Application of Particle Filters to Vision-Based Orientation Estimation using Harmonic Analysis. In *International Conference on Robotics and Automation (ICRA)*, pages 2556–2561. IEEE, 2010.

Felix Schill and Robert Mahony. Estimating ego-motion in panoramic image sequences with inertial measurements. *Robotics Research*, 70:87–101, 2011.

Bernhard Schölkopf and Alexander J. Smola. *Learning with kernels: Support vector machines, regularization, optimization, and beyond*. MIT Press, 2002.

D. W. Scott. *Multivariate density estimation: theory, practice, and visualization*. Wiley-Interscience, 1992. ISBN 0471547700.

G. A. F. Seber. *Multivariate Observations*. John Wiley & Sons, 1984.

H. A. Sedgwick. *The geometry of spatial layout in pictorial representation*, volume 1, pages 33–90. Academic Press New York, 1980.

H. A. Sedgwick. *Space Perception*, volume 1, page 21. Wiley-Interscience, 1986.

T. Serre, L. Wolf, and T. Poggio. Object Recognition with Features Inspired by Visual Cortex. In *International Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 994–1000. IEEE, 2005.

J. Shotton, M. Johnson, and R. Cipolla. Semantic Texton Forests for Image Categorization and Segmentation. In *International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8. IEEE, 2008.

Jamie Shotton, Andrew Fitzgibbon, Mat Cook, Toby Sharp, Mark Finocchio, Richard Moore, Alex Kipman, and Andrew Blake. Real-time human pose recognition in parts from single depth images, 2011.

P.E. Shrout and J.L. Fleiss. Intraclass correlations: Uses in assessing rater reliability. *Psychol Bull*, 86(2):420–428, 1979.

G. Sibley, C. Mei, I. Reid, and P. Newman. Vast-scale Outdoor Navigation Using Adaptive Relative Bundle Adjustment. *International Journal of Robotics Research*, 29(8):958–980, 2010.

Hedvig Sidenbladh. Multi-Target Particle Filtering for the Probability Hypothesis Density. In *International Conference on Information Fusion*, 2003.

J. Sivic, B. Kaneva, A. Torralba, S. Avidan, and W. T. Freeman. Creating and exploring a large photorealistic virtual space. In *First IEEE Workshop on Internet Vision*, 2008.

Michael Stark, Michael Goesele, and Bernt Schiele. Back to the Future: Learning Shape Models from 3D CAD Data. In *British Machine Vision Conference (BMVC)*, pages 1–11, 2010.

Johannes Stephan. *Implementation and Evaluation of a Continuous Latent Variable Model for Visual Ego-Motion Estimation*. Master thesis, Eberhardt Karls Universität Tübingen, 2010.

Arnold Stoper and Malcom Cohen. Judgments of eye level in light and in darkness. *Attention, Perception and Psychophysics*, 40(5):311–316, 1986.

Hao Su, Min Sun, Li Fei-Fei, and Silvio Savarese. Learning a dense multi-view representation for detection, viewpoint classification and synthesis of object categories. In *International Conference on Computer Vision (ICCV)*, pages 213–220. IEEE, 2009.

Erik B Sudderth. *Graphical Models for Visual Object Recognition and Tracking*. PhD thesis, MIT, 2006.

Vladimir Svetnik, Andy Liaw, Christopher Tong, J Christopher Culberson, Robert P Sheridan, and Bradley P Feuston. Random forest: a classification and regression tool for compound classification and QSAR modeling. *Journal of chemical information and computer sciences*, 43(6):1947–58, 2003.

Jean-Philippe Tardif, Yanis Pavlidis, and Kostas Daniilidis. Monocular visual odometry in urban environments using an omnidirectional camera. In *International Conference on Intelligent Robots and Systemsi (IROS)*, pages 2531–2538. IEEE / RSJ, 2008.

Alexander Thomas, Vittorio Ferrari, Bastian Leibe, Tinne Tuytelaars, Bernt Schiele, and Luc Van Gool. Towards Multi-View Object Class Detection. In *International Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 1589–1596. IEEE, 2006.

William B Thompson, Valentina Dilda, and Sarah H Creem-Regehr. Absolute distance perception to locations off the ground plane. *Perception*, 36:1559–1571, 2007.

S. Thorpe, D. Fize, and C. Marlot. Speed of processing in the human visual system. *Nature*, 381:520–522, 1996.

T.Y. Tian, Carlo Tomasi, and D.J. Heeger. Comparison of approaches to egomotion computation. In *International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 315–320. IEEE, 1996.

Michael E. Tipping and Christopher M. Bishop. Probabilistic Principal Component Analysis. *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, 61 (3):611–622, 1999.

A. Torralba and P. Sinha. Statistical Context Priming for Object Detection. In *International Conference on Computer Vision (ICCV)*, pages 763–770. IEEE, 2001.

L. Tremblay, D. Elliott, and J. L. Starkes. Gender differences in perception of self-orientation: Software or hardware? *Perception*, 33(3):329–337, 2004.

Chris Urmson, Joshua Anhalt, Drew Bagnell, Christopher Baker, Robert Bittner, M. N. Clark, John Dolan, Dave Duggins, Tugrul Galatali, Chris Geyer, Michele Gittleman, Sam Harbaugh, Martial Hebert, Thomas M. Howard, Sascha Kolski, Alonzo Kelly, Maxim Likhachev, Matt Mcnaughton, Nick Miller, Kevin Peterson, Brian Pilnick, Raj Rajkumar, Paul Rybski, Bryan Salesky, Young-woo Seo, Sanjiv Singh, Jarrod Snider, Anthony Stentz, William Red Whittaker, Ziv Wolkowicki, Jason Ziglar, Hong Bae, Thomas Brown, Daniel Demitrish, Bakhtiar Litkouhi, Jim Nickolaou, Varsha Sadekar, Wende Zhang, Joshua Struble, Michael Taylor, Michael Darms, and Dave Ferguson. Autonomous Driving in Urban Environments: Boss and the Urban Challenge. *Journal of Field Robotics*, 25(8):425–466, 2008.

Paul Viola and Michael Jones. Rapid object detection using a boosted cascade of simple features. In *International Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2001.

Chris S. Wallace and David L. Dowe. MML clustering of multi-state, Poisson, von Mises circular and Gaussian distributions. *Statistics and Computing*, 10(1):73–83, 2000.

Jue Wang, Bo Thiesson, Yingqing Xu, and Michael Cohen. Image and Video Segmentation by Anisotropic Kernel Mean Shift. In *European Conference on Computer Vision (ECCV)*, pages 238–249, 2004.

M Wertheimer. Experimentelle Studien über das Sehen von Bewegung. *Zeitschrift für Psychologie*, 61, 1912.

Felix Woelk and Reinhard Koch. Robust monocular detection of independent motion by a moving observer. *Complex Motion*, pages 209–222, 2007.

David H. Wolpert. Stacked generalization. *Neural Networks*, 5(2):241–259, 1992.

Bing Wu, Teng Leng Ooi, and Zijiang J. He. Perceiving distance accurately by a directional process of integrating ground information. *Nature*, 428:73–77, 2004.

Bing Wu, Zijiang He, and Teng Ooi. The linear perspective information in ground surface representation and distance judgment. *Attention, Perception, & Psychophysics*, 69 (5):654–672, 2007.

G Wyszecki and W S Stiles. *Color science: Concepts and Methods, Quantitative data, and Formulae*. Wiley-Interscience, 2000.

Jian Yao, Sanja Fidler, and Raquel Urtasun. Describing the scene as a whole: Joint object detection, scene classification and semantic segmentation. In *International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 702–709. IEEE, 2012.

Alper Yilmaz, Omar Javed, and Mubarak Shah. Object tracking. *ACM Computing Surveys*, 38(4):13, 2006.

S. E. Yuksel, J. N. Wilson, and P. D. Gader. Twenty Years of Mixture of Experts. *Transactions on Neural Networks and Learning Systems*, 23(8):1177–1193, 2012.

Jerrold H. Zar. *Biostatistical Analysis*. 5 edition, 2010.

Fan Zhang, Edwin R. Hancock, Casey Goodlett, and Guido Gerig. Probabilistic White Matter Fiber Tracking using Particle Filtering and von Mises-Fisher Sampling. *Medical Image Analysis*, 13(1):5–18, 2009.

M Zeeshan Zia, Michael Stark, Bernt Schiele, and Konrad Schindler. Revisiting 3D Geometric Models for Accurate Object Shape and Pose. In *ICCV Workshop*, pages 569–576, 2011.