Visual Detection of Independently Moving Objects by a Moving Monocular Observer

Dissertation

zur Erlangung des akademischen Grades
Doktor der Ingenieurwissenschaften
(Dr.-Ing.)
der Technischen Fakultät
der Christian-Albrechts-Universität zu Kiel

Dipl.-Phys. J. Felix Woelk

Kiel
August 27, 2008

1. Gutachter	:	Prof. Dr.-Ing. Reinhard Koch
2. Gutachter	:	Prof. Dr.-Ing. Rolf-Rainer Grigat
Datum der mündlichen Prüfung	:	10.1.2008

Danksagung

Ich möchte mich hiermit bei allen, die mir während der Durchführung dieser Arbeit zur Seite standen, ganz herzlich bedanken.

Meinem Doktorvater, Prof. Dr.-Ing. Reinhard Koch, möchte ich für das interessante Thema danken. Er führte mich freundschaftlich in die spannende Welt der Bildverarbeitung ein und ermöglichte mir dabei eine sehr freie Arbeitsweise. Prof. Dr.-Ing. Rolf-Rainer Grigat möchte ich für seine Arbeit als Gutachter danken. Weiterhin gilt mein Dank auch den Mitgliedern der Prüfungskommission, Prof. Dr.-Ing. Ulrich Heute und Prof. Dr. Thomas Wilke.

Bei Dr. Stefan Gehrig und Dr. Uwe Franke möchte ich mich für die freundschaftliche Zusammenarbeit bedanken. Sie setzten mir von Anfang an die hohen, aber dennoch erreichbaren Ziele, die für den Erfolg dieser Arbeit wichtig waren.

Meinen Kollegen, Dr. Jan-Michael Frahm, Kevin Köser, Dr. Daniel Grest, Jan-Friso Evers-Senne, Bogumil Bartczak sowie Dr. Christian Beder möchte ich für die fruchtbaren Diskussionen danken. Torge Storm und Renate Staecker danke ich dafür, dass sie mir stets unkompliziert in technischer und administrativer Hinsicht Hilfe gewährt haben. Ferner möchte ich allen Mitarbeitern der Arbeitsgruppe Multimedia Information Processing an der CAU Kiel für das freundschaftliche und gute Arbeitsklima danken. Die Ergebnisse dieser Arbeit haben immens von der sehr guten Zusammenarbeit der Gruppe profitiert.

Bei meinen Eltern möchte ich mich für die Förderung meiner wissenschaftlichen Ausbildung bedanken. Meine Mutter hat sich weiterhin sehr um die grammatikalischen und orthographischen Qualitäten dieser Arbeit verdient gemacht.

Michaela Urban möchte ich für ihr Verständnis dafür danken, dass viele gemeinsame Stunden verloren gingen. Sie war immer sehr tolerant, wenn ich meine Freizeit mal wieder der Arbeit opferte und stand mir dennoch stets mit aufmunternden Worten zur Seite.

Contents

Typographic Conventions

The typographic conventions from table 1 have been used in this work. However in rare occasions, either when the readability is affected or when customarily other fonts are used, these typographic conventions are dropped in favour of readability or in favour of the familiar fonts.

Typeface	Description	Meaning	Examples
$abcxyz$	lower case italic roman	scalar values	the polynom $ax + by + c = 0$
$\alpha\beta\gamma\mu\nu$	lower case italic greek	scalar values	the angle α
$\boldsymbol{xyz}$	lower case bold italic roman	vectors in $\mathbb{R}^2$ or $\mathbb{P}^2$	projective image point $\boldsymbol{x}$
$\boldsymbol{XYZ}$	upper case bold italic roman	vectors in $\mathbb{R}^3$ or $\mathbb{P}^3$	homogenous world point $\boldsymbol{X}$
abcxyz	lower case bold roman	vector of arbitrary length, not representing geometric entities	vector of Lagrange multipliers **u**
ABP	upper case bold roman	matrices	projection matrix **P**
$\mathcal{TO}$	upper case caligraphic	tensors with more than 2 dimensions and vector fields	trifocal tensor $\mathcal{T}$, optical flow field $\mathcal{O}$

Table 1: Typographic conventions

Lower case roman or greek italic letters are used to represent scalar value. Bold italic lower case roman letters symbolise vectors representing geometric entities in 2D: Either in Euclidean representation ($\mathbb{R}^2$) or in projective representation ($\mathbb{P}^2$). Bold italic upper case roman letters symbolise geometric entities in 3D: Again either in their Euclidean representation ($\mathbb{R}^3$) or in projective representation ($\mathbb{P}^3$). Matrices are represented with symbols in upper case roman bold letters and lower case bold roman letters represent vectors of arbitrary dimension which do not represent geometric entities. Tensors and vector fields are represented by upper case caligraphic letters.

Nomenclature

α ..... Direction difference between expected and measured translational flow

α ..... Rotation angle (car motion)

α ..... Rotation angle (egomotion estimation)

α ..... Viewing angle (collision detection)

α_e ..... Epipole error (egomotion estimation)

α_f ..... Direction of translational flow vector

α_p ..... Direction of predicted translational flow vector

α_r ..... Rotation error (egomotion estimation)

β_0 ..... Slip angle

χ^2 ..... Chi square distribution

$[\boldsymbol{a}]_\times$ ..... Skew symmetric 3 by 3 matrix containing the components of the vector $\boldsymbol{a}$

δ_A ..... Steering angle (Ackermannwinkel)

$\dot{\alpha}$ ..... Yaw rate

γ ..... Mixing parameter (MLESAC)

$\Gamma(x)$ ..... Gamma Function

κ ..... Factor for cornerness computation of the Harris corner detector

κ_i ..... Lens distortion coefficients

λ ..... Memory parameter of transition probability $p(s_t|s_{t-1})$

λ ..... Projective scaling factor

λ_i ..... Eigenvalue i

∇ ..... Nabla operator

ω ..... Rotation angle

$\omega(r)$ ..... Weight function (M-Estimator)

$\omega_{x,y,z}$ ..... Euler angle around x, y or z axis

$\Psi(r)$ ..... Influence function (M-Estimator)

$\rho(r)$ ..... Error function (M-Estimator)

$\Sigma_{\boldsymbol{xx}}$ ..... Covariance matrix of the vector $\boldsymbol{x}$

σ_x ..... Standard deviation of random variable x

τ ..... Time to contact

θ ..... Threshold of a classifier

$\theta^{(i)}$ ..... Particle i

θ_t ..... State of the system at time t

$\boldsymbol{a}_e$ ..... Randomly chosen axis perpendicular to epipole

$\boldsymbol{a}_r$ ..... Randomly chosen rotation axis

a ..... Aspect ratio

a ..... Longitudinal acceleration

a_l ..... Lateral acceleration

AIC ..... Akaike Information Criterion

b ..... Baseline of stereo camera system

$B(x|a,b)$ ..... Beta distribution

$\mathbf{C}$ ..... Conic

$\mathbf{C}^*$ ..... Dual conic

$\boldsymbol{C}$ ..... Centre of projection

$\boldsymbol{c}$ ..... Principal point

$\boldsymbol{c}_D$ ..... Centre of distortion

cdf ..... Cumulative distribution function

$\overline{\boldsymbol{d}}$ ..... Weighted mean motion of a cluster

$\boldsymbol{d}$ ..... Displacement vector

$\boldsymbol{d}_f$ ..... Translational flow vector

$\boldsymbol{d}_o$ ..... Vector between origin and center of ellipsoid

$\boldsymbol{d}_p$ ..... Predicted translational flow vector of unit length

d ..... Distance (car motion)

d ..... Length of the translational flow vector

d ..... Sum of reprojection errors

d' ..... Discriminability index d-prime

d_{ia} ..... Algebraic error

d_{ig} ..... Geometric error

DLT ..... Direct Linear Transform

DOF ..... Degrees Of Freedom

$\mathbf{E}$ ..... Essential matrix

$\boldsymbol{e}$ ..... Epipole

$E[x]$ ..... Expectation of x

EM ..... Expectation Maximisation

$\mathbf{F}$ ..... Fundamental matrix

$\boldsymbol{f}$ ..... Flow vector

$\boldsymbol{f}_p$ ..... Prediction of the translational component of the flow vector $\boldsymbol{f}$

$\boldsymbol{f}_r$ ..... Rotational component of the flow vector $\boldsymbol{f}$

$\boldsymbol{f}_t$ ..... Translational component of the flow vector $\boldsymbol{f}$

f ..... Focal length

FOC ..... Focus Of Contraction

FOE ..... Focus Of Expansion

FOV ..... Field Of View of a camera (aperture angle)

$g(\theta)$ ..... Importance distribution (particle filter))

g_{xx} ..... Sum of squared horizontal gradients

g_{xy} ..... Sum of products between vertical and horizontal gradients

g_{yy} ..... Sum of squared vertical gradients

GRIC ..... Geometric Robust Information Criterion

$\mathbf{H}$ ..... Homography

$\boldsymbol{h}_l$ ..... Longer half axis of uncertainty ellipsoid

$h_{p,\theta}(x)$ ..... Classifier

$\boldsymbol{I}$ ..... Image plane

I ..... Image intensity

iid ..... Independent, identical distributed

IMO ..... Independently Moving Object

IMPSAC .... IMPortance sampling and RANdom SAmple Consensus estimator

$\mathbf{J}(x)$ ..... Jacobian of function x

$\mathbf{J}_{\mathbf{x}}$ ..... Jacobian of the function $\mathbf{x}$

$\mathbf{K}$ ..... Calibration matrix

KLT ..... Feature tracking algorithm, named after its inventors: Kanade, Lucas and Tomasi

$\boldsymbol{l}$ ..... Epipolar line

l ..... Length of the flow vector

l ..... Wheelbase of the car

$L(r^2)$ ..... Lens distortion function

l_h ..... Distance of the barycentre from the rear axis of the car

$l_{\max}$ ..... Maximal measurable flow length

l_{SBG} ..... Maximum length of the optical flow for points belonging to the static background

LM ..... Levenberg-Marquardt minimisation algorithm

LMedS ..... Least Median of Squares estimator

LSE ..... Least Squares Estimator

LTS ..... Least Trimmed Squares estimator

$\mathbf{M}$ ..... Structure tensor

MAPSAC ... Maximum A Posteriori SAmple Consensus

MINPRAN .. MINimise the Probability of RANdomnes estimator

MLESAC .... Maximum Likelihood Estimation SAmple Consensus

MSAC ..... Maximum likelihood estimator (M-estimator) SAmple Consensus

n ..... Normalization constant for $p(l|s = \text{SBG})$

NAPSAC .... N Adjacent Points Sample Consensus estimator

NIS ..... Normalised Innovation Squared

$\mathcal{O}$ ..... Optical flow field

$\mathcal{O}_r$ ..... Translational part of optical flow vector field

$\mathcal{O}_t$ ..... Rotational part of optical flow vector field

$\boldsymbol{O}$ ..... Optical axis

O ..... Origin of the reference frame

$\mathbf{P}$ ..... Projection matrix modelling perspective projection

$\overline{p}$ ..... Mean probability of a cluster

p ..... Parity of a classifier

$p(\alpha, l|s)$ ..... Joined likelihood function for α and l

$p(l|s)$ ..... Likelihood function for flow length l

$p(s|\alpha, l)$ ..... A posteriori probability for the state of nature s

$p(s)$ ..... A priori probability for the state of nature s

$p(s_t|s_{t-1})$ .... Transition probability

pdf ..... Probability density function

PLUNDER .. Pick Least UNDEgenerate Randomly

$\boldsymbol{q}$ ..... Vector part of quaternion

$\boldsymbol{q}$ ..... Quaternion

q ..... Scalar part of Quaternion

$\mathbf{R}$ ..... Rotation matrix

R ..... Radius (car motion)

r_i ..... Residuum associated with i-th datum

R_{fn} ..... False negative rate

R_{fp} ..... False positive rate

R_{tn} ..... True negative rate

R_{tp} ..... True positive rate

RANSAC .... RAndom SAmple Consensus

ROC ..... Receiver Operating Characteristics

s ..... Skew

s ..... State of nature, i.e. either static (SBG) or moving (IMO)

SP ..... Barycenter

SBG ..... Static BackGround

SVD ..... Singular Value Decomposition

$\mathcal{T}$ ..... Trifocal tensor

$\mathcal{T}_i$ ..... Slice i of trifocal tensor

$\mathbf{T}_e$ ..... Euclidean point transformation

$\boldsymbol{t}^{\pm}$ ..... Tangent points on an ellipsoid

t ..... Time

t_c ..... Confidence in the correspondence measurements

$T_{c,d}$ ..... $T_{c,d}$ test

TTC ..... Time To Contact, time to crash, time to impact or time to collision

$\boldsymbol{u}$ ..... Undistortion function

UT ..... Unscented Transform

v ..... Speed

$v(\sigma_v, d)$ ..... Influence function

$\boldsymbol{w}$ ..... Rotation axis

$w_t^{(i)}$ ..... Weight of the i-th particle at time t

$w_{\text{imp},t}^{(i)}$ ..... Weights of i-th importance sample at time t

$\dot{x}$ ..... First temporal derivative of x

$\ddot{x}$ ..... Second temporal derivative of x

$\boldsymbol{x}_{x/y}$ ..... States of the Kalman filter

y_t ..... Observation of the system at time t

Chapter 1

Introduction

1.1 Motivation

An important part of modern automobiles are *driver assistant systems*. These systems, which support the driver in various situations, generally aim at improving traffic safety or, alternatively, at increasing the driving comfort. A number of simpler assistant systems are available for a long time and some of them became quasi standard. Examples for assistant systems aiming at traffic safety include well-known systems like the anti-lock braking system (ABS) or the electronic stability program (ESP), and a number of simpler systems which sometimes require user interaction (i.e. overspeed warning or speed delimiters with manually set limits). Comfort increasing assistant systems include for example cruise control systems or automatic window wiper interval control. These early systems rely on sensor information about the car and its state and require no information about the car environment.

The next generation of advanced driver assistant systems incorporate information about the car environment into the assistant function. Some of these systems are already on the market for some time (e.g. parking aids, lane departure warning systems, emergency braking systems) and have found their way to medium class cars, while others are still only available in luxury class automobiles or in professional vehicles like trucks and busses (e.g. adaptive cruise control, night vision, congestion assistant). These systems rely on environment information which can easily be processed. The emergency braking assistant, for example, only needs to assess the empty space in front of the car together with inertial information like speed, acceleration and steering angle.

Current development is on the border to yet another level of assistant systems. These systems are characterised by their ability to *interpret* complex data from sensors capturing the car environment. Examples for these systems are: (i) traffic sign detection and its applications (e.g. automatic overspeed warning, automatic overspeed delimiters, stop sign detection). (ii) Camera based pedestrian and cyclist detection aims at identifying especially vulnerable road users and incorporates this information in pre-crash scenario decisions and (iii) blind spot monitoring based on video, radar or lidar information is used to feed

complex models of the car environment including position and velocity of all neighbouring road users. These models are used for example for lane change assistants.

An important category of third level driver assistant systems are *intersection assistant systems*. 28 percent of all traffic accidents in Germany happen in complex intersection situations, and the majority of these accidents involve two or more road users. The availablility of a system assisting the driver in cross traffic situations would hence be a major achievement for traffic safety. A system which supports the driver in complex intersections is currently still missing, even though first prototypes for specialised scenarios exists (e.g. traffic light detection, turning assistant, intersection navigation assistant). This thesis investigates an important and obligatory aspect of intersection assistant systems: The detection of *independently moving objects*. While static objects are relatively harmless and can easily be avoided, moving objects are potentially dangerous, and thus a fast and reliable detection is vital. An overall intersection assistant system can only function when information about each individual road user and about his motion is available to the system. Other road users must be detected at rather close distances (approximately up to 50m), with high accuracy and within a large field of view ($\approx 180°$). Information about moving objects can be gathered using a number of different sensors, for example radar, laser range scanner, lidar, sonar or cameras. The decision for visual input as a basis for such a system was mainly inspired by two facts: (i) Human drivers gather information about their environment to a large extent with their eyes and consequently, mimicking nature, image sensors are a natural choice. (ii) Cameras are inexpensive and already available in recent car models, which reduces the barrier to introduce techniques based on cameras to the market.

However, the observation of complex car environments, e.g. road intersections, with camera sensors presents a difficult problem. The sensor must be able to focus and track moving objects like pedestrians and cars. Using the human head as an inspiration, cameras mounted on pan-tilt units are used for this task. The usage of pan-tilt units enlarges the field of view of the camera, and in particular when combined with a digital map, the camera system can in anticipation be directed towards the intersecting road and detect motion at an early stage. Small and light weight sensors are necessary for this application in order to enable fast camera movements. Due to this requirement, heavy stereo rigs are not suitable, and the use of a small *monocular camera system* is necessary. In the absence of stereo information, alternative algorithms have to be used to investigate the car environment. It is a simple task for a human driver to distinguish for example between a parked car and a moving car from visual input only. However, anyone who has tried to distinguish between a moving and a non-moving car with one eye closed while at the same time moving himself, will confirm that this is a challenging exercise.

A driver assistant system for intersection situations operating on image sequences is suggested in this thesis. The primary goal of this *attention guidance system* is to alert the driver to moving objects. It operates on image sequences captured by a single camera and on the associated measurements from inertial sensors. The analysis of optical flow fields

gathered from a single camera is a straightforward approach avoiding heavy and sensitive stereo rigs. This work investigates two important aspects of a visual detection system for independent motion.

The first important aspect is the computation of the egomotion of the camera. The inertial sensors used in this thesis are not sufficiently accurate for egomotion computation and hence visual cues are used. When the egomotion is known, classification of the scene into static background and independent motion is greatly simplified.

The second important aspect of the system is the detection of independent motion from monocular image sequences with known observer motion. It can be broken down into detection of independent motion using only two views and temporal integration of the results. A stochastic approach to detection and integration is suggested.

Contributions

This thesis contains five major contributions to research:

- Practical comparison of different egomotion estimation algorithms based on visual input.
- Theoretical and practical evaluation of point-based classifiers for the detection of independent motion.
- A novel Bayesian framework for detection of independent motion based on the direction classifier using occupation probability maps.[1]
- An algorithm boosting the occupation probability map. The algorithm is inspired by the particle filter and aims at denser sampling of the occupation probability map in interesting regions while at the same time maintaining real-time capabilities.[2]
- The prototype of a complete intersection assistant system has been implemented and was integrated into a real world demonstrator.[3]

1.2 Overview

The remaining part of this thesis is structured as follows:

Chapter 2 summarises basic knowledge about projective geometry, perspective cameras, multiview geometry and optical flow.

An in depth review of previous work on multibody structure and motion and on the automatic detection of independent motion (chapter 3) precedes the two chapters containing the main contributions of this work.

[1] An early version of this work has been published in Woelk et al. (2005).

[2] An early version of this work has been published in Woelk and Koch (2004).

[3] This has been published in Woelk et al. (2004); Woelk and Koch (2004); Woelk et al. (2005).

Precise and fast egomotion estimation is essential for the detection task. The underlying algorithms for the detection of image features, for the estimation of image correspondences and a comparison of different egomotion estimation algorithms are presented in chapter 4.

The second contribution of this work, which consists of the evaluation of different classifiers for independent motion, opens chapter 5. A novel Bayesian framework for independent motion detection based on the most promising classifier is suggested. It constitutes the third contribution. The framework results in an occupation probability map. The final contribution of this work describes a method boosting this probability map. The chapter is closed by the description of the system integration and experimental validation of the complete system.

The final conclusions, which include a summary as well as suggestions for future work, are given in chapter 6.

Chapter 2

Theoretical Background

After a brief introduction of projective geometry, the perspective camera model is described. The mathematical description of the imaging process using the projective camera model prefaces the section about multi-view geometry. It further describes the geometric configuration of two and three cameras viewing the same scene. Finally, the notion of optical flow is introduced.

2.1 Projective Geometry

This section gives a very short introduction of projective geometry, only touching the issues necessary for the comprehension of this thesis. A good and detailed introduction to projective geometry can be found in Hartley and Zissermann (2004).

2.1.1 Points and Lines

The projective space $\mathbb{P}^n$ is generated from the Euclidean space $\mathbb{R}^n$ by extending it about one dimension. The two-dimensional Euclidean space $\mathbb{R}^2$ can be regarded as a plane and each point in this space is uniquely identified - with respect to a certain reference frame - by a tuple (i.e. by a vector) containing its two *Euclidean coordinates* (x, y). The same point in projective coordinates $\mathbb{P}^2$ can be represented by the triplet $(\lambda x, \lambda y, \lambda)$ for any $\lambda \neq 0$. A representation of a point in projective space is called a *projective point*, and its representation is given in *homogeneous coordinates*. The different representation, with the different λ, form an equivalence class of coordinate triplets. This directly leads to the equivalence relation for two homogeneous vectors $\boldsymbol{x}$ and $\boldsymbol{y}$

$$\boldsymbol{x} \doteq \boldsymbol{y} \qquad \Longleftrightarrow \qquad \boldsymbol{x} = \lambda \boldsymbol{y}, \quad \lambda \in \mathbb{R}/0 \tag{2.1}$$

stating that two homogeneous vectors are equal if they are related by a common scale factor $\lambda \neq 0$. This concept is not restricted to two dimensions and can directly be transferred to higher dimensions, i.e. points from $\mathbb{P}^3$ are represented in homogeneous coordinates using a quadruplet.

Any 2D-line is defined by the equation $ax + by + c = 0$, and it can be identified by the triplet $\boldsymbol{l} = (a, b, c)$. Multiplying the triplet by an arbitrary scale factor $\lambda \neq 0$, however, does not alter the line. The representation of a 2D-line by the above triplet is called a *homogeneous line* or the projective representation of a 2D-line. Lines follow the same projective equality relation as points (eq. 2.1).

A 2D-line $\boldsymbol{l}$ is constructed from two points $\boldsymbol{x}$ and $\boldsymbol{y}$ using the cross product

$$\boldsymbol{l} \doteq [\boldsymbol{x}]_\times \boldsymbol{y} \doteq [\boldsymbol{y}]_\times \boldsymbol{x} \tag{2.2}$$

A point $\boldsymbol{x}$ is located on the line $\boldsymbol{l}$ iff

$$\boldsymbol{l}^T \boldsymbol{x} = \boldsymbol{x}^T \boldsymbol{l} = 0 \tag{2.3}$$

2.1.2 Infinity

The Euclidean coordinates can always be recovered from the projective coordinates by dividing through its last component. However, when the last component is zero, division is formally not allowed. This leads to a possibility to express points at infinity simply by setting their last component to zero. Because expressions for entities at infinity can be used, some nice simplifications to traditional geometry can be made. For example, in $\mathbb{P}^2$ two lines do always intersect and hence always define a point. If the lines are parallel, the point may however be at infinity. In Euclidean geometry, two lines intersect iff they are not parallel.

2.1.3 Coordinate Transformations

The representation of entities in the projective space has the additional advantage that any transformation (i.e. Euclidean, similarity, affine or projective) can be expressed using simple matrix multiplication. An Euclidean point transformation $\mathbf{T}_e$ in $\mathbb{P}^2$, for example, can be expressed using the following 3×3 matrix

$$\mathbf{T}_e = \begin{bmatrix} \mathbf{R} & \boldsymbol{c} \\ \boldsymbol{0}^T & 1 \end{bmatrix} \tag{2.4}$$

with an orthogonal 2×2 rotation matrix $\mathbf{R}$ and a 2D Euclidean translation vector $\boldsymbol{c}$. A homogeneous point is transformed by multiplication with the matrix from the left $\boldsymbol{x}' = \mathbf{T}\boldsymbol{x}$. Given a point transformation $\mathbf{T}$, the associated transformation for a line is given by the inverse transposed matrix $\boldsymbol{l}' = \mathbf{T}^{-T}\boldsymbol{l}$.

2.2 Coordinate Systems and Cameras

2.2.1 Coordinate Systems

Each geometric entity can be described in different coordinate systems. All coordinate systems in this work are right-handed. Entities in each coordinate system can be expressed

using either the Euclidean or the projective representation. The different coordinate systems are defined next:

World Coordinate System: The world coordinate system is any arbitrary Euclidean metric system. It has 3 degrees of freedom and is fixed relative to the earth. The position of the camera at the time of the first image is often used to fix the origin and orientation of the world coordinate system in this work, but other methods can also be used. The world coordinate system remains fixed during an image sequence.

Camera Centric Coordinate System: The camera centric coordinate system has also 3 degrees of freedom and is rigidly coupled with the camera. Its origin is fixed at the centre of projection of the camera. The positive z-axis points from the centre of projection towards the object and is parallel to the optical axis of the camera. The x- and y-axes are aligned with the image borders. The positive y-axis points downwards, and the positive x-axis points to the right. The camera centric coordinate system has the same scale as the world coordinate system. It is also called *camera coordinate system*. The camera coordinate system is related to the world coordinate system via an Euclidean transformation. Using projective space, the transformation of a point $\boldsymbol{X}$ can be expressed by a 4×4 matrix containing the rotation matrix $\mathbf{R}$ describing the orientation of the camera in the world coordinate system and the Euclidean position of the camera centre $\boldsymbol{C}$

$$\boldsymbol{X}_{\text{cam}} = \begin{bmatrix} \mathbf{R}^T & -\mathbf{R}^T\boldsymbol{C} \\ \boldsymbol{0}^T & 1 \end{bmatrix} \boldsymbol{X}_{\text{world}} \qquad \boldsymbol{X}_{\text{world}} = \begin{bmatrix} \mathbf{R} & \boldsymbol{C} \\ \boldsymbol{0}^T & 1 \end{bmatrix} \boldsymbol{X}_{\text{cam}} \tag{2.5}$$

Normalised Image Coordinate System: The normalised image coordinate system has 2 degrees of freedom and is rigidly coupled to the camera. Its origin is located at the principal point, i.e. the intersection of the optical axis and image plane, and its axes are aligned with the image borders. The x-axis is aligned with the horizontal border and the y-axis is aligned with the vertical axis. The positive y-axis points downwards. The normalised image coordinate system does not impose constraints on the norm of the vectors. The name is merely used to indicate the difference in scale and origin compared to the pixel coordinate system.

Pixel Coordinate System: The pixel coordinate system has 2 degrees of freedom and its origin is located at the upper left corner of the image. It has a fixed scale such that one unit equals the size of a pixel. The positive y-axis points downwards, and the positive x-axis points to the right.

2.2.2 Orthographic Camera

The *orthographic projection* is also called *parallel projection*. All viewing rays in an orthographic camera are orthogonal to the image plane and hence parallel. Figure 2.1 illustrates

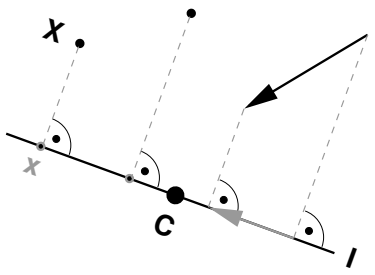


Figure 2.1: Orthographic camera projecting 3D-points $\boldsymbol{X}$ resulting in 2D-points $\boldsymbol{x}$. All viewing rays are perpendicular to the image plane I. The centre of projection $\boldsymbol{C}$ defines the origin of the camera centric coordinate system.

an orthographic camera. The orthographic camera can be modelled in Euclidean space

$$\boldsymbol{x} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{R}^T & -\mathbf{R}^T \boldsymbol{C} \end{bmatrix} \begin{pmatrix} \boldsymbol{X} \\ 1 \end{pmatrix} \tag{2.6}$$

with the 3D-point $\boldsymbol{X}$ and its 2D-projection $\boldsymbol{x}$. The camera orientation is given by the 3D-rotation matrix $\mathbf{R}$, and the camera position is given by $\boldsymbol{C}$. One important property of the orthographic camera is that the image only depends on the orientation of the camera. Translation of the camera just shifts the internal reference frame and does not alter the image when all points stay in front of the camera. The orthographic projection is mainly of theoretic interest and is often used to approximate perspective cameras with very large focal lengths or to approximate perspective projection when only a very small image patch is of interest.

2.2.3 Perspective Camera

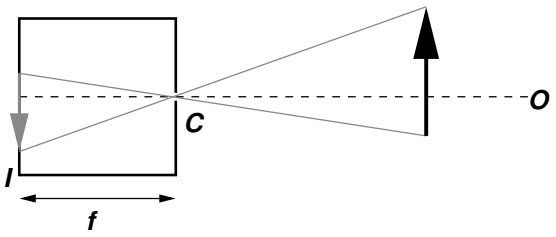


Figure 2.2: Pinhole camera. The distance between the image plane $\boldsymbol{I}$ and the centre of projection $\boldsymbol{C}$ is called the focal length f. The optical axis is denoted with $\boldsymbol{O}$.

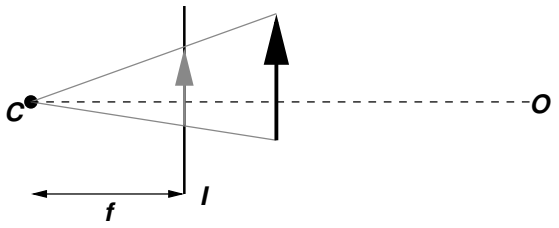


Figure 2.3: Schematic sketch of a pinhole camera. The distance between the image plane $\boldsymbol{I}$ and the centre of projection $\boldsymbol{C}$ is called the focal length f. The optical axis is denoted with $\boldsymbol{O}$. This schematic sketch has the advantage of neglecting the mirror effect of the true pinhole camera.

The most simple perspective camera model is the pinhole camera (fig. 2.2). Using the pinhole camera model, each viewing ray passes through an infinitesimally small hole at the centre of projection $\boldsymbol{C}$ before intersecting the image plane I at distance f from the camera centre $\boldsymbol{C}$. The focal length f of a pinhole camera is given by the distance between the centre of projection (the pinhole) and the image plane. Often the image plane is drawn in front of the centre of projection (fig. 2.3), circumventing the fact that the image is mirrored in the pinhole camera. The camera obscura is a technical realisation of the pinhole camera with a very small hole (typically in the range of mm).

Fixing the coordinate system such that the origin coincides with the centre of projection $\boldsymbol{C}$, the positive z-axis is perpendicular to the image plane and the x- and y-axes are parallel to the borders of the rectangular image, a ray through the point $\boldsymbol{X} = (x, y, z)^T$ intersects the image plane at $\boldsymbol{x}$.

$$\boldsymbol{x} = \begin{pmatrix} x \\ y \\ z \end{pmatrix} \frac{f}{z} = \begin{pmatrix} x\frac{f}{z} \\ y\frac{f}{z} \\ f \end{pmatrix} \tag{2.7}$$

When the coordinate system is further defined such that $f = 1$, the projection of a point on the image plane can computed by simply dividing by its z component.

Projection Matrix P

Using projective space, perspective projection can be linearly modelled by a *projection matrix* $\mathbf{P} \in \mathbb{R}^{3\times4}$

$$\boldsymbol{x} \doteq \mathbf{P}\boldsymbol{X} \tag{2.8}$$

with the sign for the projective equality from eq. 2.1. Apart from the 3D-rotation matrix $\mathbf{R}$ describing the camera orientation and the camera centre $\boldsymbol{C}$, the projection matrix contains the calibration matrix $\mathbf{K}$. The calibration matrix $\mathbf{K} \in \mathbb{R}^{3\times3}$ describes the transformation between the normalised image coordinate and pixel coordinate system of an ideal pinhole camera

$$\mathbf{K} = \begin{pmatrix} f & s & c_x \\ 0 & af & c_y \\ 0 & 0 & 1 \end{pmatrix} \tag{2.9}$$

with focal length f, aspect ratio a, skew s and principal point $\boldsymbol{c} = (c_x, c_y)^T$. A homogeneous point $\boldsymbol{x}_{\text{img}}$ in image coordinate system is related to its projective representation in pixel coordinate system $\boldsymbol{x}_{\text{pixel}}$ by

$$\boldsymbol{x}_{\text{pixel}} = \mathbf{K} \cdot \boldsymbol{x}_{\text{img}} \tag{2.10}$$

$\mathbf{K}$ can be analytically inverted:

$$\mathbf{K}^{-1} = \begin{pmatrix} \frac{1}{f} & -\frac{s}{af^2} & \frac{sc_y}{af^2} - \frac{c_x}{f} \\ 0 & \frac{1}{af} & -\frac{c_y}{af} \\ 0 & 0 & 1 \end{pmatrix} \tag{2.11}$$

And hence

$$\boldsymbol{x}_{\text{img}} = \mathbf{K}^{-1} \cdot \boldsymbol{x}_{\text{pixel}} \tag{2.12}$$

Perspective projection of a 3D-point given in projective camera coordinates can be expressed by a 3×4 identity matrix and results in the 2D-projection onto the image plane given in projective image coordinates.

$$\boldsymbol{x}_{\text{img}} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \boldsymbol{X}_{\text{cam}} \tag{2.13}$$

The projection matrix captures all coordinate system transformations (world to camera to image to pixel) in one matrix. It can be composed as follows

$$\mathbf{P} = \mathbf{K} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{R}^T & -\mathbf{R}^T \boldsymbol{C} \\ \boldsymbol{0}^T & 1 \end{bmatrix} = \mathbf{K} \left[\mathbf{R}^T \middle| -\mathbf{R}^T \boldsymbol{C} \right] \tag{2.14}$$

Using a perspective camera model, a point in normalised image coordinates in its projective representation ($\mathbb{P}^2$) can also be interpreted as a vector indicating the direction of the corresponding viewing ray in 3D Euclidean space and vice versa: Each 3D-point given in its representation in Euclidean camera coordinate system can be interpreted as the projective representation of its 2D-projection.

2.2.4 Lens Distortion

Real cameras usually differ from an ideal (e.g. pinhole) camera because of physical effects like for example lens distortion, spherical aberration, colour dispersion (chromatic aberration), astigmatism, etc. (Tippler, 1994; McGlone, 2004). In particular the effect of the lens distortion cannot be neglected for real cameras with a wide aperture angle or cheap lenses. This section describes a model for the description of the lens distortion. The correction of the lens distortion is an important part of many algorithms and an important step in this thesis.

One of the first closed form solutions to the calibration problem can be found in Tsai (1987). This chapter is however based on the lens distortion model from J. Heikkilae (1997). An implementation of the calibration can be found at Bouguet (1998).

The focal length of a real lens is dependent of the distance from the centre of distortion $\boldsymbol{c}_D = (\overline{x}, \overline{y})^T$. Because of the distance between the aperture and the lens in real cameras, this fact causes curvilinear lens distortion. The curvilinear lens distortion can be described by the distortion function $L(r^2)$

$$\boldsymbol{x}_r = L(r^2)\boldsymbol{x}. \tag{2.15}$$

with the distorted image point $\boldsymbol{x}_r$ and the ideal image point $\boldsymbol{x} = (x_x, x_y)^T$ as it would have been imaged by an ideal pinhole camera. The distortion function $L(r^2)$ only depends on the square distance $r^2 = (x_x - \overline{x})^2 + (x_y - \overline{y})^2$ from the centre of distortion $\boldsymbol{c}_D = (\overline{x}, \overline{y})$. A common assumption is that the principal point $\boldsymbol{c} = (c_x, c_y)^T$ and the centre of distortion $\boldsymbol{c}_D$ coincide. The distortion function is approximated by a polynomial

$$L(r^2) = 1 + \kappa_1 r^2 + \kappa_2 r^4 + \kappa_3 r^6 + \dots \tag{2.16}$$

Only the first three coefficients of the polynomial $\kappa_1, \ldots, \kappa_3$ are used in this model.

Another possible distortion is the tangential distortion. It is caused by the fact that the centres of curvatures of lens surfaces are not always strictly collinear (J. Heikkilae, 1997). Mathematically this can be modelled as

$$\boldsymbol{x}_t = \boldsymbol{x} + \begin{pmatrix} 2\kappa_4 x_x x_y + \kappa_5(r^2 + 2x_x^2) \\ \kappa_4(r^2 + 2x_y^2) + 2\kappa_5 x_x x_y \end{pmatrix} \tag{2.17}$$

with the tangential distortion coefficients κ_4 and κ_5.

Combining both distortion models, a point $\boldsymbol{x}$ in an ideal pinhole camera is distorted to the point $\boldsymbol{x}_d$ in the real camera

$$\boldsymbol{x}_d = (1 + \kappa_1 r^2 + \kappa_2 r^4 + \kappa_3 r^6) \begin{pmatrix} x_x \\ x_y \end{pmatrix} + \begin{pmatrix} 2\kappa_4 x_x x_y + \kappa_5(r^2 + 2x_x^2) \\ \kappa_4(r^2 + 2x_y^2) + 2\kappa_5 x_x x_y \end{pmatrix} \tag{2.18}$$

The image point $\boldsymbol{x}$ must be given in the camera coordinate system. The distortion model describes the disturbance of a pinhole camera by real lens systems. In practise, the inverse distortion function is required to correct the lens distortion and compute the ideal image point $\boldsymbol{x}$.

Correction of Lens Distortion

(a) (b) (c)

Figure 2.4: Displacement maps for lens distortion correction. Light pixel mean positive displacement, dark pixel mean negative displacement, medium pixel mean no displacement. The horizontal displacement for correction of lens distortion can be seen in image (a), the vertical displacement for correction of lens distortion can be seen in image (b). The magnitude of the displacement is visualised in image (c). The displacement maps have been generated using real camera parameter and distortion coefficients ($f_x = 839.675, f_y = af_x = 840.567, c_x = 314.05, c_y = 247.43, \kappa_1 = -0.1005250, \kappa_2 = 0.1003916, \kappa_3 = 0.0000000, \kappa_4 = 0.0023025, \kappa_5 = -0.0022120$)

The inverse distortion function is needed to compensate the lens distortion. Because the distortion function (eq. 2.18) cannot be inverted analytically, the inversion is computed iteratively. The approximate inverse distortion function (equation 2.19) is computed until

the computation converges, i.e. until the update between two iteration steps $|\boldsymbol{x}_i - \boldsymbol{x}_{i+1}|$ falls below a certain threshold t:

$$\boldsymbol{x}_{i+1} = \begin{pmatrix} x_{x,i+1} \\ x_{y,i+1} \end{pmatrix} = \frac{1}{(1 + \kappa_1 r_i^2 + \kappa_2 r_i^4 + \kappa_3 r_i^6)} (\boldsymbol{x}_d - \begin{pmatrix} 2\kappa_4 x_{x,i} x_{y,i} + \kappa_5 (r_i^2 + 2x_{x,i}^2) \\ \kappa_4 (r_i^2 + 2x_{y,i}^2) + 2\kappa_5 x_{x,i} x_{y,i} \end{pmatrix}) \quad (2.19)$$

with the squared distance from the centre of distortion (i.e. from the principal point) $r_i^2 = x_{x,i}^2 + x_{y,i}^2$. The iteration is initialised with $\boldsymbol{x} = \boldsymbol{x}_d$. The displacement map can be computed offline using this algorithm. Fast correction of lens distortion computation can be realised by a simple look up and bi-linear interpolation in the displacement maps. Figure 2.4 visualises the look up maps.

2.3 Multi-View Geometry

2.3.1 Two-View Geometry

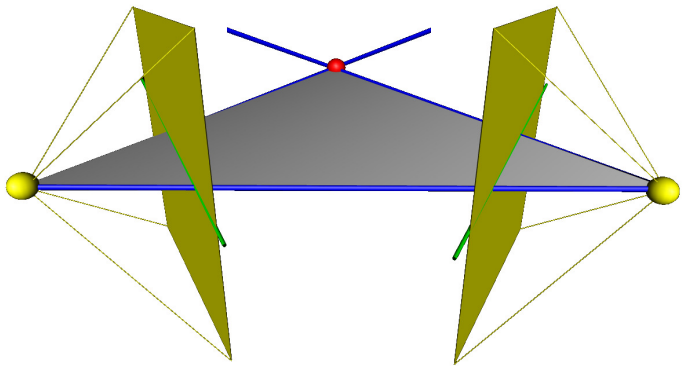


Figure 2.5: Two-view geometry. A point in space (red) is seen by two cameras (yellow). The epipolar plane (grey) is defined by the two camera centres and the point in space. The epipolar lines (green) are given by the intersection of the epipolar plane with the image planes. The epipoles (blue) are given by the projections of the other camera centre.

Epipole *e* and Epipolar Line

The two-view geometry is shown in figure 2.5. Two cameras see the same point in space (red). If the scene is static, two images taken with the same moving camera at two different times represent exactly the same geometric configuration. The epipoles are given by the intersections of the line (blue) between the two camera centres with the image planes. The

epipolar plane (grey) is defined by the point in space and the two camera centres. The epipolar lines (green) are the intersections of the epipolar plane with the image planes.

Fundamental Matrix F

The fundamental matrix $\mathbf{F}_{12}$ describes the relation between two images taken with uncalibrated cameras. It has 7 degrees of freedom and can be constructed from the epipole $\hat{\boldsymbol{e}}$ and a homography mapping $\mathbf{H}_{12}$. When the calibrations of the cameras are known, the homography can be computed as $\mathbf{H}_{12} = \mathbf{K}_2\mathbf{R}_2^{-1}\mathbf{R}_1\mathbf{K}_1^{-1}$

$$\mathbf{F}_{12} = [\hat{\boldsymbol{e}}_2]_\times \mathbf{H}_{12} \tag{2.20}$$

$\mathbf{F}_{12}$ is a 3×3 matrix mapping one image point $\hat{\boldsymbol{x}}_1$ to its corresponding epipolar line $\hat{\boldsymbol{l}}_2$ in the other image.

$$\hat{\boldsymbol{l}}_2 = \mathbf{F}_{12}\hat{\boldsymbol{x}}_1 \tag{2.21}$$

The right and left nullspaces of $\mathbf{F}$ are the epipoles

$$\hat{\boldsymbol{e}}_2^T\mathbf{F}_{12} = 0 \qquad \mathbf{F}_{12}\hat{\boldsymbol{e}}_1 = 0 \qquad \hat{\boldsymbol{e}}_1^T\mathbf{F}_{21} = 0 \qquad \mathbf{F}_{21}\hat{\boldsymbol{e}}_2 = 0 \tag{2.22}$$

The fundamental matrix between image 2 and 1 is the transpose of the fundamental matrix between image 1 and 2 (Hartley and Zissermann, 2004)

$$\mathbf{F}_{12} = \mathbf{F}_{21}^T \tag{2.23}$$

A pair of corresponding image points $\hat{\boldsymbol{x}}_1$ and $\hat{\boldsymbol{x}}_2$ obeys the fundamental constraint

$$0 = \hat{\boldsymbol{x}}_1^T\mathbf{F}_{21}\hat{\boldsymbol{x}}_2 = \hat{\boldsymbol{x}}_1^T\hat{\boldsymbol{l}}_1 \tag{2.24}$$

Degeneracies: When the camera centres coincide, and thus the cameras are related by a pure rotation, the fundamental matrix is not defined. In this case, the point correspondences can be described by a simpler model called homography (Hartley and Zissermann, 2004).

As long as the two camera centres are distinct, the fundamental matrix is uniquely defined. It is however impossible to compute a unique fundamental matrix when the set of 3D-points, from which the 2D-point correspondences are created, are located on a specific geometric configuration:

- When all 3D-points and the camera centres are located on a ruled quadric (Hartley and Zissermann, 2004), three possible solutions exist. Hence the ruled quadric is a *critical surface* for fundamental matrix estimation.
- When all 3D-points are coplanar, they are located on a degenerate quadric consisting of two planes. In this case, the point correspondences can also be described by a simpler model, namely the homography.

Essential Matrix E

With known calibration matrix $\mathbf{K}$, it is possible to do all calculations in normalised image coordinates $\boldsymbol{x} = \mathbf{K}^{-1}\hat{\boldsymbol{x}}$. The essential matrix $\mathbf{E}$ is the analogue to the fundamental matrix in this case. It is a rank 2 matrix with 5 degrees of freedom (2 for the epipole and 3 for the rotation). For known camera motion it can be composed from the epipole $\boldsymbol{e}$ and the relative rotation between the two views $\mathbf{R}$:

$$\mathbf{E}_{12} = [\boldsymbol{e}_2]_\times \mathbf{R}_{12} \tag{2.25}$$

The *essential constraint* is the equivalent to the fundamental constraint for a point correspondence in normalised image coordinates $\hat{\boldsymbol{x}}_1$ and $\hat{\boldsymbol{x}}_2$. It is given by

$$\boldsymbol{x}_2^T \mathbf{E}_{12} \boldsymbol{x}_1 = 0 \tag{2.26}$$

Because the essential matrix has rank 2, its determinant vanishes

$$\det(\mathbf{E}) = 0 \tag{2.27}$$

The internal structure of the essential matrix can be enforced by additionally ensuring the *decomposability conditions* (Kanatani, 2005)

$$||\mathbf{E}|| = \sqrt{2} \qquad ||\mathbf{E}\mathbf{E}^T|| = \sqrt{2} \tag{2.28}$$

Alternatively, the internal structure of the essential matrix can be expressed using the following constraint (Nistér, 2004; Philip, 1996)

$$\mathbf{E}\mathbf{E}^T\mathbf{E} - \frac{1}{2}\operatorname{trace}(\mathbf{E}\mathbf{E}^T)\mathbf{E} = 0 \tag{2.29}$$

Given an essential matrix $\mathbf{E}$, the relative orientation $\mathbf{R}$ and the epipole $\boldsymbol{e}$ can be extracted as follows (Kanatani, 2005):

1. The epipole - up the sign - $\boldsymbol{e}$ is the unit norm eigenvector corresponding to the smallest eigenvalue of $\mathbf{E}\mathbf{E}^T$.

2. The sign of the epipole can be estimated by ensuring

$$\sum_i (\boldsymbol{e} \times \boldsymbol{x}_i)^T \mathbf{E} \boldsymbol{x}_i' > 0 \tag{2.30}$$

   for the point correspondences between $\boldsymbol{x}_i$ and $\boldsymbol{x}_i'$.

3. The orientation $\mathbf{R}$ can be computed using the singular value decomposition (SVD) of the matrix $-[\boldsymbol{e}]_\times \mathbf{E} = \mathbf{U}\mathbf{S}\mathbf{V}^T$

$$\mathbf{R} = \mathbf{V} \operatorname{diag}(1, 1, \det(\mathbf{V}\mathbf{U}^T))\mathbf{U}^T \tag{2.31}$$

When the essential matrix is decomposable (i.e. if its eigenvalues are 1, 1, and 0), a more efficient step can be used to replace the computation of the relative orientation (Kanatani, 2005). If column vectors of the essential matrix $\mathbf{E}$ and rotation matrix $\mathbf{R}$ are given by $\mathbf{E} = (\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3)$ and $\mathbf{R} = (\mathbf{r}_1, \mathbf{r}_2, \mathbf{r}_3)$

$$\mathbf{r}_i = \mathbf{e}_i \times \mathbf{e} + \mathbf{e}_{i+1} \times \mathbf{e}_{i+2} \tag{2.32}$$

Hartley and Zissermann (2004) suggest a slightly different approach on the decomposition of the essential matrix. Given the matrices

$$\mathbf{W} = \begin{pmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{2.33}$$

and

$$\mathbf{Z} = \begin{pmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \tag{2.34}$$

and the SVD of $\mathbf{E} = \mathbf{U}\mathbf{S}\mathbf{V}^T$, the epipole (up to its sign) is given by

$$[\boldsymbol{e}]_\times = \mathbf{U}\mathbf{Z}\mathbf{U}^T \tag{2.35}$$

and the relative orientation is given either by

$$\mathbf{R} = \mathbf{U}\mathbf{W}\mathbf{V}^T \quad \text{or} \quad \mathbf{R} = \mathbf{U}\mathbf{W}^T\mathbf{V}^T \tag{2.36}$$

resulting in four possible solutions. The true solution can be extracted by ensuring that the triangulations of point correspondences result in 3D-points located in front of both cameras. The baseline can be fixed to an arbitrary value $\neq 0$ for the triangulation. This approach has the advantage that it works also when the essential matrix is not strictly decomposable, i.e. its singular values are not necessarily 1, 1 and 0, because the singular values are not used in this approach.

Degeneracies: With distinct camera centres, a unique solution can be computed from 2D-correspondences even if the generating 3D-points are located on a plane. A critical configuration for the five point algorithm from Philip (1998) is when four or five 3D-points generating the 2D-point correspondences are sitting on a straight line (Philip, 1998).

Decomposition Degeneracies: When the internal calibration of the camera is known, two solutions exist when all 3D-points are located on a plane and when the baseline is not perpendicular to the plane. However, the cheirality constraint[1] can be used to resolve the twofold ambiguity (Nistér, 2004). When the baseline is perpendicular to the plane, a unique solution exists.

[1]The cheirality constraint states that all scene points must be in front of the cameras.

2.3.2 Three-View Geometry

Trifocal Tensor $\mathcal{T}$

The trifocal tensor is the three-view analogy to the fundamental matrix in two views. The trifocal tensor completely describes the relation between three images. It can be seen as a collection of three 3×3 matrices $\mathcal{T}_i$ or, alternatively, as a three-dimensional tensor of size $3 \times 3 \times 3$ with real entries.[2] The trifocal tensor has 27 real entries but only 18 degrees of freedom (DOF) (Hartley and Zissermann, 2004). If the internal calibrations $\mathbf{K}_i$ of the cameras are known, the trifocal tensor has only 11 DOF. There are 6 DOF for the relative orientations between the cameras and 6 DOF for the relative translations of the cameras. An overall scale factor remains undetermined resulting in an overall of 11 degrees of freedom.

When the calibrations of the cameras are known, the trifocal tensor can be computed from the positions and orientations of three cameras. Without the loss of generality, the extrinsics of the cameras can be described in the camera coordinate system of the first camera. Let $\mathbf{R}_{12}^T = [\mathbf{a}_1|\mathbf{a}_2|\mathbf{a}_3]$ and $\mathbf{R}_{13}^T = [\mathbf{b}_1|\mathbf{b}_2|\mathbf{b}_3]$ be the relative orientations and $\boldsymbol{t}_{12}$ and $\boldsymbol{t}_{13}$ be the relative translations in camera coordinate system of the first camera. If $\mathbf{a}_4$ and $\mathbf{b}_4$ are defined as follows

$$\mathbf{a}_4 = -\mathbf{R}_{12}^T \cdot \boldsymbol{t}_{12} \qquad \mathbf{b}_4 = -\mathbf{R}_{13}^T \cdot \boldsymbol{t}_{13} \tag{2.37}$$

the tensor matrices $\mathcal{T}_i$ can the be constructed by (Hartley and Zissermann, 2004)

$$\mathcal{T}_i = \mathbf{a}_i \mathbf{b}_4^T - \mathbf{a}_4 \mathbf{b}_i^T \tag{2.38}$$

The equivalent to the fundamental constraint in three views is the trilinear relation for point-point-point correspondences

$$[\boldsymbol{x}']_\times (\sum_i x_i \mathcal{T}_i)[\boldsymbol{x}'']_\times = \mathbf{0} \tag{2.39}$$

with the point correspondence between $\boldsymbol{x} = (x_1, x_2, x_3)^T$, $\boldsymbol{x}'$ and $\boldsymbol{x}''$ and the 3×3 zero matrix $\mathbf{0}$. Trilinear relations for other geometric entities also exist (Hartley and Zissermann, 2004).

2.4 Optical Flow

Optical flow is the projection of a 3D motion field. Consider a static camera observing a moving scene. The motion of each 3D-point in the scene between two frames can be expressed using a 3D motion vector. The projection of these motion vectors into the image is the optical flow field. It is a 2D-vector field indicating the apparent motion of each 2D-point in the image.

[2] A three-dimensional tensor can be seen as an three-dimensional collection of numbers. In this case a cube with edges of length 3 and a real number in each cell.

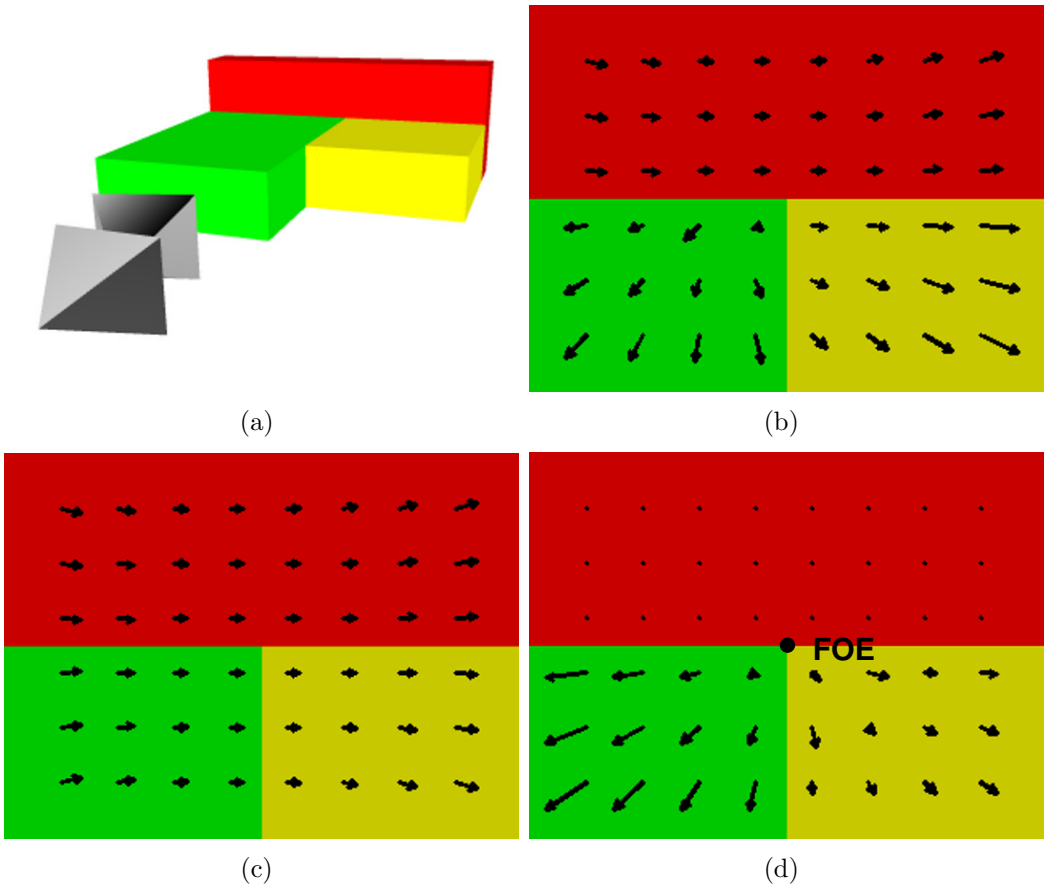


Figure 2.6: Theoretical flow field for a simple scene. The 3D-scene is shown in (a). The scene consists of 3 blocks with different distance from the cameras. The camera, displayed as small pyramids, translates towards the scene while rotating around the vertical axis. The flow field $\mathcal{O}$ as induced by this movement is shown in (b). Its rotational component $\mathcal{O}_r$ (c) and translational component $\mathcal{O}_t$ (d) with the Focus of Expansion (FOE) are shown in the bottom row.

Consider the optical flow induced by the translational velocity $\boldsymbol{V}_c$ of a camera observing a rigid static scene. Changing the frame of reference using a Galilean transformation (Tippler, 1994) does not alter the imaging process, and hence the same flow field arises from a camera moving with $\boldsymbol{V}'_c = \boldsymbol{V}_c + \boldsymbol{V}_a$ viewing the same rigid scene moving with $\boldsymbol{V}'_s = \boldsymbol{V}_a$. Only the relative velocity $\boldsymbol{V}_r = \boldsymbol{V}'_c - \boldsymbol{V}'_s = \boldsymbol{V}_c$ between camera and object determines the optical flow field. Similar considerations lead to the fact that only the relative rotation between camera and object is relevant for the optical flow field. In the following, all considered motions and rotations are relative.

Obviously, the notion of optical flow can be applied independently to any relative motion between a camera and an observed object. Assuming a rigid and static scene and one or more rigid objects moving in the scene, observed by a moving camera, it is therefore feasible to apply the notion of optical flow independently to both the observed scene and the observed objects.

2.4.1 Composition of Optical Flow

An arbitrary rigid motion can always be described by a rotation $\mathbf{R}$ followed by a translation $\boldsymbol{t}$. Hence the optical flow field $\mathcal{O}$ induced by the motion is the superposition of the optical flow field $\mathcal{O}_r$ resulting from the rotation $\mathbf{R}$ and the optical flow field $\mathcal{O}_t$ resulting from the translation $\boldsymbol{t}$ (see fig. 2.6). Optical flow resulting from a rotation is independent of the scene geometry and can be computed directly if the rotation is known. A flow field induced by pure translation has simple geometric properties: All flow vectors point radially away from the focus of expansion (FOE) and radially towards the focus of contraction (FOC). The locations of the FOE and FOC are determined by the camera translation and do not necessarily lie in the image. The length of each flow vector depends on the distance of the projected object from the camera centre and is hence scene-dependent. See for example fig. 2.6(d), the FOE is located in the image centre.

Mathematically, both parts of the flow field can be distinguished using the notions of curl and divergence from vector analysis. The curl of the translational part of the optical flow vanishes

$$\operatorname{curl}(\mathcal{O}_t) = \boldsymbol{0} \qquad \operatorname{curl}(\mathcal{O}_r) \neq \boldsymbol{0} \tag{2.40}$$

with the curl of a vector field $\boldsymbol{X}$

$$\operatorname{curl}(\boldsymbol{X}) = \operatorname{curl}\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} \frac{\partial z}{\partial \mathbf{e}_y} - \frac{\partial y}{\partial \mathbf{e}_z} \\ \frac{\partial x}{\partial \mathbf{e}_z} - \frac{\partial z}{\partial \mathbf{e}_x} \\ \frac{\partial y}{\partial \mathbf{e}_x} - \frac{\partial x}{\partial \mathbf{e}_y} \end{pmatrix} \tag{2.41}$$

given in Cartesian coordinates with basis of unit vectors $\mathbf{e}_x, \mathbf{e}_y, \mathbf{e}_z$. Or, in slightly abusive notation, $\operatorname{curl}(\boldsymbol{X}) = \nabla \times \boldsymbol{X}$ with the nabla operator ∇ defined as usual:

$$\nabla = \begin{pmatrix} \frac{\partial}{\partial \mathbf{e}_x} \\ \frac{\partial}{\partial \mathbf{e}_y} \\ \frac{\partial}{\partial \mathbf{e}_z} \end{pmatrix} \tag{2.42}$$

The divergence of the rotational part of the optical flow field vanishes

$$\operatorname{div}(\mathcal{O}_r) = \boldsymbol{0} \tag{2.43}$$

with divergence of a vector field $\boldsymbol{X}$

$$\operatorname{div}(\boldsymbol{X}) = \operatorname{div}\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \frac{\partial x}{\partial \mathbf{e}_x} + \frac{\partial y}{\partial \mathbf{e}_y} + \frac{\partial z}{\partial \mathbf{e}_z} \tag{2.44}$$

given in Cartesian coordinates with basis of unit vectors $\mathbf{e}_x, \mathbf{e}_y, \mathbf{e}_z$ or, in slightly abusive notation $\operatorname{div}(\boldsymbol{X}) = \nabla^T \boldsymbol{X}$.

Note that the divergence of the translational part of the vector field does not necessarily exist. When, for example, the camera moves parallel to the image plane, all flow vectors

are exactly parallel to each other and hence the divergence of the translational part of the optical flow field vanishes.

When the field of view (FOV) of a camera is narrow, the distinction between rotation around any axis perpendicular to the optical axis of the camera and translation becomes ambiguous. When the camera is rotating around an axis $\boldsymbol{A}_r$ perpendicular to the optical axis $\boldsymbol{O}$, the rotational field is approximately constant in the image and thus resembles the translational flow field induced by translation parallel to the vector $\boldsymbol{O} \times \boldsymbol{A}_r$ while viewing a scene with constant depth. The rotation around the optical axis is however not affected by this ambiguity.

2.4.2 Parameter Estimation and Error Model

The estimation of the underlying parameters θ of a process is called *parameter estimation*. Often the parameters are hidden and cannot be measured directly. Input to parameter estimation algorithms are a number of measurements y which are connected to the parameters by a measurement function f

$$y = f(\theta) \tag{2.45}$$

Typically, the measurement process f cannot be modelled in all details, and hence a simpler approximation $\hat{f}$ of the measurement process is used

$$y = \hat{f}(\theta) + \nu \tag{2.46}$$

leading to an error ν. The errors resulting from the simplified measurement model can be described as a random variable with some underlying distribution. In many cases, the normal distribution is a good approximation. The distribution of ν is generally unknown and can be estimated from the data.

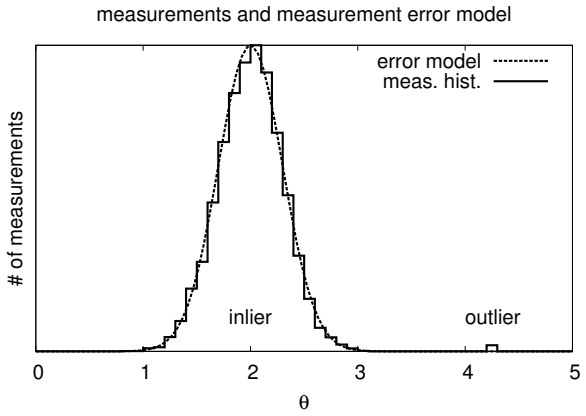


Figure 2.7: Measurement of a scalar value $\theta = 2.0$. The measurement process of the inlier can be described by a normal distribution with mean 2.0 and standard deviation 0.3. The measurement at 4.2 cannot be explained by the measurement model and is hence called an outlier.

Outlier: The occurrence of gross measurement errors is a common problem encountered in parameter estimation practise. Gross measurement errors cannot be explained by equation 2.46. These errors occur, for example, when underlying assumptions about the measurement process are not fulfilled. Measurements which can be explained by equation 2.46 are called *inlier* and measurements corrupted by gross errors are called *outlier*. Figure 2.7 illustrates the histogram of measurements of a scalar value x. Most measurements can be modelled by a normal distribution with mean at 2.0 and standard deviation $\sigma = 0.3$. The measurement at $\theta = 4.2$ cannot be easily explained by the above noise model and is hence declared to be an outlier. The occurrence of a significant number of outliers is typical for computer vision applications. Examples include correspondences between 2D image points or correspondences between 3D-points and their projections onto the image plane.

Leverage Point: A *leverage point* is an outlier with a large effect on the estimated parameters. The difference between common outlier and a leverage point is illustrated in figure 2.8. The parameters of a line are estimated from 5 points which include one outlier. The points and the final line are drawn, the true position of the outlier is indicated by a circle. The estimate of the line is not influenced by the outlier in figure 2.8(a). The outlier in figure 2.8(b) on the right side acts as a leverage point. It has a large effect on the estimated parameters and actually tilts the inclination of the line.

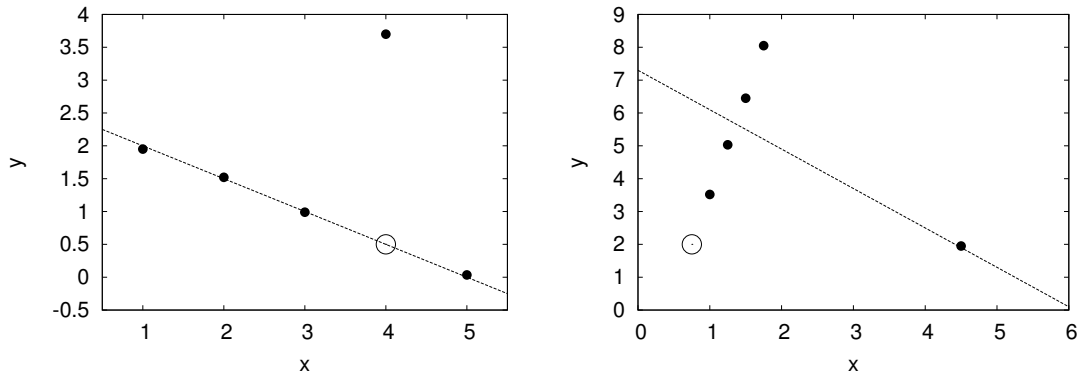


Figure 2.8: (a) Robustness of L_1 regression with respect to an outlier in y direction and (b) sensitivity of L_1 regression to a leverage point. Similar illustrations can be found in Rousseeuw and Leroy (1987)

Robust Estimators are algorithms capable of dealing with outliers in measurements. An overview over robust parameter estimation methods working in the presence of outliers is given in appendix C.1.

Breakdown Point: The breakdown point of an estimator is an important commonly used concept for the evaluation of a robust estimator: Given an estimator T and a sample Z, the breakdown point is "... the smallest fraction of (arbitrary) contamination (in Z) that can cause the estimator T to take on values arbitrarily far from $T(Z)$." (Rousseeuw and Leroy, 1987). From this definition follows the breakdown point of $\frac{1}{N}$ (which is equal to one datum) for a least squares estimator with N data points.

Chapter 3

Previous Work on Detection of Independent Motion

Detection of independently moving objects from visual cues has been subject to many research efforts. Grouping of the research can be based on the hardware setup: Stationary cameras, stereo cameras and monocular freely moving cameras have been used for the task. The subject of this thesis is the detection of independent motion with a monocular, freely moving camera, and hence the main focus in this summary of previous work is laid on literature using such a hardware setup. A short overview for stationary cameras and stereo cameras is given nonetheless.

3.1 Stationary Camera

Stationary cameras are popular for example in surveillance and video conferencing systems. Sometimes the cameras are mounted on pan-tilt units.

The case of a single stationary rotating camera is also important for intersection assistant systems. The situation of a stationary camera occurs when the car is stopped, for example in front of a traffic light or in front of a stop sign. The assistant system must be able to cope with this situation and incorporate the information gained with a stationary camera into the decision process.

When using a single stationary and possibly rotating camera, background subtraction techniques are often used, for example in Evers-Senne et al. (2002). Some authors have used Bayesian approaches for online background model generation (Hayman and Eklundh, 2003).

The approach suggested in this thesis is, however, not based on background subtraction techniques. The algorithm uses a common approach for stationary and freely moving cameras based on the flow length. One advantage of this unifying approach is the smooth transition between the two states (stationary and moving) of the camera.

3.2 Stereo Camera Systems

Stereo camera setups resemble the human perception system with its two eyes and seem the natural approach for visual perception. In comparison to single camera systems, they have the distinctive advantage of being able to estimate 3D-geometry independent from the observer or object motion. This advantage comes, however, at the cost of a higher sensitivity to calibration, a higher price and bigger space requirement. Therefore, the usability of monocular cameras for visual intersection systems is investigated in this thesis. A short summary of existing stereo systems for detection of independent motion is given nonetheless.

Heinrich (2002) formulated the *flow-depth constraint* for the detection of independent motion using a stereo camera setup. Given a standard stereo rig with focal length f, baseline b, the disparity d of a 3D-point $(X, Y, Z)^T$ is given by

$$d = \frac{fb}{Z} \tag{3.1}$$

When the camera is translating along its optical axis with velocity $\dot{Z}$, the optical flow vector $\boldsymbol{x} = (\dot{x}, \dot{y})^T$ of the same 3D-point is constrained by

$$\frac{\dot{x}}{x} = \frac{\dot{Z}}{Z} \qquad \frac{\dot{y}}{y} = \frac{\dot{Z}}{Z} \tag{3.2}$$

The flow-depth constraints are derived from equation 3.1 and 3.2

$$\frac{\dot{x}}{d} = \frac{\dot{Z}}{bf}x \qquad \frac{\dot{y}}{d} = \frac{\dot{Z}}{bf}y \tag{3.3}$$

A threshold t_{fd} on the error of the quotients is derived from the known accuracies of the flow and disparity measurements, and independent motion is detected when the equalities of equations 3.3 violated by more than t_{fd}. The authors suggest compensation of camera rotation using a matched filter method enabling operation on real image sequences with camera rotation.

In 2006 Franke et al. approached the problem from a slightly different angle. They estimate the 3D-position and velocity of a point by integrating subsequent measurements of the 2D-and 3D-point position over time using multiple Kalman filters (Kalman, 1960; Welch and Bishop, 2001). Because the state space of the Kalman filters consists of the 3D-point position plus the 3D-velocity vector, they name this approach 6D-vision. Rapid convergence to the true state of nature is achieved by using multiple Kalman filters with different initialisations. The normalised innovation squared (NIS) is used to identify the optimal Kalman filter. The NIS is given by the Mahalanobis distance between the measurement and the predicted measurement. The uncertainty of this difference is given by the innovation covariance matrix. The authors state that the accuracy of the inertial sensors of the car provides sufficient information about egomotion and do not compensate for rotation. The proposed system runs comfortably in real time.

Argyros and Orphanoudakis (1997) detect independent motion as outliers in a robust estimation process using the least median of squares technique (Rousseeuw and Leroy, 1987). They estimate the external calibration of a stereo setup and the motion parameters using normal flow measurements as input. Normal flow denotes the projection of the optical flow vector on the direction of the image gradient. A local voting process smooths the resulting classification map and fills gaps in positions with insufficient spatial structure for estimation of optical flow.

An interesting generalisation to the standard stereo approach is presented in Sturm (2002). The stereo setup is unconstrained such that its internal calibration is neither known nor fixed, reducing stereo to two time-synchronous views of a 3D-scene. The work also focuses on multiple motions under the constraint that each motion lies in one of a pencil of planes.

Very good practical results have been presented particularly by the research group from DaimlerChrysler. Their algorithms are able to detect moving objects reliably and track them through time with high frame rates. One disadvantage of these algorithms is that they rely on expensive and sensitive stereo camera systems. This thesis investigates the possibility to avoid stereo camera systems and detect independently moving objects using a single monocular camera.

3.3 Monocular Freely Moving Camera

Algorithms for visual detection of independent motion with a monocular moving camera can be grouped in purely *correspondence based* algorithms and algorithms incorporating *context* information. The majority of scientific work focuses on the former group. Three major subclasses of correspondence based algorithms for the detection of independent motion can be identified:

Embedding into high dimensional polynomial spaces is used to get a linear description of the multibody structure and motion problem.

Factorisation approaches are based on the factorisation of a huge matrix containing point correspondences from many images.

Recursive algorithms typically alternate between robust motion estimation and classification of correspondences in inlier and outlier.

While early work often (but not exclusively) belongs to the class of algorithms either alternating between or recursively applying clustering and motion estimation (for example Torr 1998), recent research efforts focused on the two "direct" estimates of multibody structure and motion: Factorisation and embedding. Factorisation algorithms are based on the work by Costeira and Kanade (1998) and recover structure and (multiple) motions by factorising a large matrix constructed from all correspondences into structure and motion. Embedding approaches are based on the work of Wolf and Shashua (2001) and linearise the problem by embedding it into high dimensional polynomial spaces.

The detection of independent motion from monocular image sequences captured by a moving camera is closely related to egomotion estimation. Egomotion is the relative motion between camera and background. If, however, the camera moves and the background is static or if the background moves and the camera is static, is merely a question of the viewpoint and does neither change the image sequence nor does it have any effect on the algorithms. Taking the viewpoint of static camera annihilates the distinction of the background motion (egomotion) from other motions. It is hence common to drop the notion of egomotion and simply speak of *multiple motions*. In many cases the dominant motion is caused by the relative motion between camera and background. When all motions are generated by rigid objects with diffuse reflecting surfaces, the problem of independent motion detection can be solved by multibody structure and motion. In addition to simple detection of independent motion, multibody structure and motion estimates the parameters of the independent motion. While recovery of structure and motion is well understood in the case when only a single rigid motion is visible in the images, even in the case of noisy image measurements, dealing with multiple motions is still subject to many recent research efforts.

After a short introduction into the three main groups of algorithms, a brief review of early estimation algorithms including recursive algorithms follows. The two groups of direct algorithms are reviewed in detail. Finally a summary of work using contextual information is given.

3.3.1 State of the Art

A brief overview over the state of art in detection of independent motion respective multibody structure and motion is given in this section. For the finer details of the algorithms refer to the following sections.

State of the art approaches to independent motion detection can be categorised in three major groups. All algorithms in these groups recover the structure (i.e. the 3D-points) and the motions from a set of corresponding image points. The first group works on two images and approaches the problem by *embedding* it in high dimensional polynomial spaces which results in linear solutions. The second group uses correspondences over many views for the creation of the trajectory matrix which is block-wise *factorised* into structure and motion. The third group approaches the problem *recursively* and is the most pragmatical: The dominant motion is computed from the pool of correspondences over two or three views. Correspondences consistent with this motion are removed from the pool and the process is repeated revealing the second most dominant motion. This is repeated until too little correspondences remain for further detection.

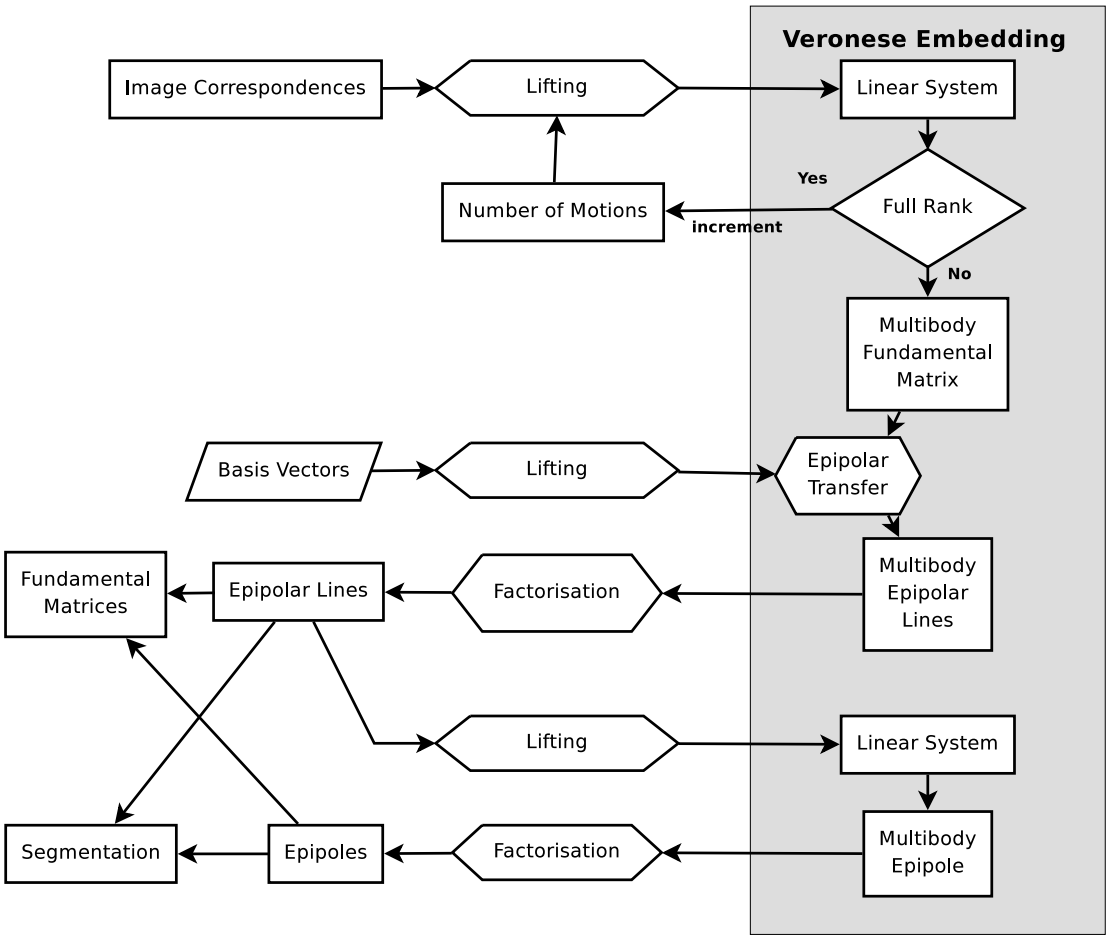


Figure 3.1: Multibody structure and motion by embedding in high dimensional polynomial space. See text for description.

Embedding

One approach to the multibody structure and motion is to linearise the problem by embedding in a high dimensional space[1] (Ma et al., 2004). Similar to the fact that perspective projection becomes linear in the projective space, the estimation of the *multibody fundamental matrix* becomes linear in the correct *Veronese embedding*. The choice of the embedding depends on the number of independent motions. The number of motions can be determined by subsequently trying different embeddings until a rank constraint on the resulting linear system is satisfied. The multibody fundamental matrix is computed and a number of multibody epipolar lines can be determined using a number of basis vectors. Factorisation extracts the corresponding epipolar lines. Embedding these lines again, using

[1] A simple example for linearisation by embedding is given for explanation: Let, for example, the nonlinear equation be given by the polynomial $ax^2 + bx + c = 0$ of degree 2 in x with coefficients a, b and c. When lifting the scalar x into two-dimensional polynomial space whose axes are given by x and x^2, the polynomial becomes linear in the two unknowns x^2 and x: $(ab)(x^2 x)^T = -c$.

the Veronese map, results in a linear equation system for the unknown multibody epipole. The individual epipoles can be extracted by factorisation from the multibody epipole. The individual essential matrices can be computed using the epipoles and epipolar lines. The complete process is visualised in figure 3.1.

Advantages:

- Direct, algebraic solution for less than 5 motions, numeric solution for more than 5 motions.
- No clustering necessary.
- Instantaneous estimation on two views.
- Statistic nonlinear optimisation is possible.

Disadvantages:

- No special motions (e.g. pure rotations or homographies).
- Only for distinct epipoles.
- Many correspondences needed.
- No experiences with noisy data.
- Only rigid body motion.

Factorisation

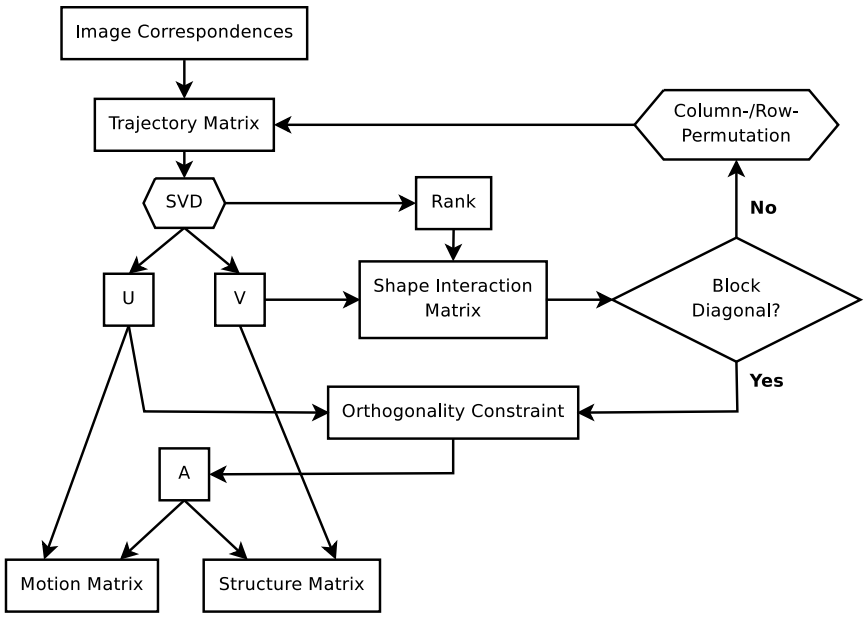


Figure 3.2: Multibody structure and motion by factorisation of trajectory matrix. See text for description.

Factorisation approaches (Costeira and Kanade, 1998) typically use correspondences over many views (e.g. the complete sequence) and transform them into the *trajectory matrix*. Assuming orthogonal projection, a factorisation of the motion matrix exists where the left factor represents the motion parameters and the right part represents the shape.

When the correspondences are sorted according to their motion, the shape matrix takes on block diagonal form. The *shape interaction matrix* is computed using the first r columns of the right orthogonal matrix of the singular value decomposition[2] (SVD) of the trajectory matrix with rank of the shape interaction matrix r. Permuting columns and rows, the shape interaction matrix is brought into block diagonal form and the resulting permutation is applied to the trajectory matrix which is afterwards again decomposed using SVD. The factorisation by SVD is not unique, however knowledge about the expected internal structure of the motion matrix can be used to calculate the invertible transformation $\mathbf{A}$ between the current factorisation and the final motion and the final shape matrix.

Advantages:

- Direct solutions for an arbitrary number of motions.
- Special motions possible.

Disadvantages:

- Direct solution only for linear projection models
- Difficult permutation problem.
- Only rigid body motion.
- Operates on complete sequence.

Recursive

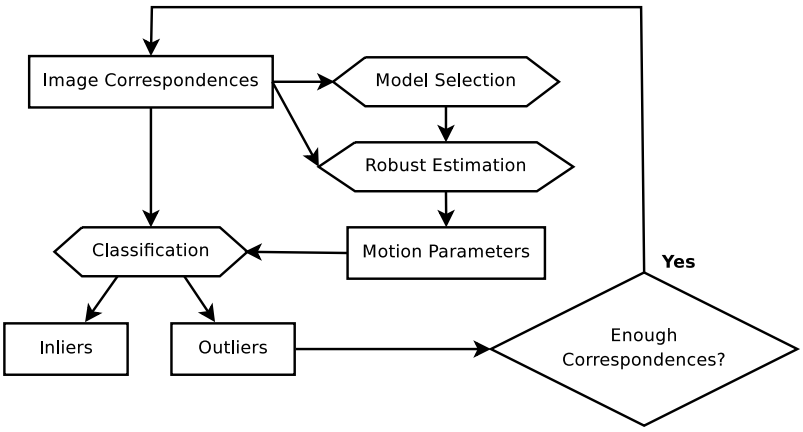


Figure 3.3: Multibody structure and motion by recursively applying robust egomotion estimation methods. See text for description.

The oldest and most simple approach to multibody structure and motion respective detection of independent motion is a recursive process (Torr and Murray, 1993): First, robust estimation of the motion parameters is performed. This results in the parameters of the egomotion under the assumption that the majority of the correspondences are located

[2]The SVD is a numerical method for decomposition of a matrix $\mathbf{B} = \mathbf{USV}$ into two orthogonal matrices $\mathbf{U}$ and $\mathbf{V}$ and a diagonal matrix $\mathbf{S}$ containing the singular values of $\mathbf{B}$ on the main diagonal.

on the background. Afterwards, the correspondences are classified into inliers and outliers. When enough correspondences are classified as outliers, robust estimation of the motion parameters can be performed with these outliers, yielding the parameters of the predominant independent motion. This process can recursively be repeated until the parameters of all motions are recovered. A model selection stage can be incorporated into the process allowing the estimation of degenerate motions.

Advantages:

- Robust estimation is well understood.
- Operates on two views.
- Little correspondences needed.

Disadvantages:

- Clustering only with rigid body motion.

3.3.2 Early Work on Detection of Independent Motion

Man representatives of early work on detection of independent motion either placed restrictions on the camera movement (Clarke and Zisserman, 1996; Torr and Murray, 1993; Enkelmann, 1990; Sinclair and Boufama, 1994; Carlsson and Eklundh, 1990), or classified the flow fields into a small number of basic camera movements (Nelson, 1991). The motion of a camera mounted in an automobile is generally not restricted, and hence these approaches are not investigated in detail.

Another sort of algorithm uses image constraints to detect independent motion. Smith (1995), for example, clusters points based on the similarity of their optical flow vectors in the image. This approach results in cluster boundaries at depth discontinuities. The 3D-geometry is neglected and the algorithm is hence condemned to failure in many non-trivial cases.

Torr (1995) was the first scientist suggesting the usage of 3D-geometry for visual point-based detection of independent motion. Under the assumption of rigid objects, he suggested clustering points based on the fundamental matrix using two views or based on the trifocal tensor using three views. He further pointed out that separate consideration of outliers and degeneracy leads to suboptimal results in the estimate of the fundamental matrix resp. trifocal tensor and hence in classification of the points. Torr suggested a robust algorithm for the joint detection of outliers and degeneracy dubbed PLUNDER (Pick Least UNDEgenerate Randomly). He suggested two algorithms for the detection of independent motion: A Bayesian approach and the recursive application of the PLUNDER algorithm.

The Bayesian approach initialises 500 candidate clusters based on fundamental matrices and assigns correspondences to them. Clusters with too few correspondences are pruned. The segmentation is obtained using multiple hypothesis testing maximising the posterior likelihood. A special cluster containing all outliers is always present in the final set and contributes to the segmentation likelihood. Segmentation with less clusters are favoured by introducing a penalty term for the number of clusters.

The recursive application of PLUNDER works in a greedy way. The predominant and possibly degenerate motion is computed using the PLUNDER algorithm. The set of correspondences consistent with this motion are removed from the pool of correspondences if the set is big enough. This process is repeated until the number of correspondences that can be assigned to a cluster drops below a threshold.

MacLean (1996) suggested the use of the EM algorithm on mixture models for detection of independent motion. The EM algorithm alternating estimates segmentation and the motion parameter. A basic difficulty of the EM algorithm is that prior knowledge about the number of nodes is required. The author circumvents this difficulty by spawning new motion processes when an object enters or leaves the field of view.

Algorithms Using Projections of Flow Fields: In the 90s, one big problem was the limited computational resources, and therefore algorithms reducing the dimensionality of the problem have been developed. This was achieved by investigating projections of optical flow fields (Fejes and Davis, 1998, 1997a,b). The simple geometric properties exhibited by restricted projections of flow fields allow the either partial or complete decoupling of structure and motion. By investigating projections of flow fields, the problem of outlier detection could be reduced to robust line fitting.

An arbitrary projection direction $\boldsymbol{p}$ is chosen and all flow vectors are projected onto it. The *parallel restriction* is given by 1D-sampling of the projected flow field along a line through the FOE parallel to the projection direction $\boldsymbol{p}$. The *orthogonal restriction* is computed by 1D-sampling of the projected flow field along a line through the FOE orthogonal to $\boldsymbol{p}$. Plotting the length of the projected flow vectors vs. their position on the respective lines results in two scatter plots (see figure 3.4(c) and 3.4(d)).

When the field of view (FOV) is small, the parallel restriction exhibits the *divergence property* and the orthogonal restriction exhibits the *linear property*. These properties are directly related to the translational and rotational part of the flow field: The orthogonal restriction is independent of the translational flow field and its slope only depends on the rotation around the camera axis. When no rotation around the optical axis occurs and when the FOV is small, the rotational part of the flow field is approximately constant in the image. The parallel restriction is independent of the rotation around the camera axis, and hence the rotational flow field only shows as a constant offset in the restriction. However, the scene structure (i.e. the scene depth) leads to the divergence of the parallel restriction.

Early work uses the linear property for projection directions passing through the image centre and estimate the rotation, while at the same time constraining the position of the FOE (Daniilidis and Thomas, 1996; Fermüller and Aloimonos, 1995; Silva and Satos-Victor, 1996).

An algorithm for the computation of the motion parameters for the case of cameras with small FOV and FOE location inside the image was developed by Fejes and Davis (1997b). The parallel restrictions stemming from multiple hypothesis about the FOE positions were investigated and the best fit was chosen. This can also be interpreted as minimising the

points with negative depth (Fejes and Davis, 1997b). The partial FOE position can be recovered from the best fitting parallel restriction, and hence the appropriate orthogonal restriction can be constructed. From this orthogonal restriction the unknown part of the FOE position is recovered. The rotation parameters were also extracted from the parallel and orthogonal restrictions.

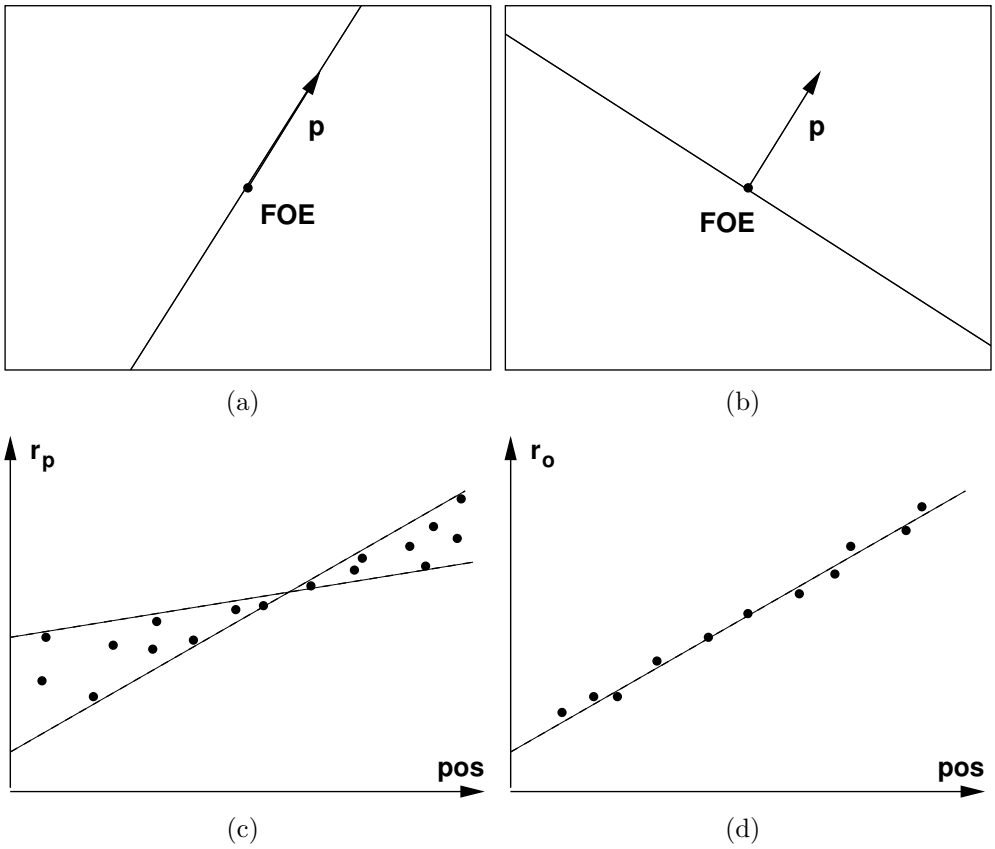


Figure 3.4: Parallel and orthogonal restriction of projections of the flow field on an arbitrary projection direction $\boldsymbol{p}$. The positions in the image where flow measurements are evaluated are restricted to a line through the focus of expansion parallel (a) and orthogonal (b) to the projection direction $\boldsymbol{p}$. The magnitude of the resulting projections of the flow vectors is plotted versus the position on the respective line for the parallel restriction (c) and for the orthogonal restriction (d).

When the rotational flow component is not small compared to the translational part or when the FOV is big, the erroneous estimate of the rotational component of the optical flow can be used to correct (de-rotate) the flow field, and the algorithm can be used as described above.

The projections of the flow fields can either be used to extract the motion parameters or to detect independent motion. The detection of independent motion can in this case be reduced to the detection of outliers in a robust line fitting procedure.

When FOE is not located inside the image, its exact position cannot be estimated. Only a qualitative estimate of the FOE can be guessed. The authors argue, that the qualitative estimate is sufficient for some structure and motion algorithms (Fejes and Davis, 1997b).

Even though the suggested algorithms may be very fast, the basic assumptions stem from a very restricted setup (narrow FOV and FOE location inside the image) and the extension to the general case either requires iterative procedures (wide FOV) or is even not possible (FOE outside the image).

3.3.3 Factorisation Methods

Factorisation methods aim at recovering structure and motion from a set of point correspondences over many views. After a brief overview over the factorisation methods, their first application to the task of independent motion detection of is reviewed in detail.

Factorisation methods for static scenes recover structure and motion by decomposing a large matrix containing correspondences from many images and many points into one matrix describing the structure of the scene and one matrix describing the camera motion. The methods were initially suggested by Kanade's group (Poelman and Kanade, 1993; Tomasi and Kanade, 1991b, 1992). This work was still restricted to linear camera models (scaled orthographic, paraperspective or affine). Contributions from other authors relaxed the constraint of linear projection models for the factorisation approach (Yu et al., 1996; Christy and Horaud, 1996; Han and Kanade, 1999b) or recovered shape and motion linearly in the projective space (Han and Kanade, 2003). Later Han and Kanade (2001, 2003, 1999a) used assumptions of linear motions to enable reconstruction of the trajectory when only few points are located on each object.

Basis of all factorisation approaches is the singular value decomposition of very large matrices, a computationally heavy burden. Information from the complete image sequence is used to construct these large matrices. Morita and Kanade (1994) suggested an sequential approach recursively estimating shape and motion at each frame overcoming this limitation.

In 1998 Costeira and Kanade extended the factorisation methods for static scenes to multiple motions. Assuming an orthographic camera model, Costeira and Kanade (1998) presented a factorisation algorithm recovering shape and motion without any prior assumptions about the number of objects or clustering of the 2D-feature point correspondences. The factorisation method can be easily extended to any linear projection model. The algorithm is claimed to be tolerant against moderate perspective projection. Formulating the orthographic projection of a homogeneous 3D-point $\boldsymbol{X}_i$ in the camera centric coordinate system to an Euclidean 2D-point $\boldsymbol{x}_{fi}$ results in

$$\boldsymbol{x}_{fi} = \begin{pmatrix} u_{fi} \\ v_{fi} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} \mathbf{R}_f & \boldsymbol{t}_f \\ \boldsymbol{0}^T & 1 \end{pmatrix} \boldsymbol{X}_i \tag{3.4}$$

with rotation $\mathbf{R}_f$ and translation $\boldsymbol{t}_f$ dependent on the frame f. Building a single matrix for multiple features for a single rigid motion results in the matrix $\mathbf{W} = [\frac{\mathbf{U}}{\mathbf{V}}]$ which was

later denoted *trajectory matrix* by Irani (2002)

$$\mathbf{W} = \begin{pmatrix} u_{11} & \dots & u_{1N} \\ \vdots & & \vdots \\ u_{F1} & \dots & u_{FN} \\ v_{11} & \dots & v_{1N} \\ \vdots & & \vdots \\ v_{F1} & \dots & v_{FN} \end{pmatrix} = \begin{pmatrix} \boldsymbol{i}_1^T & t_{x1} \\ \vdots & \vdots \\ \boldsymbol{i}_F^T & t_{xF} \\ \boldsymbol{j}_1^T & t_{y1} \\ \vdots & \vdots \\ \boldsymbol{j}_F^T & t_{yF} \end{pmatrix} \begin{pmatrix} \boldsymbol{X}_1 & \dots & \boldsymbol{X}_N \end{pmatrix} = \mathbf{MS} \tag{3.5}$$

The trajectory matrix can be factorised into the motion matrix $\mathbf{M}$ and the shape matrix $\mathbf{S}$. The motion matrix $\mathbf{M}$ consists of the first two rows $\boldsymbol{i}_k^T$ and $\boldsymbol{j}_k^T$ of each rotation matrix $\mathbf{R}_k$ and the first two entries of each translation vector t_{xk} and t_{yk}. Using the singular value decomposition of $\mathbf{W} = \mathbf{U}\Sigma\mathbf{V}^T$ results in such a decomposition by defining $\hat{\mathbf{M}} = \mathbf{U}\Sigma^{1/2}$ and $\hat{\mathbf{S}} = \Sigma^{1/2}\mathbf{V}^T$. This decomposition is however not unique because for any invertible 4×4 matrix $\mathbf{A}$, $\mathbf{W} = (\hat{\mathbf{M}}\mathbf{A})(\mathbf{A}^{-1}\hat{\mathbf{S}}) = \mathbf{M}'\mathbf{S}'$ also represents a valid solution. Using internal structure of the motion matrix $\mathbf{M}$ (i.e. the orthogonality between $\boldsymbol{i}_i$ and $\boldsymbol{j}_i$), an appropriate matrix $\mathbf{A}$ for the recovery of structure and motion can be found.

When multiple motions are present and when the features are sorted, the shape matrix $\mathbf{S}$ takes on block diagonal form

$$\mathbf{W}^* = \begin{pmatrix} \mathbf{M}_1 & \mathbf{M}_2 \end{pmatrix} \begin{pmatrix} \mathbf{S}_1 & \mathbf{0} \\ \mathbf{0} & \mathbf{S}_2 \end{pmatrix} \tag{3.6}$$

The algorithm boils down to find a column permutation of $\mathbf{W}$ determining the canonical form of the measurement matrix $\mathbf{W}^*$. The canonical form $\mathbf{W}^*$ can be factorised into the block diagonal shape matrix $\mathbf{S}^*$ and the motion matrix $\mathbf{M}^*$. The *shape interaction matrix* $\mathbf{Q} = \mathbf{V}\mathbf{V}^T$ is introduced for this purpose. When the $\mathbf{W}$ has not full rank (i.e. when one of the motion system is over-constrained[3]), only the first $r = \text{rank}(\mathbf{W})$ columns of $\mathbf{V}$ are used for computation of $\boldsymbol{Q}$. The canonical form $\mathbf{W}^*$ can be found permuting columns of the trajectory matrix $\mathbf{W}$, and hence the factorisation of the canonical form $\mathbf{V}^{*T}$ results from applying the same set of permutations to $\mathbf{V}^T$. It is shown that by bringing $\mathbf{Q}$ in block diagonal form permuting columns and rows, the canonical form $\mathbf{V}^{*T}$ can be found by applying the same permutations to $\boldsymbol{V}^T$. Using a single block from $\mathbf{V}^{*T}$ at a time, the recovery of shape and motion proceeds as in the case of a single motion.

The shape interaction matrix $\mathbf{Q}$ is invariant to object motion, image scale, change in reference frame ,and its structure is invariant to the number of objects. A greedy hill-climbing algorithm sorting $\mathbf{Q}$ into canonical form is presented, and block detection is done using a noise model for the image correspondences.

Gear (1998) suggested a combined rank estimation and factorisation based on the *reduced row echelon form* of the trajectory matrix $\mathbf{W}$. The reduced row echelon form of $\mathbf{W}$ is

[3]For example when more than 4 points are present on a full 3D-object, more than 3 point on a planar 3D-object, or more than 2 points on a linear 3D-object.

computed using **QR** decomposition with column pivoting followed by Gauss Jordan elimination on the upper triangular matrix **R**. The matrix has now the reduced row echelon form

$$\begin{pmatrix} 1 & 0 & \cdots & 0 & f_{1,r+1} & \cdots & f_{1,n} \\ 0 & 1 & \cdots & 0 & f_{2,r+1} & \cdots & f_{2,n} \\ \vdots & & \ddots & & \cdots & & \\ 0 & 0 & \cdots & 1 & f_{r,r+1} & \cdots & f_{r,n} \end{pmatrix} \tag{3.7}$$

with the rank r of the original motion matrix **M** build using n frames. The columns of the reduced row echelon form correspond to 3D-points. When the correspondence data is exact, linear independence between points and thereby the grouping into independent subspaces is easily established by looking for zero elements in the entries of the column vectors of $f_{r,n}$. Column vectors having non-zero elements in the same row correspond to the same subspace and hence to the same rigid object. When the correspondence data is corrupted by noise, motion matrix has in general full rank. The rows of the reduced row echelon form matrix are represented using a bipartite graph and a probabilistic algorithm selecting most likely rank, and the most likely partition from a number of candidate ranks and partitions is suggested.

Kanatani (2001) reformulated the factorisation method from Costeira and Kanade (1998) in a purely mathematical fashion. He gave mathematical proof for the block structure of the shape interaction matrix and used a process which he called *dimension correction* while permuting columns and rows of the shape interaction matrix. Dimension correction works as follows: After clustering more than d points[4] together, an optimal subspace is fitted to them, and the points are replaced by their projections onto the subspace. This techniques reduces the noise if the clustering is correct. Afterwards, model selection (i.e. geometric AIC) is used to fuse multiple groups. To get rid of possibly misclassified points in these groups, least median of squares techniques are used rejecting outlying data. The resulting algorithms outperforms previous approaches with respect to accuracy and reaches an accuracy near a globally derived bound on synthetic data. It is claimed to be robust.

Irani (2002) suggested the employment of factorisation methods to constrain the 2D-correspondence estimation process itself. The *displacement field matrix* $[\mathbf{U}|\mathbf{V}]$ is introduced for this purpose, complementing the trajectory matrix $\mathbf{W} = [\frac{\mathbf{U}}{\mathbf{V}}]$ from Tomasi and Kanade (1991b). Using the image brightness constancy constraint, or alternatively the KLT equation, the trajectory matrix can be constructed using image brightness and derivatives thereof only. Another distinction between displacement and trajectory matrix is that the displacement matrix contains an entry for every pixel in the image, while the original trajectory matrix only contains the trajectories of carefully computed 2D-feature points. Afterwards, factorisation methods similar to Gear (1998) and Tomasi and Kanade (1991b) are used to compute the optical flow field using both trajectory and displacement matrix. Anadan and Irani (2002) found out that it is possible to incorporate uncertainty information into the factorisation process. The proposed method only works with either linear

[4]The minimal number of points depends on the motion model: Planar $d = 3$ or full 3D $d = 4$.

camera models or is restricted to instantaneous motion model[5] for perspective cameras. However, a workaround for perspective rotating and zooming cameras is given when a dominant plane is visible in the images. It basically works by using homography corrected images in which big rotations and big zoom changes are compensated. This workaround belongs to a group of algorithms called *plane + parallax*.

The above work from Irani (2002) and Anadan and Irani (2002) has been extended to the multibody case by Zelnik-Manor et al. (2006). The proposed algorithm gives new insights into motion segmentation, because it clusters points having a consistent temporal behaviour rather than points belonging to the same rigid motion model. This makes it feasible to cluster points belonging to non rigid motion as well as connected objects whose single parts move with different rigid motions. To achieve such a clustering, the multibody segmentation algorithm from Costeira and Kanade (1998) is applied to the displacement field matrix $[U|V]$. In doing so ,information about directional uncertainty can be incorporated into the factorisation process by transforming the raw data into covariance weighted data space. This process resembles the transformation used in the Mahalanobis distance. One further advantage is that clustering is conducted per pixel, rather than on distinctive feature points.

Estimation techniques using uncalibrated cameras are without question the most challenging, because no restrictions at all are placed on the images or image sequences. Nonetheless, interesting results can be achieved using factorisation approaches on uncalibrated sequences. For example, it has been shown that the presence of multiple motions stabilises the self calibration process (Fitzgibbon and Zimmerman, 2000).

Yan and Pollefeys (2006) also work on the motion matrix $\mathbf{M}$, but only extract a clustering of the trajectories from it. After rank correction and normalisation of the column vectors of $\mathbf{M}$, linear subspaces are estimated using local sampling. The affinity between two points is defined as the principal angle between their local subspaces. An *affinity matrix* is computed for all trajectories. Spectral clustering based on this matrix is performed resulting in the final segmentation. The authors claim that the algorithm is robust to a moderate number of outliers and argue that outliers only affect the estimate locally. Outlier detection is done after segmentation. Their algorithm is independent on the motion type and works with independent, articulated, rigid, non-rigid, degenerate and non-degenerate motions.

3.3.4 Linearisation by Embedding in Higher Dimensional Spaces

Recent research approaches the problem of multiple motion estimation by embedding the problem in high dimensional spaces. The algorithms investigate two views of a static scene containing one or more rigidly and independently moving objects. Each of these motions can be explained by either a fundamental or an essential matrix. The ultimate goal of the

[5]The instantaneous motion model for perspective cameras is valid for small rotations and small field of views.

algorithms is the estimation of the number of different motions, their parameters and the segmentation.

In 2001 Wolf and Shashua introduced the 2-body multi-linear constraint which is valid for two rigid motions in two views. Each point correspondence $\boldsymbol{p}_1$, $\boldsymbol{p}_2$ in this setup must obey the 2-body multi-linear constraint

$$(\boldsymbol{p}_1^T \mathbf{F}_1 \boldsymbol{p}_2)(\boldsymbol{p}_1^T \mathbf{F}_2 \boldsymbol{p}_2) = 0 \tag{3.8}$$

Each projective point $\boldsymbol{p} = [x, y, w]^T$ can be "lifted" onto a 6-dimensional projective space resulting in $\hat{\boldsymbol{p}} = [x^2, xy, y^2, xw, yw, w^2]^T$. Note that the symmetric matrix $\boldsymbol{p}\boldsymbol{p}^T$ consists of the same entries as the lifted vector $\hat{\boldsymbol{p}}$. The segmentation matrix $\mathbf{S} \in \mathbb{R}^{6\times 6}$ is defined such that

$$\hat{\boldsymbol{p}}_1^T \mathbf{S} \hat{\boldsymbol{p}}_2 = 0 \tag{3.9}$$

A linear equation system can be set up using DLT (direct linear transform) techniques, and the entries of the segmentation matrix can be computed when at least 35 point correspondences are given. Note that $\mathbf{S}\hat{\boldsymbol{p}}$ represents a degenerate conic (consisting of two lines) defined by the two epipolar lines $\mathbf{F}_1\boldsymbol{p}$ and $\mathbf{F}_2\boldsymbol{p}$. Wolf and Shashua (2001) show that the segmentation matrix exhibits similar properties as the fundamental matrix. For example, the lifted epipoles $\hat{\boldsymbol{e}}_1$ and $\hat{\boldsymbol{e}}_2$ are in the right nullspace of the segmentation matrix

$$\mathbf{S}\hat{\boldsymbol{e}}_1 = 0 \qquad \text{and} \qquad \mathbf{S}\hat{\boldsymbol{e}}_2 = 0 \tag{3.10}$$

The rank of the segmentation matrix can be used to differ between coinciding epipoles and distinct epipoles of the two motions. When the epipoles of the two motions are distinct, the segmentation matrix has rank 4. Otherwise it has only rank 3. An algorithm for the recovery of the fundamental matrices and the epipoles from the segmentation matrix is given. When the epipoles of the two motions are distinct, they can be recovered directly, otherwise the fundamental matrices must be computed, and the epipoles are afterwards extracted from the fundamental matrices.

The epipoles lie in the nullspace of $\mathbf{S}$, and hence they are given by a linear combination of the nullspace vectors $\hat{\boldsymbol{n}}_1$ and $\hat{\boldsymbol{n}}_2$. This can also be expressed using the (symmetric) matrix representation $\mathbf{N}_1 = \hat{\boldsymbol{n}}_1\hat{\boldsymbol{n}}_1^T$ and $\mathbf{N}_2 = \hat{\boldsymbol{n}}_2\hat{\boldsymbol{n}}_2^T$ of the null vectors $\hat{\boldsymbol{n}}_1$ and $\hat{\boldsymbol{n}}_2$

$$\mathbf{E}_1 = \mathbf{N}_1 + \lambda_1\mathbf{N}_2 = \boldsymbol{e}_1\boldsymbol{e}_1^T \qquad \text{and} \qquad \mathbf{E}_2 = \mathbf{N}_1 + \lambda_2\mathbf{N}_2 = \boldsymbol{e}_2\boldsymbol{e}_2^T \tag{3.11}$$

with the symmetric matrices $\mathbf{E}_1 = \hat{\boldsymbol{e}}_1\hat{\boldsymbol{e}}_1^T$ and $\mathbf{E}_2 = \hat{\boldsymbol{e}}_2\hat{\boldsymbol{e}}_2^T$ representation of the epipoles $\boldsymbol{e}_1$ and $\boldsymbol{e}_2$ "lifted" onto the 6-dimensional projective space $\hat{\boldsymbol{e}}_1$ and $\hat{\boldsymbol{e}}_2$. $\mathbf{E}_1$ and $\mathbf{E}_2$ both have rank 1, and hence their determinants vanish ($\det(\mathbf{E}_1) = 0$ and $\det(\mathbf{E}_2) = 0$). Furthermore, the determinants of arbitrary minors $\mathbf{M}_1$ and $\mathbf{M}_2$ of $\mathbf{E}_1$ and $\mathbf{E}_2$ must also vanish resulting in the constraint

$$\det(\mathbf{M}_1 + \lambda_1\mathbf{M}_2) = 0 \qquad \text{and} \qquad \det(\mathbf{M}_1 + \lambda_2\mathbf{M}_2) = 0 \tag{3.12}$$

The resulting 9 second order polynomials all share the same roots. The epipoles can be extracted by computing the norm of the polynomials (viewing the coefficients as a vector) and use the mean of polynomials above a certain threshold.

When the two motions share the epipole, the fundamental matrices must be recovered from the segmentation matrix in order to get the epipoles. The fundamental matrices are recovered column-wise. The special point $\boldsymbol{p}_x = [1,0,0]^T$ and its corresponding lifted entity are used for the extraction of the first columns $\hat{\boldsymbol{p}}_x = [1,0,0,0,0,0]^T$ from $\mathbf{S}$. The degenerate conic

$$\mathbf{D} = \mathbf{S}\hat{\boldsymbol{p}_x} \tag{3.13}$$

consists of the union of the two epipolar lines $\boldsymbol{l}_1 = \mathbf{F}_1\boldsymbol{p}_x$ and $\boldsymbol{l}_2 = \mathbf{F}_2\boldsymbol{p}_x$. At the same time, the two lines $\boldsymbol{l}_1$ and $\boldsymbol{l}_2$ are the first columns of $\mathbf{F}_1$ and $\mathbf{F}_2$. The second and third columns can be recovered in a similar fashion.

Problems arise with the labelling and the relative scale of the columns. Up to now, it is not clear which column belongs to which fundamental matrix and - because of the projective nature of the equations - the relative scale of the columns is also unclear. The scale problem can be solved by recovering the first row of the fundamental matrix using the transpose of equation 3.13. The labelling problem can only be solved by computing the eight different candidate solutions and their corresponding segmentation matrices. Either the candidate solution with the biggest similarity to the measured segmentation matrix $\mathbf{S}$ can be chosen or the best solution is determined by applying the candidate segmentation matrix $\mathbf{S}'$ and the measured segmentation matrix $\mathbf{S}$ to a set of randomly generated points and to measure similarity between resulting fundamental lines.

Segmentation of the points can be done as follows: Given a point $\hat{\boldsymbol{p}}$, compute the corresponding conic $\mathbf{D} = \mathbf{S}\hat{\boldsymbol{p}}$ and extract the epipolar lines from it. Segmentation is done by checking on which epipolar line the corresponding point $\boldsymbol{p}_2$ is located in the second image.

Starting in 2002, Vidal and his colleagues generalised the 2-body multi-linear constraint (equation 3.8) to an arbitrary number of rigid motions resulting in the multibody epipolar constraint Vidal et al. (2002, 2006)

$$\Pi_{i=1}^{n}(\boldsymbol{x}_1^T\mathbf{F}_i\boldsymbol{x}_2) = 0 \tag{3.14}$$

with n fundamental matrices $\mathbf{F}_i$ for the n different rigidly moving objects and the point correspondence between $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$. Note that the above expression is a polynomial of degree n in $\boldsymbol{x}$. It can also be expressed in bilinear form (in $M_n = \binom{n+2}{2}$-dimensional space)

$$\nu_n(\boldsymbol{x}_2)^T\mathbf{F}_m\nu_n(\boldsymbol{x}_1) = 0 \tag{3.15}$$

where $\mathbf{F}_m$ is the *multibody fundamental matrix* and $\nu_x(\boldsymbol{x}) : \mathbb{P}^2 \rightarrow \mathbb{P}^{M_n-1}$ is the so called *Veronese map* of degree n embedding a vector $\boldsymbol{x}$ in n-dimensional polynomial space. Equation 3.15 can be rewritten as linear problem

$$\mathbf{L}_i\boldsymbol{f} = \begin{pmatrix} [\nu_i(\boldsymbol{x}_2^1)^T \otimes \nu_i(\boldsymbol{x}_1^1)]^T \\ [\nu_i(\boldsymbol{x}_2^2)^T \otimes \nu_i(\boldsymbol{x}_1^2)]^T \\ \vdots \\ [\nu_i(\boldsymbol{x}_2^N)^T \otimes \nu_i(\boldsymbol{x}_1^N)]^T \end{pmatrix} \boldsymbol{f} = \boldsymbol{0} \tag{3.16}$$

with the vector $\boldsymbol{f} \in \mathbb{R}^{M_n^2}$ consisting of the stacked columns of the multibody fundamental matrix $\mathbf{F}_m$, the matrix $\mathbf{L}_i \in \mathbb{R}^{N \times M_n^2}$ and the Kronecker product $\otimes$. A rank constraint on the matrix $\mathbf{L}_i$ allows the estimation of the number of different motions simply by subsequently constructing candidate matrices $\mathbf{L}_i$ and checking their rank. The rank constraint is given by

$$\operatorname{rank}(\mathbf{L}_i) = M_n^2 - 1 \tag{3.17}$$

Equivalent to Wolf and Shashua (2001), the linear estimation of the multibody fundamental matrix is possible when enough point correspondences are given (see table 3.1).

# motions n	# required point correspondences N
1	8
2	35
3	99
4	225

Table 3.1: Minimum required number of point correspondences N for the linear estimation of the multibody fundamental matrix depending on the number of different motions n

The author notes the limitations imposed by the very large number of required correspondences and suggests three different strategies to overcome these difficulties in future work:

- Obviously the number of points could also be reduced when the internal structure of the multibody fundamental matrix is considered during its estimation equivalently to the fact that the 8 point algorithm for the fundamental matrix can be reduced to the seven point algorithm when the internal constraint $\det(\mathbf{F})$ is used.
- Under the assumption of constant motions, more than two views can be used for the estimation process.
- When the rotations are very small and the motions are hence approximately pure translations, the number of required correspondences is also greatly reduced.

In Vidal and Sastry (2003), the estimation of the fundamental matrices is reformulated as a simple nonlinear optimisation problem. By estimating the number of motions in advance using the rank constraint on $\mathbf{L}_i$ and by the usage of a single objective function for multiple motions, the typical estimation scheme iterating between grouping of the points and estimation of the individual fundamental matrices can be circumvented. The estimation of the multiple fundamental matrices is reformulated as a nonlinear optimisation problem minimising a single objective function. It is shown that the objective function is identical to the optimal objective function as given by Ma et al. (2001) in the case of a single motion.

Lately, the multibody estimation procedures from Wolf and Shashua (2001) and Vidal et al. (2002) were expanded by a model selection stage (Schindler and Suter, 2005; Schindler

et al., 2006). A common problem of structure from motion algorithms is the occurrence of degenerate cases, for example, when all 3D-points are located on a planar surface. The fundamental matrix is not defined in this case, and all correspondences can be described by a model with less degrees of freedom, the homography **H**. Model selection aims at automatic determination of the correct model for the given data, i.e. it tries to decide whether a homography is sufficient to describe the data or if a fundamental matrix is needed.

Using two views of several rigidly moving objects, Schindler and Suter (2005) generate several thousand candidate models (i.e. **F** and **H** matrices). Only candidate models with a mean error smaller than 4 pixels are passed to the model selection stage, where the number of independently moving objects and the intrinsic dimension (fundamental matrix vs. homography) of the model describing a particular motion are automatically determined. The model selection stage is based on the GRIC (Torr, 2002) criterion.

This approach is generalised in Schindler et al. (2006) to more than two views. In contrast to Schindler and Suter (2005), calibrated cameras are used and the candidate motions are computed using the algorithms from Nistér (2004). The spatial coherence of points is exploited using a heuristic local sampling scheme, and inlier sets are computed for each candidate motion. Candidate motions computed for two frames are clustered based on their inlier set: Inliers are specified using a Boolean vector whose length corresponds to the number of tracks. The Hamming distance between these vectors is used as similarity measure for clustering. Afterwards, clusters are replaced by their "means" which are computed from all inliers belonging to more than 50% of all cluster members. Typically less than 10 candidate motions remain after clustering, and these are linked throughout the sequence based on their inlier set leading to a large number of candidate motion chains. A MDL (minimum description length) like approach selecting the best motion chain from the candidate motions precedes a specialised multi-branch optimisation for determination of the motion chain with the minimal description length. Motion segementation is now reduced to the problem of disambiguating points over time.

3.3.5 Work Using Contextual Information

Another class of algorithms uses contextual information, Smith et al. (2000), for example, computes the segmentation under the assumption of layered motion and visible edges at motion boundaries by tracking edges along their normals. An expectation maximisation (EM) algorithm (Dellaert, 2002) is used to iteratively estimate two motions and group the edges to one of the two models. When the EM is near convergence, the dependency between neighbouring edge points is modelled using a Markov chain model. The final solution is computed using simulated annealing. The label of each edge is flipped and the label likelihood is computed. The algorithm is applied to a sequence of images, and probabilities are temporally integrated.

In Ogale et al. (2005) segmentation is computed using dense flow fields and information about occluded regions. Three distinct possibilities for the detection of independent motions are characterised:

- *Motion direction conflict*: Flow direction is different from expectation.
- *Ordinal depth conflict*: Occlusion indicates that object 1 is closer to the camera than object 2 while flow indicates opposite (i.e. flow of object 2 is longer than flow of object 1).
- *Cardinal depth conflict*: No detection possible without addition knowledge (e.g. stereo, 3D-geometry, etc.).

In order to determine the depth order of regions, occlusions must be merged to one of their neighbouring regions. It is important to know *who occluded what* instead of *what was occluded*. To achieve this, three frames are used for merging occluded regions to one of its neighbouring regions using logical reasoning and temporal consistency.

3.4 Summary and Relation to this Thesis

Detection of independent motion from a single freely moving camera is a wide field of research. A lot of different camera models, both calibrated and uncalibrated, have been used for this task. Recently, two approaches for direct estimation of the number of objects and the underlying motions and structures came into the focus of the research community: Factorisation approaches and algorithms based on embedding the basic equations in high dimensional spaces.

Factorisation approaches suffer from their computational complexity which significantly hinders their usage in real time systems. Their need for many views of the scene would introduce an undesired latency into the detection process. Their algorithmic complexity makes it difficult to cope with outliers in the process.

Embedding approaches require many correspondences for the estimation of independent motion. The number of required correspondences is a major disadvantage of these approaches.

Even though both approaches seem very promising, real time implementations on data heavily corrupted by noise are still missing. Further on, the detection of non-rigid objects cannot be achieved by both groups of algorithms, with the notable exception of the work from Zelnik-Manor et al. (2006). For the above stated reasons, the direct approaches are not used in this work, and a simple algorithm based on the essential constraint is suggested.

Relation between Torr's work and this Thesis: The algorithm proposed in this thesis is similar to the algorithms in the group recursive approaches. It is in particular closely related to the work in Torr (1995). The main differences are:

Camera Calibration The basic difference of the proposed algorithm to the work of Torr is that in this thesis a calibrated camera is used. One important implication thereof is that less degenerate cases exist. When using uncalibrated cameras, configurations, e.g. where all 3D-points are located on a plane, are degenerate. With a calibrated

camera, the only degenerate case consists of a purely rotational relative camera motion. This case can easily be detected for the egomotion of the camera using the car inertial sensors. Purely rotational relative motion between camera and independently moving objects is highly unlikely in traffic scenarios, especially when the egomotion of the camera itself is translating. In the only possible configuration, camera and object move with identical velocities in the same direction on parallel trajectories, and one object rotates around its centre. Because degenerate cases are unlikely, the simple recursive application of robust estimation algorithms for the essential matrix is sufficient, and model selection needs not be pursued in this thesis.

Motion Model Torr's work focuses on the segmentation using rigid motions only, while this work does not constrain the type of independent motion. It could hence be used as a preliminary stage resulting in prior information for Torr's algorithms.

Boosting the Measurements The novel approach boosting the measurements results in many correspondences on independent motion and thus avoids the problems from Torr's thesis with motions where too few correspondences are measured.

Chapter 4

Egomotion Estimation

The system for the detection of independently moving objects, which is presented in this thesis, is based on image point correspondences. The correspondences are classified as either belonging to the static background or as independently moving based on information about the relative motion between static background and the camera. Fast and sound computation of (i) correspondences and (ii) the egomotion is hence essential for successful detection of independent motion. An algorithm for image point correspondence estimation is chosen in the following section. Afterwards the computation of egomotion is described in section 4.2.

4.1 Correspondence Estimation

Image-based detection of independent motion requires the knowledge about 2D-correspondences, i.e. correspondences between points in the image which are projections of the same 3D-points at different times. A lot of methods for the determination of such correspondences exist. A basic difference between the algorithms can be made regarding the density of the estimates. Optical flow algorithms aim at dense correspondence estimation, i.e. a correspondence is given for every pixel in the image apart from regions which are occluded in one of the images (Barron et al., 1994; Stein, 2004). Tracking algorithms aim at estimation of correspondences for a set of prominent image features only. The algorithm for detection of independent motion does not rely on a specific algorithm but on a set of corresponding points. For speed reasons, a tracking algorithm is chosen for correspondence estimation.

First, the correspondence estimation method is described, and afterwards algorithms for the detection of prominent point features are presented.

4.1.1 Gradient-Based Minimisation of Intensity Difference

The iterative image registration algorithm from Lucas and Kanade (1981) is described in this section. Correspondence estimation in image sequences is also called *feature tracking*.

The algorithm operates on small patches in temporal sequences of images. Each patch is assumed to be the projection of a rigid object with diffuse reflecting surface in a scene with constant illumination. Because small patches, i.e. 11×11 pixel, are used, perspective effects are small and the projection can be approximated by an orthographic camera model. When changes in camera orientation are small, the image intensities in two subsequent images are approximately related by a pure translation $\boldsymbol{d}$. The image intensity I is a function of the position $\boldsymbol{x}$ and the time t, and the intensities between two subsequent images at time t and $t + \mathrm{d}t$ are related by

$$I(\boldsymbol{x}, t) = I(\boldsymbol{x} - \boldsymbol{d}, t + \mathrm{d}t) \quad \Longleftrightarrow \quad \frac{\mathrm{d}I}{\mathrm{d}t} = 0 \tag{4.1}$$

This leads to the "Image-Brightness-Constancy-Equation"

$$\frac{\mathrm{d}I}{\mathrm{d}t} = \frac{\partial I}{\partial \boldsymbol{x}}\frac{\partial \boldsymbol{x}}{\partial t} + \frac{\partial I}{\partial t} = \nabla I\,\boldsymbol{d} + \Delta I = 0 \tag{4.2}$$

with the displacement vector $\boldsymbol{d} = \frac{\partial \boldsymbol{x}}{\partial t} = (\frac{dx}{dt}, \frac{dy}{dt})^T$, the spatial gradient $\nabla I = (\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y})^T$ and the intensity difference $\Delta I = I(x, t = 1) - I(x, t = 0)$.[1]

Equation 4.2 has two unknowns and hence cannot be solved using only a single measurement. Assuming that neighbouring points in a support window W obey the same displacement results in the following equation system

$$\mathbf{A}\ \boldsymbol{d} - \mathbf{b} = \begin{pmatrix} g_{x,0} & g_{y,0} \\ g_{x,1} & g_{y,1} \\ g_{x,2} & g_{y,2} \\ \vdots & \vdots \end{pmatrix}\ \boldsymbol{d} - \begin{pmatrix} -\Delta I_0 \\ -\Delta I_1 \\ -\Delta I_2 \\ \vdots \end{pmatrix} = \boldsymbol{0} \tag{4.4}$$

The usage of the support window has a second positive effect: It enlarges the convergence radius of the minimisation. Multiplying by $\mathbf{A}^T$ from the left leads to the normal equation

$$\begin{gathered} \mathbf{A}^T\mathbf{A}\,\boldsymbol{d} = \mathbf{A}^T\mathbf{b} \\ \begin{pmatrix} \sum_W g_x^2 & \sum_W g_x g_y \\ \sum_W g_x g_y & \sum_W g_y^2 \end{pmatrix} \boldsymbol{d} = \begin{pmatrix} \sum_W \Delta I g_x \\ \sum_W \Delta I g_y \end{pmatrix} \end{gathered} \tag{4.5}$$

A unique solution exists if the matrix $\mathbf{A}^T\mathbf{A}$ is invertible. It can be given in closed form

$$\boldsymbol{d} = (\mathbf{A}^T\mathbf{A})^{-1}\ \mathbf{A}^T\mathbf{b} = \begin{pmatrix} \sum_W g_x^2 & \sum_W g_x g_y \\ \sum_W g_x g_y & \sum_W g_y^2 \end{pmatrix}^{-1} \begin{pmatrix} \sum_W \Delta I g_x \\ \sum_W \Delta I g_y \end{pmatrix} \tag{4.6}$$

[1] For small displacement vectors $\boldsymbol{d}$ equation 4.2 can also be derived using the linear part of a Taylor series

$$I(\boldsymbol{x}, t = 0) = I(\boldsymbol{x} - \boldsymbol{d}, t = 1) \approx I(\boldsymbol{x}, t = 1) + \frac{\mathrm{d}I}{\mathrm{d}\boldsymbol{x}}\boldsymbol{d} \tag{4.3}$$

.

The matrix $\mathbf{A}^T\mathbf{A}$ is also called structure tensor $\mathbf{M}$

$$\mathbf{M} = \mathbf{A}^T\mathbf{A} = \begin{pmatrix} \sum_W g_x^2 & \sum_W g_x g_y \\ \sum_W g_x g_y & \sum_W g_y^2 \end{pmatrix} \tag{4.7}$$

The image brightness function is generally not linear, and hence the solution from equation 4.6 is only approximately true. It can be refined by using the algorithm iteratively until it converges (i.e. the displacement converges against 0).

Big displacements cannot be determined using this approach because the image brightness function is generally not linear and hence the algorithm is used on a Gaussian pyramid of the images. First, the displacement is guessed on the smallest image of the pyramid. The result is used as initial guess on the next bigger image and so forth.

Several modifications and enhancements have been suggested to the algorithm, amongst others the use of an affine similarity function for quality monitoring of feature tracks (Shi and Tomasi, 1994) and the usage of an robust outlier rejection criterion called X84 (Tommasini et al., 1998).

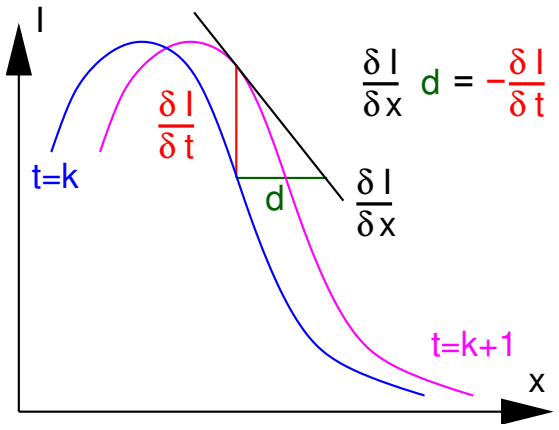


Figure 4.1: One-dimensional intensity function at two different times k (blue) and $k+1$ (magenta) as it is seen by a slit camera. The displacement d can be computed using equation 4.6. The spatial derivative $\partial I/\partial x$ and the temporal derivative $\partial I/\partial t$ are used to compute the displacement d. The displacement is not necessarily equal to the true displacement because of the linearisation of the intensity function.

The algorithm can be visualised for the one-dimensional case (i.e. a slit camera). The 1D-intensity function I is drawn for $t = k$ in blue and $t = k + 1$ in magenta in figure 4.1. The spatial location is given on the abscissa, and the intensity is given on the ordinate. The intensity functions at the two times k and $k + 1$ only differ by a displacement. The displacement d is computed using the spatial intensity gradient and the temporal intensity difference. For nonlinear intensity functions the computed displacement d (equation 4.6) is generally not equal to the true displacement, and hence the algorithm is applied iteratively until it converges.

Covariance Approximation

When the structure tensor is invertible, the covariance of the displacement vector $\boldsymbol{d}$ can be approximated using the approximation from section C.2

$$\Sigma_{\boldsymbol{dd}} \approx \frac{\hat{\mathbf{r}}^T\hat{\mathbf{r}}}{w-2-1}(\mathbf{A}^T\mathbf{A})^{-1} \tag{4.8}$$

with the number of equations w in each support window W, the residual error vector $\hat{\mathbf{r}} \approx \mathbf{A}\hat{\boldsymbol{d}} - \mathbf{b}$ and the structure tensor $\mathbf{A}^T\mathbf{A}$. The covariance matrix of the point in the current image $\Sigma_{\boldsymbol{x}_1\boldsymbol{x}_1}$ is given by the sum of the covariance matrix of the displacement vector $\Sigma_{\boldsymbol{dd}}$ and the covariance matrix of the point in the previous image $\Sigma_{\boldsymbol{x}_0\boldsymbol{x}_0}$

$$\Sigma_{\boldsymbol{x}_1\boldsymbol{x}_1} = \Sigma_{\boldsymbol{x}_0\boldsymbol{x}_0} + \Sigma_{\boldsymbol{dd}} \tag{4.9}$$

Because the new feature position $\boldsymbol{x}_1$ is given by simple addition of the old position and the displacement, equation 4.9 simply states the error propagation for linear functions (see appendix D.3).

The Aperture Problem

(a) (b)

Figure 4.2: Visualisation of the aperture problem: Computation of the displacement vector of a 2D-line segment. The line is drawn in blue at time t and in red at time $t + dt$. (a) If the complete line including the endpoints is used, a human can easily guess the displacement vector. (b) If however only a small support window (black) is used when computing the displacement, a unique solution cannot be determined.

When using point-based methods for correspondence estimation, the aperture problem cannot be neglected. The estimation method requires an invertible structure tensor $\mathbf{M}$. The structure tensor becomes singular (and hence non invertible) when either no structure is visible or when all points in the support window have the same gradient direction. For visualisation purposes the displacement of a line segment is investigated. Figure 4.2 shows the line at time t (blue) and at time $t + dt$ (red). The true displacement (green) is obvious when looking at the complete line segment (a). The algorithm only uses a small support window (black, b). When only information from this support window is used,

a displacement cannot be determined uniquely. A family of possible displacements exist (green arrows, b). This is called the aperture problem.

4.1.2 Feature Detection

Feature detection identifies salient features in the image. Image features are for example corners, edges, regions, blobs, ridges, etc. In this work features are used for the identification of promising regions for correspondence computation. Spatial structure is essential for reliable correspondence computation, and hence a natural decision for structure-based features is made ruling out blobs and regions. Contrary to edges and ridges, corner features circumvent the aperture problem described in section 4.1.1. They are hence chosen for the tracking task.

Different approaches exist in the literature for the estimation process. They can be roughly separated into two classes: Algorithms based on the structure tensor and other algorithms. In the following, a short overview summarising existing corner detection algorithms is presented, and afterwards the choice of a specific algorithm is motivated.

Non Structure-Tensor-Based Corner Detection

SUSAN Corner Detector: The Smallest Univalue Segment Assimilating Nucleus (SUSAN) corner detector (Smith and Brady, 1995) investigates image structure without considering the structure tensor. Pixel in a circular region whose intensity values are similar to the intensity at the centre of the region are identified. A corner is found when three conditions are fulfilled: Firstly, the number of identified pixels must be below a threshold. Secondly, the centroid of the identified pixels must be far from the centre of the region, and thirdly, all pixels on the line connecting the centre and the centroid must be identified.

FAST Corner Detector: The Feature from Accelerated Segment Test (FAST) corner detector (Rosten and Drummond, 2005, 2006) investigates pixel on a circle around the centre. A corner is found when at least n contiguous pixel on the circle differ by at least t from the centre. The contiguous pixel on the circle must have either all bigger intensity values or all smaller intensity values when the centre is located on a corner.

Wang and Brady Corner Detector: The corner detector of Wang and Brady (1995) computes a cornerness measure by regarding the intensity as a surface and computing the curvature along an image edge.

Structure Tensor Based Corner Detection

Several structure tensor based methods for corner detection exist. The computation of a "cornerness" measure using the structure tensor $\mathbf{M}$ for each pixel is common to all these methods. Afterwards, a non maximum suppression extracts the corners from the cornerness

function. Subpixel accuracy can be achieved for example by using a parabola approximation of the cornerness around the location in question and computing the maximum of the parabola.

Harris Corner Detector: The Harris corner detector (Harris and Stephens, 1988) computes the cornerness c_H as

$$c_H = \det(\mathbf{M}) - \kappa \operatorname{trace}(\mathbf{M})^2 \tag{4.10}$$

Förstner Corner Detector: The Förstner corner detector (Förstner, 1986) computes the cornerness c_F as

$$c_F = \frac{\det(\mathbf{M})}{\operatorname{trace}(\mathbf{M})} \tag{4.11}$$

KLT Corner Detector: In 1981 Lucas and Kanade suggested an iterative gradient-based tracking algorithm (Lucas and Kanade, 1981) neglecting the question of feature selection. The resulting algorithm has been explained in detail in section 4.1.1. In 1991 Tomasi and Kanade (1991a) derived an optimal solution for the feature selection problem based on the tracking algorithm from Lucas and Kanade (1981). The bundle of feature selection and image registration is called the KLT (Kanade Lucas Tomasi) algorithm.

The KLT corner detector (Tomasi and Kanade, 1991a) computes the cornerness c_K as the smaller of the two eigenvalues λ_1 and λ_2 of the structure tensor

$$c_K = \min(\lambda_1, \lambda_2) \tag{4.12}$$

The structure tensor $\mathbf{M}$ is a symmetric real 2×2 matrix

$$\mathbf{M} = \begin{pmatrix} g_{xx} & g_{xy} \\ g_{xy} & g_{yy} \end{pmatrix} \tag{4.13}$$

with the abbreviations $g_{xx} = \sum_W (\frac{\partial I}{\partial x})^2$, $g_{yy} = \sum_W (\frac{\partial I}{\partial y})^2$ and $g_{xy} = \sum_W \frac{\partial I}{\partial x}\frac{\partial I}{\partial y}$ summing first order spatial derivatives over a support window W. Sometimes weights depending on the location in the support window are used in the sum. The eigenvalues $\lambda_{1,2}$ of the structure tensor can be computed by solving for the roots of the characteristic polynomial explicitly

$$\lambda_{1,2} = \frac{g_{xx} + g_{yy}}{2} \pm \frac{1}{2}\sqrt{\frac{(g_{xx} + g_{yy})^2}{4} - g_{xy}^2} \tag{4.14}$$

The support window can be classified based on the two eigenvalues: Two small eigenvalues correspond to approximately uniform regions without significant spatial structure, a large and a small eigenvalue correspond to regions with a dominant linear structure and two small eigenvalues correspond to regions rich in texture, e.g. corners, salt and pepper textures.

Köthes Improvements to Structure Tensor Computation: Köthe (2003) showed that Shannon's sampling theorem is violated when the structure tensor is computed using the original image resolution[2]. He suggests that the structure tensor must be computed using an image with higher resolution than the original image to avoid aliasing. Secondly, anisotropic filtering is used to enhance the localisation accuracy of corners. Hourglass-shaped filters with orientation according to the gradient are used for this task. Thirdly, an algorithm for integrated junction/corner and edge detection is proposed. The structure tensor is decomposed into two parts, one part representing the intrinsic 1D-properties (corner information) at the current location and the other part representing the intrinsic 2D-properties (edge information).

Selection of Feature Detection Algorithm

The KLT corner detector is selected for the final system, because it is based on the tracking equations and hence optimally collaborates with the chosen tracking algorithm. Even though the sampling theorem is theoretically violated by structure tensor computation, the practical effect is small and leads only to a small error in the initial position of the corner. Because only relative correspondences are of concern for the estimation of the essential matrix and because upsampling of the image imposes a significant computational cost, this improvement suggested by Köthe is not used in this thesis.

4.1.3 Conclusions

The selected tracking algorithm Lucas and Kanade (1981) and the corner detector Tomasi and Kanade (1991a) constitute an optimally integrated feature tracking system. Because the original idea is rather old, a large number of additional improvements have been suggested for the original tracking algorithm, including for example affine motion models and compensation of illumination changes. However, since only very small image patches of 15×15 pixels are used in this work, none of these improvements is necessary.

[2]The gradient of an image is usually computed by convolution with two derivative filters, for example spatial derivatives of a 2D-Gaussian can be used as convolution mask. The gradient tensor is defined as the outer product of the gradient vector with itself. The structure tensor can be seen as the averaged gradient tensor, i.e. a convolution of the gradient tensor image with a low pass filter mask, for example a Gaussian mask or a box filter. When the original image was properly sampled at Nyquist rate, the original signal was bandlimited, and hence the derivatives are still bandlimited. However, multiplying two bandlimited signals doubles the bandwidth and hence the resulting signal (i.e. the structure tensor) must be represented at half the sampling frequency to avoid aliasing. The image can also be represented at the double sampling frequency alternatively.

4.2 Egomotion Estimation

The detection of independent motion requires knowledge about the camera motion. The objective is to compute the vehicle motion between two successive images of a sequence. First the computation of the camera motion using vehicle inertial sensors is described in section 4.2.1. It is, however, not sufficient, because important sensors, e.g providing information about roll movement, are missing. Egomotion from car inertial sensor data can nonetheless be used as initial guess to speed up the image-based egomotion estimation. The computation of the essential matrix E from point correspondences is described in section 4.2.2.

4.2.1 Vehicle Inertial Sensors

Because the camera is mounted in an automobile, vehicle inertial sensors measuring

- speed v,
- steering angle δ_A,
- yaw rate $\dot{\alpha}$,
- lateral acceleration a_l,
- longitudinal acceleration a,
- time t and
- DGPS position and speed data

can be used to estimate the egomotion of the camera relative to the static scene. The accuracy of the speed sensor is assumed to be 0.13 m/s. The accuracy of the steering angle sensor is given by 0.14°, the accuracy of the yaw rate sensor is given as 0.3°/s, and the accuracy of the longitudinal and lateral acceleration sensors are assumed to be 0.01 m/s^2. The timestamps are assumed to have an accuracy of 3 ms, and the DGPS position data is assumed to be 1.1 m. See section 5.5.1 for a detailed description of the inertial sensors.

It is necessary to compute the egomotion of the vehicle for the computation of the egomotion of the camera. The derivation of the camera motion from the vehicle motion is described first.

Because of the lack of information about vertical motion, it is assumed that the vehicle is moving in the x-z-plane. The vehicle coordinate system is defined such that the positive z-axis coincides with the forward direction of the vehicle and the positive x-axis coincides with the "right" direction of the vehicle. In this section, the world coordinate system is defined using the vehicle coordinate system at the time of the first frame. The transformation from vehicle to world coordinate system is given by the Euclidean transformation

$$\mathbf{T}_{CW} = \begin{pmatrix} \mathbf{R}_{CW}(t) & \boldsymbol{C}(t) \\ \boldsymbol{0}^T & 1 \end{pmatrix} \tag{4.15}$$

with the rotation matrix $\mathbf{R}_{CW}(t) \in \mathbb{R}^{3\times3}$ given by the orientation and the position of the vehicle in the world coordinate system $\boldsymbol{C}(t) \in \mathbb{R}^3$. Obviously, both the orientation of the vehicle and the position of the vehicle change with time and hence need to be updated after every new vehicle motion estimation.

In this work three simple algorithms have been used to estimate the egomotion of the vehicle. The computation of the vehicle motion using the steering angle or yaw rate and the computation from differential GPS data are described next.

Steering angle

The demonstrator which has been used in this thesis is called urban traffic assistant (UTA)[3]. It is equipped with an steering angle sensor. The steering angle is directly measured with an accuracy of $\approx 0.14°$. However, the zero position is determined manually introducing a constant but unknown offset in the measurement. The vehicle dynamics can be modelled

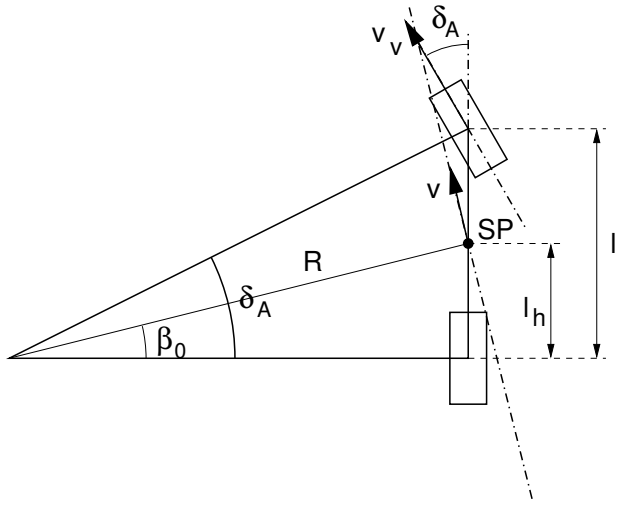


Figure 4.3: Geometry of simple vehicle model for circular driving under static conditions. The steering angle δ_A can be used to compute the radius R via the side slip angle β_0. The vehicle has a wheelbase l, and the barycentre SP is located at a distance l_h from the rear axis. (Figure similar to Zomotor (1991))

using a simple single track model. When the lateral acceleration is very small, the lateral slip on the wheels can safely be neglected. For experiments with low speed and therefore with low lateral acceleration, it is sufficient to use this simple vehicle model for circular driving under static conditions (Zomotor, 1991). Figure 4.3 shows the geometry of the simple vehicle model.

The steering angle δ_A (also called Ackermannwinkel) can be used to compute the side slip angle β_0 if the wheelbase of the vehicle l and the distance of the barycentre from the

[3]For a detailed description refer to section 5.5.1.

rear axis l_h are known (Zomotor, 1991)

$$\beta_0 \approx \frac{l_h}{R} \approx \delta_A \frac{l_h}{l} \tag{4.16}$$

Hereby the magnitude of the angles δ_A and β_0 are assumed to be small, and hence the usual linearisation approximations for trigonometric functions hold (i.e. $\sin(\delta_A) \approx \delta_A$ and $\cos(\delta_A) \approx 1$). Re-arranging terms results in the radius of the circular motion R

$$R \approx \frac{l}{\delta_A} \tag{4.17}$$

Without the linearisation of the trigonometric functions the side slip angle becomes

$$\beta_0 = \tan\left(\frac{l_h \tan(\delta_A)}{l}\right) \tag{4.18}$$

and the radius is given by

$$R = \operatorname{sgn}(\delta_A)\sqrt{R_\nu^2 + l_h^2} \tag{4.19}$$

with $R_\nu = \frac{l_h}{\tan(\beta_0)}$.

Assuming constant longitudinal acceleration a, the distance d that the vehicle moved between the two images can be computed using the velocity v and the time interval Δt between two images as

$$d = v\Delta t + \frac{1}{2}a(\Delta t)^2 \tag{4.20}$$

The angle α of the circle segment on which the vehicle is moving is given by

$$\alpha = \frac{d}{R} \tag{4.21}$$

In the vehicle coordinate system, the new position of the vehicle $\boldsymbol{B}_C$ is given by

$$\boldsymbol{B}_C = R \begin{pmatrix} \cos(\alpha) - 1 \\ 0 \\ \sin(\alpha) \end{pmatrix} \tag{4.22}$$

Equation 4.22 has a singularity at $\alpha = 0$. For small angles δ_A the trigonometric functions are approximated by their Taylor series up to the quadratic term resulting in the new position of the vehicle

$$\boldsymbol{B}_C = \frac{d}{\alpha} \begin{pmatrix} 1 - \frac{\alpha^2}{2} - 1 \\ 0 \\ \alpha \end{pmatrix} = d \begin{pmatrix} -\operatorname{sgn}(\alpha)\frac{\alpha}{2} \\ 0 \\ 1 \end{pmatrix} \tag{4.23}$$

The transformation into the world coordinate system is given by equation 4.15.

The impact of the linearisation (equations 4.17 and 4.23) are however small. This has been empirically tested by computing the vehicle position on a closed sequence of 745 measurements twice, once using the linearisation and once using the exact equations. The total range covered in the vehicle in the test sequence is approximately 720 m, and the vehicle speed ranges from 4.5 to 11.3 m/s. The difference of the final positions of the exact and the linearised approach is only 1.5 m, and hence the linearisation can safely be used with speed ranges up to ≈ 10 m/s.

Yaw Rate

The yaw rate sensor installed in UTA has an accuracy of $\approx 0.3°/s$. During the experiments the yaw rate sensor exhibited however sensitivity to changes in temperature resulting in significant drift. The drift could be minimised by ensuring approximately constant temperature during operation.

The yaw rate $\dot{\alpha}$ can be used directly to compute the angular motion of the vehicle using the time between two images Δt. The angle α is given by

$$\alpha = \dot{\alpha}\Delta t \tag{4.24}$$

The distance d that the vehicle has moved can be computed using equation 4.20 and the radius of the circle segment is given by

$$R = \frac{d}{\alpha} \tag{4.25}$$

For the computation of the new position of the vehicle in the vehicle coordinate system equation 4.22 can be used. The transformation into the world coordinate system is given by equation 4.15. Equation 4.25 has a singularity at $\alpha = 0$. For small angles α the trigonometric function are approximated by their Taylor series up to the quadratic term resulting in the new position of the vehicle

$$\boldsymbol{B}_C = \frac{d}{\alpha}\begin{pmatrix} 1 - \frac{\alpha^2}{2} - 1 \\ 0 \\ \alpha \end{pmatrix} = d\begin{pmatrix} -\operatorname{sgn}(\alpha)\frac{\alpha}{2} \\ 0 \\ 1 \end{pmatrix} \tag{4.26}$$

Global Positioning System (GPS)

The GPS system works by computing the distance to three or more satellites by measuring the signal transmission times. When the positions of the satellites are known, the own position can be computed.

A differential GPS system (DGPS) is available in the demonstrator vehicle. It is based on the ordinary GPS system and a network of ground based reference stations broadcasting the difference between the position as indicated by the satellite system and the known position of the station. A DGPS receiver corrects the satellite based measurements by the error from the nearest broadcast station.

Additionally to the position, the velocity of the receiver can also be computed using the Doppler frequency shift.

The differential GPS system can achieve accuracies of less than a meter near a broadcast station. The error grows with distance to the broadcast station approximately by 0.2 m per 100 km distance to the station (Monteiro et al., 2004). The closest broadcast station has a distance of ≈ 50 km the site where the following experiment was conducted, resulting in an error estimate for the DGPS method of 1.1 m. In theory the DGPS method measures of the absolute position of the vehicle and hence the error estimate is independent of the history.

DGPS Measurements are available with approximately 1 Hertz. The raw DGPS measurements are not used because of the low sample rate and the low relative accuracy for example. Instead, the absolute position of the vehicle is computed from all vehicle inertial sensors using a Kalman filter (Gern, 2000).

Computing the Camera Motion from Vehicle Motion

The previous sections describe the estimation of the motion of the barycentre of the vehicle. The camera is however not located at the barycentre. It has a known offset $\boldsymbol{t}_{CC}$ and orientation R_{CC} relative to the vehicle. The transformation between vehicle and camera coordinate system is given by

$$\mathbf{T}_{CC} = \begin{pmatrix} \mathbf{R}_{CC}(t) & \boldsymbol{t}_{CC}(t) \\ \boldsymbol{0}^T & 1 \end{pmatrix} \tag{4.27}$$

When the camera is mounted on a pan-tilt unit, the offset $\mathbf{R}_{CC}$ and orientation $\boldsymbol{t}_{CC}$ may change with time, but in this thesis it is assumed to be known and constant.

Temporal Alignment of Vehicle Inertial Sensors and Images: The measurements from the vehicle inertial sensors and image capturing is an asynchronous process. Therefore it is necessary to align the data temporarily. Fortunately, both the images and the vehicle sensor data are marked with time stamps. A very simple approach is used: The vehicle inertial sensor data are linearly interpolated to obtain approximate values synchronous with the image frames.

Comparison of Methods Using Vehicle Inertial Sensors

The different methods (see above) of vehicle egomotion computation are compared on the closed "Zollberg" sequence (i.e. the camera position at the last frame of the sequence is approximately the same as at the first frame). The speed in this sequence ranges from 4.5 to 11.3 m/s. Figure 4.4 shows some images from the "Zollberg" sequence. Figure 4.5 shows a bird's-eye view on the tracks computed using the different methods. The position uncertainties for the last positions of the different methods are indicated by covariance ellipsoids. The uncertainties are derived using the unscented transform (section D.3) from

Figure 4.4: Some selected frames from the closed loop "Zollberg" sequence.

the known accuracies of the inertial sensors. Only the GPS gives absolute position information. The motion information from all other inertial sensors is only relative, and hence the uncertainties are accumulated over the sequence.

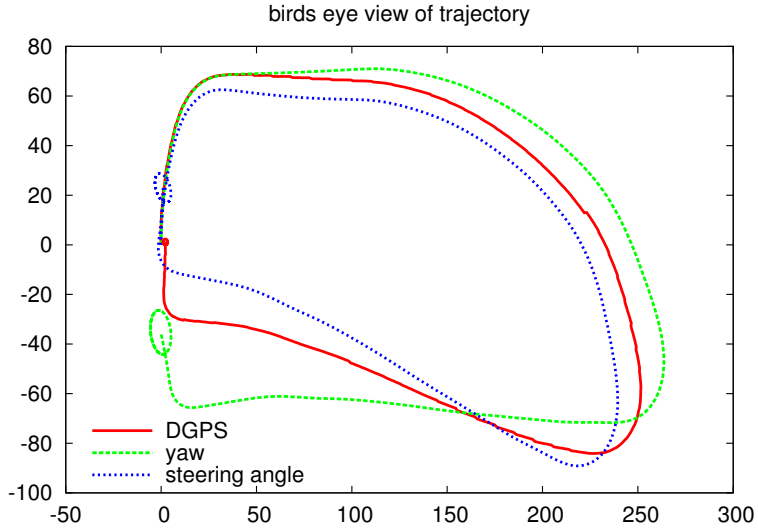


Figure 4.5: Comparison of the different methods for egomotion computation using vehicle inertial sensors. The DGPS method is given in red, the steering angle method is drawn blue and the yaw rate method is drawn green. A bird's-eye view on the positions as computed from the "Zollberg" sequence is shown.

Only the GPS method achieves global consistency and hence a closed track. This can be explained by the fact, that in contrast to the other methods the DGPS method measures absolute positions. The steering angle method overestimates the yaw rotation. This could be explained by the fact that the slip of the wheels cannot be neglected in the speed ranges used in this sequence. The manual determination of the zero steering angle introduces a further significant uncertainty into the estimation process. The yaw rate method underestimates the yaw rotation. This could be explained by an offset in the yaw rate measurements. The specification indicates that the offset is in the range of

$\pm 3°$. The overall time which passed between first and last frame of the sequence was 74.5s resulting in a maximum angular error of 223°. Obviously the offset was much smaller in the case observed here.

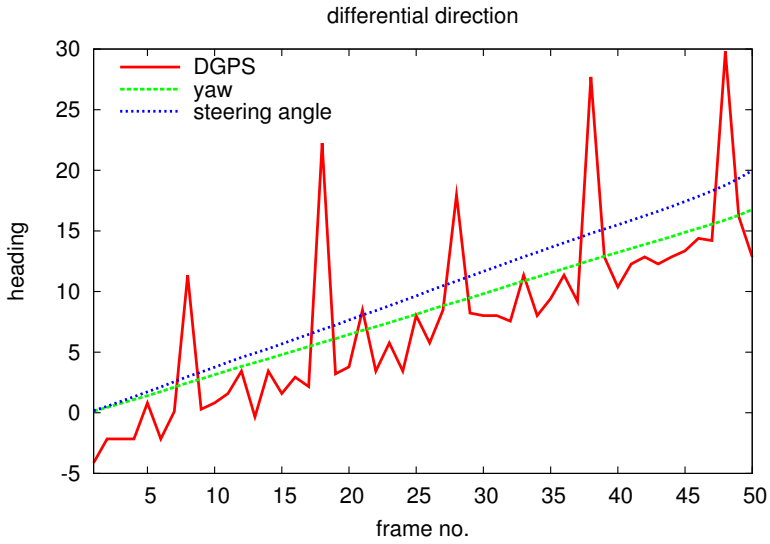


Figure 4.6: Comparison of the different methods for egomotion computation using vehicle inertial sensors. The DGPS method is given in red, the steering angle method is drawn blue and the yaw rate method is drawn green. The heading angle of the car is plotted vs. the frame number. The heading angle is computed using the difference vector of the subsequent positions.

Global consistency is however not the main application of the car inertial data in this thesis, and hence the local consistency of the estimators must be investigated. The heading angle is computed from two subsequent position measurements and plotted vs. the frame number (figure 4.6). There are no abrupt direction changes and because of the moments of inertia, the car motion is smooth in the examined sequence "Zollberg". The steering angle and yaw rate methods exhibit superior performance over the DGPS method when the primary aim is to obtain local consistent positions. A significant difference between the steering angle method and the yaw rate method could not be identified. The yaw rate algorithm has been chosen for the final system.

Two problems arise when using the car inertial sensors for the visual detection of independent motion. Firstly, images and inertial sensors are not synchronised and hence the sensor measurements are not sufficiently accurate for prediction of optical flow with a precision of a tenth of a pixel. And secondly, information about roll and pitch axes is not provided by inertial sensors. It is therefore obligatory to use image information for the refinement of the egomotion. Different algorithms for image-based egomotion estimation are compared next.

4.2.2 Essential Matrix Estimation

The essential matrix only contains the minimal necessary motion information for detection of independently moving objects: The motion direction (i.e. the epipole) and the relative rotation. This information is generally sufficient for detection of independent motion, however in the case when the absolute camera position is needed, the information from the essential matrix can be augmented by the distance computed from integrated velocity measurements. The computation of the essential matrix from image point correspondences is described in this section. Basis for all algorithms is a number of N point correspondences $\boldsymbol{x}_i = (x_{ix}, x_{iy}, x_{iw})^T$ and $\boldsymbol{x}_i^{'} = (x'_{ix}, x'_{iy}, x'_{iw})^T$ between the two images. A number of different approaches of essential matrix computation are empirically compared regarding accuracy and computational requirements. Two direct methods, the linear 8 point algorithm (Hartley and Zissermann, 2004) and the 5 point algorithm from Nistér (2003a), are compared with four nonlinear algorithms. The Levenberg-Marquardt (LM) algorithm (Hartley and Zissermann, 2004) is used to minimise three different objective functions: The geometric error, an angle-based objective function and the Mahalanobis distance between point and corresponding epipolar line. Another minimisation algorithm, the Gauss-Helmert model (McGlone, 2004), is used to minimise the algebraic error. The Gauss-Helmert model has the advantage that internal constraints between the parameters, e.g. the unit-norm constraint on a quaternion describing a rotation, can be directly enforced during estimation.

Linear: Following Hartley and Zissermann (2004), the linear algorithm is based on the linear estimation of a fundamental matrix $\mathbf{F}$. The resulting linear system has 8 unknowns and hence 8 point correspondences are required for the estimation process. Afterwards the intrinsic constraints of essential matrices are enforced using singular value decomposition (SVD). An essential matrix has two identical eigenvalues and the third eigenvalue is zero (Hartley and Zissermann, 2004)[4]. Given the decomposition of the fundamental matrix $\mathbf{F} = \mathbf{U}\mathbf{S}\mathbf{V}^T$, the nearest essential matrix $\mathbf{E}$ to $\mathbf{F}$ in Frobenius norm is computed by replacing the two nonzero eigenvalues in the diagonal matrix $\mathbf{S}$ by their mean. The resulting diagonal matrix $\mathbf{S}'$ is used to construct the essential matrix $\mathbf{E} = \mathbf{U}\mathbf{S}'\mathbf{V}^T$ (Hartley and Zissermann, 2004).

5 Point Algorithm: The 5 point algorithm described next was invented by Nistér (2003a). Its main advantage is that it works even if all point correspondences are stemming from 3D-points located on a plane. First the essential constraint is used to compute the 4

[4]In fact, Kanatani (2005) claims that an essential matrix $\mathbf{E}$ can only be decomposed into a unit vector $\boldsymbol{e}$ and a rotation matrix $\mathbf{R}$ such that $\mathbf{E} = [\boldsymbol{e}]_\times \mathbf{R}$ if and only if its singular values are 1, 1 and zero. However, this decomposability constraint is not enforced in this algorithm.

vectors spanning the solution space by rewriting $\boldsymbol{x}'^T\mathbf{E}\boldsymbol{x} = 0$ (equation 2.26) as

$$\begin{pmatrix} \tilde{\boldsymbol{x}}_1^T \\ \tilde{\boldsymbol{x}}_2^T \\ \tilde{\boldsymbol{x}}_3^T \\ \tilde{\boldsymbol{x}}_4^T \\ \tilde{\boldsymbol{x}}_5^T \end{pmatrix} \tilde{\boldsymbol{E}} = 0 \tag{4.28}$$

with the vector

$$\tilde{\boldsymbol{E}} = (E_{11}, E_{12}, E_{13}, E_{21}, E_{22}, E_{23}, E_{31}, E_{32}, E_{33})^T$$

consisting of the stacked columns of the essential matrix and the vector

$$\tilde{\boldsymbol{x}} = (x_x x'_x, x_y x'_x, x_w x'_x, x_x x'_y, x_y x'_y, x_w x'_y, x_x x'_w, x_y x'_w, x_w x'_w)^T$$

consisting of combinations of the elements of point correspondence $\boldsymbol{x}$ and $\boldsymbol{x}'$. The solution space is spanned by the four vectors $\tilde{\boldsymbol{X}}$, $\tilde{\boldsymbol{Y}}$, $\tilde{\boldsymbol{Z}}$ and $\tilde{\boldsymbol{W}}$, and hence the essential matrix is a linear combination of the four corresponding matrices $\mathbf{X}$, $\mathbf{Y}$, $\mathbf{Z}$ and $\mathbf{W}$

$$\mathbf{E} = x\mathbf{X} + y\mathbf{Y} + z\mathbf{Z} + w\mathbf{W} \tag{4.29}$$

Because the essential matrix is only defined up to scale, the last coefficient w is – without the loss of generality – assumed to be 1. Inserting equation 4.29 into the cubic constraints (2.27)

$$\det(\mathbf{E}) = 0$$

and (2.29)

$$\mathbf{E}\mathbf{E}^T\mathbf{E} - \frac{1}{2}\operatorname{trace}(\mathbf{E}\mathbf{E}^T)\mathbf{E} = 0$$

and performing Gauss-Jordan elimination with partial pivoting results in the equation system

$$\begin{pmatrix} < a > \\ < b > \\ < c > \\ < d > \\ < e > \\ < f > \\ < g > \\ < h > \\ < i > \\ < j > \end{pmatrix} \begin{pmatrix} x^3 \\ y^3 \\ x^2y \\ xy^2 \\ x^2z \\ x^2 \\ y^2z \\ y^2 \\ xyz \\ xy \\ x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & . & . & . & . & . & . & . & . & . & [2] & [2] & [3] \\ & 1 & . & . & . & . & . & . & . & . & [2] & [2] & [3] \\ & & 1 & . & . & . & . & . & . & . & [2] & [2] & [3] \\ & & & 1 & . & . & . & . & . & . & [2] & [2] & [3] \\ & & & & 1 & & & & & & [2] & [2] & [3] \\ & & & & & 1 & & & & & [2] & [2] & [3] \\ & & & & & & 1 & & & & [2] & [2] & [3] \\ & & & & & & & 1 & & & [2] & [2] & [3] \\ & & & & & & & & 1 & & [2] & [2] & [3] \\ & & & & & & & & & 1 & [2] & [2] & [3] \end{pmatrix} \begin{pmatrix} x^3 \\ y^3 \\ x^2y \\ xy^2 \\ x^2z \\ x^2 \\ y^2z \\ y^2 \\ xyz \\ xy \\ x \\ y \\ 1 \end{pmatrix} = \boldsymbol{0} \tag{4.30}$$

with scalar values denoted by a dot (.), polynomials of degree N in z denoted by $[N]$. The rows of the equation system are named from $< a >$ to $< j >$. Subtracting rows from equation system 4.30 results in

$$\mathbf{B}\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} < e > -z < f > \\ < g > -z < h > \\ < i > -z < j > \end{pmatrix}\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} [3] & [3] & [4] \\ [3] & [3] & [4] \\ [3] & [3] & [4] \end{pmatrix}\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \boldsymbol{0} \tag{4.31}$$

where polynomials in z of degree N are again denoted by $[N]$. A solution exists iff

$$\det(\mathbf{B}) \neq 0 \tag{4.32}$$

resulting in a polynomial of degree 10 in z. The real roots of this polynomial are computed numerically, and for each root the accompanying values of x and y are computed using equation 4.31. The resulting essential matrices are obtained, and the relative orientation and the epipole can be extracted as described in section 2.3.1.

Geometric Error: The distances of the feature points $\boldsymbol{x}_{i1} = (x_{i1,x}, x_{i1,y}, x_{i1,w})^T$ and $\boldsymbol{x}_{i2} = (x_{i2,x}, x_{i2,y}, x_{i2,w})^T$ from the corresponding epipolar lines in the images are minimised in this algorithm leading to the objective function

$$\mathbf{E}(\boldsymbol{e}_\alpha, \boldsymbol{q}) = \arg\min_{\boldsymbol{e}_\alpha, \boldsymbol{q}} \sum_{i=1}^{N} \frac{\boldsymbol{x}_{i1}^T \boldsymbol{l}_{i1}}{(l_{i1,x}^2 + l_{i1,y}^2) x_{i1,w}^2} + \frac{\boldsymbol{x}_{i2}^T \boldsymbol{l}_{i2}}{(l_{i2,x}^2 + l_{i2,y}^2) x_{i2,w}^2} \tag{4.33}$$

where the epipolar lines $\boldsymbol{l}_{ik} = (l_{i1,x}, l_{i1,y}, l_{i1,w})^T$ are given by $\boldsymbol{l}_{i1} = E_{21}(\boldsymbol{e}_\alpha, \boldsymbol{q})\boldsymbol{x}_{i2}$ and $\boldsymbol{l}_{i2} = E_{12}(\boldsymbol{e}_\alpha, \boldsymbol{q})\boldsymbol{x}_{i1}$. Equation 4.33 is solved iteratively using the Levenberg Marquardt algorithm. The essential matrix is parametrised as a quaternion $\boldsymbol{q}$ describing the rotation and sphere angles $\boldsymbol{e}_\alpha$ describing the epipole. Numerical techniques are used for the minimisation of equation 4.33, and hence an initial guess sufficiently close to the solution must be provided to the algorithm. Contrary to the other estimation algorithms, a robust cost function is employed in the iteration process, resulting in better convergence even if a minor percentage of outlying data is still present in the correspondences. The Huber cost function is used (cf. appendix C.1).

Angles: Two angles are minimised in this objective function:

1. The homogeneous representation of a 2D-point in projective space is a 3-dimensional vector. It can also be interpreted as a line through the origin in 3D Euclidean space. The homogeneous representation of a 2D-line in projective space also is a 3D-vector. It can be interpreted as the normal vector to a plane through the origin. Each homogeneous vector is only defined up to scale, and hence unit length 3D-vectors are used in the computation. The normalisation to length 1 is dropped for the sake of clarity in the following. The scalar product between projective point and projective line $\boldsymbol{l}_2 = \mathbf{E}\boldsymbol{x}_1$ is equal to the cosine of the angle between the two vectors when unit length vectors are used. In this case the angle α between Euclidean line $\boldsymbol{x}_2$ and Euclidean plane with normal $\boldsymbol{l}_2$ is given by $\alpha_2 = \pi/2 - \text{acos}(\boldsymbol{x}_2^T \boldsymbol{l}_2) = \text{asin}(\boldsymbol{x}_2^T \boldsymbol{l}_2)$.

2. The 2D-projective line $\boldsymbol{t}$ representing the translational part of each correspondence vector is given by $\boldsymbol{t}_2 = [\boldsymbol{x}_2]_\times \mathbf{R}\boldsymbol{x}_1$. The angle between the epipolar line $\boldsymbol{l}_2$ and $\boldsymbol{t}_2$ is given by $\beta_2 = \text{acos}(\boldsymbol{l}_2^T \boldsymbol{t}_2)$. This angle is only well defined when the translational flow has not length zero. The length of the translational part of the correspondence is given by $\iota_2 = |\boldsymbol{x}_2 - \mathbf{R}^T\boldsymbol{x}_1|$. It is used as a weight for β_2 in the objective function.

The objective function is given by

$$\mathbf{E}(\boldsymbol{e}_\alpha, \boldsymbol{q}) = \arg\min_{\boldsymbol{e}_\alpha, \boldsymbol{q}} \left[\sum_{i=1}^{N} \alpha_1^2 + \alpha_2^2 + \iota_1^2\beta_1^2 + \iota_2^2\beta_2^2 \right] \tag{4.34}$$

The complementary angles α_1 and β_1 are computed by interchanging the roles of the points $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$. Note that all points and lines must be normalised to length 1 for the computation of the angles.

The parametrisation of the essential matrix is again given by a quaternion $\boldsymbol{q}$ describing the rotation and sphere angles $\boldsymbol{e}_\alpha$ describing the epipole. N is the number of point correspondences used in the estimation process. Equation 4.34 is solved using the Levenberg-Marquardt algorithm.

Mahalanobis Distance: A covariance matrix can be obtained for every interest point in the images (see sections 4.1.1 and 4.1.2). The knowledge about these uncertainties can be incorporated into the estimation process by minimising the Mahalanobis distances between the points and the respective epipolar lines.

Given a point correspondence $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ in projective space and a fixed essential matrix $\mathbf{E}_{12}$, the corresponding epipolar line to $\boldsymbol{x}_1$ in image 2, $\boldsymbol{l}_2$ is given by

$$\boldsymbol{l}_2 = \mathbf{E}_{12}\boldsymbol{x}_1 \tag{4.35}$$

The construction process is linear in the point $\boldsymbol{x}_1$ and the covariance matrix of the line $\boldsymbol{\Sigma}_{\boldsymbol{ll}}$ can be computed rigorously using Gaussian error propagation (section D.3)

$$\boldsymbol{\Sigma}_{\boldsymbol{l}_2\boldsymbol{l}_2} = \mathbf{E}_{12}\boldsymbol{\Sigma}_{\boldsymbol{x}_1\boldsymbol{x}_1}\mathbf{E}_{12}^T \tag{4.36}$$

The algebraic distance d_a between the point $\boldsymbol{x}_2$ and the line $\boldsymbol{l}_2$ is given by

$$d_a^2(\boldsymbol{x}_2, \boldsymbol{l}_2) = \boldsymbol{l}_2^T\boldsymbol{x}_2 = \boldsymbol{x}_2^T\boldsymbol{l}_2 \tag{4.37}$$

The algebraic distance d_a can be seen as a function with arguments $\boldsymbol{x}_2$ and $\boldsymbol{l}_2$. The Jacobian $\mathbf{J}(d_a)$ of the distance function is given by

$$\mathbf{J}(d_a) = [\boldsymbol{x}_2^T | \boldsymbol{l}_2^T] \tag{4.38}$$

If the line $\boldsymbol{l}_2$ and the point $\boldsymbol{x}_2$ are uncorrelated, the joint covariance of line and point $\boldsymbol{\Sigma}_{\boldsymbol{xlxl}}$ has block diagonal form

$$\boldsymbol{\Sigma}_{\boldsymbol{xlxl}} = \begin{pmatrix} \boldsymbol{\Sigma}_{\boldsymbol{x}_2\boldsymbol{x}_2} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Sigma}_{\boldsymbol{l}_2\boldsymbol{l}_2} \end{pmatrix} \tag{4.39}$$

The covariance matrix[5] of the algebraic distance σ_{d_a} can again be computed rigorously using Gaussian error propagation

$$\sigma_{d_a}^2 = \mathbf{J}(d_a)\boldsymbol{\Sigma}_{\boldsymbol{xlxl}}\mathbf{J}(d_a)^T \tag{4.40}$$

Hence the Mahalanobis distance d_{m2}^2 (see section D.4) between line $\boldsymbol{l}_2$ and point $\boldsymbol{x}_2$ is given by

$$d_{m2}^2 = \frac{d_a^2}{\sigma_{d_a}^2} \tag{4.41}$$

Using the fact that $\mathbf{E}_{21} = \mathbf{E}_{12}^T$, the Mahalanobis distance d_{m1}^2 between the line $\boldsymbol{l}_1$ and the point $\boldsymbol{x}_1$ can also be computed, resulting in two error measures for every point correspondence. The Levenberg-Marquardt algorithm is applied to compute the essential matrix using these error measures. Because the essential matrix has only five degrees of freedom (section 2.3.1), a appropriate parametrisation has to be chosen for good results. The epipole is parametrised using the two sphere angles, and the orientation is parametrised using a unit quaternion (see section A.2). Following Förstner (2005), it is very important to use normalised vectors to avoid numerical problems. The normalisation of a vector $\boldsymbol{x}$

$$\hat{\boldsymbol{x}} = \frac{1}{|\boldsymbol{x}|}\boldsymbol{x} \tag{4.42}$$

has an influence on its covariance matrix and its Jacobian $\mathbf{J}_{\text{norm}}$ is given by (Förstner, 2005)

$$\mathbf{J}_{\text{norm}} = \frac{1}{|\boldsymbol{x}|}\left(\mathbb{I} - \frac{\boldsymbol{x}\boldsymbol{x}^T}{\boldsymbol{x}^T\boldsymbol{x}}\right) \tag{4.43}$$

Hence the covariance matrix of the normalised vector can be approximated by

$$\boldsymbol{\Sigma}_{\hat{\boldsymbol{x}}\hat{\boldsymbol{x}}} = \mathbf{J}_{\text{norm}}\boldsymbol{\Sigma}_{\boldsymbol{xx}}\mathbf{J}_{\text{norm}}^T \tag{4.44}$$

Algebraic Error: The essential matrix is computed by minimisation of the algebraic error using the Gauss-Helmert model (McGlone, 2004). The essential matrix is parametrised as a unit vector describing the epipole and a unit quaternion describing the relative orientation. Two constraints among the unknowns are directly required by this parametrisation: The epipole and the quaternion both must have length one

$$\mathbf{h}(\mathbf{p}) = \mathbf{h}(\boldsymbol{e}, \boldsymbol{q}) = \begin{pmatrix} \boldsymbol{e}^T\boldsymbol{e} - 1 \\ \boldsymbol{q}^T\boldsymbol{q} - 1 \end{pmatrix} = \boldsymbol{0} \tag{4.45}$$

The functional relationships between the observed image point correspondences and the parameters are given by the essential constraint (equation 2.26)

$$\mathbf{g}(\mathbf{p}, \mathbf{l}) = \mathbf{g}(\boldsymbol{e}, \boldsymbol{q}, \boldsymbol{x}_{i,1}, \boldsymbol{x}_{i,2}, \ldots) = \begin{pmatrix} \boldsymbol{x}_{1,2}^T\mathbf{E}(\boldsymbol{e}, \boldsymbol{q})\boldsymbol{x}_{1,1} \\ \vdots \\ \boldsymbol{x}_{n,2}^T\mathbf{E}(\boldsymbol{e}, \boldsymbol{q})\boldsymbol{x}_{n,1} \end{pmatrix} = \boldsymbol{0} \tag{4.46}$$

[5]The covariance matrix boils down to a variance in this case, because the algebraic distance has exactly one dimension.

These relationships are also called algebraic error. Equations 4.45 and 4.46 are nonlinear and hence the solution is computed iteratively starting from the initial guess of the parameters $\hat{\mathbf{p}}^{(0)}$ and the approximate observations $\hat{\mathbf{l}}^{(0)}$. The corrections to the parameters $\Delta\hat{\mathbf{p}}^{(\nu)}$ and the fitted observations $\hat{\mathbf{l}}^{(\nu)} = \hat{\mathbf{l}}^{(0)} + \hat{\mathbf{v}}$ in the ν-th iteration step can be derived using the technique of Lagrange multipliers $\mathbf{u}$ resulting in (McGlone, 2004)

$$\begin{bmatrix} \mathbf{A}^T(\mathbf{B}^T\boldsymbol{\Sigma}_{\mathbf{ll}}\mathbf{B})^{-1}\mathbf{A} & \mathbf{H} \\ \mathbf{H}^T & \mathbf{0} \end{bmatrix} \begin{bmatrix} \Delta\hat{\mathbf{p}} \\ \mathbf{u} \end{bmatrix} = \begin{bmatrix} \mathbf{A}^T(\mathbf{B}^T\boldsymbol{\Sigma}_{\mathbf{ll}}\mathbf{B})^{-1}\mathbf{r}_g \\ \mathbf{r}_h \end{bmatrix} \tag{4.47}$$

and

$$\hat{\mathbf{v}} = \boldsymbol{\Sigma}_{\mathbf{ll}}\mathbf{B}(\mathbf{B}^T\boldsymbol{\Sigma}_{\mathbf{ll}}\mathbf{B})^{-1}(\mathbf{c}_g - \mathbf{A}\Delta\hat{\mathbf{p}}) \tag{4.48}$$

with the Jacobians

$$\mathbf{A} = \frac{\partial \mathbf{g}(\mathbf{p},\mathbf{l})}{\partial \mathbf{p}} \qquad \mathbf{B} = \left(\frac{\partial \mathbf{g}(\mathbf{p},\mathbf{l})}{\partial \mathbf{l}}\right)^T \qquad \mathbf{H} = \left(\frac{\partial \mathbf{h}(\mathbf{p})}{\partial \mathbf{p}}\right)^T \tag{4.49}$$

and the residual vectors

$$\mathbf{r}_g = -\mathbf{g}(\hat{\mathbf{p}}^{(0)}, \hat{\mathbf{l}}^{(0)}) - \mathbf{B}^T(\mathbf{l} - \hat{\mathbf{l}}^{(0)}) \qquad \mathbf{r}_h = -\mathbf{h}(\hat{\mathbf{p}}^{(0)}) \tag{4.50}$$

The Jacobians $\mathbf{A}$ and $\mathbf{B}$ are computed using the analytic expressions given in appendix C.3 and the Jacobian $\mathbf{H}$

$$\frac{\partial \mathbf{h}(\mathbf{p})}{\partial \mathbf{p}} = 2\begin{pmatrix} e_x & e_y & e_z & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & q_w & q_x & q_y & q_z \end{pmatrix} \tag{4.51}$$

with the epipole $\boldsymbol{e} = (e_x, e_y, e_z)^T$ and the quaternion $\boldsymbol{q} = (q_w, q_x, q_y, q_z)^T$. When the observations are samples from a normal distribution with covariance matrix $\boldsymbol{\Sigma}_{\mathbf{ll}}$, the resulting estimate is the maximum likelihood estimate.

Robust Estimation of Essential Matrix

All of the above methods are least squares estimators. Least squares estimators however have a breakdown point of 1 (cf. appendix C.1). Outliers frequently occur in the correspondence generation process using real images, and thus a robust estimation technique must be used. The RANSAC algorithm (cf. appendix C.1) can be used in this case. However, with a significant portion of outliers the RANSAC algorithm becomes very slow, because lots of samples need to be investigated. The slowness of the RANSAC algorithm results from the fact that samples are investigated depth first, i.e. samples are drawn subsequently, and each sample is scored against all measurements before the next sample is drawn. Traditionally the score in the RANSAC algorithm is the inlier count, even though robust functions like for example the log likelihood is also used (Torr and Zisserman, 1996) (See appendix C.1 for a detailed review of random sampling algorithms and score functions.). Here the log likelihood is chosen as score function. Lately Nistér (2003b) suggested the preemptive RANSAC algorithm which uses a breadth first search for the solution. In this

algorithm a sample consists of 6 points out of which 5 points are used for the computation of the solutions using the 5 point algorithm from Nistér (2003a). The 6th point is used for solution verification. This can be seen as a very simple form of preemption.

A fixed number of d samples are randomly generated, and the $T_{c,d}$ test is used to reject unlikely solutions. The $T_{c,d}$ test rejects solutions when less than c out of d data points are inliers. The investigations of Nistér (2003b) reveal values of $d = 8$ and $c = 1$ to be optimal with respect to computation time for the relative pose problem.

Afterwards all remaining poses are subsequently scored against the next correspondence. When the number of scored correspondences is a multiple of $M = 100$ correspondences, the remaining solutions are sorted according to their score and the worse half of the solutions is rejected. When all correspondences have been used for scoring, the best remaining hypothesis is returned.

It is important to randomly select the order in which point correspondences are scored. Otherwise, when samples were for example spatially ordered, it would become increasingly possible that a solution is rejected because of the occurrence of an independently moving object in the image.

A slightly adopted scheme of the preemptive RANSAC is used in this thesis:

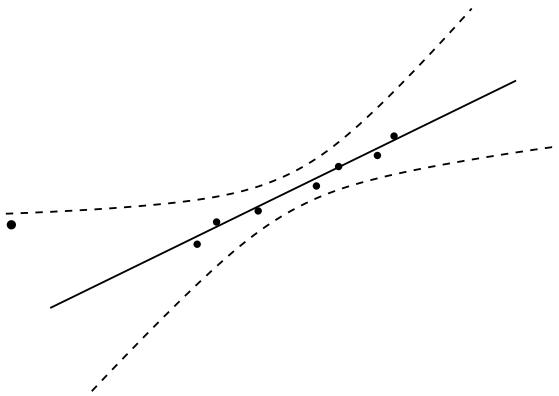


Figure 4.7: Illustration of the leverage problem (Beder, 2007): Point line incidence is used as an example. The line (solid) and its confidence region (dashed) is shown. Even though the bigger point at the left is located inside the confidence region, it still acts as a leverage point which would change the line significantly when it would be included in least squares estimation of the line from the points. This problem can be avoided by ensuring that points which are tested for incidence lie inside or close to the convex hull of the point from which the line was computed originally.

- Only 5 points are randomly chosen for each sample, and hence only the $T_{c,d}$ test - and not a 6th sample - is used for initial hypothesis rejection.
- For sample generation five points are chosen randomly such that the convex hull of the 2D-points in the image is large. The solution used to score all other correspondences has thus a valid region covering the complete image, and the leverage problem is avoided. Figure 4.7 illustrates the leverage problem.

- The solutions are evaluated block-wise. The maximum number of evaluated solutions is split into blocks with a minimum number of solutions. The first block is evaluated as in the original preemptive RANSAC algorithm. When a certain score is achieved by the best solution, further evaluation is aborted, and the best solution is returned. When the minimum score is not met, the next block of solutions is evaluated until a maximal number of overall evaluated solutions is reached.

- The car inertial sensors and their uncertainties are used to compute an initial guess of the camera egomotion. This egomotion and its uncertainties are used to reject solutions even before the $T_{c,d}$ test is used. For this purpose the Mahalanobis distance between the solution from the car inertial sensors and the solutions from the 5 samples is computed. When the Mahalanobis distance exceeds a certain threshold, the solution is rejected.

Model Selection

The essential matrix is only defined when the camera centres are distinct. When the camera centres coincide, the estimation of the essential matrix becomes unstable and should no longer be used. In this case it is necessary to use the simpler model of pure camera rotation.

With additional sensor measurements of the camera motion, the case of pure camera rotation can easily be detected and dealt with. When no auxiliary information about the camera motion is present, the pure rotational case must be detected from the data and treated separately. The detection of the pure rotational case in the absence of auxiliary information is conducted by *model selection*.

Model selection aims at identifying the model which best fits the data while at the same time avoiding overfitting. When data is fitted using a complicated model even if it could also be fitted by a simpler model without loss of accuracy, the data is *overfitted*. A lot of different algorithms for model selection have been given in the literature. Examples are the Akaike Information Criterion (AIC), the Bayesian Information Criterion (BIC), the Minimum Description Length (MDL) and the Geometric Robust Information Criterion (GRIC).

Comparison of Essential Matrix Estimation Algorithms

The algorithms described in the previous section are compared regarding accuracy and speed on a synthetic image sequence (fig. 4.8). The preemptive RANSAC algorithm is applied to compute an initial solution which is used to remove outlier from the correspondences using the incidence test (see appendix D.4). Afterwards, the remaining correspondences are used for the computation of a refined solution. Each algorithm is started with exactly the same initial guess on exactly the same set of correspondences.

Error measures: When comparing two essential matrices $\mathbf{E} = [\boldsymbol{e}]_\times \mathbf{R}$ and $\mathbf{E}' = [\boldsymbol{e}']_\times \mathbf{R}'$, e.g. an estimated essential matrix and the ground truth essential matrix, two error measures are specified in this thesis. The first error measure is the direction difference of the

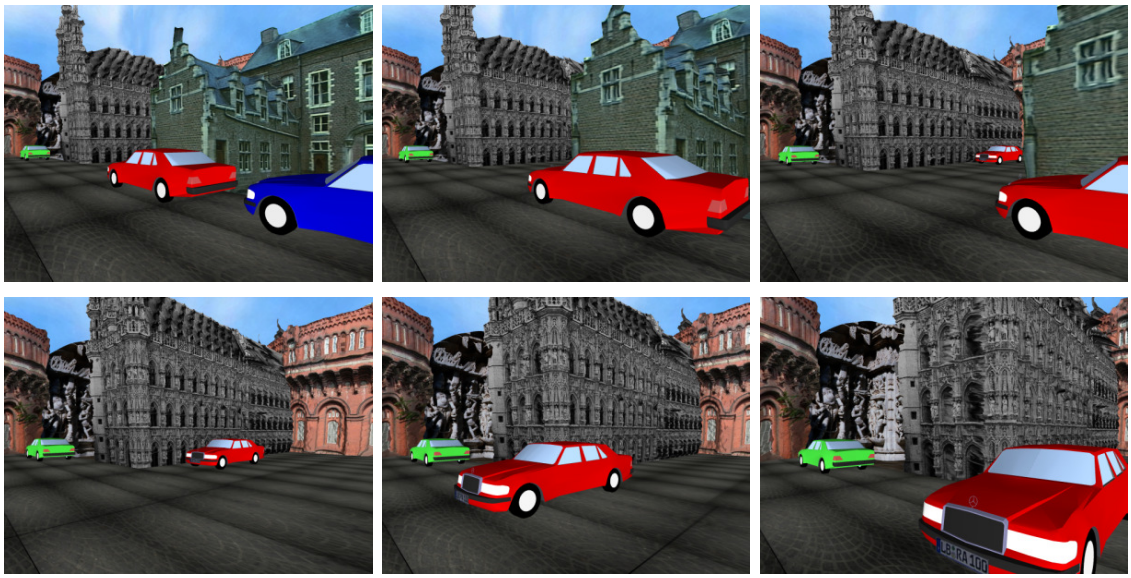


Figure 4.8: Some selected frames from the synthetic image sequence.

two epipoles $\Delta \boldsymbol{e}$

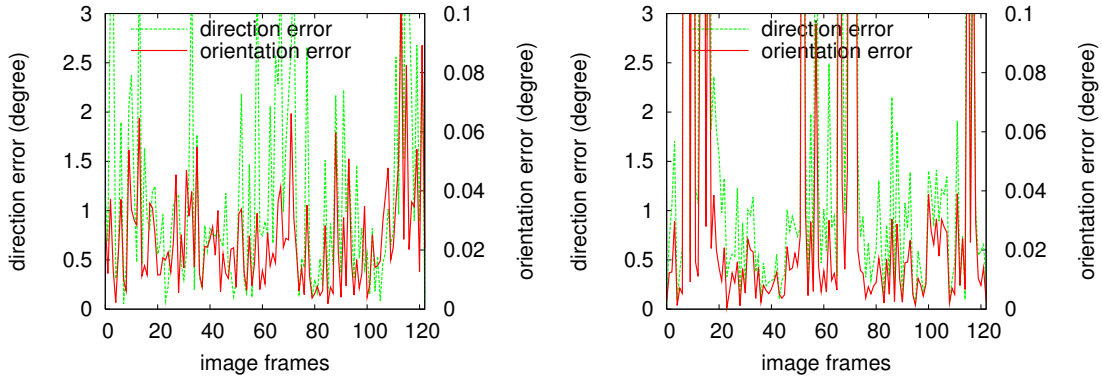
$$\Delta \boldsymbol{e} = \operatorname{acos}\left(\left(\frac{\boldsymbol{e}}{|\boldsymbol{e}|}\right)^T \frac{\boldsymbol{e}'}{|\boldsymbol{e}'|}\right) \tag{4.52}$$

where the epipoles $\boldsymbol{e}$ and $\boldsymbol{e}'$ are given as 3D-vectors. The second error measure $\Delta\mathbf{R}$ relates to the difference in the relative orientations given by the rotation matrices $\mathbf{R}$ and $\mathbf{R}'$.

$$\Delta\mathbf{R} = \| \operatorname{acos}\left(\frac{1}{2}\operatorname{trace}(\mathbf{R}^T\mathbf{R}') - \frac{1}{2}\right) \| \tag{4.53}$$

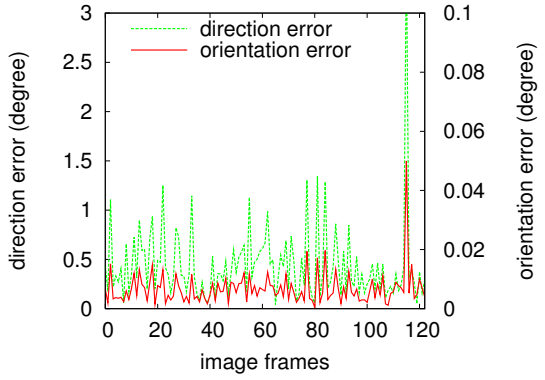
The rotation matrix $\mathbf{R}^T\mathbf{R}'$ describes the difference between the two relative orientation changes $\mathbf{R}$ and $\mathbf{R}'$. This difference in rotation (given by $\mathbf{R}^T\mathbf{R}'$) can be expressed as rotation axis and rotation angle. The rotation angle α is related to the matrix by $\operatorname{trace}\left(\mathbf{R}^T\mathbf{R}'\right) = 1 + 2\cos\alpha$ (McGlone, 2004), and hence the absolute rotation angle $\|\alpha\|$ is specified by equation 4.53.

Accuracy: The different estimation methods are compared using a sequence of synthetic generated images. The camera moves in a direction which is approximately 40 degrees rotated around a vertical axis with respect to the optical axis. Figure 4.8 shows some selected frames from the synthetic sequence. The scene consists of 124 frames and resembles an inner city intersection situation. Some cars are parked on the right side at the kerbside and a red car comes down an intersecting road from the right. The image point correspondences are computed using the KLT algorithm (section 4.1.1). The preemptive RANSAC algorithm (section 4.2.2) is applied to compute an initial solution which is used as an initial guess and for the elimination of outliers from the correspondences. The initial guess of the

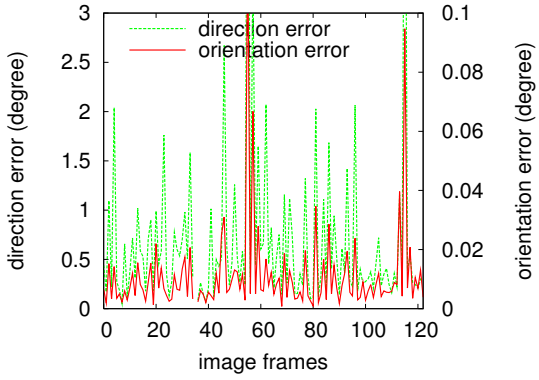


(a) Linear estimator

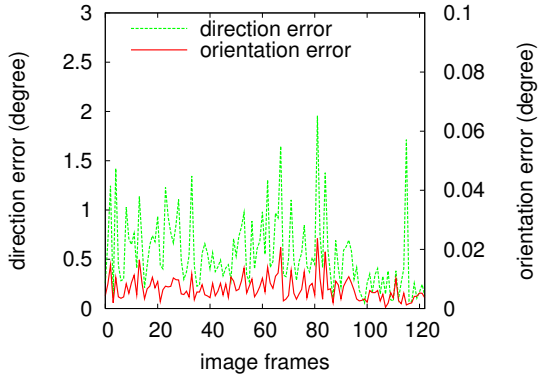
(b) 5 point



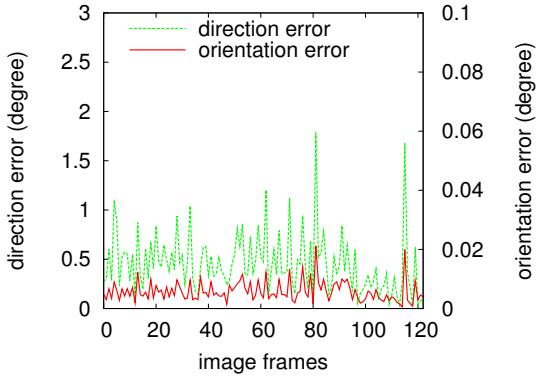
(c) Geometric error (LM, Huber cost function)



(d) Angles (LM)



(e) Algebraic Error (Gauss-Helmert)



(f) Mahalanobis distance (LM)

Figure 4.9: Comparison of essential matrix estimation algorithms.

essential matrix and the remaining inliers are fed into the different estimation algorithms. The error is computed with respect to known ground truth data. The two errors (eqs. 4.52 and 4.53) are plotted over the sequence for each estimator in figure 4.9. The two direct methods, i.e. the 5 point algorithms and the linear algorithm, do not perform as reliable as the other, nonlinear algorithms. The angle estimation algorithm does also not seem to be very promising. The remaining 3 nonlinear algorithms give similar results with minor differences.

Estimation Method	$t[ms]$
Linear	16
5 point	19
Geometric error (LM, robust cost function)	10
Angles (LM)	55
Algebraic error (Gauss-Helmert)	137
Mahalanobis distance (LM)	118

Table 4.1: Average computation time t of the different estimation algorithms using 300 synthetically generated point correspondences. The time measurements were conducted on a Pentium 4 with 3.2 GHz.

Computational Requirements: To investigate the computational requirements, the different algorithms are run using 300 synthetically generated point correspondences with normally distributed noise in both images. The computational requirements of the different algorithms are given in table 4.1. The algorithms using the covariance matrices (i.e. Algebraic error and Mahalanobis distance) are an order of magnitude slower than the other algorithms. The fastest algorithm is the nonlinear LM estimation using the geometric error. It also gives consistent results over the sequence and thus is chosen for the final algorithm.

4.2.3 Influence of Camera Calibration Errors

The camera calibration data is not exactly known because it stems from an estimation process. The influences of errors in the camera calibration data are investigated in this section. First the Jacobians for the error propagation are derived theoretically, afterwards the influence of calibration errors is investigated empirically.

Propagation of Calibration Errors

The projection rays in the camera coordinate system are used for the image-based estimation of the essential matrix. The projection of a world point into the image is described in chapter 2. The inverse function is of interest in this section, i.e. the estimation of the viewing rays in the camera coordinate system to every observed image point in the pixel

coordinate system. Let the undistortion function be denoted by $\boldsymbol{u}(\boldsymbol{x}_{\text{PixelCoo}})$. The undistortion function describing the computation of the projection ray in camera coordinates $\boldsymbol{x}_{\text{CamCoo}}$ from the pixel coordinates $\boldsymbol{x}_{\text{PixelCoo}}$ can be split in two parts.

$$\boldsymbol{x}_{\text{CamCoo}} = \boldsymbol{u}(\kappa_1, \kappa_2, f, a, \boldsymbol{c}, \boldsymbol{x}_{\text{PixelCoo}}) = \boldsymbol{L}^{-1}(\kappa_1, \kappa_2, \boldsymbol{k}^{-1}(f, a, \boldsymbol{c}, \boldsymbol{x}_{\text{PixelCoo}})) \tag{4.54}$$

The inner function, $\boldsymbol{k}^{-1} = \mathbf{K}^{-1}\boldsymbol{x}_{\text{PixelCoo}}$, is the multiplication of the point in pixel coordinates with the inverse calibration matrix $\mathbf{K}^{-1}$. This is a linear operation and accounts for the camera model. The calibration matrix $\mathbf{K}$ depends on the focal length f, the aspect ration a, the principal point $\boldsymbol{c} = (c_x, c_y)^T$, and the skew s. It is multiplied with the point coordinates $\boldsymbol{x}_{\text{PixelCoo}} = (x_x, x_y, x_w)^T$. Typically, the skew is zero in modern cameras and is hence neglected in the following derivation.

The outer function, the inverse distortion function $\boldsymbol{L}^{-1}$, models the nonlinearities in the lens (see section 2.2.4). The distortion function $\boldsymbol{L}$ (equation 2.16) depends on the coefficients of the polynomial describing the radial distortion κ_1 and κ_2 and on $\boldsymbol{x}_{\text{CamCooIdeal}} = (x_{xi}, x_{yi}, 1)^T$. Because $\boldsymbol{L}$ cannot be inverted analytically, it is approximated by

$$\hat{\boldsymbol{L}}^{-1}(\kappa_1, \kappa_2, x_{xi}, x_{yi}) \approx \frac{1}{1 + \kappa_1 r^2 + \kappa_2 r^4} \begin{pmatrix} x_{xi} \\ x_{yi} \\ 1 \end{pmatrix} \qquad \text{with} \qquad r^2 = x_{xi}^2 + x_{yi}^2 \tag{4.55}$$

To study the undistortion function, its derivatives with respect to the distortion parameters κ_1, κ_2 and with respect to the projection parameters f, a and $\boldsymbol{c}$ are investigated. The derivatives are the entries of the Jacobian of $\boldsymbol{u}$ which in turn can be used for linear error propagation. Using the chain role for vector valued functions results in the Jacobian

$$\mathbf{J}(\boldsymbol{u}) = \frac{\partial \boldsymbol{u}}{\partial \kappa_1, \kappa_2, f, a, c_x, c_y, x_x, x_y} = \Big[\frac{\partial \hat{\boldsymbol{L}}^{-1}}{\partial \kappa_1, \kappa_2} \Big| \frac{\partial \hat{\boldsymbol{L}}^{-1}}{\partial \boldsymbol{k}^{-1}} \frac{\partial \boldsymbol{k}^{-1}}{\partial f, a, c_x, c_y, x_x, x_y}\Big] \tag{4.56}$$

with

$$\frac{\partial \hat{\boldsymbol{L}}^{-1}}{\partial \kappa_1, \kappa_2} = \Big(\frac{1}{1 + \kappa_1 r^2 + \kappa_2 r^4}\Big)^2 \begin{pmatrix} -x_{xi} r^2 & -x_{xi} r^4 \\ -x_{yi} r^2 & -x_{yi} r^4 \\ -r^2 & -r^4 \end{pmatrix} \tag{4.57}$$

,

$$\frac{\partial \hat{\boldsymbol{L}}^{-1}}{\partial \boldsymbol{k}^{-1}} = \Big(\frac{1}{1 + \kappa_1 r^2 + \kappa_2 r^4}\Big)^2 \begin{pmatrix} \frac{1}{1+\kappa_1 r^2 + \kappa_2 r^4} - x_{xi}^2 \upsilon & -x_{xi} x_{yi} \upsilon \\ -x_{xi} x_{yi} \upsilon & \frac{1}{1+\kappa_1 r^2 + \kappa_2 r^4} - x_{yi}^2 \upsilon \\ -x_{xi} \upsilon & -x_{yi} \upsilon \end{pmatrix} \tag{4.58}$$

with $\upsilon = (2\kappa_1 + 4\kappa_2 r^2)$ and

$$\frac{\partial \boldsymbol{k}^{-1}}{\partial f, a, c_x, c_y, x_x, x_y} = \begin{pmatrix} \frac{c_x x_w - x_x}{f^2} & 0 & -\frac{1}{f} & 0 & \frac{1}{f} & 0 \\ \frac{c_y x_w - x_y}{a f^2} & \frac{c_y - x_y}{a^2 f} & 0 & -\frac{1}{af} & 0 & \frac{1}{af} \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \tag{4.59}$$

When the uncertainty of the calibration data is known, linear error propagation can be used to compute the uncertainty of the point position in camera coordinates. Because

the uncertainty in the camera calibration data is independent of the point correspondence uncertainty, the joint covariance matrix $\boldsymbol{\Sigma}_{cxcx}$ of the point and the camera calibration data takes on a block diagonal form

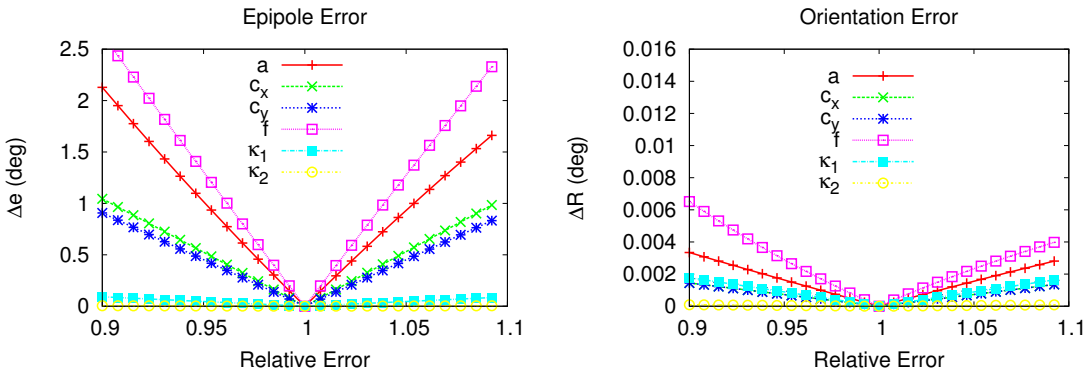
$$\boldsymbol{\Sigma}_{cxcx} = \begin{pmatrix} \boldsymbol{\Sigma}_{cc} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Sigma}_{xx} \end{pmatrix} \tag{4.60}$$

The covariance matrix of a projection ray in the camera coordinate system $\boldsymbol{\Sigma}_{cc}$ is given by

$$\boldsymbol{\Sigma}_{cc} \approx \mathbf{J}(\boldsymbol{u})\boldsymbol{\Sigma}_{cxcx}\mathbf{J}(\boldsymbol{u})^T \tag{4.61}$$

The uncertainties of the calibration parameters are however unknown and are hence not used. The sensitivity of the essential matrix estimation algorithms to errors in the calibration parameters are investigated in the next section.

Empirical Investigation

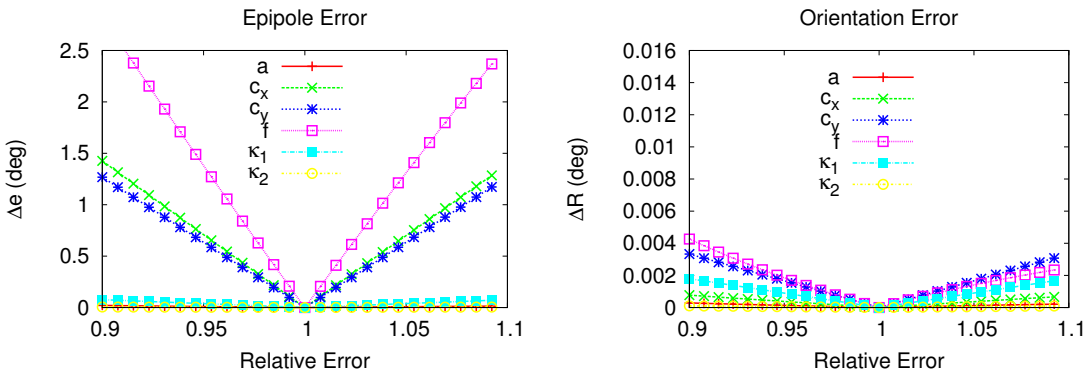


(a) Epipole error (b) Orientation error

Figure 4.10: Sensitivity of essential matrix estimation to calibration errors using a general motion with relative translation vector $(0.1, -0.05, 0.08)$ and relative orientation around the axes of the coordinate system about $(1.7°,\ 0.6°,\ -1.7°)$. The influence of calibration errors to (a) the accuracy of the estimated direction of the epipole $\Delta\boldsymbol{e}$ and (b) the accuracy of the estimated relative rotation $\Delta\mathbf{R}$ is plotted.

In this section errors resulting from inaccurately known camera calibration data are investigated empirically using synthetic data. 100 3D-points are generated randomly in the common viewing frustum of two cameras. The distance of the points from the cameras is chosen between 5 and 30 units. The camera calibration parameters of the virtual cameras are chosen identical to the real camera used in the car, i.e. image size 640×480 pixel, focal length $f = 837.5$ pixel, aspect ration $a = 1.0$, principal point $\boldsymbol{c} = (319.53, 244.84)^T$ and distortion parameters $\kappa_1 = -0.0889658$, $\kappa_2 = 0.0194259$, $\kappa_4 = 0.0015841$ and $\kappa_5 = 0.0002699$. The 3D-points are projected into the images. Afterwards, the 3D-viewing rays are computed from 2D-point positions using modified calibration parameters. Modification of

$\pm 10\%$ of the original value of the calibration parameters are investigated. These viewing rays are fed to the essential matrix estimation algorithm and the results are compared to the known ground truth data. The comparison uses the same two error measures from equations 4.52 and 4.53. The different essential matrix estimation algorithms did not show qualitativly different behaviour, and hence only the results from the non-linear estimator using the geometric errors are shown exemplarily. Four different camera motions are investigated. The sensivity of the essential matrix estimation to errors in the calibration parameters for a general motion with relative translation vector $(0.1, -0.05, 0.08)$ and relative orientation $(1.7°, 0.6°, -1.7°)$ in Euler angles are shown in figure 4.10. The tangential distortion parameters κ_4 and κ_5 are very small and therefore had no influence on the estimation. For the sake of clarity, the sensivity of essential matrix estimation to κ_4 and κ_5 is not shown in the following figures. Focal length, aspect ratio and principal point have the biggest influence on the estimation accuracy. The influence of the radial lens distortion parameters can safely be neglected. Only κ_1 has a minor influence on the orientation accuracy.



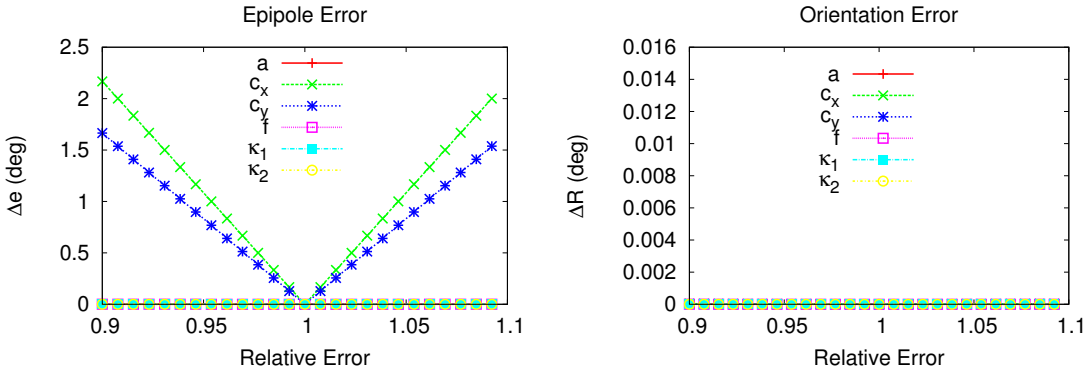
(a) Epipole error (b) Orientation error

Figure 4.11: Sensitivity of essential matrix estimation to calibration errors using a pure translation in the x-z plane. The influence of calibration errors to (a) the accuracy of the estimated direction of the epipole $\Delta \boldsymbol{e}$ and (b) the accuracy of the estimated relative rotation $\Delta \mathbf{R}$ is plotted.

The second motion is a pure translation in the x-z plane with relative translation vector $(-0.129, 0, 0.153)^T$ and zero relative rotation. This motion has been chosen because it resembles the scenario where a car travels straight ahead and the camera attached to the car is rotated around 40° to one side. The results are shown in figure 4.11. Again focal length, aspect ratio and principal point have the biggest influence on the estimation accuracy.

The case where the camera translation is parallel to the optical axes of the cameras is shown in figure 4.12. As one would expect, focal length and aspect ratio have no significant influence on the accuracy of the estimation process. Only the principal point influences the estimation accuracy.

With lateral translation (figure 4.13), mainly the horizontal position of the principal



(a) Epipole error (b) Orientation error

Figure 4.12: Sensitivity of essential matrix estimation to calibration errors for a forward translation about $(0, 0, 0.1)^T$ and zero rotation. The influence of calibration errors to (a) the accuracy of the estimated direction of the epipole $\Delta \boldsymbol{e}$ and (b) the accuracy of the estimated relative rotation $\Delta \mathbf{R}$ is plotted.

point and the focal length are relevant to the estimation accuracy of the essential matrix.

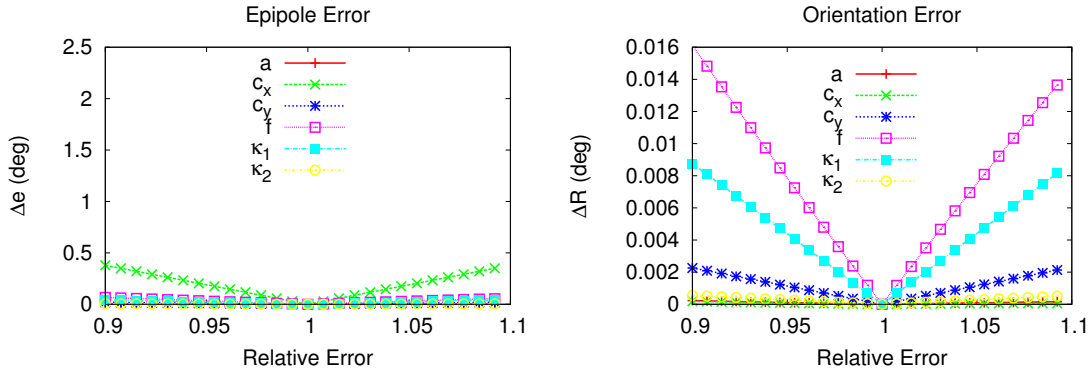
4.3 Summary

The computation of the two mandatory inputs, i.e. image point correspondences and egomotion of the camera, to the detection system has been investigated in this section.

The detection of independent motion is based on image point correspondences and hence a fast and reliable algorithm for the computation of such correspondences has been chosen. It works by first identifying promising regions for correspondence estimation using a corner detector and secondly estimating the correspondences by gradient-based minimisation of the image intensity differences. Even though a specific algorithm for correspondence estimation has been chosen, the detection algorithm is independent of this choice and can operate on correspondence data from any algorithm.

The relative motion between camera and static background, the egomotion, can be computed using either vehicle inertial sensor data or image information. Three different methods using different car inertial sensors were investigated. None of them provided full egomotion information, because for example roll and pitch sensors are missing in the car. The egomotion computation from sensor data is, however, very fast and can be used as prior knowledge for the image-based egomotion estimation. This prior information firstly speeds up the image-based estimation process and secondly makes the estimation process more reliable.

Six different estimation algorithms for egomotion computation from image point correspondences have been compared in this thesis. They are based on different error measures, and the best algorithm for the given purpose was identified. The best algorithm qualifies by speed and robustness against outliers in the input data. It computes egomotion non-



(a) Epipole error (b) Orientation error

Figure 4.13: Sensitivity of essential matrix estimation to calibration errors for a lateral translation about $(0.1, 0, 0)^T$ and zero rotation. The influence of calibration errors to (a) the accuracy of the estimated direction of the epipole $\Delta \boldsymbol{e}$ and (b) the accuracy of the estimated relative rotation $\Delta \mathbf{R}$ is plotted.

linearily using the Levenberg-Marquardt approach and is based on a robust cost function of the geometric error.

Each point correspondence which is located on a moving object is an outlier to the detection system, and hence the presence of a significant fraction of outliers must be anticipated. Even though the best algorithm is based on a robust cost function, it fails in the presence of many outliers, and therefore a random sampling scheme (i.e. the preemptive RANSAC) is used to identify the majority of the outliers and only feed the remaining inliers to the chosen estimation algorithm. The final image-based estimation algorithm for egomotion is fast and very robust even with a significant fraction of correlated outliers in the correspondences.

All image-based egomotion estimation algorithms make use of the internal calibration of the camera. The internal parameters (i.e. focal length, aspect ratio, principal point and radial distortion) are estimated in advance using a special calibration pattern. Because these internal calibration parameters are estimated values themselves, they may also be corrupted by noise. The influence of errors in the internal calibration parameters to the egomotion estimation has been investigated, and the focal length and the principal point have been identified as the main influential parameters for the given setup. Particular accuracy is hence suggested in the computation of these parameters.

The detection of independent motion is based on the results of the two algorithms described in this section: (i) Image point correspondences and (ii) egomotion. The next chapter suggests a novel framework for point-based detection of independent motion.

Chapter 5

Detection of Independent Motion

All independently moving objects are potentially dangerous in traffic situations and hence their detection should be aimed at. Examples include many different object classes, for example pets, pedestrians, cyclists and cars, but also small objects like the child's toy ball rolling onto the street from behind a parked car. Obviously, the ball itself is not dangerous but it should be detected nonetheless to alert the driver to the possibility of the child running inattentively behind the ball. The warning system should be able to detect any moving object regardless of the class to which it belongs, and thus the ideal warning system should not include any assumptions about the object class.

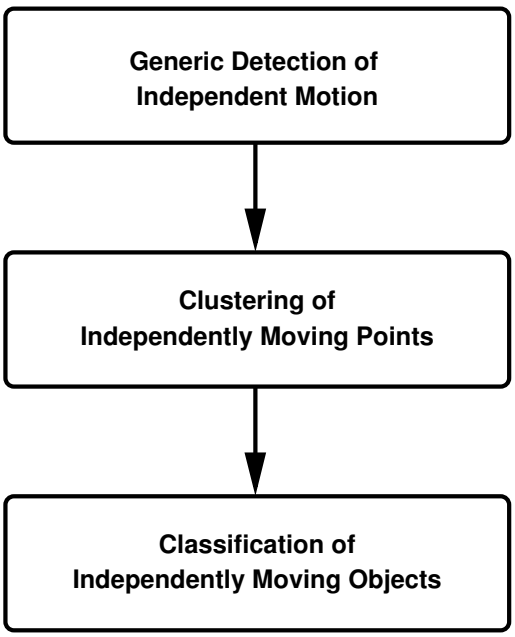


Figure 5.1: Generic warning system operating in three stages. See text for details.

Furthermore, an optimal warning system should be able to operate in real-time, regardless of the environment in which the vehicle is moving, whether on a flat surface or on a bumpy road, going up- or downhill, going straight or turning, with static obstacles or with clear view of the surrounding area. The design of the warning system should therefore include no assumption about the environment.

A generic warning system could consist of three stages, the first stage detecting independent motion in arbitrary environment without any model assumption. The second stage clusters points belonging to the same object, and finally the third stage classifies the objects and their motions into harmless, dangerous and unknown (see figure 5.1).

The first stage of such a system is described in sections 5.1 and 5.2. Different model-less, point-based detection methods for independent motion are evaluated in section 5.1. Afterwards a Bayesian framework for the pro-

posed point-based detection method is formulated in section 5.2 including a novel approach boosting the point-based estimation results and times. A simple approach for the clustering stage is suggested in section 5.3. Object classification is a highly complex area and is thus excluded in this thesis. Investigations about the trajectories of independently moving objects are presented in section 5.4.

5.1 Comparison of Point-Based Detection Methods

Given knowledge about the egomotion of the camera, the discriminative power of different point-based criteria for detection of independent motion is investigated. For the investigation, each criterion is regarded as a classifier[1] discriminating between points belonging to a certain motion model and points which do not belong to such a motion model. Each point is classified separately based solely on its correspondence(s) to the preceding frame(s). First, the notion of classifier and the detection rates are introduced more formally in section 5.1.1. The different classifiers (i.e. the detection criteria) are explained in section 5.1.2, and a detailed comparison based on ROC curves (Receiver Operating Characteristics) is conducted in section 5.1.3.

5.1.1 Classifier

The traditional task of a classifier $h(x)$ is the discrimination between two classes A and B. This usually incorporates the calculation of some value $f(x)$ from the data x and the adjacent classification into one class by the use of a threshold θ and a parity $p \in \{1, -1\}$:

$$h_{p,\theta}(x) = \begin{cases} \text{class } A & \text{if } f(x) \cdot p < \theta \cdot p \\ \text{class } B & \text{otherwise} \end{cases} \tag{5.1}$$

Often, the meaning of class B is equivalent to "not belonging to class A".

Let's assume an amount of n data x_i whereof m belong to class A, while $k = n - m$ do not belong to class A. The classifier $h_{p,\theta}(x)$ only discriminates between "belonging to A" and "not belonging to A". Four major rates describe the performance of the classifier $h_{p,\theta}(x)$:

The detection rate or *true positive rate* R_{tp} of a classifier h is the ratio between the number of data $\overline{m}_h$ correctly classified by $h_{p,\theta}(x)$ into class A and the true number of data m belonging to A:

$$R_{\text{tp}} = \frac{\overline{m}_h}{m} \tag{5.2}$$

[1] Note the difference to the classifier in the third stage of the generic warning system. The classifier from the third stage classifies *objects consisting of multiple points* into harmless, dangerous and unknown. The classifier in this section classifies *a single point correspondence* into independently moving or static background.

The false positive rate R_{fp} is the ratio between the number of data $\overline{k}_h$ falsely classified by $h_{p,\theta}(x)$ into class A and the true number of data $n-m$ not belonging to A:

$$R_{\text{fp}} = \frac{\overline{k}_h}{n-m} \tag{5.3}$$

The false negative rate R_{fn} is the ratio between the number of data $\hat{m}_h$ falsely classified by $h_{p,\theta}(x)$ not into class A and the true number of data m belonging to A:

$$R_{\text{fn}} = \frac{\hat{m}_h}{m} = 1.0 - R_{\text{tp}} \tag{5.4}$$

The true negative rate R_{tn} is the ratio between the number of data $\hat{k}_h$ correctly classified by $h_{p,\theta}(x)$ not into class A and the true number of data $n-m$ not belonging to A:

$$R_{\text{tn}} = \frac{\hat{k}_h}{n-m} = 1.0 - R_{\text{fp}} \tag{5.5}$$

5.1.2 Detection Methods

In this section the different detection methods for independent motion are explained in detail. Each detection method can be regarded as a classifier and hence consists of a function f, a parity p and a threshold θ according to the definition of a classifier from the previous section (equation 5.1). In this case, the different functions $f(\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3, \boldsymbol{C}_{12}, \boldsymbol{C}_{23}, \mathbf{R}_{12}, \mathbf{R}_{23})$ compute a scalar value from the point correspondences between $\boldsymbol{x}_1$, $\boldsymbol{x}_2$ and $\boldsymbol{x}_3$ using the known relative camera poses given by the relative translations $\boldsymbol{C}_{12}$ and $\boldsymbol{C}_{23}$ and the relative orientation changes $\mathbf{R}_{12}$ and $\mathbf{R}_{23}$. Different versions of the function f are described and compared next. Note that not all versions of f need point correspondences and relative pose changes between three views.

It is assumed that the intrinsics of the camera are known and fixed. The extrinsics of the cameras and the derived values (i.e. the relative orientation and the epipole or the relative translation) are also known, even though these values result from an estimation process and hence may be corrupted by some noise. The image correspondences are also measured values and may hence be corrupted by noise. A point correspondence is tested by computing a scalar value using a version of the function f and comparing this value to an expected value from an underlying model (i.e. the model that the point belongs to a specific relative motion). The expected value is zero in all cases in this thesis. Thresholding the difference between the expected value and the outcome of f results in the classification result. Five different functions f for the computation of the scalar value are introduced next.

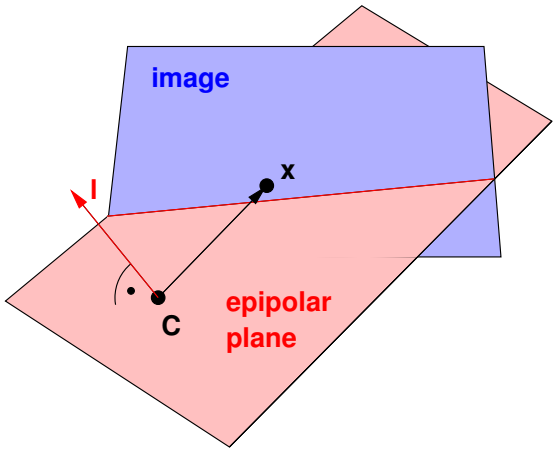


Figure 5.2: The epipolar line $\boldsymbol{l}$ (red) and the camera centre $\boldsymbol{C}$ define the epipolar plane in projective space. The scalar product between the normal vector on the epipolar plane and the 2D-point is called the *algebraic error*. The algebraic error is not directly related to the angle between the two vectors because either vector can have an arbitrary length.

Algebraic Error

The fundamental constraint holds for all point correspondences $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ between image 1 and 2

$$\hat{\boldsymbol{x}}_2^T \mathbf{F}_{12} \hat{\boldsymbol{x}}_1 = 0 \tag{5.6}$$

With known camera calibration, the essential matrix $\mathbf{E}$ can be used instead of the fundamental matrix

$$\boldsymbol{x}_2^T \mathbf{E}_{12} \boldsymbol{x}_1 = 0 \qquad \text{with} \qquad \boldsymbol{x}_i = \mathbf{K}^{-1} \hat{\boldsymbol{x}}_i \tag{5.7}$$

When the extrinsics and hence $\mathbf{E}$ or correspondences are contaminated by noise, equation 5.7 no longer vanishes, but instead results in a scalar value $d_{ia}^2 = \boldsymbol{x}_2^T \mathbf{E}_{12} \boldsymbol{x}_1 \neq 0$. The residual d_{ia} is also called the *algebraic error*. d_{ia} is, however, not the distance of the point from the epipolar line $\boldsymbol{l} = \mathbf{E}\boldsymbol{x}$ in the image. Its relation to the distance between point and epipolar line is given in equation 5.9. The geometric relations between point and epipolar line leading to the algebraic error are illustrated in figure 5.2.

Geometric Error

The *geometric error* is the distance of the point from the corresponding epipolar line in the image plane. The epipolar lines $\boldsymbol{l}_1$ and $\boldsymbol{l}_2$ for a point correspondence between $\boldsymbol{x}_1 = (x_{1x}, x_{1y}, x_{1w})^T$ and $\boldsymbol{x}_2 = (x_{2x}, x_{2y}, x_{2w})^T$ are given by $\boldsymbol{l}_1 = (l_{1x}, l_{1y}, l_{1z})^T = \mathbf{E}_{21}\boldsymbol{x}_2$ and $\boldsymbol{l}_2 = (l_{2x}, l_{2y}, l_{2z})^T = \mathbf{E}_{12}\boldsymbol{x}_1$. The distance d_{ig} of the point $\boldsymbol{x}_i$ to the epipolar line $\boldsymbol{l}_i$ in the image plane can be computed by (Hartley and Zissermann, 2004)

$$d_{ig}^2 = \frac{\boldsymbol{x}_i^T \boldsymbol{l}_i}{(l_{ix}^2 + l_{iy}^2) x_{iw}^2} \tag{5.8}$$

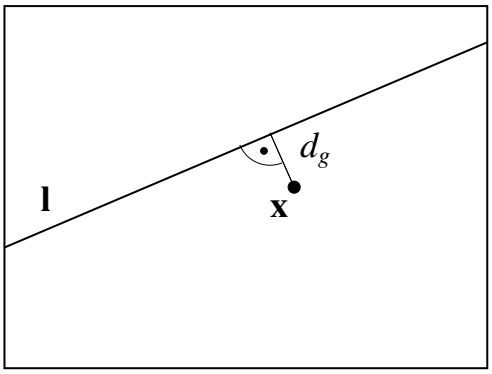


Figure 5.3: The geometric error is given by the Euclidean distance d_g between the point $\boldsymbol{x}$ and the corresponding epipolar line $\boldsymbol{l}$.

In contrast to the fundamental constraint, the distance of the point from the epipolar line is always measured in the image plane. The distance d_{ig} is also called the *geometric error*.

Assuming homogenised points (i.e. $x_{1w} = x_{2w} = 1$), the algebraic error d_{ia} is related to the geometric error d_{ig} by a scale factor λ_i

$$d_{ia} = \lambda_i d_{ig} = \sqrt{l_{ix}^2 + l_{iy}^2} d_{ig} \tag{5.9}$$

with $\lambda_i = \sqrt{l_{ix}^2 + l_{iy}^2}$. For a given essential matrix $\mathbf{E}$, the epipolar line $\boldsymbol{l}$ belonging to a point is different for every point, and hence the scale factor λ_i is also different for every point.

Correspondence direction

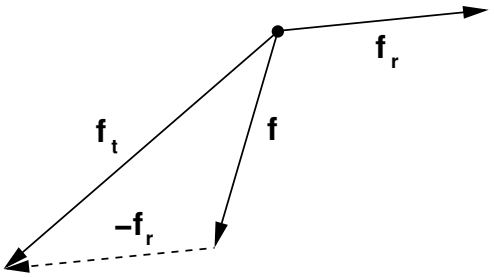


Figure 5.4: Rotation compensation (de-rotation) of a single correspondence vector $\boldsymbol{f}$. $\boldsymbol{f}_r$ is the rotational component as calculated from the known camera rotation and the position in the image, and $\boldsymbol{f}_t$ is the translational correspondence component.

A flow vector $\boldsymbol{f} \in \mathbb{R}^2$ can be computed from each image correspondence $\boldsymbol{x}_1$, $\boldsymbol{x}_2$

$$\boldsymbol{f} = \boldsymbol{x}_2 - \boldsymbol{x}_1 \tag{5.10}$$

Each optical flow vector $\boldsymbol{f}$ is composed of two parts: On part resulting from the relative translation $\boldsymbol{f}_t$ between camera and 3D-point and the other part resulting from the relative rotation $\boldsymbol{f}_r$ between camera and 3D-point. Under the assumption that the 3D-point belongs to the static background, there are two approaches for the computation of the translational part of the optical flow, when the camera motion relative to the static background is known:

1. Since the rotational component of the flow field at the image position $\boldsymbol{x}_2$ is independent of the 3D-scene structure, it can be calculated analytically. Using projective space, the point $\boldsymbol{x}_2 \in \mathbb{P}^2$ can be rotated using a homography[2] $\mathbf{H}$ resulting in the rotation corrected point $\boldsymbol{x}'_2 = \mathbf{H}\boldsymbol{x}_2 = \mathbf{R}_{12}^T\boldsymbol{x}_2$. The rotational component of the flow field can be computed using again the Euclidean representations of the point $\boldsymbol{x}_2$ and the rotation corrected point $\boldsymbol{x}'_2$

$$\boldsymbol{f}_r(\boldsymbol{x}_2) = \boldsymbol{x}_2 - \boldsymbol{x}'_2 \tag{5.11}$$

   The translational component of the flow field can then be computed from the flow measured in the images $\boldsymbol{f}$ and the rotational component $\boldsymbol{f}_r$ recovered from the known relative camera rotation

$$\boldsymbol{f}_t = \boldsymbol{f} - \boldsymbol{f}_r \tag{5.12}$$

   The process is called *de-rotation* or *rotation correction* of the correspondence (see fig. 5.4).

2. The direction of the translational part can also be computed using only the known camera translation, because pure translational flow fields exhibit simple geometric properties (see also section 2.4). Each flow vector (correspondence vector) points radially away from the focus of expansion (FOE) or radially towards the focus of contraction (FOC). The epipole $\boldsymbol{e}$ is either the FOE or the FOC, depending on the motion direction. The predicted correspondence direction vector $\boldsymbol{f}_p$ can easily be computed for each image location when the epipole is known

$$\boldsymbol{f}_p(\boldsymbol{x}) = \pm\,(\boldsymbol{x} - \boldsymbol{e}) \tag{5.13}$$

   with the sign depending on the fact if the epipole is the focus of expansion or the focus of contraction. Note that $\boldsymbol{f}_p$ only contains information about the direction of the translational flow, not about its magnitude. $\boldsymbol{f}_p$ is called the *predicted direction of the translational flow.*

Comparing the directions of the two vectors $\boldsymbol{f}_t$ and $\boldsymbol{f}_p$, for example by using the angle between the vectors, yields points where the underlying assumption (i.e. known relative motion between camera and 3D-point) is violated. In practise, a threshold t over the angle

[2]The homography $\mathbf{H}$ simply consists of the rotation matrix describing the relative orientation change between the two images, because normalised image coordinates are used to describe the point correspondence.

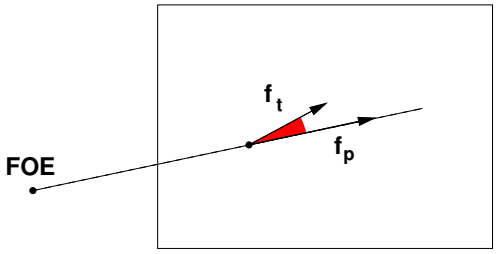


Figure 5.5: The difference in direction between predicted translational flow $\boldsymbol{f}_p$ and rotation compensated measured flow $\boldsymbol{f}_t$ can be used to detect independent motion. The predicted direction can be computed using the position of the focus of expansion (FOE), the point position and the relative rotation between the two images.

between the flow vector and the predicted direction is used as a criterion for the detection of independently moving objects. This is illustrated in figure 5.5. The angle α between the expected direction of the translational correspondence component $\boldsymbol{f}_p$ and the measured direction of the translational correspondence component $\boldsymbol{f}_t$ is given by

$$\cos(\alpha) = \frac{\boldsymbol{f}_p^T \boldsymbol{f}_t}{|\boldsymbol{f}_p||\boldsymbol{f}_t|} \tag{5.14}$$

Reprojection Error

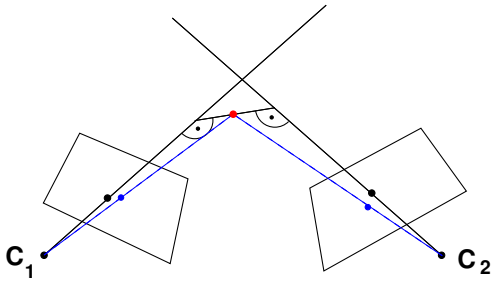


Figure 5.6: A 3D-point is seen by two cameras C_1 and C_2. Generally, the viewing rays of the corresponding 2D-points in the images (black) do not intersect in space. Optimal triangulation determines the 3D-point (red). The 3D-point is projected into the images resulting in slightly different 2D-points (blue). The sum of distances between the blue 2D-points and the original 2D-points (black) is the reprojection error.

The gold standard error measure for the computation of the fundamental matrix (Hartley and Zissermann, 2004) can also be used for detection of independent motion:

- Instantiate two projection matrices $\mathbf{P}_1 = [\mathbf{I}|\boldsymbol{0}]$ and $\mathbf{P}_2 = [\mathbf{R}_{12}^T|\boldsymbol{e}_2]$ from the parametrisation of the essential matrix

- Triangulate a 3D-point $\boldsymbol{X}$ for the point correspondence using the optimal triangulation method described in Hartley and Zissermann (2004).
- Compute sum d of the reprojection errors in the image plane

$$d = |\mathbf{P}_1(\boldsymbol{X}) - \boldsymbol{x}_1| + |\mathbf{P}_2(\boldsymbol{X}) - \boldsymbol{x}_2| \tag{5.15}$$

  with the reprojected image points $\mathbf{P}_i(\boldsymbol{X}) \in \mathbb{R}^2$ and the original image points $\boldsymbol{x}_i \in \mathbb{R}^2$.

The reprojection error d vanishes for correspondences affiliated with the motion model of the camera and hence classification can be striven for by thresholding the reprojection error d.

Trifocal Tensor

The trilinear relation for point-point-point correspondences (equation 2.39) results in a zero 3 by 3 matrix $\mathbf{D}$ for ideal point correspondences consistent with the trifocal tensor belonging to the underlying model. Two different detection methods were investigated:

1. Thresholding the biggest absolute element of the matrix $\mathbf{D}$ results in the first classifier and
2. thresholding the square Frobenius norm $||\mathbf{D}||_F^2$ of the matrix $\mathbf{D}$ results in the second classifier. The square Frobenius norm is given by the sum of the squared elements of the matrix.

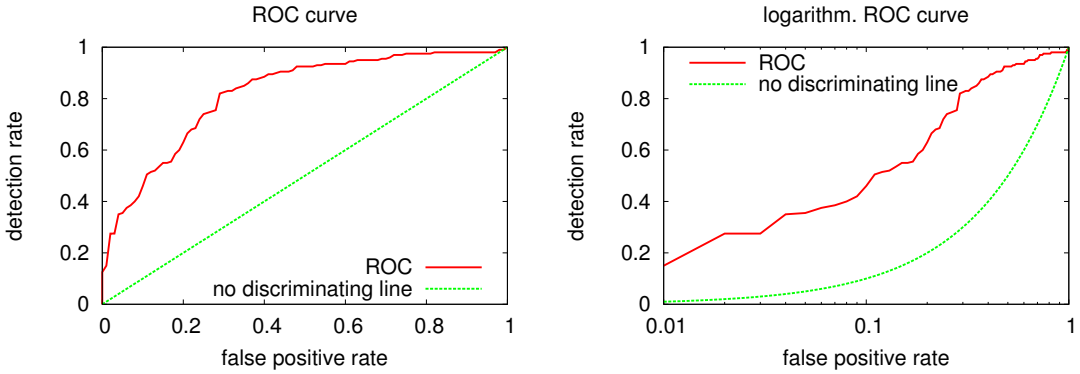
Both detection methods gave comparable results, and hence only the classifier using the maximum entry as threshold is shown for the sake of clarity.

5.1.3 Experimental Comparison of Detection Methods

As explained above, each detection method from the previous section is basis for a classifier. These different classifiers are compared next. First the criterion for the comparison, the ROC curve, is introduced and afterwards the generation of synthetic image point correspondences, which are used for the comparison, is described.

Receiver Operating Characteristics (ROC) Curves

Receiver operating characteristics (ROC) are a way of describing the performance of a classifier (Langdon, 2003). The ROC consists of a graph in which the true positive rate is plotted against the false positive rate (see fig. 5.7). This curve can be obtained by varying the threshold of the classifier θ. The points $(0, 0)$ and $(1, 1)$ always belong to the ROC curve. The point $(0, 0)$ represents the working point θ_0 where no positives are detected, and the point $(1, 1)$ represents the working point θ_a where no negatives are detected. The random guessing classifier represents a straight line, which is also called the no-discrimination line, between the points $(0, 0)$ and $(1, 1)$. Three different ways to summarise ROC curves exist:



(a) (b)

Figure 5.7: (a) Example for receiver operating characteristics (ROC) curve. The detection rate of a classifier based on the geometric error is plotted against the false positive rate. The no-discriminating line is equivalent to a purely random classifier. (b) The false positive rate is sometimes also shown on a logarithmic scale.

- The *area* between the no-discrimination line and the ROC curve or, alternatively, the area under the ROC curve can be used as summary. A value of 1.0 represents the optimal classifier, and a value of 0.5 represents the randomly guessing classifier when the area under the ROC curve is chosen.

- The *discriminability index* d-prime d' is based on the assumption of normally distributed populations of the two classes A and B with identical variance σ^2 and means $\overline{x}_A$ and $\overline{x}_B$. Each instance of a class is represented by a vector, and the distribution of all members of a certain class can be described by these multivariate Gaussian distribution. Then d' is defined as (Heeger, 1998)

$$d' = \frac{|\overline{x}_A - \overline{x}_B|}{\sigma} \tag{5.16}$$

  d' can also be calculated from the false positive rate $R_{\text{fp},0.5}$ corresponding to the detection rate 0.5. When the detection rate is 0.5, the threshold is exactly given by the mean of the positives, and the corresponding false positive rate can be expressed using of the cumulative distribution function of the normal distribution

$$F(x) = R_{\text{fp},0.5} = \frac{1}{\sigma\sqrt{2\pi}} \int_{-\infty}^{-d} e^{-\frac{\nu^2}{2\sigma^2}} d\nu \tag{5.17}$$

  with the distance d between the mean of the positives $\overline{x}_A$ and the mean of the negatives $\overline{x}_B$. Substituting ν by $\frac{\hat{\nu}}{\sigma}$ results in

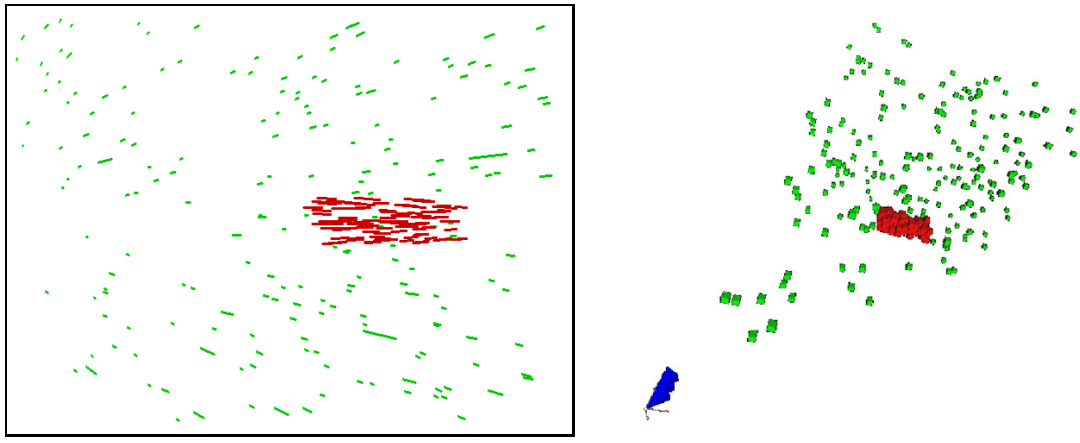
$$R_{\text{fp},0.5} = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{-d'} e^{-\frac{\hat{\nu}^2}{2}} d\hat{\nu} \tag{5.18}$$

with the unknown discriminability index d'. Equation 5.18 cannot be solved analytically and must therefore be approximated numerically.

- The *intercept* of the ROC curve with the line between the points $(0, 1)$ and $(1, 0)$ can be used as a summary. It is sufficient to consider only the false positive rate of the intercept, since the true positive rate is unambiguously determined by the false positive rate. Smaller false positive rates of the intercept indicate better classifiers.

The underlying distributions are unknown and not necessarily Gaussian, and hence the discriminability index d' cannot be used. The intercept of the ROC curve with the line between $(0, 1)$ and $(1, 0)$ does not fully capture the appearance of the ROC curve and therefore the area under the ROC curve is chosen for the following comparison.

Generation of Synthetic Correspondences



(a) (b)

Figure 5.8: (a) Synthetic point correspondences in an image of 640 × 480 pixels and (b) a 3D-view of the scene used for generation of the 2D-2D-correspondences. The correspondences from points belonging to the static background scene are drawn in green, the correspondences resulting from the moving object are drawn in red. The 3D-geometry of the scene for the first image is shown on the right (b). The cameras are indicated by blue pyramids, the moving points are drawn in red and the static points are drawn in green. The epipole is located at the left side outside the image. The camera rotates around the vertical axis. Because of the special geometric configuration of cameras and moving object, flow vectors from the background have a dominant flow direction to the right, and flow vectors on the moving object have a dominant flow direction to the left.

Synthetic image correspondences are generated from two sets of 3D-points. The first set of 3D-points $\boldsymbol{X}_i \in \mathcal{P}_s$ is assumed to be static in space, and the second set of 3D-points $\boldsymbol{X}'_i(t) \in \mathcal{P}_m(t)$ is assumed to be rigidly moving, i.e.

$$\boldsymbol{X}'_i(t) = \mathbf{T}(t)\ \boldsymbol{X}'_i(0) \tag{5.19}$$

with the Euclidean transformation $\mathbf{T}(t)$ dependent of time t. The static points in $\mathcal{P}_s$ are uniformly distributed in the intersection of the viewing frustums of the cameras. They are hence seen in all images, and near and far points are clipped such that the distances of the remaining points to the cameras are all in the range between 1 and 30 length units. The points on the moving object in the set $\mathcal{P}_m(0)$ are generated randomly according to a uniform distribution in a user-given bounding box in space. The points in $\mathcal{P}_s$ and the points in $\mathcal{P}_m(0)$ remain fixed over the sequence.

A moving camera with user-supplied, fixed intrinsics $\mathbf{P}(t)$ projects the 3D-points of the two sets into the image points $\boldsymbol{x}_i(t)$ and $\boldsymbol{x}'_i(t)$. The camera motion is also supplied by the user. The static image points are given by

$$\boldsymbol{x}_i(t) = \mathbf{P}(t)\ \boldsymbol{X}_i \tag{5.20}$$

and the projected points of the moving object are given by

$$\boldsymbol{x}'_i(t) = \mathbf{P}(t)\ \mathbf{T}(t)\ \boldsymbol{X}'_i(0) \tag{5.21}$$

The intrinsics of the camera are chosen such that they resemble the intrinsic of the camera in the demonstrator (image size 640×480 pixel, focal length 840 pixel, zero skew and principal point at $(320, 240)$) without modelling the lens distortion effects. A random variable $\boldsymbol{n}_i$ drawn from a 2D normal distribution $\mathcal{N}(\boldsymbol{n}_i|\boldsymbol{0}, \sigma_i^2\mathbb{I})$ with diagonal covariance matrix and user-supplied variance σ_i^2 is added to all 2D image point positions. The 2D random variable $\boldsymbol{n}_i$ models the noise in the 2D-point correspondences resulting from limited accuracy of the feature tracking algorithm and the influence of image intensity noise.

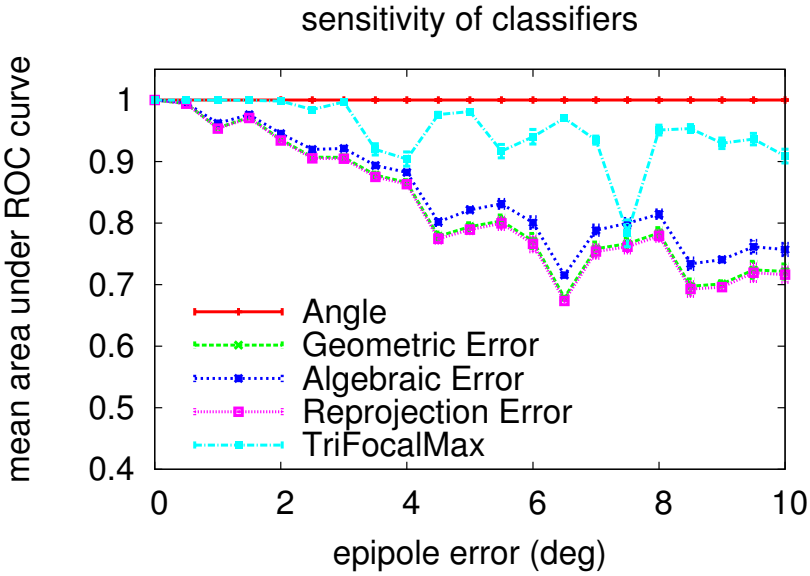
In the algorithm suggested in this thesis, the camera extrinsics are either computed using sensors or estimated using one of the algorithms in section 4.2 and are hence corrupted by noise. The influence of the limited accuracy of this estimation process can be investigated by adding the rotational error of user controlled magnitude α_r around a randomly chosen axis $\boldsymbol{a}_r$. The epipole direction can also be artificially corrupted by rotating around an axis with random direction perpendicular to the epipole direction $\boldsymbol{a}_e$. The magnitude of the epipole α_e error can again be varied. The errors are applied such that the relative quantities (i.e. the epipole and the relative orientation) can be corrupted by specific magnitudes. The final noisy point correspondences are generated as follows

$$\boldsymbol{x}_i(t) = \mathbf{P}(t, \mathbf{P}(t-1), \boldsymbol{a}_r, \alpha_r, \boldsymbol{a}_e, \alpha_e)\ \boldsymbol{X}_i + \boldsymbol{n}_i(\sigma_i^2) \tag{5.22}$$

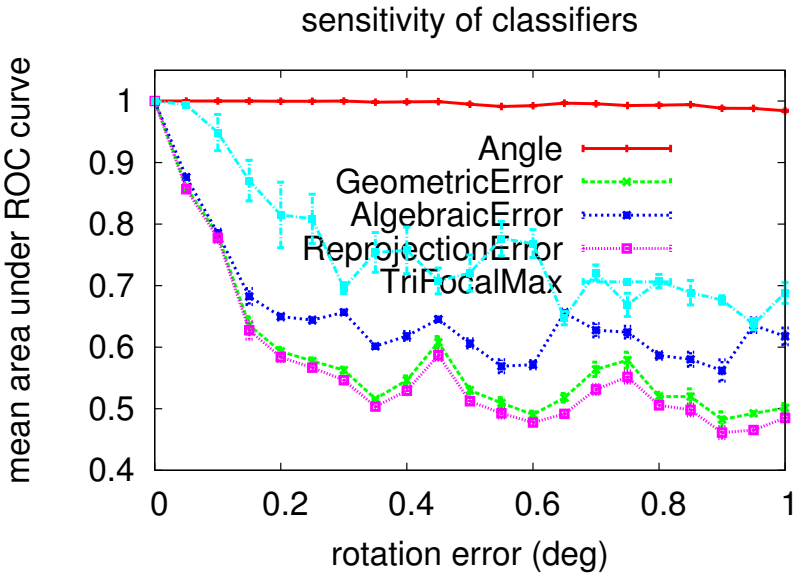
and

$$\boldsymbol{x}'_i(t) = \mathbf{P}(t, \mathbf{P}(t-1), \boldsymbol{a}_r, \alpha_r, \boldsymbol{a}_e, \alpha_e)\ \mathbf{T}(t)\ \boldsymbol{X}'_i(0) + \boldsymbol{n}_i(\sigma_i^2) \tag{5.23}$$

with the noisy projection matrices $\mathbf{P}(t, \mathbf{P}(t-1), \boldsymbol{a}_r, \alpha_r, \boldsymbol{a}_e, \alpha_e)$ and the additional noise $\boldsymbol{n}_i(\sigma_i^2)$. Figure 5.8 shows an exemplary image visualising the synthetic point correspondences and a 3D-view of the generating synthetic scene.



(a) Sensitivity of classifiers to epipole errors.



(b) Sensitivity of classifiers to orientation errors.

Figure 5.9: Sensitivity of the different classifiers to errors in the underlying model. The angular error of the (a) epipole and (b) orientation axis is given on the abscissa. The mean area under ROC curve is given on the ordinate. The mean area is computed using 30 runs with different direction of the epipole error and (b) different orientation error axis, the magnitude of the error however remained constant. The std. deviation of the area under the ROC curve is indicated using error bars. See paragraph 5.1.3 for detailed description of the underlying error model.

Comparison

A scene with synthetic correspondences where the configuration of the 3D-entities resembles a real intersection situation is used for the comparison. The car with the camera is moving straight ahead with 0.5 units per image frame. The camera is oriented around the yaw axis about approximately 29° with respect to the car and points to the right. This geometric setup corresponds to an epipole position at approximately $(-139, 240)^T$ in pixel coordinates (neglecting camera position noise). A second car coming down an intersecting street from the right is on an intersecting trajectory with the first car. The second car is initially located 15 units in lateral direction and 20 units in longitudinal direction from the first car with the camera. The second car moves with 1 length unit per frame in a direction perpendicular to the trajectory of the first car towards the trajectory. Figure 5.8 shows the configuration. There are 200 points belonging to the static scene and 100 points on the moving object. The points are easy to classify in this scenario, because the direction of the optical flow from the moving object is roughly opposite (from right to left) to the direction of the flow from the static background (from left to right).

The classifiers based on the detection functions f from the previous section (5.1.2) are compared under a variety of error conditions using the synthetic scene described above. All results are generated by averaging over 30 randomly generated scenes with the same configuration, but with different error directions $\boldsymbol{a}_r$ and $\boldsymbol{a}_e$. The mean area under the ROC curve is used as a quality measure. Because all classification algorithms depend on the correctness of the underlying motion model, the sensitivity of the classifiers to errors in this motion model is analysed by disturbing the motion model (namely the epipole direction and relative rotation) and investigating the performance of the classifiers. Figure 5.9(a) compares the sensitivity of the different classifiers to errors in epipole direction of the underlying model. Figure 5.9(b) compares the sensitivity of the different classifiers to errors in relative rotation of the underlying model. The classifiers based on the angular error show a superior performance compared to the other classifiers.

To understand this result, visualise an simple example: A purely translating camera moves forwards along its optical axis. If the camera sees a static scene, all flow vectors point away from the epipole (the epipole is a FOE in this case). If however the camera looks at a second object which moves on a parallel trajectory but faster than the camera, the relative motion between camera and moving object results in the same epipole position as from the camera motion. The relative distance between object and camera grows and the epipole of the moving object is a FOC: All flow vectors on the object point towards the epipole. This situation is visualised in figure 5.10.

As long as there is no correspondence noise and no uncertainty in the motions, the fundamental constraint based on the camera motion is precisely fulfilled even for the flow vector from the moving object, because all flow vectors stay on their corresponding epipolar lines. The classifier based on the correspondence direction is able to distinguish between the two classes, while the other classifiers are not. The classifier based on the reprojection error could separate between the two classes by checking if the triangulated point is located

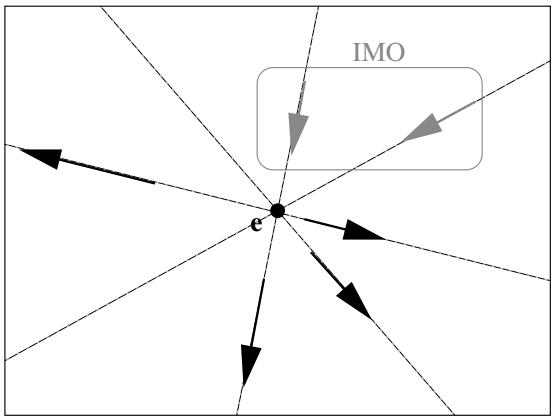


Figure 5.10: The camera motion and the relative motion between camera and independently moving object (IMO) share the same epipole $\boldsymbol{e}$. Camera and IMO are moving on parallel trajectories in the same direction and do not rotate. The IMO is faster than the camera and hence the epipole of the IMO is a focus of contraction (FOC), and the epipole of the camera is a focus of expansion (FOE).

in front of the camera. The classifiers based on trifocal tensor, geometric and algebraic error cannot discriminate between the two classes.

A second source for errors stems from the limited accuracy of the correspondence estimation. This is approximated by adding 2D normally distributed errors to the point correspondences. The performance of the classifiers is analysed under a variety of different noise levels. Figure 5.11 compares results. The classifiers based on the angular error show again superior performance compared the other classifiers.

Investigations about the sensitivity of the classifiers to variations in all three sources for errors at the same time were also conducted. The results of these investigations did, however, not reveal qualitatively new results and are hence not shown.

Computation Time: The average computation time of the proposed detection method per 300 point correspondences is given in table 5.1. The computation times were not dependent on the noise levels. The exemplary computation times used in the table were measured with zero noise.

Discussion

The angular based classifier performed superior on the chosen sequence. The discriminative capacity of the angular classifiers remained very good, even when the underlying model was corrupted by noise. One could argue, that the geometric setup is somewhat ideal for the angular classifiers because the correspondence vectors on the moving object point nearly in the opposite direction as the correspondence vectors of the static scene. This geometric

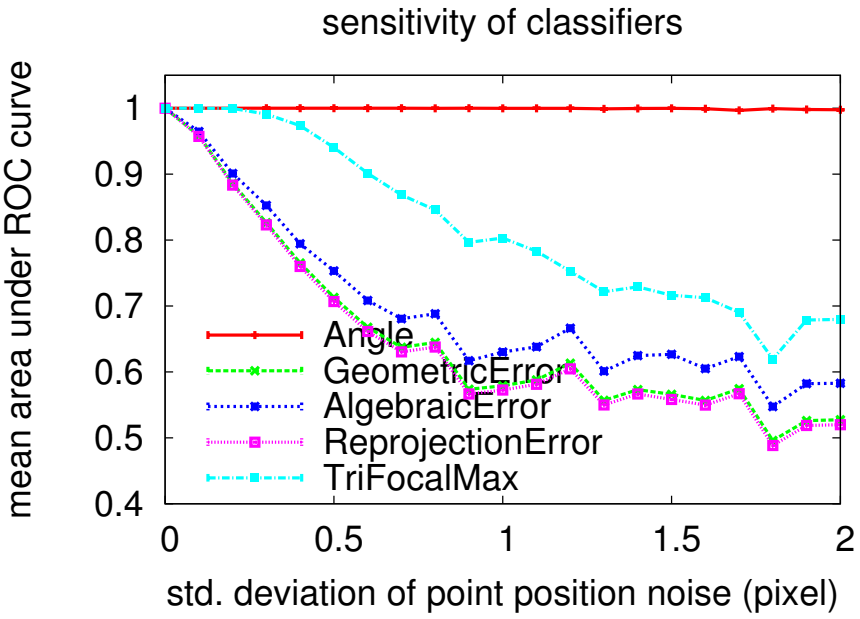


Figure 5.11: Sensitivity of the different classifiers to errors in the correspondence estimation. The std. deviation of the position error is given on the abscissa. The mean area under ROC curve is given on the ordinate. The mean area is computed using 30 runs. The std. deviation of the area under the ROC curve is indicated using error bars. See paragraph 5.1.3 for detailed description of the underlying error model.

setup is however very common in real traffic situations, and hence the chosen synthetic scene is of relevance.

A second geometric setup is also investigated in which the only difference to the previous setup is that the intersecting car has a slower velocity (i.e. with 0.25 units per frame) and hence the correspondences from this car have "flipped" their direction compared to the first scenario. The results from this setup are shown in figure 5.12. The angular classifiers no longer perform significantly better than the other classifiers. The trifocal-tensor-based classifiers are slightly superior to the other classifiers in this setup. However, the differences between the classifiers are not as pronounced as in the first setup.

To summarise, the angular classification algorithm is less sensitive to errors in the underlying model and in the correspondences in particular in the special geometric configuration when the correspondence vectors on the moving object have approximately the opposite direction of the correspondence vectors of the static scene. In general, the sensitivity of the angular based classification algorithm is comparable to or even slightly better than the other algorithms using only two images. The trifocal tensor based algorithms are generally less sensitive than the algorithms based on geometric, algebraic and reprojection error, but on the other hand they need more computation time. The angular classification

Detection Method	$t[\mu s]$	σ_t
Angle	121	3.5
Geometric Error	22.0	0.8
Algebraic Error	14.5	1.1
Reprojection Error	26648.6	379.3
TriFocalMax	825.6	23.3

Table 5.1: Computation time t of the different classification methods per 300 point correspondences. The timings are averages over 1200 classifications of 300 point correspondences. The time measurements were conducted on a Pentium 4 with 3.2 GHz. The standard deviation σ_t of the time measurements is given in the third column.

algorithm has been chosen for the final system.

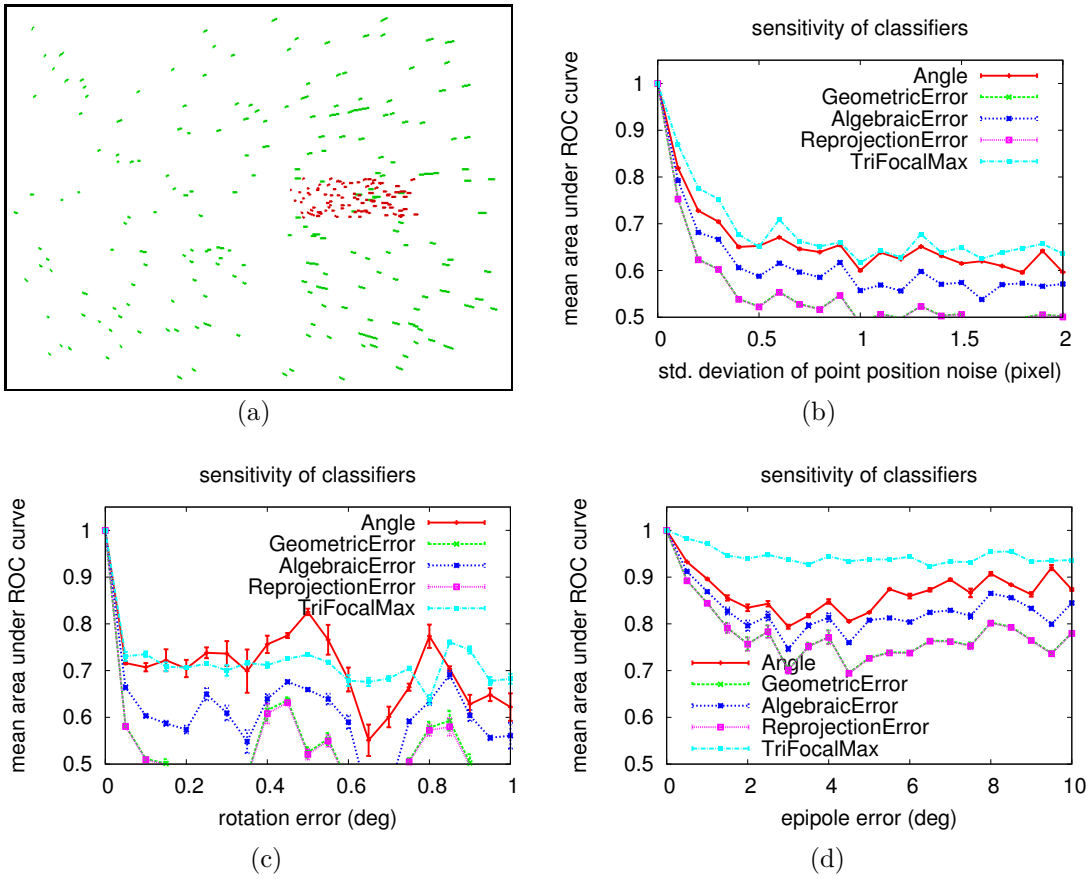


Figure 5.12: Comparison of classifiers on second scenario. (a) Exemplary correspondences for the second scenario. (b) Investigation of the sensitivity of the different classifiers to errors in the correspondence estimation and (c,d) errors in the underlying model. The std. deviation of the area under the ROC curve is indicated using error bars. For a detailed description refer to figures 5.8, 5.11, 5.9(b) and 5.9(a)

Degenerate Cases: A number of degenerate cases, where the detection of independent motion fails even theoretically without further knowledge, are identified next. Because of their slow performance, trifocal tensor based algorithms are excluded from the following considerations. Since every motion has its own FOE, an additional FOE exists for every motion between the camera and an independently moving object. Three cases exist where the detection of an independently moving object fails:

- If the FOE of the camera motion and the FOE of the relative motion between the camera and the moving object coincide, the predicted directions of the translational flow fields are the same, and hence the object is not detectable as independently moving. A scenario for this setup is a frontal collision trajectory between the camera and the moving object, another scenario would be any motion parallel to the camera with a velocity less than the camera velocity (overtaking).

- Collinearity: If the object and the two FOEs are collinear and the object is not located between the two FOEs, the predicted directions of the translational flow fields are the same, and hence the object is not detectable as independently moving. This scenario is strictly true for infinitesimal small objects. As soon as the object extends beyond the line connecting the foci of expansion, the collinearity does no longer hold for the complete object. This fact is illustrated in figure 5.13. Even though the collinearity condition holds for point B, point A on the same moving object could theoretically be detected.

- If there is no relative motion and no relative orientation change between the camera and the moving object, there is no optical flow, and the object is hence not detectable as independently moving by means of investigating the optical flow. An obvious scenario is an object moving with the same velocity parallel to the camera path. This scenario is generally of little importance because no collision danger exists.

These degenerate cases are independent of the classification algorithm.

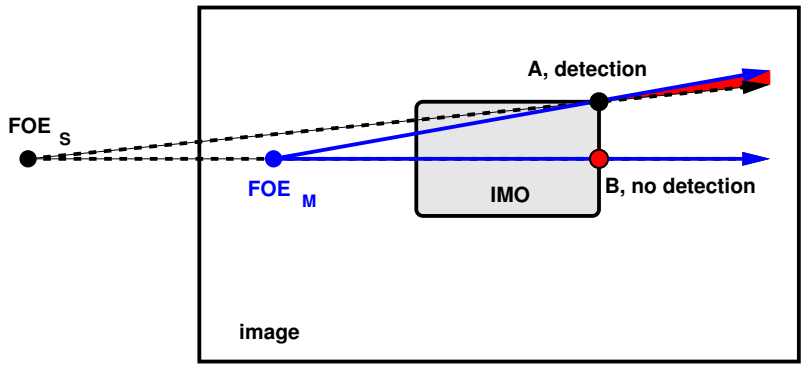


Figure 5.13: Collinearity: The point A on the independently moving object (IMO) is not collinear with the two foci of expansion (FOEs) and can be detected. The point B is collinear with the two FOEs, and hence no detection is possible.

5.2 Bayesian Framework for Independent Motion Detection

The previous section compared classifiers for detection of independent motion. A binary decision is made by the classifiers based on a single image point correspondence: The correspondence is classified as either being the projection of an independently moving object (IMO) or as being the projection of a static background object. Both states are mutually exclusive. In this aspect the classifiers directly model the underlying *state of nature* s: The object, of which the pixel is the projection, either moves or belongs to the static background (SBG). Contrary to the previous section, this section focuses on temporal and spatial integration of information. The objective of this section is the maintenance of an occupation map filled with sequentially captured information about independent motion from multiple image point correspondences. At first glance this seems straightforward, at closer inspection there are, however, a number of open questions concerning this approach:

1. What about locations where no correspondence has been measured? Which state of nature should be recorded in these positions?
2. Each classifier from the previous section is based on a certain threshold. One problem with the classification is that the choice of this threshold is, however, non-trivial. What is the best choice for the threshold?
3. Another open question is temporal integration, which seems to be an essential part of human perception (Sekuler et al., 2004): Image correspondences are measured, and each new image correspondence leads to new information about the same underlying state of nature. What is the best way to combine these pieces of information in a consistent manner? If, for example, the correspondence from the last frame indicates independent motion and the correspondence from the current frame indicates static background, which correspondence is more reliable?

Probability theory, and in particular Bayes law, provides a consistent approach dealing with these questions. It allows prediction of the state of nature, temporal integration of measurements, and postpones the decision about the threshold until it is needed. Locations where no correspondence is measured, can be filled with the prediction from the last time step when it is available or with a global prior. Probability theory transforms the question of the threshold of the classifiers to the more descriptive domain of probabilities and so eases the choice of the threshold for the user. Bayes law in combination with the Chapman-Kolmogorov prediction equation provides a consistent and simple approach for temporal integration.

Bayesian Approach: The previous section revealed that the classifier which is best suited for the detection of independent motion is based on the direction difference between predicted and measured translational flow α. When the camera is static, for example when

the car waits in front of a traffic light, the length of the optical flow l is an appropriate measurement for the detection of independent motion. A probabilistic Bayesian framework for detection of independent motion based on these two different cues is suggested. It aims at maintaining an *occupation probability map*. Each pixel in the occupation probability map[3] represents the conditional probability $p(s = \text{IMO}|\alpha, l)$ that the pixel position is the projection of an independently moving object based on α and l. The entries in the occupation probability image are conditioned on the two cues:

1. The direction difference α between expected and measured translational flow
2. The flow length l

The probability that the pixel belongs to the static background is given by the complement of the entries in the occupation probability map

$$p(s = \text{SBG}|\alpha, l) = 1 - p(s = \text{IMO}|\alpha, l) \tag{5.24}$$

It is very hard to find a plausible model for the conditional probability $p(s|\alpha, l)$ and hence, as usual, Bayes law (see appendix D.1) is used to derive this probability from the associated likelihood function $p(\alpha, l|s)$ and the prior $p(s)$

$$p(s|\alpha, l) = \frac{p(\alpha, l|s)p(s)}{p(\alpha, l)} \tag{5.25}$$

Or, in nomenclature of statistics: The *posterior* $p(s|\alpha, l)$ is computed using the *likelihood function* $p(\alpha, l|s)$, the *prior* $p(s)$ and the normalisation $p(\alpha, l) = \sum_s p(\alpha, l|s)p(s)$. See appendix D for a brief introduction to statistics.

The *likelihood function* represents the "inverse" probability, i.e. the probability of a certain direction difference and a certain flow length $p(\alpha, l|s)$ given knowledge about the state of nature s (i.e. pixel is projection of IMO or SBG). Two different state of natures exist, and hence two different likelihood models are suggested: One under the assumption that the point belongs to the static background $p(\alpha, l|s = \text{SBG})$, and one under the assumption that the point belongs to independent motion $p(\alpha, l|s = \text{IMO})$. Both models are needed, for example for the normalisation $p(\alpha, l) = \sum_s p(\alpha, l|s)p(s)$ in Bayes law. Combined likelihood functions are difficult to model, and the most simple approach is to assume independence between α and l. This is equivalent to the independence assumption in the so-called "naive" ore simple Bayes classifier. Even though the independence assumption is very often violated in practise, naive Bayes classification gives remarkably good results (Elkan, 1997), and hence the independence assumption is adopted for this

[3]Note that the occupation probability map does not constitute a probability density function because it does not necessarily integrate to one. The occupation probability does not integrate to one because there is **no** restriction, (i) that the projection of each IMO covers only a single pixel and (ii) that only a single independently moving object exists. Normalisation to one would not be a valid operation because multiple pixels are allowed to be projections of moving objects.

case. When both variables are independent, the combined likelihood function is given by $p(\alpha, l|s) = p(\alpha|s)p(l|s)$ and the posterior becomes

$$p(s|\alpha, l) = \frac{p(\alpha|s)p(l|s)p(s)}{p(\alpha, l)} \tag{5.26}$$

with the normalisation

$$\begin{aligned} p(\alpha, l) = \quad & p(\alpha|s = \text{IMO})p(l|s = \text{IMO})p(s = \text{IMO}) \quad + \\ & p(\alpha|s = \text{SBG})p(l|s = \text{SBG})p(s = \text{SBG}) \end{aligned} \tag{5.27}$$

The *prior* $p(s)$ describes external knowledge about the system state (i.e. IMO or SBG) and can be used for temporal integration.

The two likelihood models are described in the next sections, and the usage of the prior for temporal integration is described in section 5.2.4.

5.2.1 Likelihood Models Conditioned on Independent Motion

Direction Difference

When a point is located on a moving object and when no further knowledge about the object motion is present, no information about the flow direction can be made, and hence a uniform distribution in the interval between $-\pi$ and π is chosen. The likelihood for a certain direction difference α between the predicted and the measured translation flow direction, given the fact that the point correspondence is a projection of an independently moving object (IMO), is hence modelled as

$$p(\alpha|s = \text{IMO}) = \frac{1}{2\pi} \tag{5.28}$$

Flow Length

Without further knowledge about the relative motion between object and camera, no information about the flow length can be made, and hence the likelihood of the flow length conditioned on the fact that the point is a projection of an independently moving object $p(l|s = \text{IMO})$ is modelled as a uniform distribution in the interval between zero and the maximal measurable flow length $l_{\max}$

$$p(l|s = \text{IMO}) = \frac{1}{l_{\max}} \tag{5.29}$$

The maximal measurable flow length $l_{\max}$ depends on the algorithm used for correspondence estimation (e.g. 40 pixel).

Combination

The combined likelihood model for direction difference and flow length, conditioned on the fact that the point is the projection of independent motion, is given by

$$p(\alpha, l|s = \text{IMO}) = p(\alpha|s = \text{IMO})p(l|s = \text{IMO}) = \frac{1}{2\pi l_{\max}} \tag{5.30}$$

5.2.2 Likelihood Model for Direction Difference for Static Background

The likelihood model for a certain direction difference between the expected translational flow and the measured translational flow, conditioned on the fact that the point is a projection of static background, is described in this section. Basic parameters of the model are the direction difference α and its uncertainty σ_α. After the description of the likelihood model, the derivation of the direction difference α and the associated uncertainty σ_α is explained.

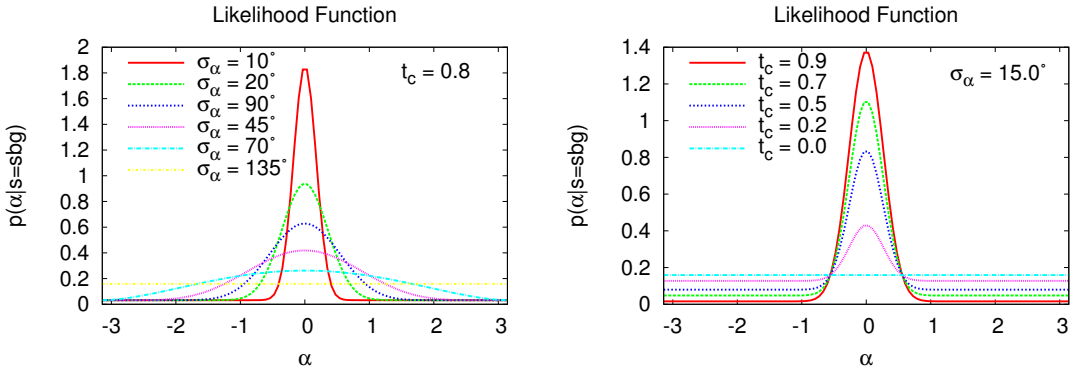
Likelihood Model

When the 3D-point is located on the static background and when the camera motion relative to the background is known, the direction of the associated translational flow can be predicted. Nonetheless, the measured direction can be different from the predicted direction even though the underlying assumption (i.e. point is SBG) is not violated. This difference in direction results for example from inaccuracies in the 2D-correspondences or from noise in the relative camera motions. The uncertainties of the correspondences and the uncertainties of the relative camera poses can be modelled by normal distributions, and it is straight forward to model the likelihood for a certain direction difference also using a normal distribution. There is, however, a problem with this approach: The direction difference is bound to lie in the interval between $-\pi$ and π, and a normal distribution would result in a non-zero likelihood for values outside of this interval. This does clearly not model the reality since the probability for a direction difference of more than π is zero. For this reason, the likelihood function is modelled using the beta distribution, which is only defined on an interval and hence better approximates the reality.

Under the assumption that the point correspondence stems from the static background (SBG), the likelihood function for the direction difference α is modelled as

$$p(\alpha|s = \text{SBG}) = \frac{t_c}{2\pi} B\left(\frac{\alpha}{2\pi} + \frac{1}{2} \mid a, a\right) + \frac{1 - t_c}{2\pi} \tag{5.31}$$

with the confidence in the correctness of the correspondences $t_c \in [0, 1]$ and the beta distribution $B(x|a, b)$ (see appendix D.2 for detailed description of beta function). The confidence in the correspondence correctness remains a parameter of the final algorithm and is set by the user (e.g. to 0.9). Note that the direction difference α is scaled and shifted before fed to the beta distribution. This is necessary because the beta distribution is only



(a) (b)

Figure 5.14: Likelihood function $p(\alpha|s = \text{SBG})$ for direction difference α under the assumption that the correspondence stems from a point on the static background. (a) Likelihood functions with different uncertainty of the direction difference σ_α are shown on the left. (b) Likelihood functions with different confidence in the correspondence measurements t_c are shown on the right.

defined in the interval $[0, 1]$ of the parameter x while the direction difference is bound to the interval $[\pi, -\pi)$. The parameter a of the beta distribution is chosen depending on the variance of the direction difference measurement σ_α^2 as

$$a = \begin{cases} \frac{4\pi^2}{8\sigma_\alpha^2} - \frac{1}{2} & \text{if } \frac{4\pi^2}{8\sigma_\alpha^2} - \frac{1}{2} >= 1 \\ 1 & \text{otherwise} \end{cases} \tag{5.32}$$

The variance of the beta distribution $B(x|a, b)$ is given by (Figueiredo, 2004)

$$\sigma^2_{B(x|a,b)} = \frac{ab}{(a+b)^2(a+b+1)} \tag{5.33}$$

The choice of a and b thus results in a variance of the beta distribution

$$\sigma^2_{B(x|a,a)} = \frac{\sigma_\alpha^2}{4\pi^2} \tag{5.34}$$

and hence the standard deviation of the "scaled" beta distribution $B(\frac{\alpha}{2\pi}|a, a)$ is σ_α. Figure 5.14 illustrates the likelihood function $p(\alpha|\text{SBG})$ for different choices of t_c and σ_α. Note that σ_α is computed from the data. The computation of α and σ_α are explained next.

Parameters of the Likelihood Model

Computation of Direction Difference: Given a correspondence $\boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathbb{R}^2$ between the projections of a 3D-point in two images and the relative orientation of both images, the correspondence can be de-rotated (see section 5.1.2). The relative orientation is an

estimated value and hence afflicted with uncertainty. This uncertainty is incorporated into the de-rotation process using the unscented transform (see appendix D.3). De-rotation of $\boldsymbol{x}_2$ results in the point $\hat{\boldsymbol{x}}_2$ and the associated covariance matrix $\Sigma_{\hat{\boldsymbol{x}}_2\hat{\boldsymbol{x}}_2}$. The direction of the translational flow vector $\boldsymbol{d}_f$ is given by

$$\boldsymbol{d}_f = \begin{pmatrix} d_{fx} \\ d_{fy} \end{pmatrix} = \frac{\hat{\boldsymbol{x}}_2 - \boldsymbol{x}_1}{|\hat{\boldsymbol{x}}_2 - \boldsymbol{x}_1|} \tag{5.35}$$

when the magnitude of the translational flow is not zero. When it is zero, a direction difference cannot be established, and the likelihood $p(\alpha|s = \text{SBG})$ is set to $\frac{1}{2\pi}$ resulting in a posterior which is independent of α. The direction is expressed as an orientation angle α_f

$$\alpha_f = \operatorname{atan}\left(\frac{d_{fy}}{d_{fx}}\right) \tag{5.36}$$

When the signums of the components of the direction vector $\boldsymbol{d}_f$ are used, a unique direction $\alpha_f \in [-\pi, \pi)$ can be determined. The direction angle is the angle between $\boldsymbol{d}_f$ and the x-axis.

The vector containing the predicted direction $\boldsymbol{d}_p$ for the point $\boldsymbol{x}_1$ can be computed when the epipole $\boldsymbol{e} \in \mathbb{R}^2$ is known and when the distance between the point $\boldsymbol{x}_1$ and the epipole is not zero

$$\boldsymbol{d}_p = \begin{pmatrix} d_{px} \\ d_{py} \end{pmatrix} = \frac{\boldsymbol{x}_1 - \boldsymbol{e}}{|\boldsymbol{x}_1 - \boldsymbol{e}|} \tag{5.37}$$

When the distance between point and epipole is zero, a direction difference cannot be predicted, and the likelihood function $p(\alpha|s = \text{SBG})$ is set to $\frac{1}{2\pi}$ resulting in a posterior which is independent of α. The direction is again expressed as an orientation angle α_p

$$\alpha_p = \operatorname{atan}\left(\frac{d_{py}}{d_{px}}\right) \tag{5.38}$$

The direction difference α between the expected translational flow and measured translational flow is given by

$$\alpha = \alpha_p - \alpha_f \tag{5.39}$$

Obviously, α can only take values between $-\pi$ and π. Of course the direction difference could also be computed using the scalar product between $\boldsymbol{d}_f$ and $\boldsymbol{d}_p$. The computation of the associated direction uncertainty σ_α is described next.

Uncertainty of Direction Difference: The computation of uncertainty of direction difference σ_α is a nonlinear problem. Three different approaches to this problem are compared. Common to all approaches is the computation of the uncertainty of the two difference vectors $\boldsymbol{d}_f$ and $\boldsymbol{d}_p$ (equations 5.35 and 5.37). The uncertainties, given by the covariance matrices $\boldsymbol{\Sigma}_{\boldsymbol{d}_f\boldsymbol{d}_f}$ and $\boldsymbol{\Sigma}_{\boldsymbol{d}_p\boldsymbol{d}_p}$, are computed using Gaussian error propagation

$$\boldsymbol{\Sigma}_{\boldsymbol{d}_f\boldsymbol{d}_f} = \boldsymbol{\Sigma}_{\boldsymbol{x}_1\boldsymbol{x}_1} + \boldsymbol{\Sigma}_{\hat{\boldsymbol{x}}_2\hat{\boldsymbol{x}}_2} \qquad \boldsymbol{\Sigma}_{\boldsymbol{d}_p\boldsymbol{d}_p} = \boldsymbol{\Sigma}_{\boldsymbol{x}_1\boldsymbol{x}_1} + \boldsymbol{\Sigma}_{\boldsymbol{e}\boldsymbol{e}} \tag{5.40}$$

The three algorithms differ in the derivation of the uncertainties σ_{α_f} and σ_{α_p} of the two orientation angles α_f and α_p. The final computation of the uncertainty in direction difference σ_α is again based on Gaussian error propagation and common to all three approaches

$$\sigma_\alpha^2 = \sigma_{\alpha_p}^2 + \sigma_{\alpha_f}^2 \tag{5.41}$$

The three different approaches for the computation of the uncertainty σ_{α_o} of a direction angle α_o from a vector $\boldsymbol{d}_o = (d_x, d_y)^T$ and the associated covariance matrix $\boldsymbol{\Sigma}_{\boldsymbol{d}_o\boldsymbol{d}_o}$ are described and compared next. The subscript $_o$ is used as a placeholder for either $_f$, in case of measured translational flow, or $_p$, in case of predicted translational flow.

1. Linear Error Propagation: According to equations 5.36 and 5.38 the computation of the orientation angle has the form

$$\alpha_o = \operatorname{atan}\left(\frac{d_y}{d_x}\right) \tag{5.42}$$

Using Gaussian error propagation, the variance of the angular direction is given by

$$\sigma_{\alpha_o}^2 = \mathbf{J}_{\alpha_o} \boldsymbol{\Sigma}_{\boldsymbol{d}_o\boldsymbol{d}_o} \mathbf{J}_{\alpha_o}^T \tag{5.43}$$

with the Jacobian of equation 5.42

$$\mathbf{J}_{\alpha_o} = \frac{1}{1 + \frac{d_y^2}{d_x^2}} \begin{pmatrix} -\frac{d_y}{d_x^2} & \frac{1}{d_x} \end{pmatrix} \tag{5.44}$$

When the uncertainty of the direction vector $\boldsymbol{\Sigma}_{\boldsymbol{d}_o\boldsymbol{d}_o}$ is large compared to the length of the direction vector, the errors induced by the linearisation from the Gaussian error propagation can no longer be neglected. This situation occurs for example for points with very small flow vectors or points very close to the epipole.

2. Geometric Error Propagation: The second approach to error propagation is based on geometric relations between the vector $\boldsymbol{d}_o$ and its uncertainty ellipsoid $\boldsymbol{\Sigma}_{\boldsymbol{d}_o\boldsymbol{d}_o}$. The uncertainty ellipsoid is defined by its half axes, which are given by the eigenvectors of the covariance matrix scaled such that their lengths are equal to the square root of the corresponding eigenvalues[4]. The angular uncertainties $\sigma_{\alpha_o}^+$ and $\sigma_{\alpha_o}^-$ resulting from uncertainty of the vector $\boldsymbol{\Sigma}_{\boldsymbol{d}_o\boldsymbol{d}_o}$ can be derived as follows: The tangents to the uncertainty ellipsoid going through the origin O and the vector $\boldsymbol{d}_o$ span the angular uncertainties (see figure 5.15). Two different algorithms, a numeric and an analytic algorithm, for the computation of the tangent points $\boldsymbol{t}^+$ and $\boldsymbol{t}^-$ are described in appendix A.4. For a given tangent point $\boldsymbol{t}^\pm$, the angle $\sigma_{\alpha_o}^\pm$ spanned by $\boldsymbol{t}^\pm$, the origin and $\boldsymbol{d}_o$, can be computed by

$$\sigma_{\alpha_o}^\pm = \operatorname{acos}\left(\frac{\boldsymbol{d}_o^T \boldsymbol{t}^\pm}{|\boldsymbol{t}_i||\boldsymbol{d}_o|}\right) \tag{5.45}$$

[4]When the errors in x and y are independent (i.e. when the axes of the ellipsoid are aligned with the coordinate system), the covariance matrix is diagonal with entries σ_{xx}^2 and σ_{yy}^2, and the half axes have the length of the standard deviations σ_{xx} and σ_{yy}.

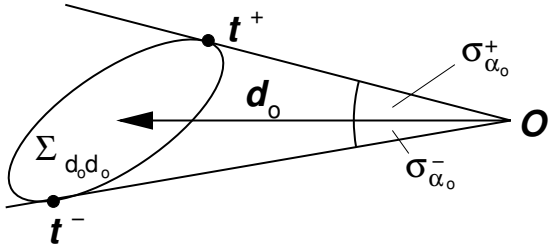


Figure 5.15: Geometric derivation of the angular uncertainties σ_α^+ and σ_α^- from the vector $\boldsymbol{d}_o$ and its associated uncertainty $\boldsymbol{\Sigma}_{\boldsymbol{d}_o\boldsymbol{d}_o}$.

3. Approximate Error Propagation: Let the uncertainty ellipsoid be again defined as in the previous approach. Using the worst case assumption that the longer half axis $\boldsymbol{h}_l$ of the uncertainty ellipsoid $\Sigma_{\boldsymbol{d}_o\boldsymbol{d}_o}$ is perpendicular to the tangent through the origin O, results in the approximate angular uncertainty (see figure 5.16)

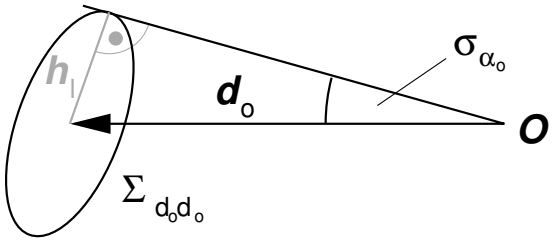


Figure 5.16: Approximating the angular uncertainty σ_{α_o} using the worst case assumption that the longer half axis $\boldsymbol{h}_l$ of the ellipsoid $\Sigma_{\boldsymbol{d}_o\boldsymbol{d}_o}$ is perpendicular to the tangent through the origin.

$$\sigma_{\alpha_o} \approx \operatorname{asin}\left(\frac{|\boldsymbol{h}_l|}{|\boldsymbol{d}_o|}\right) \tag{5.46}$$

Comparison: Two different scenarios are chosen to investigate the properties of the different error propagation algorithms.

1. Fixed covariance matrix: The covariance matrix is held fixed in the first scenario. It is inclined by 45° to the coordinate system and its half axes have lengths 1 and 0.5 units. The length d of the translational flow vector $\boldsymbol{d}_o$ is varied. This is illustrated in figure 5.17(a). Figure 5.18 shows the results of the comparison in form of the left and right angular uncertainties $\sigma_{\alpha_o}^{\pm}$ for different lengths of $\boldsymbol{d}_o$. The "approximate" algorithm gives an approximated upper bound for the error.

2. Fixed vector: In the second scenario the length of the flow vector is fixed at 5 units and the inclination angle of the covariance matrix changes. The half axes of the covariance matrix are the same as in the first scenario. This scenario is illustrated in figure

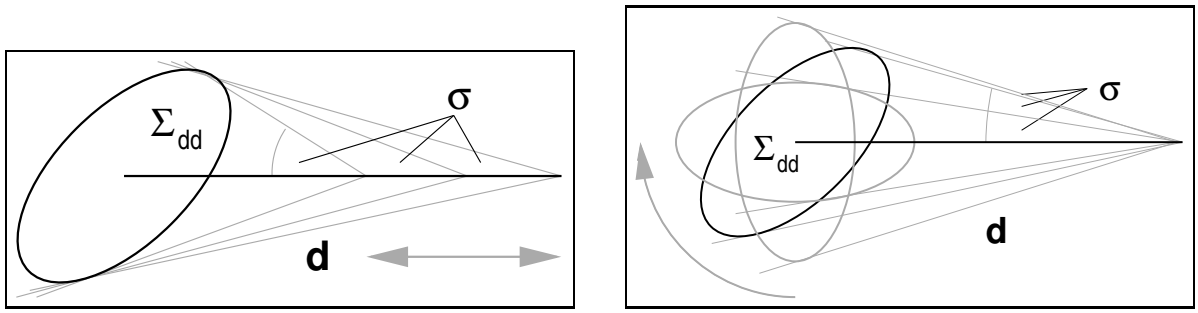


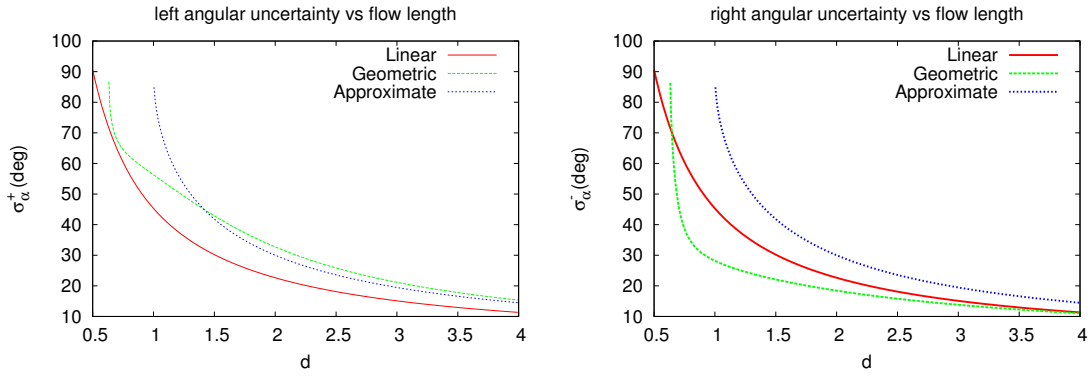
Figure 5.17: Visualisation of the two scenarios for comparison of error propagation algorithms for uncertainty of direction of translational flow. (a) The influence of the length d of the translational flow is investigated. The covariance matrix of the flow $\mathbf{\Sigma}_{dd}$ is held fixed in this scenario. (b) The influence of covariance matrix orientation is investigated with this scenario. The length of the translational flow d is fixed.

5.17(b). Figure 5.19 shows the results of the comparison. Note the differences between linear error propagation and geometric error propagation. The linear error propagation results in a symmetric uncertainty while the geometric error propagation captures the geometric nonlinearities of the transformation.

Algorithm	time [μs]
Linear	2.2
Geometric (analytic)	10.9
Geometric (numeric)	4.8
Approximate	2.1

Table 5.2: Computational requirements of error propagation algorithms. The time measurements were conducted on a Pentium M with 1.5 GHz.

The computational requirements of the 4 different algorithms for error propagation are summarised in table 5.2. The numeric geometric algorithm is chosen for the error propagation in the final algorithm, because it is relatively fast and deals with the nonlinearities.



(a) (b)

Figure 5.18: Comparison of error propagation algorithms. The covariance matrix of the point is held fixed with an inclination of 45° to the coordinate system, while the length d of the flow vector $\boldsymbol{d}_o$ varies. See text for detailed explanation.

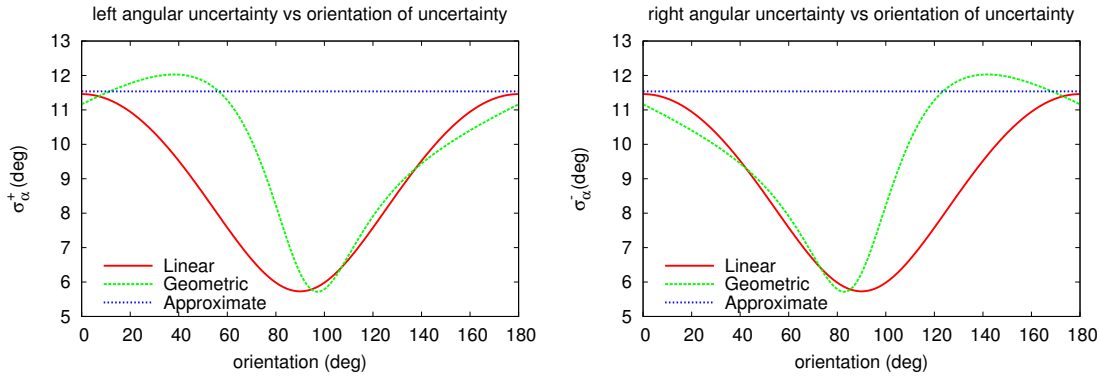
5.2.3 Likelihood Model for Flow Length for Static Background

The flow length l can also be used as a cue for independent motion. When a minimum distance of the objects from the camera is given, for example by the bonnet, a maximum length for the translational part of the optical flow can be computed from the known camera motion under the assumption that the correspondence belongs to the static background. First the likelihood model and afterwards the derivation of the parameters of the models l_{SBG} and σ_l is explained.

Likelihood Model

Given a minimum distance of the camera from all scene points, a maximum length l_{SBG} of the optical flow for points belonging to the static background can be derived. The probability that a flow vector belongs to the static background is significantly lower for flow vectors longer than this boundary than for flow vectors shorter than this boundary. Without further knowledge about the scene geometry, no further information is available, and hence all flow vectors shorter than the boundary should share the same probability. For the same reason, all flow vectors longer than the boundary should also share the same probability. Due to uncertainties in 2D-correspondences and camera poses, the boundary is also uncertain.

The logistic function (see appendix B.2) models these facts and it is hence chosen to model the likelihood for flow length $p(l|s = \text{SBG})$ under the assumption that the image point is a projection of the static background. The logistic function can be interpreted as a uniform distribution for an interval between 0 and a "soft" boundary at l_{SBG}. The "softness" of the boundary is determined by the uncertainty σ_l of the difference between l_{SBG} and l. The maximal flow length l_{SBG} of background points and the uncertainty σ_l of



(a) (b)

Figure 5.19: Comparison of error propagation algorithms. The length of the flow vector is fixed at 5 units and the orientation of the covariance matrix varies. See text for detailed explanation.

the difference between the measured flow length and l_{SBG} are parameters of the likelihood function

$$p(l|s = \text{SBG}) = t_c \frac{1}{n} \frac{1}{1 + \exp\left(\frac{l - l_{\text{SBG}}}{\sigma_l}\right)} + (1 - t_c) \frac{1}{l_{\max}} \tag{5.47}$$

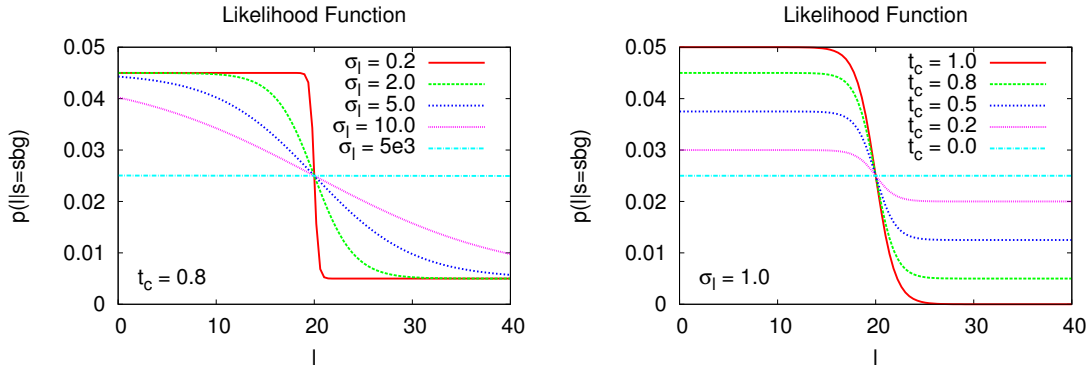
with the confidence in the flow measurements $t_c \in [0, 1]$ (e.g. $t_c = 0.9$) and the normalisation constant n

$$n = \int_0^{l_{\max}} \frac{1}{1 + \exp\left(\frac{l - l_{\text{SBG}}}{\sigma_l}\right)} dl = l_{\max} - \sigma_l \left[\ln\left(1 + e^{\frac{l_{\max} - l_{\text{SBG}}}{\sigma_l}}\right) - \ln\left(1 + e^{\frac{-l_{\text{SBG}}}{\sigma_l}}\right) \right] \tag{5.48}$$

The normalisation constant accounts for the fact that a probability distribution must integrate to one over the interval between 0 and $l_{\max}$. Figure 5.20 visualises the likelihood function for different values of σ_l and t_c.

Parameters of the Likelihood Model

Computation of Maximal Translational Flow Length: The maximal theoretic possible flow length l_{SBG} is computed for a specific image point $\boldsymbol{x}_1$. It depends on the position of the image point and on the camera positions and orientations given by the projection matrices $\mathbf{P}_1 = [\mathbf{R}_1^T | -\mathbf{R}_1^T \boldsymbol{C}_1]$ and $\mathbf{P}_2 = [\mathbf{R}_2^T | -\mathbf{R}_2^T \boldsymbol{C}_2]$. The camera poses and uncertainties can be extracted from the known essential matrix and the car inertial sensor data, namely velocity and time. Using the minimal distance $d_{\min}$ of the scene from the camera, a hypothetically closest 3D-point to the cameras $\boldsymbol{X}_h$ is generated, such that it projects onto the image point $\boldsymbol{x}_1$ and has distance $d_{\min}$ from centre of camera 1 and a bigger distance to the centre of camera 2. If this constraint cannot be fulfilled, the role of



(a) (b)

Figure 5.20: Likelihood function $p(l|s = \text{SBG})$ for flow length l under the assumption that the correspondence stems from a point on the static background. The maximal flow length of background points l_{SBG} is 20 in the graphs. (a) Likelihood functions with different uncertainties of maximal flow length from static background σ_l are shown on the left. (b) Likelihood functions with different confidences in the correspondence measurements t_c are shown on the right.

both cameras and the role of both 2D-points must be exchanged.

$$\begin{aligned} |\boldsymbol{X}_h - \boldsymbol{C}_1| &= d_{\min} \\ |\boldsymbol{X}_h - \boldsymbol{C}_2| &\geq d_{\min} \\ \mathbf{P}_1(\boldsymbol{X}_h) &= \boldsymbol{x}_1 \end{aligned} \tag{5.49}$$

Using the projections of $\boldsymbol{X}_h$, the maximum flow length for static background l_{SBG} is given by

$$l_{\text{SBG}} = |\mathbf{P}_1(\boldsymbol{X}_h) - \mathbf{P}_2(\boldsymbol{X}_h)| \tag{5.50}$$

with the projections $\mathbf{P}_1(\boldsymbol{X}_h) \in \mathbb{R}^2$ and $\mathbf{P}_2(\boldsymbol{X}_h) \in \mathbb{R}^2$ of the 3D-point .

Uncertainty of Maximal Translational Flow Length: The uncertainty of the maximal flow length for points from the static background depends on the uncertainties from the camera poses (i.e. the uncertainties in the camera positions and in the camera orientations). Again the unscented transform (see appendix D.3), is used for error propagation resulting in $\sigma_{l_{\text{SBG}}}$.

A second source for uncertainty is the covariance matrix of the translational flow vector $\boldsymbol{\Sigma}_{\boldsymbol{d}_f \boldsymbol{d}_f}$ (see equation 5.40). Using linear error propagation (appendix D.3) the uncertainty $\sigma_{l_{\text{length}}}$ of length of $\boldsymbol{d}_f$ is given by

$$\sigma^2_{l_{\text{length}}} = \mathbf{J}_{\text{norm}} \boldsymbol{\Sigma}_{\boldsymbol{d}_f \boldsymbol{d}_f} \mathbf{J}^T_{\text{norm}} \tag{5.51}$$

with the Jacobian $\mathbf{J}_{\text{norm}}$ of the vector norm $\left|(d_x, d_y)^T\right| = \sqrt{d_x^2 + d_y^2}$

$$\mathbf{J}_{\text{norm}} = \frac{1}{\sqrt{d_x^2 + d_y^2}} \begin{pmatrix} d_x & d_y \end{pmatrix} \tag{5.52}$$

The uncertainty σ_l of the difference between the maximal flow length of static background objects l_{SBG} and the actual flow length l is computed using again linear error propagation. It has uncertainty $\sigma_l^2 = \sigma_{l_{\mathrm{SBG}}}^2 + \sigma_{l_{\mathrm{length}}}^2$. This uncertainty is used in the likelihood model (equation 5.47).

5.2.4 Temporal Integration

Temporal integration seems to be an essential part of human perception (Sekuler et al., 2004). This is consistent with the fact that the distinction between moving object and static background is usually valid for a relatively long period of a time. It is certainly possible that a moving object stops and becomes part of the static scene or that a static object starts moving, but these events are rather rare in typical traffic situations. When the distinction between moving and non-moving is valid for several frames, it is advantageous to capture as much information as possible to come to a well-founded decision about each pixel. Temporal integration combines information about a single object from several frames in a consistent manner and thus provides an easy way of enhancing the reliability of information.

Spatial smoothing could be considered as an alternative to temporal integration. It is, however, more complicated to model: What should be the scale of the spatial smoothing? If the scale on one hand is too small, it has no effect, especially when only sparse information is present. If on the other hand the scale is too large, small objects like for example the children's toy ball, might not be detected. When fixing a scale of the smoothing, implicit assumptions about the expected size of the images of moving objects are made. Apart from the size of the object itself, the size of its image depends on the distance, the focal length and other parameters, and these parameters are subject to change even within a rather short period of time. Temporal smoothing is easier to model. The only question to answer concerns the probability that the state of an object changes. Therefore temporal integration is favoured in this thesis.

Let the state of the system at time t be denoted by s_t and let the measurements at time t be denoted by α_t and l_t. When the system is modelled as a Markov process, the transition probability $p(s_{t+1}|s_0, s_1, \ldots, s_t) = p(s_{t+1}|s_t)$ only depends on the current state and not on the history. The transition probability models the chance of a change in the state of the system, i.e. the probability that a moving object stops, or the probability that a object starts to move.

Temporal integration aims at accumulating the information from each time step by replacing the prior $p(s_t)$ in Bayes law with the prediction of the state $p(s_t|\alpha_{t-1}, l_{t-1})$ from the last time step. This is done using the well-known Chapman-Kolmogorov prediction

equation (Doucet et al., 2001)

$$\begin{aligned} p(s_t|\alpha_{t-1}, l_{t-1}) &= \int p(s_t|s_{t-1}) p(s_{t-1}|\alpha_{t-1}, l_{t-1})\ ds_{t-1} \\ &= p(s_t|s_{t-1} = \text{IMO}) p(s_{t-1} = \text{IMO}|\alpha_{t-1}, l_{t-1}) + \\ &\quad p(s_t|s_{t-1} = \text{SBG}) p(s_{t-1} = \text{SBG}|\alpha_{t-1}, l_{t-1}) \end{aligned} \tag{5.53}$$

Replacing the prior in 5.26 by equation 5.53 results in the update equation

$$p(s_t|\alpha_t, l_t) = \frac{p(\alpha_t, l_t|s_t) p(s_t|\alpha_{t-1}, l_{t-1})}{\sum_{s_t} p(\alpha_t, l_t|s_t) p(s_t|\alpha_{t-1}, l_{t-1})} \tag{5.54}$$

The transition probability $p(s_t|s_{t-1})$ is modelled as

$$p(s_t|s_{t-1}) = \begin{cases} \lambda & \text{if } s_t = s_{t-1} \\ 1 - \lambda & \text{if } s_t \neq s_{t-1} \end{cases} \tag{5.55}$$

with the transition probability $\lambda \in [0, 1]$. λ can be interpreted as a memory factor: When λ is 0.5, the predicted occupation probability is non-informative (i.e. $p(s_t|\alpha_{t-1}, l_{t-1}) = 0.5$), resulting in a posterior which is independent on the history. In this case the system has no memory, and the occupation probability depends solely on the current measurements.

When λ is 1.0, the system does not forget anything and the prior is given by the occupation probability from the last time step. The resulting posterior is given by the Bayesian combination with the occupation probability based on the current measurements.

$\lambda = 0.0$ indicates that the system has definitely changed its state since the last time step. The system has also a memory, and it is assumed that the prior is given by one minus the occupation probability from the last time step. The resulting posterior is then given by the Bayesian combination with the occupation probability based on the current measurements.

Reasonable values for λ for the given application are in the range between 0.5 and 1.0.

5.2.5 Estimation of Occupation Probability

Since determining highly accurate correspondences with subpixel precision is a computationally expensive operation, the number of correspondence measurements has to be restricted in real-time environments. Therefore the occupation probability map is only very sparsely populated.

However, when computing correspondences at sparse locations, one would like to capture as much information about independently moving objects as possible. A novel algorithm boosting the density of the occupation probability map is suggested. It is inspired by the particle filter algorithm and results in adaptive sampling of the occupation probability function such that the sampling rate is higher in positions where the occupation probability is high. Positions where correspondences are measured are determined partly using a vector of random variables, which are distributed according to the cornerness function, partly by propagating positions from the last time step using a particle filter style approach and partly using stable features on the static background.

First the general idea of the particle filter is reviewed and afterwards its adaptation to the problem at hand is described.

Particle Filter

The idea of the particle filter has been first published in 1949 by Metropolis and Ulam. Since then it has been only sporadically mentioned in the literature up to the rediscovery in 1993 by Gordon et al. A large variety of papers on particle filters has been written since. A good introduction can be found in Doucet et al. (2001). In 1998 Isard and Blake (1998a,b) introduced this technique under the name of CONDENSATION (CONditional DENSity propagATION) into the field of computer vision for tracking tasks. Lately a lot of effort went into improvements of particle filters to overcome certain limitations and problems (Hue et al., 2002; Khan et al., 2004; Isard and McCormick, 2001; Vermaak et al., 2003). However, only little work has been done in the field of using such probabilistic techniques for the investigation and interpretation of optical flow fields: In Black and Fleet (1999) motion discontinuities are tracked using optical flow and the CONDENSATION algorithm, and in 2002 Zelek used a particle filter to predict and therefore speedup a correlation based optical flow algorithm.

In this subsection the general concept of the particle filter algorithm is summarised. The particle filter algorithm is designed to handle the task of propagating any probability density function (pdf) over time by representing it by a set of weighted samples.

Bayesian Filtering: Let θ_t denote the unobserved state of the system at the discrete time $t \in \mathbb{N}$. The a priori probability density distribution is given by $p(\theta_t)$. Let further denote y_t the observation of our system at time t. Given a likelihood function $p(y_t|\theta_t)$, modelling the observation process, the posterior distribution can be computed using Bayes' law

$$p(\theta_t|y_t) = \frac{p(y_t|\theta_t)p(\theta_t)}{\int p(y_t|\theta_t)p(\theta_t)\ d\theta} \tag{5.56}$$

where the marginalisation $p(y_t) = \int p(y_t|\theta_t)p(\theta_t)\ d\theta$ can be seen as a normalisation factor.

Temporal Propagation: Modelling the system as a Markov process results in the conditional transition probability $p(\theta_{t+1}|\theta_0, \theta_1, \ldots, \theta_t) = p(\theta_{t+1}|\theta_t)$. In other words: The state of the system in the next time step only depends on the current state of the system and not on the history. Since $p(\theta_t)$ is at no time known exactly (it is unobserved), the best estimate for $p(\theta_t)$ can be calculated by using the estimate from the last time step $p(\theta_{t-1}|y_{t-1})$ and the transition probability $p(\theta_t|\theta_{t-1})$ resulting in the Chapman-Kolmogorov prediction equation (Doucet et al., 2001)

$$p(\theta_t|y_{t-1}) = \int p(\theta_t|\theta_{t-1})p(\theta_{t-1}|y_{t-1})\ d\theta \tag{5.57}$$

Plugging this estimate into Bayes' law (equation 5.56) results in the update equation

$$p(\theta_t|y_t) = \frac{p(y_t|\theta_t)p(\theta_t|y_{t-1})}{\int p(y_t|\theta_t)p(\theta_t|y_{t-1})\ d\theta} \tag{5.58}$$

Generally the prediction and update equations cannot be computed in closed form since they require the evaluation of complex and possibly multidimensional integrals.

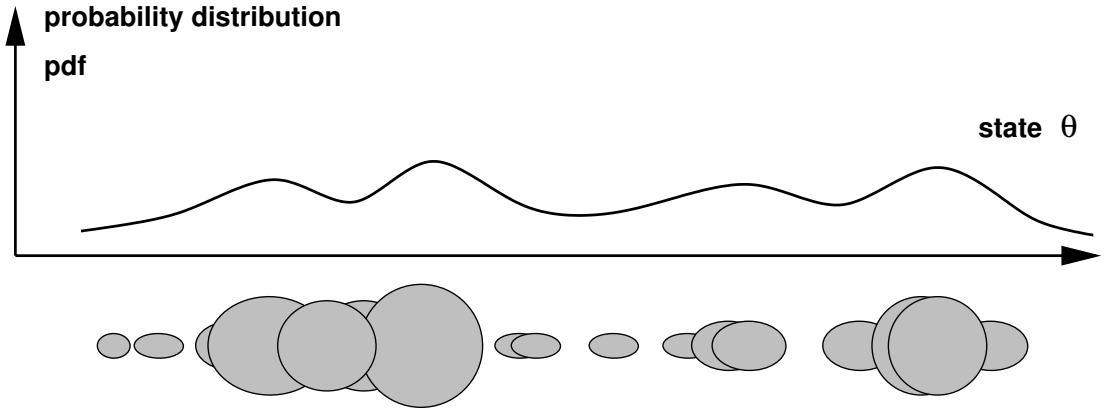


Figure 5.21: Representation of multimodal, one-dimensional probability density function through a set of weighted samples. The centres of the blobs represent the position and the size of the blobs represent the weight of the samples $s^{(n)}$. (Illustration is similar to Isard and Blake (1998a))

Particle Filter Algorithm: One way of circumventing the computation of the possibly highly complex integrals in equations 5.57 and 5.58 is to apply Monte Carlo techniques. This leads to the particle filter algorithm where the distributions are approximated by a set of N particles $\theta^{(i)}$ and their weights $w^{(i)}$. The approximation of a one-dimensional distribution by a set of particles is illustrated in figure 5.21. A theoretical description of

- Initialise (t=0):
  Generate N independent identical distributed (iid) samples $\theta_0^{(i)}, i \in \{1, \ldots, N\}$ from the user given initial distribution $p(\theta_0)$.
- Iterate:
  1. Predict $\theta_{t+1}^{(i)}$ by sampling from $p(\theta_{t+1}|\theta_t^{(i)})$
  2. Evaluate weights $w_{t+1}^{(i)} = p(y_{t+1}|\theta_{t+1}^{(i)})$
  3. Normalise the weights.
  4. Resample N times with replacement from the samples $\theta_{t+1}^{(i)}$ according to the weight $w_{t+1}^{(i)}$.
  5. Set $t = t + 1$ and repeat iteration (goto 1).

Figure 5.22: Theoretical description of the particle filter algorithm following Doucet et al. (2001).

the particle filter algorithm is given in figure 5.22. Practically the propagation of pdfs over time using their representation as a set of weighted particles reduces to the construction of the "new" sample set $\{\theta_t^{(n)}, w_t^{(n)}, n = 1, \ldots, N\}$ at time t from the "old" sample set $\{\theta_{t-1}^{(n)}, w_{t-1}^{(n)}, n = 1, \ldots, N\}$ at time $t-1$. The conditional transition probability $p(\theta_{t+1}|\theta_t)$ is usually given by a motion model function $\mathbf{f}(\theta)$ plus diffusion.

1. Select sample $\theta_t^{(n)}$ with probability $w_{t-1}^{(n)}$ and assign it to the new sample $\theta_t^{'(n)}$. Select N new samples $\theta_t^{'(n)}$. Samples can be selected several times.
2. Predict the new position of the sample in the state space by applying the motion model $\mathbf{f}(\theta)$ to each sample $\theta_t^{''(n)} = \mathbf{f}(\theta_t^{'(n)})$.
3. Diffuse by adding noise to each sample $\theta_t^{(n)} = s_t^{''(n)} + \nu_t^{(n)}$.,where $\nu_t^{(n)}$ is a vector of normally distributed random variables with covariance matrix $\mathbf{B}$.
4. Weight each sample by making a measurement (i.e. evaluate the likelihood function) at its position $\theta_t^{(n)} = p(y_t|\theta_t = \theta_t^{(n)})$.
5. Normalise the weights such that their sum equals one.

To reduce the degeneracy problem[5] (Marchetti et al., 2006) and to allow reinitialisation if the object is lost, the sample selection step in the iteration process can be modified such that a fraction of the samples are chosen by sampling from an importance distribution $g(\theta)$ (Doucet et al., 2001). The weights of these importance samples $w_{\text{imp},t}^{(i)}$ must then be

[5]The degeneracy problem describes the fact that after a few iterations all but very few samples have a negligible weight and hence the diversity of the sample population is greatly reduced.

corrected to achieve a consistent representation of the posterior density (Isard and Blake, 1998b)

$$w_{\text{imp},t+1}^{(i)} = \frac{p(y_{t+1}|\theta_{t+1}^{(i)})}{g(\theta)} \tag{5.59}$$

Fig. 5.23 shows a graphical representation of one iteration step in the particle filter with the modified resampling step. Note the clustering effect of the particle filter: Particles are denser in regions with high probability density. For a thorough discussion of particle filters see (Doucet et al., 2001).

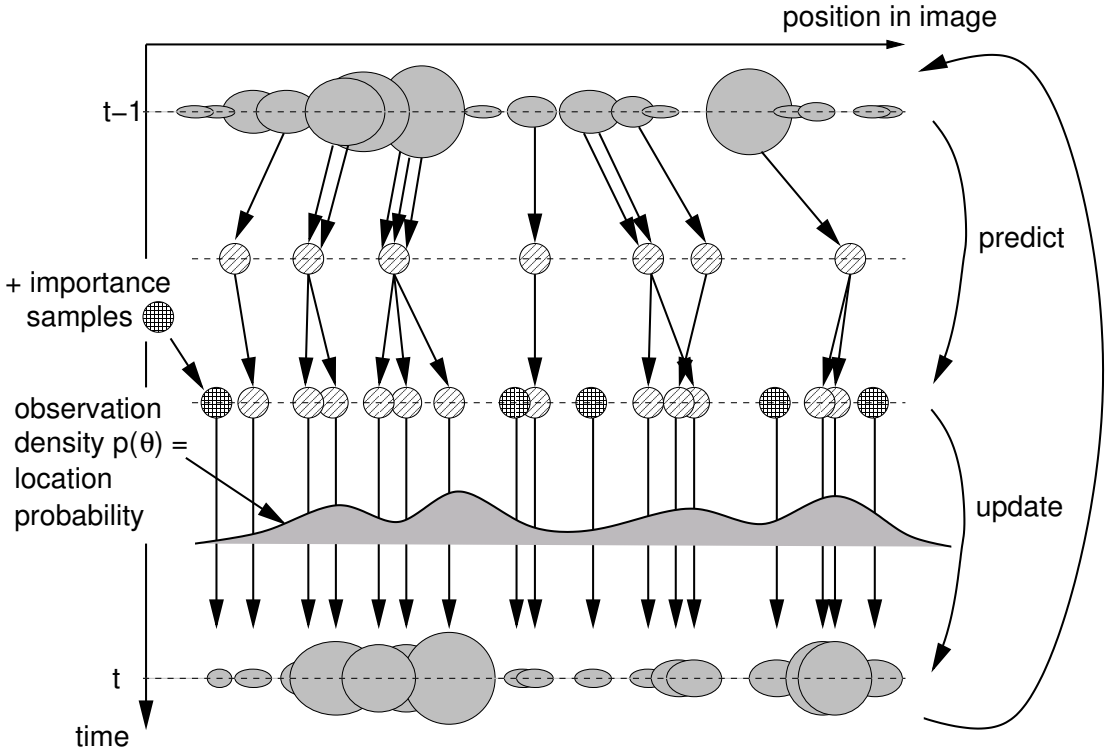


Figure 5.23: Graphical representation of a particle filter cycle with importance sampling. The one-dimensional state space is given on the horizontal axis. The vertical axis represents the time. The new samples are selected from the samples of the last time step (top) according to their weight (size of blobs). After applying the motion model, diffusion is added. These two steps represent the prediction step. The importance samples are added and weights are calculated by evaluating the observation density at the sample positions (update). The weights of the importance samples are corrected according to eq. 5.59 in this step. The resulting samples are used as input to the next time step. (Figure similar to (Doucet et al., 2001)). The "clustering effect" of the particle filter can be observed.

5.2.6 Boosting the Occupation Probability Image

The particle filter algorithm is used to propagate pdfs over time and usually only the first and second moments of the posterior pdf are of interest. However, in this case only the clustering property of the particle filter algorithm is desired. Primary goal of the algorithm is to boost the knowledge about independent motion by measuring more correspondences on independently moving objects. Points where correspondence measurements are made stem from three separate pools:

1. Stable features on the static background: These points are needed for the computation of the camera motion with respect to the static scene. They are determined using a corner detector and have some user given minimum distance between each other. Points are kept in this pool when their flow vector is consistent with the camera motion and rejected from this pool when it is not. Stable points can build long correspondence chains over several images.

2. Particle filter determined positions: These positions are purely determined by sampling from the occupation probability for independent motion. Because of the resampling, only correspondence chains between maximal two images are established.

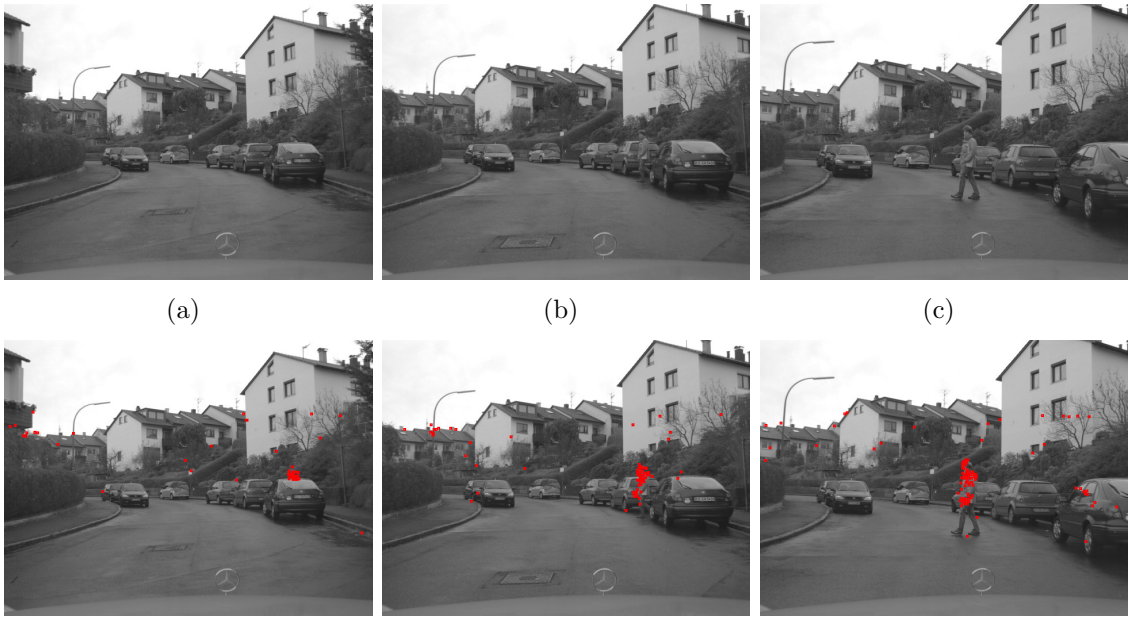
3. Initialisation positions: These positions are chosen by sampling from the cornerness image. They are chosen independently in each new image and give a chance for initialisation of new objects or rediscovery when the object has been lost (e.g. by occlusion).

The number of positions in each pool is user chosen and remains fixed. Figure 5.24 illustrates the boosting effect of the particle filter. Positions from the particle filter and initialisation pool are marked red. They cluster on the moving pedestrian.

Detailed descriptions of the generation of positions where correspondences are measured follow next.

Initialisation Positions: The initialisation positions are generated as follows: An integral cornerness vector is computed starting at the top left and progressing row major towards the bottom of the image. Each entry contains the sum of all preceding cornerness and the current position coordinates. Positions where the cornerness is below a certain threshold are neglected, because a minimal cornerness is needed for successful correspondence computation. The final vector is automatically sorted according to the cornerness sum. A uniform distributed random number between zero and the sum of cornerness value of the last entry in the vector is generated. The position from the entry of the vector corresponding to this random number determines the initialisation position. This is equivalent to (thresholded) sampling of the cornerness function.

Particle Filter Positions: Each sample lives in a two-dimensional state space representing a position in the image. The new particle positions are determined sequentially as follows:



(a) (b) (c)

(d) (e) (f)

Figure 5.24: Clustering effect of the particle filter algorithm. Point positions are chosen by sampling from the occupation probability image for independent motion and do hence cluster on the independently moving pedestrian. The original images are shown in the top row, the corresponding images with the particle positions marked in red are shown in the bottom row. Only positions from the particle filter pool and the initialisation pool are marked.

- Compute weights for each sample by first computing the 2D correspondence at the sample position and afterwards evaluating the occupation likelihood function (equation 5.26). Weights are computed for positions from all three pools. Obviously the pool with positions determined by the particle filter is initially empty.

- Generate new sample positions by sampling from old positions according to their weight such that samples with higher weight are chosen with higher probability. Samples from the three pools have equal rights in the sampling process.

- Predicting the new position is easy. It is the end point of the measured correspondence and therefore no motion model is necessary.

- Add normally distributed random diffusion to the new position. The new position is only accepted

  1. when the cornerness at the new sample position is above a certain threshold and
  2. when the new position has a minimum distance from already determined positions.

  Point 1 enhances the success rate of the correspondence measurements at the next time step, and point 2 avoids double correspondence computations. When the new position is rejected, a new diffusion is randomly generated and the conditions are checked again. This is repeated up to a maximum number of tries. When the maximum number of tries is reached, the particle is rejected. This is important to have an upper bound on execution time.

Temporal Integration

The advantage of the adapted particle filter algorithm is the enhanced measurement density on independently moving objects. One disadvantage is the temporal incoherency of the correspondence measurements. The maximum length of a correspondence chain in the particle filter pool is two frames. Longer correspondence chains are only produced by chance. Obviously this poses a problem for the Bayesian sequential estimation of the occupation probability as suggested in section 5.2.4. One way of circumventing this problem is described next.

Correspondences are estimated using a support window of a certain size (typically 7×7 pixel on the biggest pyramid level) and one underlying assumption of the correspondence estimation process is, that all pixels in the support window have the same displacement vector. With the same assumption, the estimated occupation likelihood would be valid for each pixel in the support window. This assumption is certainly not strictly true, in particular for regions with depth discontinuities. As a trade-off, an exponentially decaying influence function $v(\sigma_v, d) = e^{-d^2/\sigma_v^2}$ has been chosen. It has parameter σ_v and decays exponentially with the distance d to the centre point where the correspondence and the

occupation likelihood has been measured. The occupation likelihood in the vicinity of the centre point where it have been measured is given by

$$p(s, d, \sigma_v) = v(\sigma_v, d)(p(s|\alpha, l) - \frac{1}{2}) + \frac{1}{2} \tag{5.60}$$

with the measured occupation probability $p(s|\alpha, l)$ for independent motion, and occupation probability $p(s, d, \sigma_v)$ for independent motion at distance d. In theory the influence function has a non-zero value even at infinite distance to the centre, in practise the influence is set to zero once the influence function falls below a certain threshold.

By using the influence function, the occupation probability map contains a small circle instead of a single pixel for each correspondence measurement. As long as the particle filter determines the position of the next measurement to be inside the circle, temporal integration is possible, even over more than two frames. However, the further the next particle is located from the centre point, the smaller is the temporal integration because of the decay of the influence function.

5.2.7 Algorithm Summary

Figure 5.25 illustrates the suggested framework for sequential Bayesian estimation of the occupation probability map for independent motion. It is explained top to bottom. A corner detector is run on the image resulting in the cornerness image and the positions for the pool of stable features (top right). Initialisation positions are generated by randomly sampling the thresholded cornerness function. Image correspondences are measured at the given positions from all three position pools using the next image. The preemptive RANSAC algorithm is used to estimate the relative camera motion parameters E with respect to the static background using only correspondences from the pool of stable features. The likelihood function is evaluated for each correspondence resulting in weights for each position and the occupation likelihood map. Now two feedback loops are implemented:

- The first consisting of the adopted particle filter algorithm determining the positions where correspondences will be measured in the next time step and
- the second consisting of the Bayesian integration of the occupation probability map.

Figure 5.26 shows the final probability occupation map for independent motion.

Challenging Correspondence Data Regions where repeatedly incorrect correspondence measurements occur are incorrectly marked with a high occupation probability. This happens, e.g. when a correspondence is measured at intersections of two high gradient edges at different depths. The lamp post in the left region of figure 5.26(c) is an example for such an "imaginary corner". A magnification of the challenging region around the lamp post is shown in figure 5.27 on the left.

Reflective surfaces violate the rigidity constraint on the static scene. When a part of the rigid background is observed by means of a reflection on a surface, it moves apparently

independent from the part of the background scene, which is observed directly. This phenomenon has geometric reasons. It can for example be observed on the right side of the figure 5.26(c). The parked car on the right side is very clean and reflects parts of the sky, a tree and a house. The correspondences measured on the reflecting surface of the car are not consistent with the relative motion from the directly observed background scene. These correspondences are hence marked as independently moving. Fortunately the reflection is not very clear, and hence not a lot of correspondences were measured on the car resulting in a rather low occupation probability for independent motion.

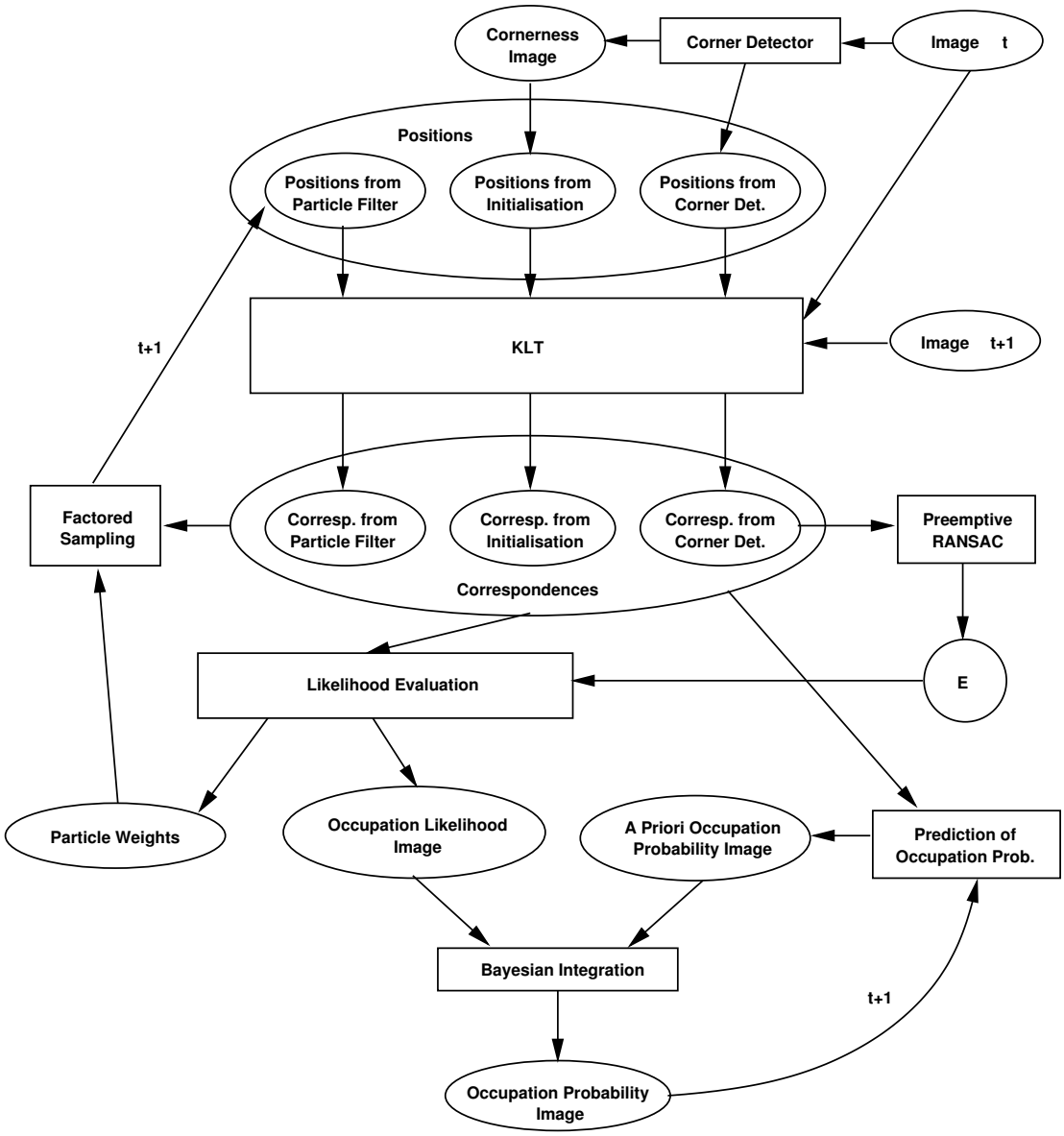


Figure 5.25: Schematic illustration of the Bayesian framework for the computation of the occupation probability map. See text for details.

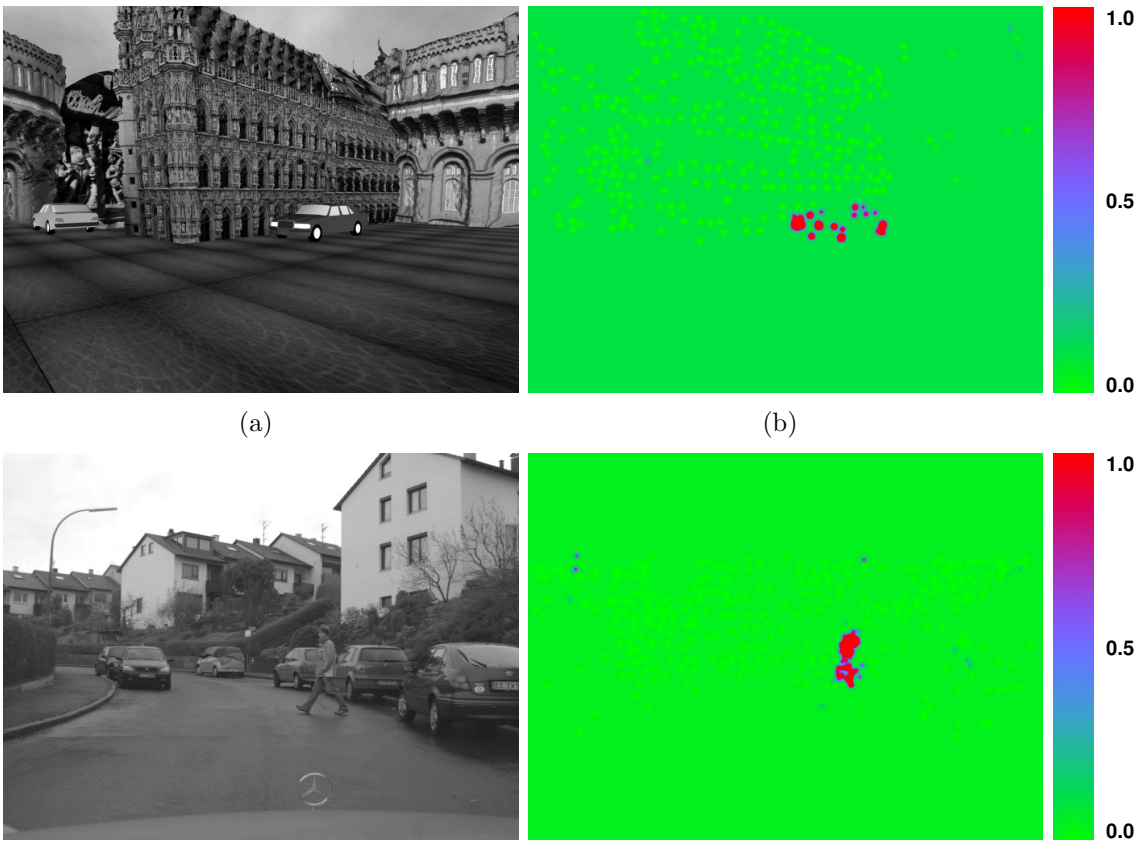


Figure 5.26: Final occupation probability maps for independent motion. The corresponding input image is shown on the left (a and c), the occupation probability map is shown on the right (b and d). Red indicates high and green indicates low probability for independent motion.

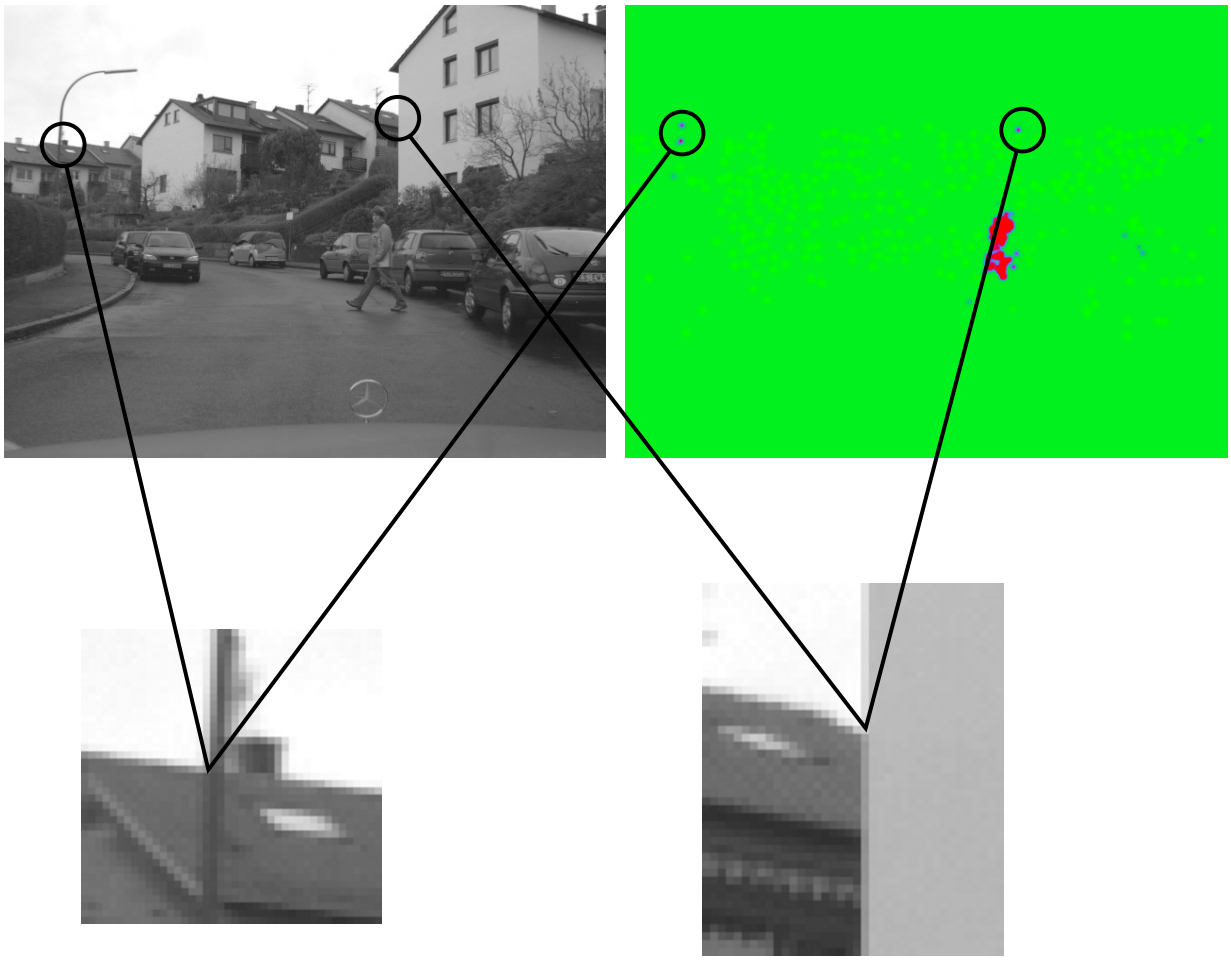


Figure 5.27: Magnification of two challenging regions for correspondence estimation. The lamp post at the left and the roof at the centre of the image 5.26(c). "Imaginary corners" are found at the intersection of high gradient edges at different depth leading to wrong image correspondences.

5.3 Clustering

Generally multiple correspondence measurements are made on each independently moving object. Clustering of measurements can on the one hand improve the reliability of the detection and on the other hand allow predictions about the "if" and "when" of a collision under certain assumptions.

After a brief review of existing methods, an algorithm connecting multiple independent detections of independently moving points to a single moving object is presented.

5.3.1 Previous Work

Figure 5.28: A frame of the well-known "flower garden" sequence (courtesy of Michael J. Black).

Clustering algorithms can be sorted in two major groups: One group is geometry-based and the other group is appearance-based. Appearance-based clustering exploit the appearance properties and hence works on the images directly. It can involve motion, texture, gradient and other clues. A vast number of research on appearance-based clustering is available, for example Cremers and Soatto (2003); Wong and Spetsakis (2004); Wong et al. (2004). Appearance-based clustering does not necessarily cluster points belonging to a rigid motion. For example the well-known "flower garden" sequence (figure 5.28) is usually segmented in for and background, even though the sequence is generated by a single rigid motion. Because of this undesired property, appearance-based approaches are not investigated in this work. 3D-geometry is usually not recovered, with the exception of Torr et al. (2001) who approximates the scene by planar object in 3D-space.

Geometry-based clustering algorithms on the other hand operate on correspondences and exploit their geometric coherence, for example Torr and Murray (1993); Costeira and Kanade (1998); Wolf and Shashua (2001); Fejes and Davis (1998). These approaches have been described in detail in chapter 3. Since all of these approaches are either slow, require knowledge of the complete sequence or do only work with rigid objects and thus exclude important object classes like pedestrians or cyclists, a different approach using spatial coherency is used in this work.

5.3.2 Clustering Using Spatial Coherence[6]

In order to transform the pixel-wise probability representation to few moving objects, a clustering algorithm is used. Thresholding the occupation probability map leads to a

[6]This section has been previously published in Woelk et al. (2005).

binary image. After running a dilation algorithm for connecting regions close to each other, a labelling algorithm is applied. The labelling algorithm identifies and labels connected regions by processing the image pixel per pixel and looking at the neighbours of the current pixel. The bounding boxes are extracted for each of the regions. Relevant characterisation is computed for every region, namely position, size, number of correspondences in it, mean probability and weighted mean motion. Position, size and number of correspondences N in every region can be extracted straight forward from the bounding boxes. The mean probability and the mean motion are calculated using only the correspondences in the region. Let $\boldsymbol{d}_i = \boldsymbol{x}_{i,t} - \boldsymbol{x}_{i,t-1}$ denote the motion given by a point correspondence between $\boldsymbol{x}_{i,t-1}$ and $\boldsymbol{x}_{i,t}$ and p_i denote the corresponding entry in the probability image from time t at position $\boldsymbol{x}_{i,t}$. The weighted mean motion $\overline{\boldsymbol{d}}$ is calculated by

$$\overline{\boldsymbol{d}} = \frac{\sum_{i=1}^{N} p_i \boldsymbol{d}_i}{\sum_{i=1}^{N} p_i} \tag{5.61}$$

and the mean probability $\overline{p}$ is calculated by

$$\overline{p} = \frac{\sum_{i=1}^{N} p_i}{N} \tag{5.62}$$

Kalman Tracking of Clusters: The idea behind this algorithm is that a separate Kalman filter (Kalman, 1960) is assigned to every moving object. The 2D-position and the velocity of the moving object in the image are used as system state in the Kalman filters. The list of regions from the clustering algorithm is used as measurements. Since multiple measurements and Kalman filters might be involved, each region must be assigned to a specific Kalman filter instance from the list of all Kalman filters. This list is obtained in the following manner in every time step:

1. The prediction step is performed for every Kalman filter in the list.
2. For every Kalman filter the closest region to its current position is searched. If this region is closer than either three times the Kalman filter variance or closer than a certain distance (e.g. 7 pixel), it is used as measurement for the update step. If the region is at a further distance, no update step is performed.
3. Kalman filters which have not been updated in the first three subsequent time steps of their lifetime are removed. Kalman filters which have not been updated for the last 3 cycles are also deleted.
4. A new Kalman filter is added to the list for every region which has not been assigned to an already existing Kalman filter.
5. If two Kalman filters adopt positions closer than a certain distance (e.g. 7 pixels) to each other, the younger Kalman filter is deleted.

The details of the Kalman filter implementation are presented in the subsequent paragraphs.

The Kalman filter (Kalman, 1960) is a set of mathematical equations that provides an efficient solution to the discrete-data linear filtering problem. For the algorithm presented here, the notation convention of Welch and Bishop (2001) is used, where the basic filter equations can also be found.

System Description: Since only image coordinates (no 3D-information) for independently moving clusters are available, the cluster centres are estimated in image coordinates. Approximating the system by a simple point mass model (as often used in physics) the x- and y-motion are independent, yielding two independent Kalman filters per cluster. The main advantage of two independent Kalman filters per cluster is the computational speed. The following states are estimated

$$\boldsymbol{x}_x = \begin{bmatrix} x & v_x & a_x \end{bmatrix}^T, \qquad \boldsymbol{x}_y = \begin{bmatrix} y & v_y & a_y \end{bmatrix}^T \tag{5.63}$$

where x, v_x, and a_x are position, velocity, and acceleration of the cluster (analogous for y). Measurements are the cluster positions and the average optical flow of the contributing flow vectors as cluster velocity.

$$\boldsymbol{z}_x = \begin{bmatrix} x_{\text{cluster}} \\ v_{x,\text{cluster}} \end{bmatrix}, \qquad \boldsymbol{z}_y = \begin{bmatrix} y_{\text{cluster}} \\ v_{y,\text{cluster}} \end{bmatrix} \tag{5.64}$$

For the simple filter used in this work, the control vector describing external system input is 0.

Process description: The connection between measurements and system variables is straightforward:

$$\dot{x} = v_x, \qquad \dot{v_x} = a_x, \qquad \dot{a_x} = 0, \tag{5.65}$$

The same equations apply for the y direction.

Measurement Description: The measurement update step incorporates the new measurements. The measurements x_{cluster} and $v_{x,\text{cluster}}$ can be easily expressed in terms of state variables:

$$h(\vec{x})_x = \begin{bmatrix} x_{\text{cluster}} \\ v_{x,\text{cluster}} \end{bmatrix} = \begin{bmatrix} x \\ v_x \end{bmatrix}. \tag{5.66}$$

Again, the same equations apply for the y direction. These equations and their derivatives w.r.t. the state vector are the input to the Kalman filter. Time update is performed at each time step. Measurement update steps are performed whenever new measurements are available.

Measurement variances are estimated to be 7 pixels for the cluster position due to the uncertainties in detecting different parts of the object in different frames. The uncertainty

of the cluster velocity is obtained from optical flow measurements with an conservatively estimated uncertainty of 2 pixels. The system uncertainties are estimated with 2 pixels, and the initial covariances are conservatively set to 7 pixels and 7 pixels/s, respectively. The acceleration state is not considered at this point, but could be estimated if needed.

5.3.3 Estimation of Independent Motion

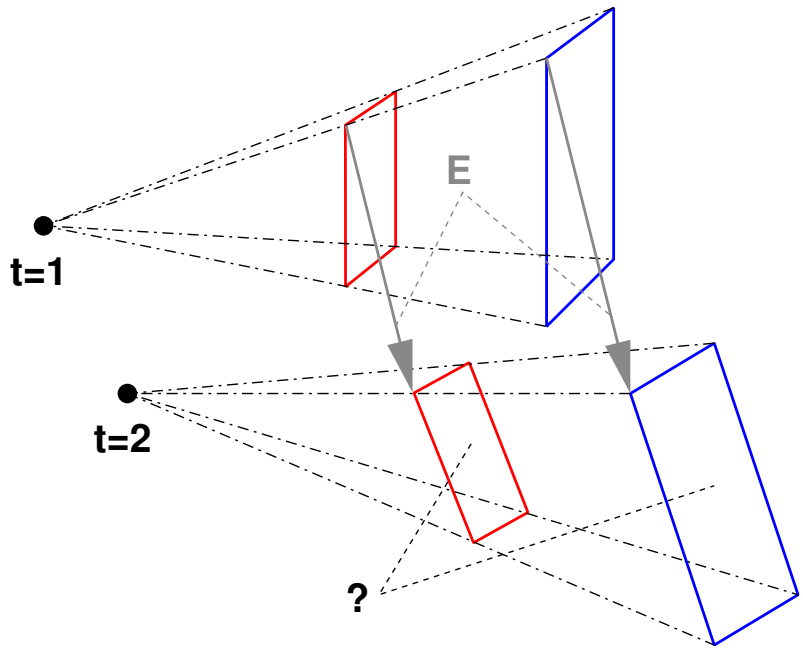


Figure 5.29: Impossibility of trajectory extraction from image point correspondences in the general case: A moving camera (black) with known motion parameters sees the red object at times $t = 1$ and $t = 2$. When the object is rigid, the essential matrix E, describing the relative motion between the red object and the camera, can be extracted from the image point correspondences. However, the essential matrix does not capture the magnitude of the relative translation, but only the direction of the relative translation. The blue object, which differs from the red object only by its scale, moves according to the same essential matrix and both objects result in the same image points. Using only the images, it is hence impossible to determine the scale of the viewed object, and without the correct scale, the extraction of the trajectory of the object remains impossible. This ambiguity subsists when more images are available.

It is generally impossible to compute the trajectory of a single moving object using a sequence of images from a single camera without further knowledge or assumptions (Avidan and Shashua, 2000; Han and Kanade, 2000, 2003). This is illustrated in figure 5.29. The red and the blue objects both result in exactly the same images, and hence size and velocity of the object cannot be determined. This impossibility is independent of the number of views. Note that the direction of the motion vector can be recovered when the object is rigid and when enough point correspondences on the object are measured, i.e. when an

essential matrix for the relative motion between object and camera can be estimated: The relative motion direction is given by the epipole.

The above stated problem is equivalent to the well-known scale problem of scene reconstruction using a calibrated camera: Without further knowledge about the scene or camera motion, structure and motion algorithms reconstruct scene and camera path only up to scale. Only the relative motion of object and camera is of concern for the reconstruction. When both, the object and the camera, move in a static scene, this relative motion is unknown, even if the camera motion is known with respect to the static scene, e.g. from inertial sensors. Hence, without further knowledge or assumptions about either scene geometry or camera motion, the geometry of scene and camera positions can only be reconstructed up to scale.

Common assumptions used to circumvent this problem are for example the object size or a common ground plane (Sturm, 2002). In certain special geometric configurations occlusion information can be used for reasoning about the object position relative to the static scene (Ogale et al., 2005). Avidan and Shashua (2000) investigate the possibility to recover the trajectory when the shape of the trajectory is known. Solutions for straight line trajectories and trajectories from the family of planar conic intersections are presented. This approach and in particular the solution for straight lines seems promising at first glance, however, when taking a closer look it turns out that the solution degenerates when the motion of the camera is also linear and when both trajectories live in the surface of the same ruled quadric. A common example for this situation is given when the camera and the point trajectories are coplanar. In this case the nullspace to the problem describes the common plane, and thus every line on the plane represents a valid solution. The authors state that using multiple points moving on parallel lines would not contribute significantly to the solution when the points are close to each other.

Even though the question about the exact location of the independently moving object in space cannot be answered in general, two important statements about the relation between observer and object can be made: Under certain assumptions it can be predicted *if* the camera and the object will collide and *when* the collision will occur in this case. This is explained in detail in the next sections.

5.4 Collision Detection

A very simple collision test is presented first. Afterwards, estimation of the time to contact is presented.

5.4.1 Constant Bearing

A collision detection which is known since centuries as the sailor's test for collision (or constant bearing) is described next. It is illustrated in figure 5.30. If the angle α, under which an object B is seen from an object A, remains constant over time ($\alpha = \alpha'$) and the apparent object size is growing, then a collision will take place. This is equivalent to the

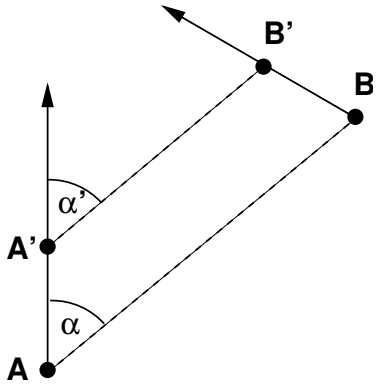


Figure 5.30: Sailor's test for collision: If the angle α, under which an object B is seen from an object A, moving itself, remains constant over time ($\alpha = \alpha'$), a collision will take place. This is also called a constant bearing.

fact that the FOE of the relative motion between the camera and the moving object lies within the growing picture of an object in the image. Note that the prediction is only valid with constant velocities and linear movements. An interesting fact is that the prediction is correct, even if the objects are constantly accelerating or decelerating in the direction of their translation vector as long as the objects do not come to a complete halt before the point of impact.

5.4.2 Time to Contact

Objects on a collision course create a characteristic looming spatio-temporal expanding pattern on the observer's retina (Sekuler et al., 2004). If the object is static and the observer is moving or if the observer is static and the object is moving, just depends on the choice of the reference frame as long as the motions are not accelerated. Animals of such different species as fiddler crabs, chicks, monkeys as well as newborn human infants try to avoid artificially created expanding patterns (Sekuler et al., 2004). The reaction of newborn children who have never encountered this stimulus before indicates that their reaction is based on instinct rather than learnt behaviour. With constant motions, the time until an object reaches the observer is given by the ratio between the distance and the relative directional velocity. However, information about distance and relative velocity is generally not available to the observer, and in the 1980s the exploitation of the dynamic changes of the images of the object was suggested. The time to contact τ was originally introduced by Lee in 1980 as the ratio between the size of the image of the object and the rate of change of the image size (Sekuler et al., 2004). τ denotes the expected remaining time until an object reaches the observer. The time to contact (TTC) is positive when two objects approach each other, and negative when the two objects are retreating. Time to contact has also been denoted as time to collision, time to crash or time to impact (van Leeuwen, 2002; Meyer and Bouthemy, 1992; Colombo and del Bimbo, 1999).

The exact definition of the time when an object reaches the observer differs slightly in the literature. It is always defined as the time when the trajectory of the object intersects a plane through the centre of projection of the camera. The normal to the plane and with it its orientation is given either by the optical axis, by the motion vector of the camera or by the difference vector between object and camera (van Leeuwen, 2002; Colombo and del Bimbo, 1999).

Time to contact estimates can be based on dense optical flow fields (Meyer and Bouthemy, 1992), sparse point correspondences, areas (Cipolla and Blake, 1992), contours (Colombo and del Bimbo, 1999; Cipolla and Blake, 1992) and surfaces. Two approaches using sparse point correspondences are explained next.

1. Derivative

Given the coordinates of 3D-point $\boldsymbol{X} = (X, Y, Z)^T$ and a simple perspective camera model with focal length f, the 2D-coordinates of the projection of the 3D-point onto the image plane $\boldsymbol{x} = (x, y)^T$ are determined by

$$fX - xZ = 0 \qquad \text{and} \qquad fY - yZ = 0 \tag{5.67}$$

Computing the second derivatives with respect to time under the assumption of purely translational motions leads to

$$f\ddot{X} - \dot{x}\dot{Z} - x\ddot{Z} - \ddot{x}Z - \dot{x}\dot{Z} = 0 \qquad \text{and} \qquad f\ddot{Y} - \dot{y}\dot{Z} - y\ddot{Z} - \ddot{y}Z - \dot{y}\dot{Z} = 0 \tag{5.68}$$

where first temporal derivatives are denoted by a dot, e.g. $\dot{x} = \frac{dx}{dt}$, and second temporal derivatives denoted by two dots, e.g. $\ddot{x} = \frac{d^2x}{dt^2}$. Assuming constant 3D-velocity, equations 5.68 simplify to

$$-\ddot{x}Z - 2\dot{x}\dot{Z} = 0 \qquad \text{and} \qquad -\ddot{y}Z - 2\dot{y}\dot{Z} = 0 \tag{5.69}$$

Using equations 5.69, the time to collision τ is given by

$$\tau = \frac{Z}{\dot{Z}} = -2\frac{\dot{x}}{\ddot{x}} = -2\frac{\dot{y}}{\ddot{y}} \tag{5.70}$$

Note that only the ratio of image velocity (optical flow) and image acceleration (i.e. optical acceleration) is needed to compute the time to contact. Knowledge about focal length or scene geometry is not necessary. Investigations about the accuracy of TTC estimates are presented next.

Assuming a certain accuracy $\sigma_{\dot{x}}$ of the correspondence estimation process, the accuracy of the optical acceleration is given by

$$\sigma_{\ddot{x}}^2 = 2\sigma_{\dot{x}}^2 \tag{5.71}$$

Using linear error propagation, the variance of the time to contact can be approximated by

$$\sigma_\tau^2 \approx \sigma_{\dot{x}}^2 \left[\left(\frac{\partial t_c}{\partial \dot{x}} \right)^2 + 2 \left(\frac{\partial t_c}{\partial \ddot{x}} \right)^2 \right] = \sigma_{\dot{x}}^2 \left[\left(\frac{2}{\ddot{x}} \right)^2 + 2 \left(\frac{-2\dot{x}}{\ddot{x}^2} \right)^2 \right] \tag{5.72}$$

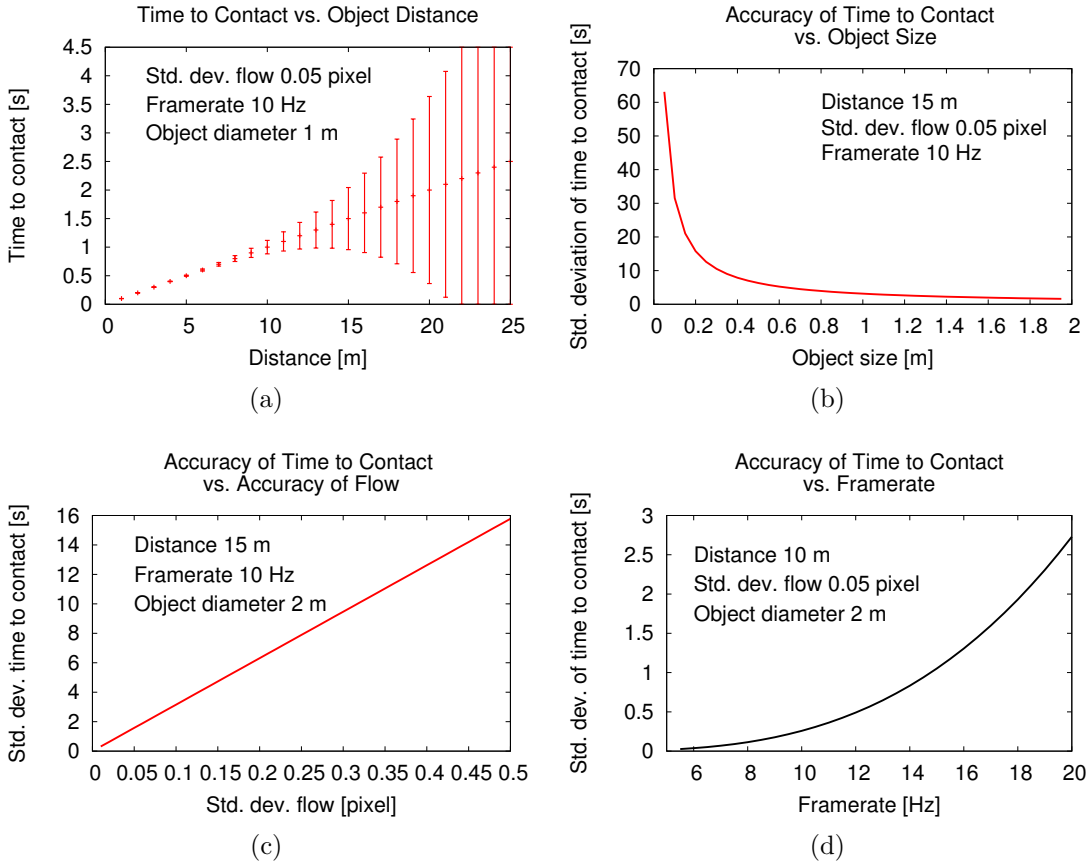


Figure 5.31: Theoretic bound on accuracy for time to contact estimation. (a) The TTC and the associated standard deviation are plotted vs. the distance of the objects from the camera. (b) The standard deviation of time to contact is plotted against the object size. (c) Standard deviation of TTC vs. accuracy of flow measurement and (d) standard deviation of TTC vs. frame rate. See text for details.

Figure 5.31 illustrates theoretic accuracy σ_τ of the TTC estimate depending on different variables.

The estimates of τ and its standard deviation σ_τ are plotted versus the distance of the object from the camera in figure 5.31(a). The camera with a focal length identical to the camera in the car (≈ 840 pixel) translates with 30 m/s along its optical axis while capturing images at 10 Hz. The camera is initially located at the origin and the point in question is located at $(0.5, 0, 30)$. The standard deviation is computed using the assumption that the correspondences can be measured with an accuracy of 0.05 pixel (i.e. $\sigma_{\dot{x}} = 0.05$ pixel).

The standard deviation of TTC is plotted versus the object diameter in figure 5.31(b). The initial distance of the object from the camera is 15 m, the point in question is located at the border of the object, and frame rate and accuracy of flow measurement are again 10 Hz and 0.05 pixel.

The standard deviation of TTC is plotted versus the accuracy of the flow measurements

in figure 5.31(c). The frame rate is again 10 Hz, the object size is fixed at 2 m and the initial distance between object and camera is 15 m. The standard deviation of TTC is plotted versus the frame rate in figure 5.31(d). The object size is again fixed at 2 m, the initial distance between object and camera is 10 m, and the accuracy of the flow measurements is 0.05 pixel.

To summarise: Big object images, large displacements and high accuracy of the correspondence estimate favour the accuracy of the TTC estimate. The derivation of τ using a different approach is described next.

2. Global

Given time dependent coordinates of a 3D-point $\boldsymbol{X} = (X + t\dot{X}, Y + t\dot{Y}, Z + t\dot{Z})^T$ and a simple perspective camera model with focal length f, the 2D-coordinates of the projection of the 3D-point onto the image plane $\boldsymbol{x} = (x, y)^T$ are determined by

$$f(X + t\dot{X}) - x(Z + t\dot{Z}) = 0 \qquad \text{and} \qquad f(Y + t\dot{Y}) - y(Z + t\dot{Z}) = 0 \tag{5.73}$$

Dividing by $\dot{Z}$ results in an equation in the three unknowns $\frac{X}{\dot{Z}}$, $\frac{\dot{X}}{\dot{Z}}$ and $\frac{Z}{\dot{Z}}$

$$f\frac{X}{\dot{Z}} + t\frac{\dot{X}}{\dot{Z}} - x\frac{Z}{\dot{Z}} - x = 0 \qquad f\frac{Y}{\dot{Z}} + t\frac{\dot{Y}}{\dot{Z}} - y\frac{Z}{\dot{Z}} - y = 0 \tag{5.74}$$

with time dependent projections of the 3D-point $x = x(t) = x_t$ and $y = y(t) = y_t$. Assuming again constant velocity, a linear equation system can be constructed using measurements at three different times $t = -1, 0, 1$. Eliminating $\frac{X}{\dot{Z}}$ and $\frac{\dot{X}}{\dot{Z}}$, the TTC is given by

$$\tau = \frac{Z}{\dot{Z}} = \frac{x_1 - x_{-1}}{x_1 - 2x_0 + x_{-1}} = \frac{y_1 - y_{-1}}{y_1 - 2y_0 + y_{-1}} \tag{5.75}$$

Assuming a certain accuracy $\sigma_{\dot{x}}$ of the correspondence estimation process, the uncertainty of the point x_0 is zero and the standard deviation of the points x_{-1} and x_1 is given by $\sigma_{\dot{x}}$. Using again linear error propagation, the variance σ_τ^2 of the TTC estimate is given by

$$\sigma_\tau^2 \approx \sigma_{\dot{x}}^2 \left[\Big(\frac{(x_1 - 2x_0 + x_{-1}) - (x_1 - x_{-1})}{(x_1 - 2x_0 + x_{-1})^2} \Big)^2 + \Big(\frac{-(x_1 - 2x_0 + x_{-1}) + (x_1 - x_{-1})}{(x_1 - 2x_0 + x_{-1})^2} \Big)^2 \right] \tag{5.76}$$

Obviously both approximations (equation 5.72 and 5.76) of the variance of the TTC exhibit the same qualitative behaviour. The influence of optical acceleration is inverse quadratic, and the influence of the optical flow is linear.

Relaxing the Constraints

The TTC estimate implies purely translational relative motion between camera and moving object. In general, the relative motion includes a rotational part. When the independently

moving object is rigid and when enough point correspondences are measured, the essential matrix for the relative motion between object and camera can be estimated. The relative orientation change can be extracted from the essential matrix and the correspondences can be de-rotated resulting in the translational part of the flow. This translational part can be used for time to contact estimation.

5.5 System Integration

First a comprehensive system description is given, afterwards the results from the integrated system are presented. Finally, the live demonstration of the system within the final presentation of the INVENT project (German: Intelligenter Verkehr und Nutzergerechte Technik - intelligent traffic and user friendly technology) is described.

5.5.1 System Description

(a) (b)

Figure 5.32: The UTA (Urban Traffic Assistant) demonstrator from Daimler Chrysler AG (a). Only one of the two PTU mounted cameras is used in this work. A closeup view of the pan-tilt unit mounted camera (b) (courtesy of S. Gehrig).

The system has been tested using the Urban Traffic Assistant (UTA) demonstrator from Daimler Chrysler AG. The setup of UTA includes a digital camera mounted on a pan-tilt unit (PTU), GPS, map data, internal velocity and yaw rate sensors, etc. For a detailed description of UTA refer to (Gehrig et al., 2003). Pictures of the cameras used in UTA are shown in figure 5.32. The fusion of GPS and map data can be used to announce the geometry of an approaching intersection to the vision system. The camera then focuses on the intersection. Using the known egomotion of the camera, independently moving objects are detected and the driver's attention can be directed towards them. A schematic illustration of the system topology is given in figure 5.33. In the following the important components of the demonstrator are described.

Inertial Sensors

The Demonstrator is equipped with standard inertial sensors:

Velocity UTA is equipped with a standard digital speed sensor which functions by measuring distance units per time interval. A distance unit is $\frac{1}{48}$th of the diameter of the

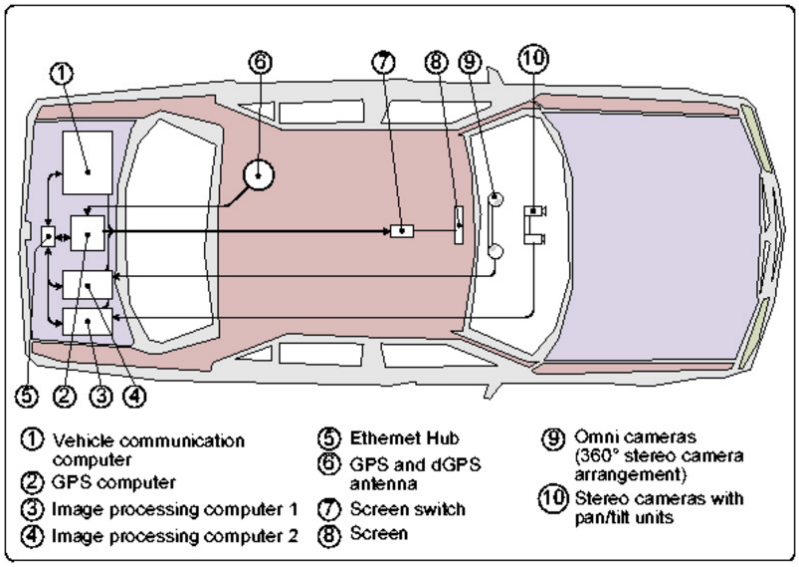


Figure 5.33: Schematic illustration of the systems used in the UTA demonstrator from DaimlerChrysler AG. Only one camera of the stereo system (10) is used in this thesis (courtesy of Gehrig et al. (2003)).

wheels resulting in approximately 4 cm when the perimeter of the wheel is assumed to be 2 m. The time interval is not exactly known, however a value of 20 ms (10 Hz) seems plausible[7]. From these values the upper bound for the error of the speed sensor is estimated as ± 0.4 m/s. Assuming normal distribution of the measurements, this corresponds to a standard deviation of ≈ 0.13 m/s.

Longitudinal Acceleration The longitudinal acceleration is not measured but computed by the integrated control unit as the derivative of the velocity. Only an internally filtered longitudinal acceleration value is available. When simple numeric derivation would be used for the computation of the longitudinal acceleration, its upper error bound would be given by twice the accuracy of the velocity measurement divided by the time interval resulting in 0.08 m/s^2 with the above assumptions. However, filtering improves the accuracy (at least in steady state conditions) and hence the standard deviation is assumed to be 0.01 m/s^2.

Lateral Acceleration Standard lateral accelerometers are present in the demonstrator. This sensors are also used for the ESP (German: Elektronisches Stabilitätsprogramm) system. The accuracy of the lateral acceleration sensor is approximated by 0.01 m/s^2.

Steering Angle The steering angle is measured at the steering wheel. The ratio between steering wheel angle and steering angle is constant and can hence be directly converted. UTA is equipped with a high performance incremental sensor in addition

[7] A clock frequency of 10 Hz results in a minimum possible velocity measurement of ≈ 0.4 m/s ≈ 1.5 km/h.

to the standard steering wheel angle sensor. Its zero position is however manually determined resulting in a possible bias in the measurements. The accuracy of the steering angle is given by $\approx 0.14°$.

The inertial sensors are available via the vehicle computer (see Figure 5.33 (1)).

Yaw rate sensor

A yaw rate sensor of type DRS-MMS 1.0 from Bosch (Fetzer, 1998) is installed in the UTA demonstrator. It is assembled on an attenuation board. Its resolution is specified as $0.3°/s$. The offset specifications are separately given for the first 10 minutes of operation and after 10 minutes of operation:

	$t < 10\text{min}$	$t > 10\text{min}$
Offset	$\pm 2°/s$	$\pm 3°/s$
Offset change	$\pm 0.1°/s/\text{min}$	$\pm 0.2°/s/\text{min}$

The yaw rate sensor is available via the vehicle computer (see Figure 5.33 (1)).

DGPS

A differential GPS system (figure 5.33 (6)) is integrated in the vehicle. Its measurements are available with approximately 1 Hertz. The raw DGPS measurements are fused with the measurements from the other inertial sensors of the car using a Kalman filter, and results of this fusion are available at higher frequencies (Gern, 2000).

Digital Camera

A digital camera (figure 5.33 (10)) of type PixelFly from PCO is used in the demonstrator. It offers a dynamic range of 65.5 dB which can be captured in 12 bit. The ICX074AL charge coupled device (CCD) chip from Sony has a resolution of 640×480 pixels. A Cinegon 1.4/8 lens from Schneider Kreuznach with a nominal focal length of 8.1 mm is used in the experiments. The calibration of the camera as described in section 2.2.4 resulted in the internal camera parameters given in table 5.3.

Pan-Tilt Unit

The camera is mounted on a pan-tilt unit (figure 5.33 (10)) PTU-46-70 from Directed Perception Inc. It has a maximum speed of 60°/s and a resolution of 0.012857°. The PTU is rigidly coupled with the dashboard of the car. It is connected to the controlling computer using a standard RS232 interface.

parameter	symbol	value
focal length	f	839.435 px
skew	s	0.0
aspect ratio	a	1.0054
principal point	c_x	319.53 px
	c_y	244.84 px
radial lens distortion	κ_1	−0.0889658
	κ_2	0.0194259
tangential lens distortion	κ_4	0.0015841
	κ_5	0.0002699

Table 5.3: Internal calibration parameters for the camera lens combination in the UTA demonstrator.

Computers

A standard desktop PC (figure 5.33 (3)) is fitted into the boot of the demonstrator. It is equipped with a 3.2 GHz Pentium 4 CPU with hyper threading technology and 2 MB of cache. This PC is used for the image processing computation loop. It controls the PTU and grabs the images from the camera. The car inertial sensors are connected to an integrated, embedded vehicle communication computer (figure 5.33 (1)) whose sole purpose is the communication with the electronic system of the car using the CAN bus. It reads out the inertial sensors and can be used to trigger for example the horn or the brake. Both computers are connected via standard ethernet. For the integration of the DGPS sensor, an extra GPS computer (figure 5.33 (2)) is used. It integrates the car inertial sensors with the DGPS measurements and provides the resulting position for further usage.

5.5.2 Live Demonstration

In this section, results from this early prototype are presented. The early prototype mainly differed in three ways from the final system described in this thesis:

Temporal Integration: A simple heuristic temporal integration was used in the prototype. It basically consisted of a spatial and a temporal low pass filter.

Robust Algorithm: An ordinary RANSAC algorithm instead of the preemptive RANSAC was used in the integrated system.

Car Inertial Sensors: The car inertial sensors are used to compute an initial guess for the camera motion. This guess is the first solution which is evaluated in RANSAC algorithm.

The early prototype delivered promising results despite the simple temporal integration. In the majority of the frames, the prediction of the camera motion by the car inertial sensors

was good and no further robust computation of the egomotion was necessary. When the prediction of the camera motion was less accurate, for example at big longitudinal accelerations, the system slowed notably down due to the longer computation time of the ordinary RANSAC.

Live Demonstration

The integrated detection system in the UTA demonstrator from DaimlerChrysler AG was chosen to be part of the final presentation of the results of the German INVENT project. It represented the FUE subproject (German: Fahrumgebungserfassung und Interpretation – English: car environment capture and interpretation). The final presentation took place on the private test area of MAN company in Dachau in April 2005. A typical traffic scenario was demonstrated where an inattentive pedestrian crosses the road on which the demonstrator travels. The pedestrian is temporarily partially occluded by a car parked along the curbside. The system detects and tracks the pedestrian. For demonstration purposes the detection system triggered the horn and the electronic brake of the demonstrator once a stable cluster was tracked over 7 frames and a collision was detected. *The system was presented in over 200 live runs* to representatives from politics and industry including the German minister for education and science at that time Mrs. Edelgard Bulmahn. During these runs, no false alarms and no missed detections occurred.

5.5.3 Real World Sequences

In the following, screenshots from the system on three real world sequences are presented. The ability of the system to react to different classes of objects such as cars, pedestrians and bicycles is demonstrated exemplary using these sequences.

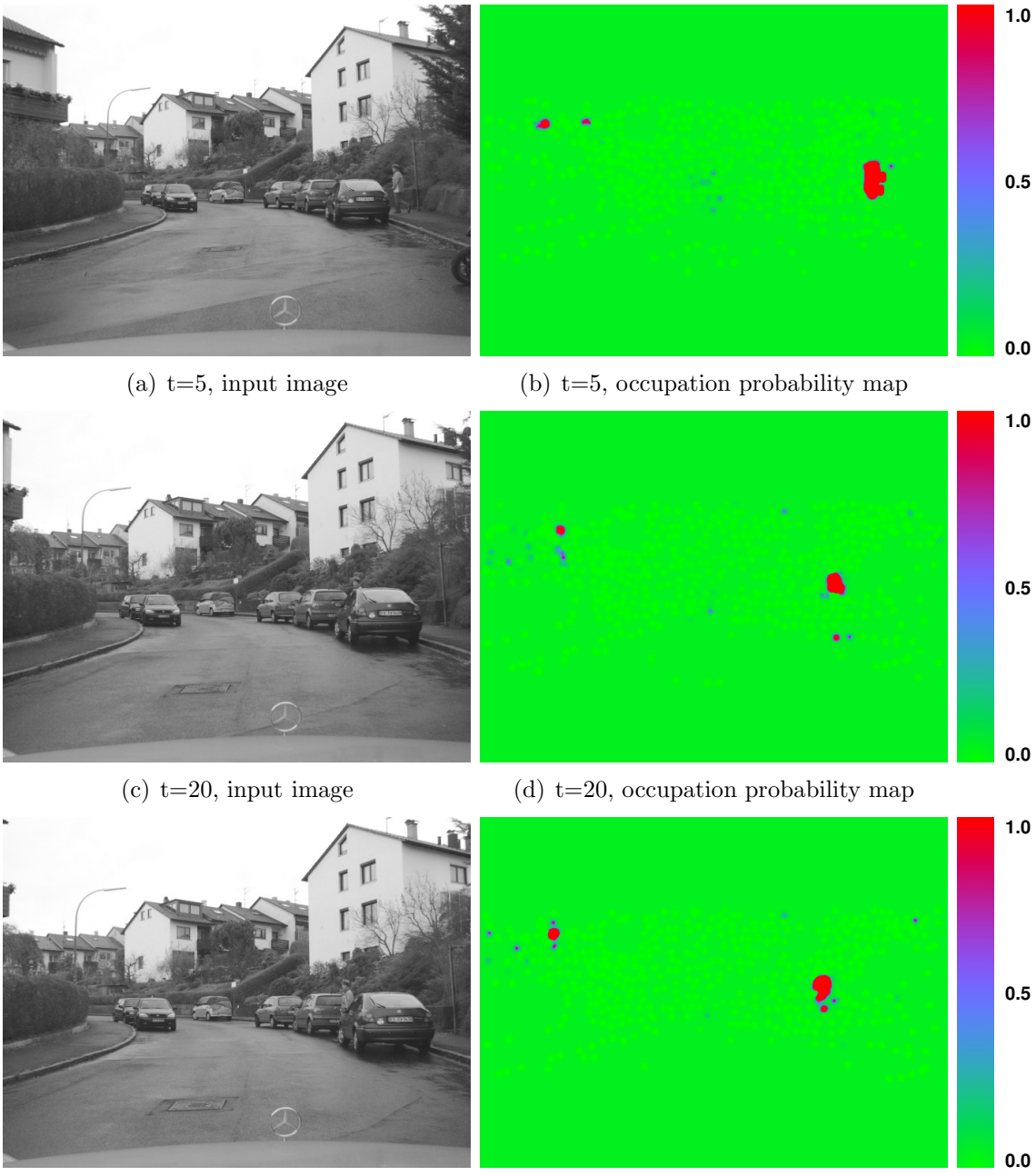
Figures 5.34 and 5.35 show the results on a sequence with a pedestrian crossing the road. Four selected original images of the sequence are shown in the left column of figure 5.34 and in 5.35 (a). The images in the right column are the corresponding occupation probability maps. The magnifications in figure 5.34 (c-f) illustrate the bounding box of a tracked cluster of high occupation probability pixels for the selected images.

Figures 5.36 and 5.37 show the results of the system on an intersection sequence with an approaching car. Again four selected original images are shown in the left column of 5.36 and in 5.37 (a). The corresponding images in the right column illustrate the associated occupation probability maps. Figure 5.37 (c-f) shows the resulting bounding box of the cluster of high occupation probability pixels which has been tracked with a Kalman filter.

Finally the images in figs. 5.38 and 5.39 show the results of the system on a real world sequence with an intersecting cyclist. Four selected original images are shown in the left column of 5.38 and in 5.39 (a). The images in the right column show the associated occupation probability maps. Figure 5.39 (c-f) demonstrates again the bounding box of a cluster of high occupation probability. The cluster has been tracked using a Kalman filter as described in section 5.3.2.

Timing

The mean computation time of the prototype in the demonstrator was 80 ± 13 ms per frame for the real world sequence with the intersecting car (Figs. 5.36 and 5.37). These timings were measured on a standard 3.0 GHz Pentium IV PC with an overall number of 500 correspondence measurements. The image size was originally 640×480 pixels, but all computations are conducted using downsampled images of 320×240 pixels. The optical flow was computed on a pyramid of size 2 and a support window size of 7×7 pixels was used for feature point tracking.

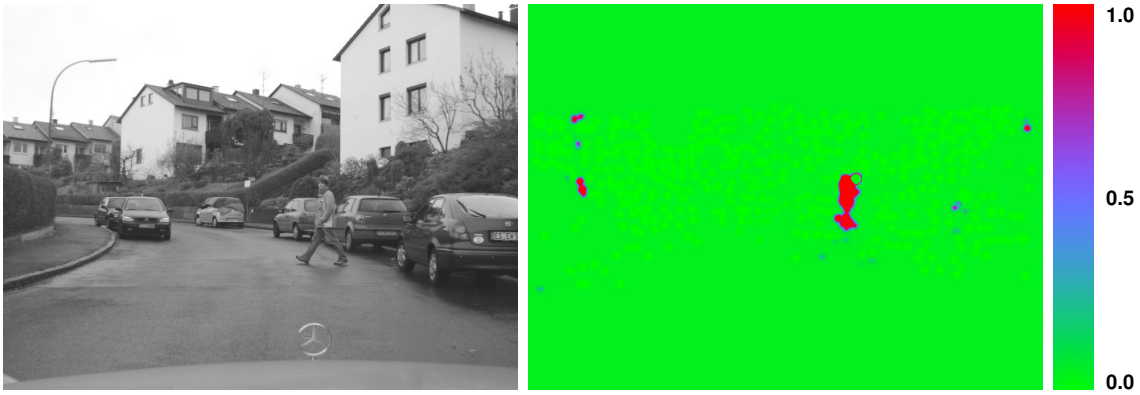


(a) t=5, input image (b) t=5, occupation probability map

(c) t=20, input image (d) t=20, occupation probability map

(e) t=23, input image (f) t=23, occupation probability map

Figure 5.34: Real world sequence with crossing pedestrian (part 1). The original images are shown in the left column (a, c, e) and the occupation probability maps are shown in the right column (b, d, f).

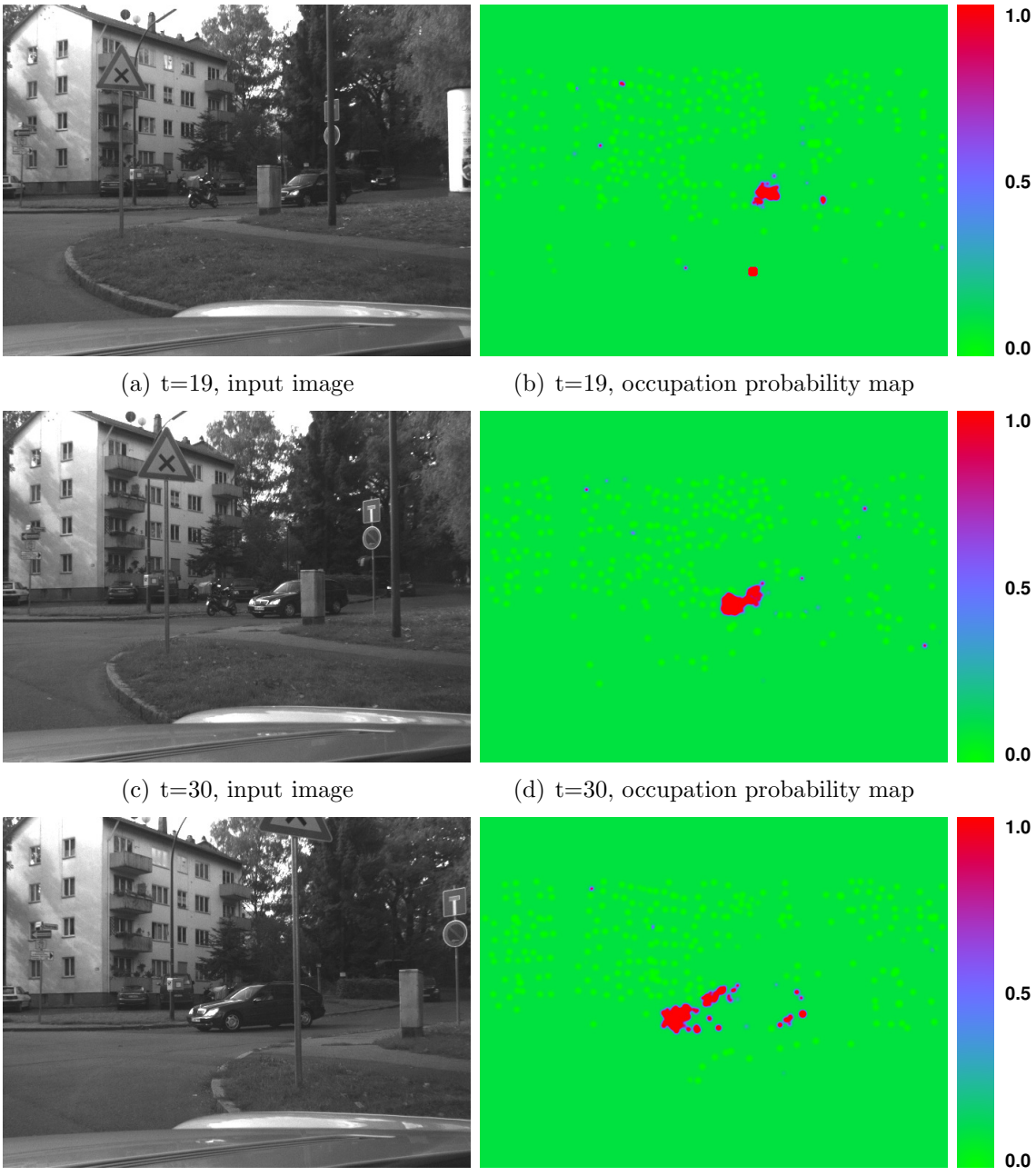


(a) t=37, input image (b) t=37, occupation probability map

(c) t=5, magnification of point cluster (d) t=20, magnification of point cluster

(e) t=23, magnification of point cluster (f) t=37, magnification of point cluster

Figure 5.35: Real world sequence with crossing pedestrian (part 2). The last original image is shown at the top left (a) and the associated occupation probability map is shown at the top right column (b). Magnifications are shown in images (c-f) to illustrate the clustering and Kalman filtering of the clusters. A warning is issued in image (e).

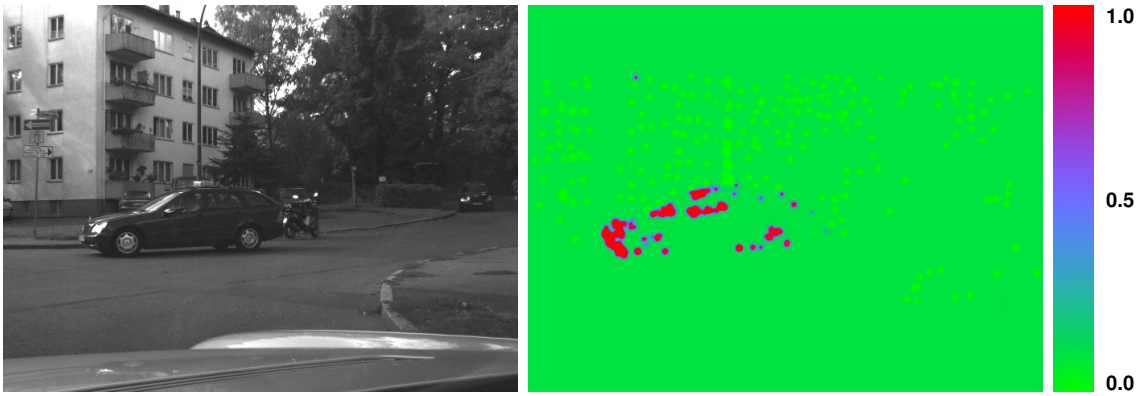


(a) t=19, input image (b) t=19, occupation probability map

(c) t=30, input image (d) t=30, occupation probability map

(e) t=51, input image (f) t=51, occupation probability map

Figure 5.36: Real world intersection sequence with crossing car (part 1). The original images are shown in the left column (a, c, e) and the occupation probability maps are shown in the right column (b, d, f).

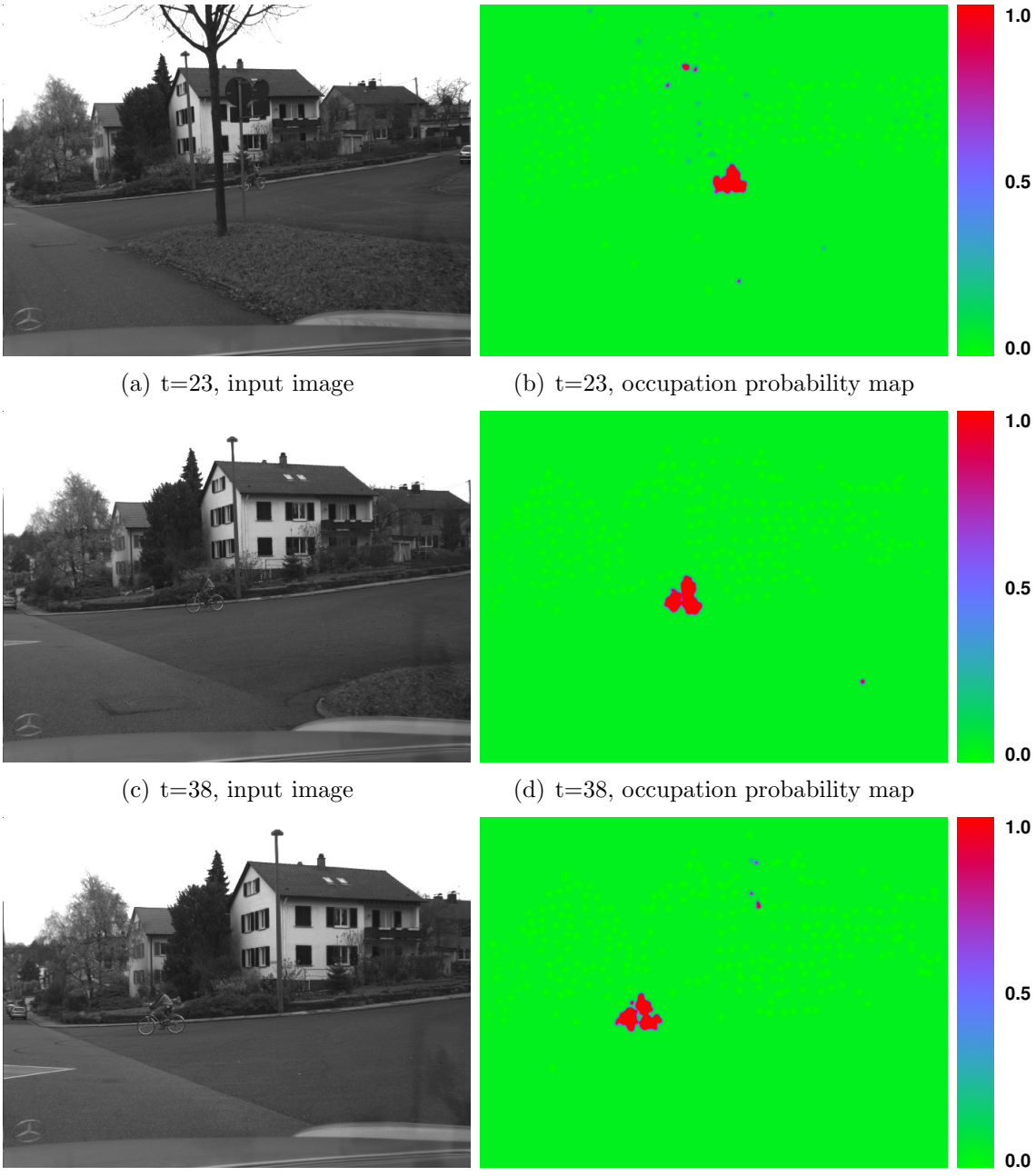


(a) t=73, input image (b) t=73, occupation probability map

(c) t=19, magnification of point cluster (d) t=30, magnification of point cluster

(e) t=51, magnification of point cluster (f) t=73, magnification of point cluster

Figure 5.37: Real world intersection sequence with crossing car (part 2). The last original image is shown at the top left (a) and the associated occupation probability map is shown at the top right column (b). Magnifications are shown in images (c-f) to illustrate the clustering and Kalman filtering of the clusters. A warning is issued in image (d).

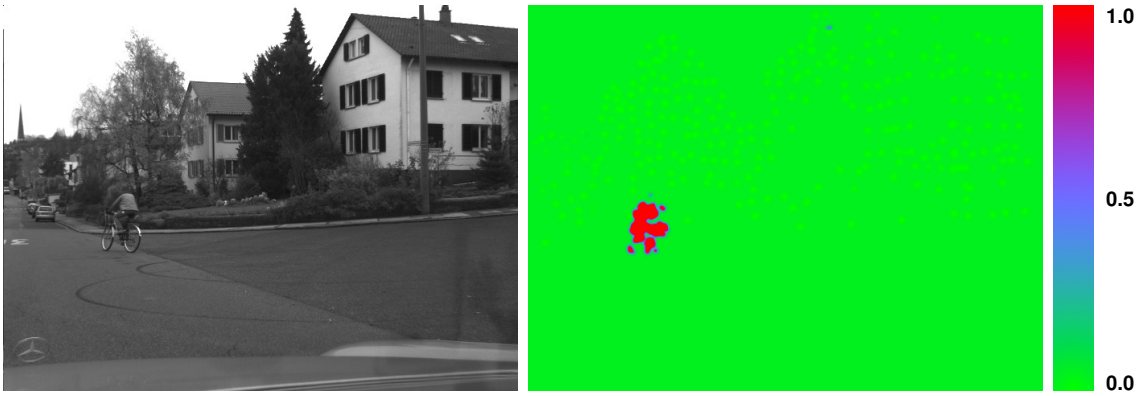


(a) t=23, input image (b) t=23, occupation probability map

(c) t=38, input image (d) t=38, occupation probability map

(e) t=48, input image (f) t=48, occupation probability map

Figure 5.38: Real world intersection sequence with crossing cyclist (part 1). The original images are shown in the left column (a, c, e) and the occupation probability maps are shown in the right column (b, d, f).



(a) t=63, input image (b) t=63, occupation probability map

(c) t=23, magnification of point cluster (d) t=38, magnification of point cluster

(e) t=48, magnification of point cluster (f) t=63, magnification of point cluster

Figure 5.39: Real world intersection sequence with crossing cyclist (part 2). The last original image is shown at the top left (a) and the associated occupation probability map is shown at the top right column (b). Magnifications are shown in images (c-f) to illustrate the clustering and Kalman filtering of the clusters.

Chapter 6

Conclusions

6.1 Summary

The development of a *driver assistant system* supporting drivers in complex intersection situations would be a major achievement for traffic safety, since many traffic accidents happen in such situations. While this is a highly complex task, which is still not accomplished, this thesis focused on one important and obligatory aspect of such systems: The visual detection of *independently moving objects*. Information about moving objects can, for example, be used in an *attention guidance system*, which is a central component of any complete intersection assistant system.

The decision to base such a system on visual input had two reasons: (i) Humans gather their information to a large extent visually and (ii) cameras are inexpensive and already widely used in luxury and professional vehicles for specific applications. Mimicking the articulated human head and eyes, agile camera systems are desirable. To avoid heavy and sensitive stereo rigs, a small and lightweight *monocular camera system* mounted on a pan-tilt unit has been chosen as input device.

In this thesis information about moving objects has been used to develop a prototype of an attention guidance system. It is based on the analysis of sequences from a single freely moving camera and on measurements from inertial sensors rigidly coupled with the camera system. The system comprises three major parts:

1. The *estimation of egomotion* of the camera relative to the static scene. Knowledge about the egomotion is essential for the detection task, and its computation must be reliable and fast.

2. The *detection of independent motion* based on image point correspondences.

3. A *Bayesian framework* to integrate the measurements in a consistent way. Temporal integration is an essential part of human perception and has been incorporated into the framework. The number of measurements on independently moving objects is boosted using an adaptive sampling scheme.

Both, the egomotion estimation methods and the detection algorithms for independent motion operate on correspondences between points in images. An algorithm for the detection and tracking of such feature points has been chosen. These correspondences are computed using an algorithm which iteratively minimises the intensity differences in a small support window (Kanade-Lucas-Tomasi feature tracking). Correspondence computation is only unambiguous if enough structure is present in the support window, and hence a corner detector is used to identify promising regions for correspondence estimation. Even though both algorithms (i.e. egomotion and detection) are independent of the particular method used for feature point tracking, the chosen tracking algorithm must be able to deal with the aperture problem and must result in full correspondences.

Egomotion:

The system determines the egomotion of the camera using a combination of car inertial sensors and image-based methods:

First a comparative study of different algorithms for egomotion computation using inertial sensors revealed the best sensors - the yawrate sensor or the steering wheel angle - and the best computation method. However, the car inertial sensors are not very accurate and do not provide information about pitch and roll motions. Therefore the full egomotion of the camera must be computed using image-based algorithms. The egomotion computation from the inertial sensors is, however, not in vain, since it can be used as prior knowledge helping to make image-based egomotion estimation faster and more robust.

Image-based egomotion estimation often suffered from the presence of highly erroneous correspondence measurements. A fast and robust algorithm, the preemptive RANSAC, has been adopted to solve this problem. The egomotion computed from the inertial sensors is used to initially reject wrong candidate solutions and thereby speed up the estimation process. The best solution and a number of correspondences which are consistent with this solution are the results of the robust estimation. All these consistent correspondences are used to refine the solution. Six different algorithms for the refinement were compared and the best algorithm (with respect to consistency, accuracy and computational requirements) was selected. It works by nonlinearily minimising the geometric error in combination with a robust cost function (Huber cost function).

The influence of camera calibration errors on the egomotion estimation was investigated and critical calibration parameters (i.e. the focal length and the principal point) for the given setup have been identified.

Detection:

Five different algorithms for detection of independent motion based on known egomotion are compared with respect to their performance and computational efficiency. The comparison was based on the area under the ROC (receiver operating characteristics) curves and revealed superior performance of the algorithm based on the direction of the translational flow: Each correspondence measurement can be decomposed in a part resulting

from camera rotation and a part resulting from camera translation. The rotational part is independent of the scene geometry and can be computed from the egomotion. The direction of the translational part can be (i) predicted solely from the known egomotion of the camera and (ii) measured by subtracting the rotational part from the measured correspondence. Comparing both directions reveals independent motion. The theoretical limits of this method have been explored and degenerate cases for this detection method have been identified.

Bayesian Framework:

A novel Bayesian framework for the detection of independent motion has been developed based on the directional detection method. It sequentially updates an occupation probability map in which the probability for independent motion is maintained in every pixel. The algorithm deduces many parameters from the data itself by means of statistical error propagation and thereby reduces the dimension of the parameter space.

Because of the real time constraints, only a relatively small number of correspondence measurements are made leading to sparse probability measurements. However, when computing correspondences at sparse locations, one would like to capture as much information about independently moving objects as possible. This is achieved by placing as many correspondence measurements as possible on moving objects. A novel algorithm *boosting* the correspondence density (and with it the information density) on independent motion was developed and integrated into the framework. The improvement of the correspondence density on moving objects is accomplished using an adaptive sampling approach. The algorithm is based on the clustering property of the particle filter and works by sequentially placing samples primarily in regions which had a high probability in the last time step. Image correspondences are measured at each sample position and the new probabilities are derived from the correspondences. The motion of the samples between two images is described by the associated correspondence vector, and the new probability is entered at the corresponding location in the occupation probability map. This approach results in a high measurements density on moving objects.

Temporal integration of the measurements is done by modelling the state of each sample (i.e. whether it is located on a moving object or not) as a Markov process. The probability for independent motion in the next time step can be predicted for each sample, due to the simple transition properties of Markov systems[1]. The motion of the sample in the image is accounted for using the correspondence measurement vector and temporal integration is conducted using Bayes law.

A prototype was build with the detection system extended by a simple spatial clustering of points belonging to a single object. Objects are tracked over time using a Kalman filter, and finally a warning is triggered when an object on a collision course is detected. The prototype of the attention guidance system has been integrated into the UTA (Urban

[1]The transition of a Markov system does only depend on its current state (which in this case is known) and not on its history (which in this case is unknown).

Traffic Assistant) demonstrator from the DaimlerChrysler AG. The prototype has been presented in over 200 successful live runs to representatives from politics and industry including the minister of education and science at that time Mrs. Edelgard Bulmahn.

6.2 Future Work

- Even though model-free detection of independent motion is important, it is not a sufficient base for a decision. The combination of classification and model-free detection system constitutes a very promising system and should be further investigated. Particularly the classification into dangerous and non-dangerous objects is still unsolved and can in my opinion only be achieved by analysis of shape and motion of the independently moving object.

- The investigation of the possibility to integrate other sensors like radar, lidar, laser range scanner etc. into the framework to enhance accuracy and reliability could be subject to further research.

- The probability for independent motion conditioned on the flow length and the direction difference $p(\mathrm{IMO}|l, \alpha)$ is an estimated value itself. In further studies, the distribution of this conditional probability could be modelled using a beta function. The two parameters a and b of the beta function could then be used to encode the probability itself and a confidence in the probability. The author is convinced, that this can be done such that the probability is given by the expectation of the beta distribution and the confidence is given by the variance of the beta distribution. Particularly the temporal integration could be very nicely done using a Kalman filter style approach. When the beta function from the last time step is used as a prior and the likelihood function is also modelled as a beta function, the posterior is then again a beta distribution because the beta distribution itself belongs to the family of conjugate priors for the beta distribution. The likelihood function could be theoretically derived from the measurements of the direction difference and the flow length.

Appendix

A Geometry

A.1 Cross Product and Skew Symmetric Matrix

Given 2 vectors $\boldsymbol{a} = (a_x, a_y, a_z)^T$ and $\boldsymbol{b} = (b_x, b_y, b_z)^T$, the cross product of the two vectors can be written as

$$\boldsymbol{a} \times \boldsymbol{b} = [\boldsymbol{a}]_\times \boldsymbol{b} = \begin{pmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{pmatrix} \boldsymbol{b} = \begin{pmatrix} a_y b_z - a_z b_y \\ a_z b_x - a_x b_z \\ a_x b_y - a_y b_x \end{pmatrix} = \left(\boldsymbol{a}^T [\boldsymbol{b}]_\times\right)^T \tag{A.1}$$

The matrix $[\boldsymbol{a}]_\times$ is a 3×3 skew symmetric matrix and hence

$$[\boldsymbol{a}]_\times^T = -[\boldsymbol{a}]_\times \tag{A.2}$$

A.2 3D-Rotation Parametrisation

Many different parametrisations for 3D-rotations exist. An overview is for example given in McGlone (2004). The parametrisations range from the simplest form using the nine entries of a rotation matrix over the minimal parametrisation using 3 Euler angles to more sophisticated parametrisations such as quaternions. Some parametrisations and their advantages and disadvantages are listed next. The quaternion has been chosen for parametrisation of rotations in this thesis because it has no singularities and combines easy concatenation and vector rotation with a relatively simple internal constraint (i.e. its norm must be one).

Rotation Matrix

3D-rotations can be given as an orthonormal 3×3 *rotation matrix* $\mathbf{R}$. It has nine real entries, and the rotation of a vector $\boldsymbol{x}$ is computed using simply the matrix vector product $\mathbf{R}\boldsymbol{x}$. Multiplication of a vector with $\mathbf{R}$ preserves the norm of the vector, i.e. $||\mathbf{R}\boldsymbol{x}|| = ||\boldsymbol{x}||$. The 3 eigenvalues of rotation matrices are given by 1, $e^{+i\omega}$, and $e^{-i\omega}$, and its determinant is always equal to 1.

Advantages:	**Disadvantages:**
• Linear equations	• Over-parametrisation
	• Difficult to enforce orthonormality
	• Difficult to interpret

Euler Angles

A minimal parametrisation of a 3D-rotation are the 3 rotation angles around the axes of the coordinate system. These angles are called *Euler angles*. The rotation matrices around the axes of the coordinate system $\mathbf{R}_x$, $\mathbf{R}_y$ and $\mathbf{R}_z$ are given as

$$\mathbf{R}_x = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(\omega_x) & -\sin(\omega_x) \\ 0 & \sin(\omega_x) & \cos(\omega_x) \end{pmatrix} \qquad \mathbf{R}_y = \begin{pmatrix} \cos(\omega_y) & 0 & \sin(\omega_y) \\ 0 & 1 & 0 \\ -\sin(\omega_y) & 0 & \cos(\omega_y) \end{pmatrix} \tag{A.3}$$

$$\mathbf{R}_z = \begin{pmatrix} \cos(\omega_z) & -\sin(\omega_z) & 0 \\ \sin(\omega_z) & \cos(\omega_z) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Because matrix multiplication is not commutative, the rotation order is of importance:

$$\mathbf{R}_x\mathbf{R}_y\mathbf{R}_z \neq \mathbf{R}_z\mathbf{R}_y\mathbf{R}_x$$

The rotation $\mathbf{R} = \mathbf{R}_x\mathbf{R}_y\mathbf{R}_z$ can be interpreted as

1. rotation around **fixed** axes with order 1. z, 2. y, 3. x or
2. rotation around **rotated** axes with order 1. x, 2. y, 3. z.

Tn this case "fixed" means that the axes of the initial coordinate system are used, and "rotated" indicates that the rotated axes are used. When the multiplication order of the individual rotation matrices is inverted, i.e. $\mathbf{R} = \mathbf{R}_z\mathbf{R}_y\mathbf{R}_x$, the interpretation is given by

1. rotation around **rotated** axes with order 1. z, 2. y, 3. x or
2. rotation around **fixed** axes with order 1. x, 2. y, 3. z

The *Gimbal lock* describes the loss of one degree of freedom. This happens when the axes of the first and the third rotation are aligned as caused by a rotation of plus or minus 90° around the second axis. Only the sum of the first and third angle determines the rotation in this case. In these cases it is impossible to express certain rotations. The Gimbal lock does only occur with big rotation angles and can safely be neglected when only small rotation changes $\ll \frac{\pi}{2}$ are estimated.

Advantages:	Disadvantages:
• Minimal parametrisation	• Singularity
• Easy to read and visualise	• Gimbal Lock
	• Ambiguity of rotation order
	• Determination of rotation angles from matrix may not be unique and unstable (McGlone, 2004)

Axis and Angle

Another parametrisation of 3D-rotations is to choose the normalised rotation axis $\boldsymbol{w}$ and the angle ω (McGlone, 2004). Derivation of axis and angle from rotation matrix can be done using eigenvalues and eigenvectors. The axis is given by the eigenvector corresponding to the eigenvalue 1, and the angle ω can be computed using the phase angle of the two complex eigenvalues $e^{+i\omega}$ and $e^{-i\omega}$ (Schmidt and Niemann, 2001)

Advantages:	Disadvantages:
• Linear equations	• Over-parametrisation
	• Ambiguity of axis at $\omega = 0$

Gibb's Vector

Another minimal parametrisation of a 3D-rotation is the *Gibb's vector* which is given by the rotation axis $\boldsymbol{w}$ with norm one multiplied by the tangent of half the rotation angle ω (Horn, 2000)

$$\boldsymbol{w}\tan(\omega/2) \tag{A.4}$$

Advantages:	Disadvantages:
• Minimal parametrisation	• Singularity at $\omega = \pi$
	• Ambiguity of axis at $\omega = 0$

Axis times Angle

Multiplying the normalised rotation axis $\boldsymbol{w}$ directly with the rotation angle ω leads to another minimal parametrisation of 3D-rotation.

Advantages:	Disadvantages:
• Minimal parametrisation	• Singularity at $\omega = 2\pi$ (Hartley and Zissermann, 2004)
	• Ambiguity of axis at $\omega = 0$

Quaternion

A widespread parametrisation of 3D-rotations are quaternions. A unit quaternion $\boldsymbol{q} = (q, \boldsymbol{q})$ with vector part $\boldsymbol{q} = \sin(\frac{\omega}{2})\boldsymbol{w}$ and scalar part $q = \cos(\frac{\omega}{2})$ describes a 3D-rotation. Quaternion rotations can be concatenated by quaternion multiplication

$$\boldsymbol{p} = \boldsymbol{q}\boldsymbol{r} = (p, \boldsymbol{p}) = (qr - \boldsymbol{q}^T\boldsymbol{r},\ r\boldsymbol{q} + q\boldsymbol{r} + [\boldsymbol{q}]_\times \boldsymbol{r}) \tag{A.5}$$

with the quaternions $\boldsymbol{p} = (p, \boldsymbol{p})$, $\boldsymbol{q} = (q, \boldsymbol{q})$ and $\boldsymbol{r} = (r, \boldsymbol{r})$. The inverse of a unit quaternion is computed by multiplying the vector part with -1. Left and right multiplication with a quaternion $\boldsymbol{q}$ and its inverse $\boldsymbol{q}^{-1}$ rotates the vector part of the $\boldsymbol{p}$

$$\boldsymbol{q}\boldsymbol{p}\boldsymbol{q}^{-1} \tag{A.6}$$

and hence rotation of an arbitrary vector $\boldsymbol{x}$ using quaternions can be conducted by constructing a quaternion whose vector part is $\boldsymbol{x}$.

Advantages:	Disadvantages:
• No singularities	• Norm 1 constraint
• Easy concatenation	• Complicated to read and visualise

Comparison

Schmidt and Niemann (2001) compare the "axis times angle" representation versus a local parametrisation related to quaternions. The local parametrisation is given by the hyperplane to the unit sphere in $\mathbb{R}^4$ at the operating point $\boldsymbol{q}_0$. Unconstrained optimisation can be used because only incremental changes to the rotation are estimated. These changes are parametrised by the basis vector of the hyperplane. Bundle adjustment for structure and motion is used to compare both parametrisations. No significant differences between the two compared parametrisations could be detected.

A.3 Conic and Dual Conic

A general *point conic* is given by the equation (Hartley and Zissermann, 2004)

$$ax^2 + bxy + cy^2 + dx + ey + f = 0 \tag{A.7}$$

Equation A.7 can also be represented using matrix $\mathbf{C}$ and vector $\boldsymbol{x} = (x, y, 1)^T$

$$0 = \boldsymbol{x}^T \cdot \mathbf{C} \cdot \boldsymbol{x} = \begin{pmatrix} x & y & 1 \end{pmatrix} \begin{pmatrix} a & \frac{b}{2} & \frac{d}{2} \\ \frac{b}{2} & c & \frac{e}{2} \\ \frac{d}{2} & \frac{e}{2} & f \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \tag{A.8}$$

Obviously the conic has 6 degrees of freedom and is a homogeneous entity. Multiplying the conic by an arbitrary real factor λ results in the same geometric entity. The tangent $\boldsymbol{l}$ to the conic $\mathbf{C}$ at $\boldsymbol{x}$ is given by $\boldsymbol{l} = \mathbf{C}\boldsymbol{x}$.

The associated *dual* or *line conic* is given by the adjoint matrix $\mathbf{C}^*$. When the conic in its matrix representation is not singular, the adjoint equals the inverse (up to scale) $\mathbf{C}^* = \mathbf{C}^{-1}$.

A conic describing an ellipse around the origin with half axes aligned with the reference frame and lengths λ_1 and λ_2 is given by

$$\mathbf{C}_c = \begin{pmatrix} \frac{1}{\lambda_1^2} & 0 & 0 \\ 0 & \frac{1}{\lambda_2^2} & 0 \\ 0 & 0 & -1 \end{pmatrix} \tag{A.9}$$

Any conic describing an ellipse can be brought into its *canonical form* (equation A.9) by a congruency transformation (Kanatani, 2005).

Under a point transformation $\mathbf{A}$ the conic transforms as $\mathbf{A}^{-T}\mathbf{C}\mathbf{A}^{-1}$ (Hartley and Zissermann, 2004).

Connection to Covariance Matrix

The isoprobability lines of a 2D normal distribution are given by ellipses whose centre coincides with the mean of the distribution. A single ellipse is hence sufficient to parametrise the complete distribution and traditionally the isoprobability line where the density has dropped to the half of its value at the peek is chosen. The half axes of the ellipse are given by the eigenvectors of the covariance matrix scaled with their associated squared eigenvalues. The position of the centre of the ellipse marks the mean of the distribution. Each ellipse thus describes a normal distribution and since ellipses are conics, each conic describing an ellipse can be interpreted as a parametrisation of the normal distribution.

The construction of a conic $\mathbf{C}$ from mean $\boldsymbol{x}$ and covariance $\boldsymbol{\Sigma}_{\boldsymbol{xx}}$ of a distribution is given by:

$$\mathbf{C} = \begin{pmatrix} \boldsymbol{\Sigma}_{\boldsymbol{xx}}^{-1} & -\boldsymbol{\Sigma}_{\boldsymbol{xx}}^{-1}\boldsymbol{x} \\ -\boldsymbol{x}^T\boldsymbol{\Sigma}_{\boldsymbol{xx}}^{-1} & \boldsymbol{x}^T\boldsymbol{\Sigma}_{\boldsymbol{xx}}^{-1}\boldsymbol{x} - 1 \end{pmatrix} \tag{A.10}$$

Proof: Under a point transformation $\mathbf{A}$ the conic transforms as $\mathbf{A}^{-T}\mathbf{C}\mathbf{A}^{-1}$ (Hartley and Zissermann, 2004). A specific similarity transformation $\mathbf{A}$ aligning the half axes of the covariance ellipse and the coordinate frame and bringing the mean into the origin can always be found

$$\mathbf{A} = \begin{pmatrix} \mathbf{R}^T & -\mathbf{R}^T\boldsymbol{x} \\ \boldsymbol{0}^T & 1 \end{pmatrix} \tag{A.11}$$

where the rotation matrix $\mathbf{R} \in \mathbb{R}^{2\times 2}$ is chosen such that it aligns the covariance ellipse with the reference frame: $\mathbf{R}\boldsymbol{\Sigma}_{\boldsymbol{xx}}\mathbf{R}^T = \mathrm{diag}(\lambda_1^2, \lambda_2^2)$. λ_1^2 and λ_2^2 are the eigenvalues of the covariance matrix.

Applying the transformation $\mathbf{A}$ (eq. A.11) to the conic $\mathbf{C}$ results in a conic $\mathbf{C}'$

$$\begin{aligned} \mathbf{C}' &= \mathbf{A}^{-T}\mathbf{C}\mathbf{A}^{-1} \\ &= \begin{pmatrix} \mathbf{R}^T & \boldsymbol{0} \\ \boldsymbol{x}^T & 1 \end{pmatrix} \begin{pmatrix} \boldsymbol{\Sigma}_{\boldsymbol{xx}}^{-1} & -\boldsymbol{\Sigma}_{\boldsymbol{xx}}^{-1}\boldsymbol{x} \\ -\boldsymbol{x}^T\boldsymbol{\Sigma}_{\boldsymbol{xx}}^{-1} & \boldsymbol{x}^T\boldsymbol{\Sigma}_{\boldsymbol{xx}}^{-1}\boldsymbol{x} - 1 \end{pmatrix} \begin{pmatrix} \mathbf{R} & \boldsymbol{x} \\ \boldsymbol{0}^T & 1 \end{pmatrix} \\ &= \begin{pmatrix} \mathbf{R}^T\boldsymbol{\Sigma}_{\boldsymbol{xx}}^{-1}\mathbf{R} & \boldsymbol{0} \\ \boldsymbol{0}^T & -1 \end{pmatrix} \\ &= \begin{pmatrix} \frac{1}{\lambda_1^2} & 0 & 0 \\ 0 & \frac{1}{\lambda_2^2} & 0 \\ 0 & 0 & -1 \end{pmatrix} \end{aligned} \tag{A.12}$$

$\mathbf{C}'$ is an ellipse conic in its canonical form (equation A.9). The canonical form of an ellipse conic describes an ellipse around the origin with half axes of lengths λ_1 and λ_2 aligned with the reference frame Kanatani (2005). The similarity transform $\mathbf{A}$ represents a change of reference frame, $\mathbf{C}$ is thus the description of an ellipse around the point $\boldsymbol{x}$ with covariance matrix $\boldsymbol{\Sigma}_{\boldsymbol{xx}}$. □

Pole and Polar

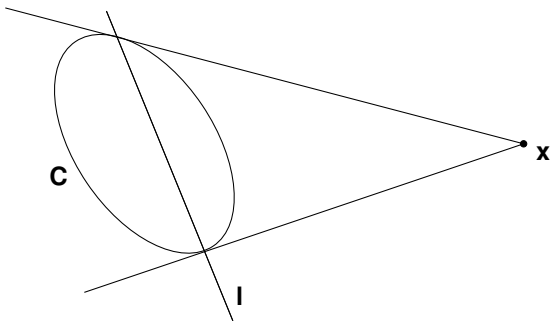


Figure A.1: Pole $\boldsymbol{x}$ and polar $\boldsymbol{l}$ to the conic $\mathbf{C}$

The line $\boldsymbol{l} = \mathbf{C}\boldsymbol{x}$ is the *polar* of the point $\boldsymbol{x}$ with respect to the conic $\mathbf{C}$ and the point $\boldsymbol{x}$ is the *pole* of $\boldsymbol{l}$ with respect to $\mathbf{C}$. The polar of $\boldsymbol{x}$ intersects the conic at the points of tangency of lines from $\boldsymbol{x}$ (Hartley and Zissermann, 2004). This is visualised in figure A.1.

A.4 Tangents to Ellipse through Point

The computation of the tangent points $\boldsymbol{t}_{1/2}$ on a given 2D-ellipse (e.g. a covariance ellipse), whose tangents meet in a given point $\boldsymbol{x}$, is described in this section. This situation is illustrated in figure 5.15. Two methods, an analytic approach and a numeric approach, are described next:

Analytic Approach

The covariance matrix can be represented by a conic $\mathbf{C}$ (see appendix A.3). The tangent points through the point $\boldsymbol{x}$ are given by the intersection of the polar of $\boldsymbol{x}$ with respect to $\mathbf{C}$ (see appendix A.3). The polar $\boldsymbol{l}_p$ is given by

$$\boldsymbol{l}_p = \mathbf{C}\boldsymbol{x} \tag{A.13}$$

The degenerate line conic $\mathbf{C}^*$ consisting of the two tangent points $\boldsymbol{t}_1$ and $\boldsymbol{t}_2$ is given by (Hartley and Zissermann, 2004)

$$\mathbf{C}^* = [\boldsymbol{l}_p]_\times \cdot \mathbf{C} \cdot [\boldsymbol{l}_p]_\times \tag{A.14}$$

The matrix $\mathbf{C}^*$ is a projective representation of a line conic and can hence be multiplied by an arbitrary scale factor $\neq 0$. Unless the bottom right entry is zero, the scale factor can be chosen such that the bottom right entry equals 2 after multiplication. When the bottom right entry is zero, the mean of the covariance matrix has a Mahalanobis distance of 1 to the origin, and another reference frame must be used.

$$\mathbf{C}^* = \begin{pmatrix} a & b & d \\ b & c & e \\ d & e & 2 \end{pmatrix} \tag{A.15}$$

A dual conic $\mathbf{C}_p^*$ consisting of 2 points $\boldsymbol{x} = (x_1, x_2, 1)^T$ and $\boldsymbol{y} = (y_1, y_2, 1)^T$ is given by (Hartley and Zissermann, 2004)

$$\mathbf{C}_p^* = \boldsymbol{x}\boldsymbol{y}^T + \boldsymbol{y}\boldsymbol{x}^T = \begin{pmatrix} 2x_1y_1 & x_1y_2 + x_2y_1 & x_1 + y_1 \\ x_1y_2 + x_2y_1 & 2x_2y_2 & x_2 + y_2 \\ x_1 + y_1 & x_2 + y_2 & 2 \end{pmatrix} \tag{A.16}$$

Setting A.15 and A.16 equal results in five equations with the 4 unknowns x_1, x_2, y_1, y_2

$$e = x_2 + y_2 \quad \rightarrow \quad x_2 = e - y_2 \tag{A.17}$$

$$d = x_1 + y_1 \quad \rightarrow \quad x_1 = d - y_1 \tag{A.18}$$

$$c = 2x_2y_2 \tag{A.19}$$

$$b = x_1y_2 + x_2y_1 \tag{A.20}$$

$$a = 2x_1y_1 \tag{A.21}$$

Using equation A.17 in A.19 and equation A.18 in A.21 results in two square equations. The two solutions for each equation can be combined to 4 possible solutions. Equation A.20 is used to find the correct combination of solutions.

Numeric Approach

Alternatively, the tangent points can also be recovered numerically. This task is easier to solve, if the ellipse is located at the origin and if its half axes are aligned with the axes of the coordinate system. A coordinate transform resulting in such a configuration can always be found (Kanatani, 2005). An arbitrary point $\boldsymbol{t}$ on an ellipse with half axes l_a and l_b is given by

$$\boldsymbol{t} = (l_a \cos(\beta), l_b \sin(\beta))^T \tag{A.22}$$

with parameter β. The normal to the point $\boldsymbol{t}$ on the ellipse is given by

$$\boldsymbol{n} = (l_b \cos(\beta), l_a \sin(\beta))^T \tag{A.23}$$

The scalar product between the normal $\boldsymbol{n}$ and $\boldsymbol{t} - \boldsymbol{x}_1$ must be zero at the tangent point, leading to

$$\begin{aligned} 0 &= (l_b \cos(\beta))(x_{1x} - l_a \cos(\beta)) + (l_a sin(\beta))(x_{1y} - l_b \sin(\beta)) \\ &= x_{1x} l_b \cos(\beta) + x_{1y} l_a \sin(\beta) - l_a l_b (\cos^2(\beta) + \sin^2(\beta)) \\ &= x_{1x} l_b \cos(\beta) + x_{1y} l_a \sin(\beta) - l_a l_b \end{aligned} \tag{A.24}$$

Newton's gradient descent is used to find the minimum of equation A.24. Two different minima are computed using two different initialisations for β. The tangent points can be computed from the solutions $\hat{\beta}_{1/2}$.

B Calculus

B.1 Gamma Function

The *gamma function* (Stöcker, 1993) is defined as

$$\Gamma(x) = \int_0^\infty t^{x-1} e^{-t} \mathrm{d}t \tag{B.1}$$

B.2 Logistic Function

The *logistic function* (Jordan, 1995) is defined as

$$p(x) = \alpha \frac{1 + me^{-x/\tau}}{1 + ne^{-x/\tau}} \tag{B.2}$$

The *standard logistic function* or *sigmoid function* is a special case of the logistic function with $\alpha = 1$, $m = 0$, $n = 1$ and $\tau = 1$

$$p(x) = \frac{1}{1 + e^{-x}} \tag{B.3}$$

Figure B.1 shows the sigmoid function.

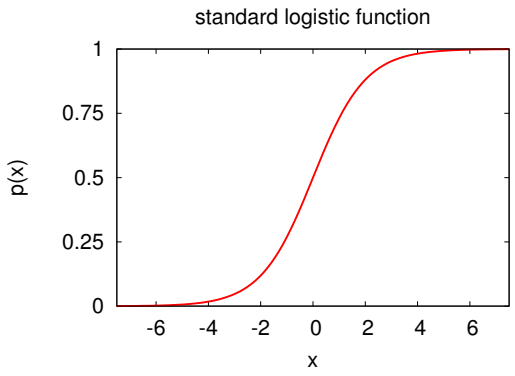


Figure B.1: The sigmoid function

Relation to Bayes law: Using Bayesian probability propagation, the posterior for a binary state of nature s can be expressed using the standard logistic function (Jordan,

1995)

$$\begin{aligned} p(s=0|x) &= \frac{p(x|s=0)p(s=0)}{p(x|s=0)p(s=0)+p(x|s=1)p(s=1)} \\ &= \frac{1}{1+e^{-\log\frac{p(x|s=0)}{p(x|s=1)}-\log\frac{p(s=0)}{p(s=1)}}} \\ &= \frac{1}{1+e^{\zeta}} \end{aligned} \tag{B.4}$$

where the exponent ζ depends on the logarithmic ratio of the likelihood functions and on the logarithmic ratio of the priors.

C Parameter Estimation

C.1 Robust Parameter Estimation Methods

The algorithms for robust parameter estimation can be classified into 4 different categories, namely:

1. Algorithms using clustering techniques.
2. M-Estimators use an iterative re-weighting technique to achieve robustness, while processing all available data.
3. Case deletion diagnostic algorithms try to identify outliers and reject them from the computation. They are based on the measurement of the influence of a single datum on the result.
4. Algorithms using random sampling techniques to achieve a solution with a minimal data set.

Clustering

A conventional approach to robust estimation is clustering in the parameter space. A famous application of this method is the Hough transform for line detection.

The parameter space is discretised in several bins, according to the desired accuracy. For each minimal data set needed for parameter calculation, the parameters are determined and the according bin in the discretised parameter space is increased. After calculating a large number of subsets, the highest peak in the parameter space represents the best supported solution. This method is well suited when a large number of data supports the solution.

In cases with several independent solutions (e.g. several lines appear in one image), the problem concerning the number of clusters and the correlation of the points in the parameter space to them remains.

how many clusters are present and which points in parameter space belong to which cluster remains. It can be solved using the fuzzy c mean algorithm (Bezdek et al., 1999).

The clustering technique is rarely used when the dimension of the parameter space is bigger than three, because size of the accumulator increases with the dimension and the required accuracy (Zhang, 1997). Even the evaluation of a parameter vector from a five-dimensional parameter space with a coarse discretisation of 10 cells per dimension would result in 10^5 bins overall.

M-Estimators

In the traditional least squares estimator (LSE), the sum of the squared residuals is minimised in order to find the best solution to a parameter vector:

$$\min \sum_i r_i^2 \tag{C.1}$$

where r_i is the residual of the ith datum. The underlying assumption behind the least squares estimator is that the noise is independent at each datum, Gaussian distributed with the same variance σ^2 at every datum and has zero mean. Under these conditions the LSE is a maximum likelihood estimator. In the presence of gross errors the assumptions about the error model are however violated, and the LSE may fail even with as little as one outlier. Other positive definite symmetric error functions $\rho(r_i)$ with a unique minimum at zero can be used instead of the squared residual, resulting in a so called M-estimator (Zhang, 1996):

$$\min \sum_i \rho(r_i) \tag{C.2}$$

The idea of this approach is to reduce the weight of outlying data and thereby approximate a kind of maximum likelihood estimator for non Gaussian error conditions. The M-Estimator can be implemented as an iterative re-weighted LSE:

A solution to eq. C.2 can be found by setting the partial derivative with respect to the parameter vector $P = (p_1, ..., p_m)^T$ to zero, and solving for p_j:

$$\sum_i \frac{\partial \rho(r_i)}{\partial r_i} \frac{\partial r_i}{\partial p_j} = 0 \quad \text{for } j = 1, \ldots, m \tag{C.3}$$

Calling $\Psi(r) = \frac{d\rho(r)}{dr}$ the influence function and defining the weight function $\omega(r) = \frac{\Psi(r)}{r}$ leads to

$$\sum_i \omega(r_i) r_i \frac{\partial r_i}{\partial p_j} = 0 \tag{C.4}$$

Integrating with respect to the parameters p_j leads to an iterative least squares estimator (Zhang, 1997)

$$\min \sum_i \omega(r_{i,k-1}) r_i^2 \tag{C.5}$$

using the weight depending on the residuals of the last iteration r_i^{k-1} in the estimation process. Several of these error functions have been investigated in the literature (see Zhang (1997, 1996) for a brief survey). Fig. C.1 gives an overview over some commonly used error functions and the according influence and weight functions. The graphic representation of these functions is shown in fig. C.2.

Case Deletion Diagnostics

Case deletion diagnostic algorithms try to identify outliers and reject them from the computation. They are based on the measurement of the influence of a single datum on the result. For instance, the question is: How would the parameters change, if we exclude the i-th datum from the calculation (Torr and Murray, 1996; Chatterjee and Hadi, 1988).

type	$\rho(x)$	$\psi(x)$	$w(x)$
L_2	$x^2/2$	x	1
L_1	$\lvert x\rvert$	$\operatorname{sgn}(x)$	$\frac{1}{\lvert x\rvert}$
$L_1 - L_2$	$2(\sqrt{1+x^2/2}-1)$	$\frac{x}{\sqrt{1+x^2/2}}$	$\frac{1}{\sqrt{1+x^2/2}}$
L_p	$\frac{\lvert x\rvert^\nu}{\nu}$	$\operatorname{sgn}(x)\lvert x\rvert^{\nu-1}$	$\lvert x\rvert^{\nu-2}$
"Fair"	$c^2[\frac{\lvert x\rvert}{c} - \log(1+\frac{\lvert x\rvert}{c})]$	$\frac{x}{1+\lvert x\rvert/c}$	$\frac{1}{1+\lvert x\rvert/c}$
Huber $\begin{cases}\text{if } \lvert x\rvert \le k \\ \text{if } \lvert x\rvert \ge k\end{cases}$	$\begin{cases}x^2/2 \\ k(\lvert x\rvert - k/2)\end{cases}$	$\begin{cases}x \\ k\operatorname{sgn}(x)\end{cases}$	$\begin{cases}1 \\ k/\lvert x\rvert\end{cases}$
Cauchy	$\frac{c^2}{2}\log(1+(x/c)^2)$	$\frac{x}{1+(x/c)^2}$	$\frac{1}{1+(x/c)^2}$
Geman-McClure	$\frac{x^2/2}{1+x^2}$	$\frac{x}{(1+x^2)^2}$	$\frac{1}{(1+x^2)^2}$
Welsch	$\frac{c^2}{2}[1-\exp(-(x/c)^2)]$	$x\exp(-(x/c)^2)$	$\exp(-(x/c)^2))$
Tukey $\begin{cases}\text{if } \lvert x\rvert \le c \\ \text{if } \lvert x\rvert > c\end{cases}$	$\begin{cases}\frac{c^2}{6}\left(1-[1-(x/c)^2]^3\right) \\ (c^2/6)\end{cases}$	$\begin{cases}x[1-(x/c)^2]^2 \\ 0\end{cases}$	$\begin{cases}[1-(x/c)^2]^2 \\ 0\end{cases}$

Figure C.1: Several commonly used M-Estimators with the according error, influence and weight functions (Images courtesy of Z. Zhang (Zhang, 1997, 1996)).

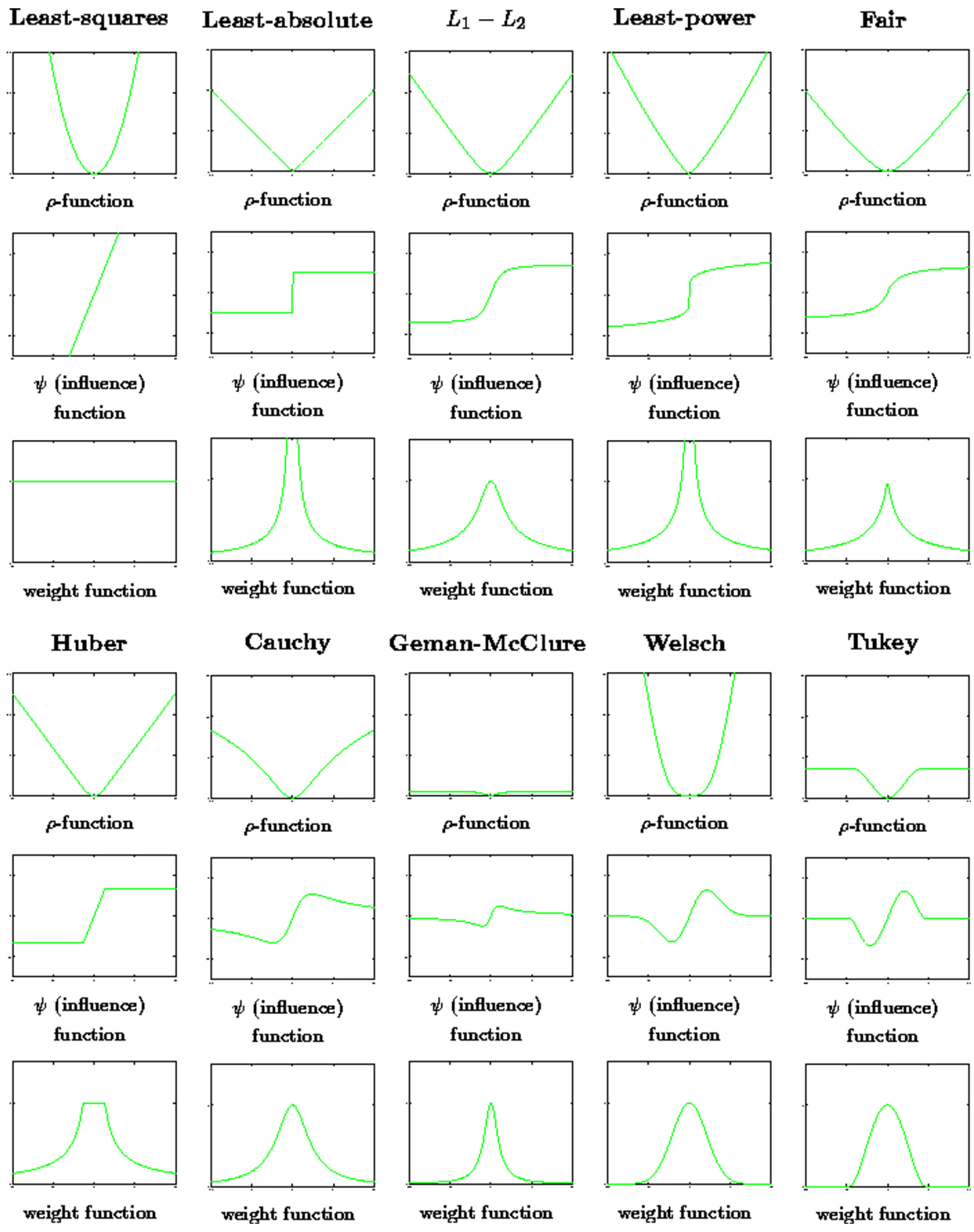


Figure C.2: Graphical representation of error, influence and weight functions of several commonly used M-Estimators (Images courtesy of Z. Zhang (Zhang, 1997, 1996)).

Random Sampling Algorithms

RANSAC: The RANSAC algorithm (Fischler and Bolles, 1981) is robust in the case of data heavily corrupted with outliers. Given that there are m data and a minimum of n of them is needed to estimate the parameter vector. The approach is very simple and works as follows:

- Randomly select a minimum set of n data and extract the parameters x from them.
- Calculate the number k of data from the overall set supporting the parameters x with respect to a given threshold t.
- If k is bigger than a given fraction, calculate the least squares solution from all data supporting x and exit with success.
- Repeat the above steps L times.
- Either use the parameters with the biggest support k, calculate the least squares solution and exit with success, or exit with failure.

This is in fact the search of a solution that minimises the cost function (Torr and Zisserman, 1996)

$$C = \sum_i \rho(r_i) \tag{C.6}$$

with

$$\rho(r_i) = \begin{cases} 0 & r_i^2 < t^2 \\ \text{const.} & r_i^2 \geq t^2 \end{cases} \tag{C.7}$$

The number of trials L needed to ensure at least one outlier-free set of data with probability z can be calculated by

$$L = \frac{\log(1-z)}{\log(1-p^n)}, \tag{C.8}$$

where p is the expected outlier fraction in the data (Fischler and Bolles, 1981).

MSAC: In the RANSAC algorithm, the penalty for gross errors is constant regardless of the actual residuum associated with the datum. This undesirable situation can be avoided with no extra cost by replacing $\rho(r_i)$ by

$$\rho_2(r_i) = \begin{cases} r_i^2 & r_i^2 < t^2 \\ \text{const.} & r_i^2 \geq t^2 \end{cases} \tag{C.9}$$

in the cost function (eq. C.6) of the RANSAC algorithm (Torr and Zisserman, 1996). The resulting algorithm is called M-estimator SAmple Consensus (MSAC) (Torr and Zisserman, 1996).

MLESAC: In the case of a simple Gaussian error model, the probability density function in the state space around the true system state $\underline{x}$ is given by

$$p(x) = \Pi_{i=1\ldots m}(\frac{1}{\sqrt{2\pi}\sigma})e^{-||x_i-\underline{x}_i||_2^2/(2\underline{\sigma}^2)} \tag{C.10}$$

where $||x||_2$ denotes the L_2 norm of the parameter vector x and $\underline{\sigma}$ denotes the variance of x. The maximum likelihood estimate of x can hence be found by minimising the negative log likelihood

$$-\sum_{i=1}^{m}\log(p(x)) = \sum_{i=1}^{m}||x_i - \underline{x}_i||_2^2 \tag{C.11}$$

In the presence of gross errors, the assumption of a Gaussian error model does not hold and a more appropriate substitution for eq. C.10 would be (Torr and Zisserman, 1996)

$$p(x) = \gamma\frac{1}{\sqrt{2\pi}\sigma}e^{-||x_i-\underline{x}_i||_2^2/(2\sigma^2)} + (1-\gamma)\frac{1}{v} \tag{C.12}$$

with the mixing parameter γ and the constant v representing a uniform distribution of the gross errors. The resulting negative log likelihood is

$$-L = -\sum_{i}\log\left(\gamma(\frac{1}{\sqrt{2\pi}\sigma})e^{-||x_i-\underline{x}_i||_2^2/(2\sigma^2)} + (1-\gamma)\frac{1}{v}\right) \tag{C.13}$$

Since the mixture parameter γ cannot be observed directly, an iterative one-dimensional search for it is done (Torr and Zisserman, 1996). However, Tordoff and Murray (2002) suggested the use of $\gamma = 0.5$ since the re-evaluation of the mixture parameter with every sample is unfair. The mixture parameter does not depend on the parameters estimated from the actual sample set, but instead is a constant prior. Extensive testing using fundamental matrix estimation revealed that better hypotheses result in higher maximum likelihood scores, regardless of the mixture parameter. Further on, Tordoff and Murray (2002) suggested the use of additional information (e.g. the correlation between two matches) to guide the sample process and thereby reduce the number of samples needed to finish the maximum likelihood estimation sample consensus (MLESAC) algorithm.

MAPSAC: The Maximum A Posteriori SAmple Consesus (MAPSAC) is the Bayesian extension of the MLESAC, incorporating a prior (Torr, 2002). The MAPSAC is similar to the MSAC algorithm. It can be approximated by maximising

$$\rho_3(\frac{r_i^2}{\sigma^2}) = \begin{cases} \frac{r_i^2}{\sigma^2} & \frac{r_i^2}{\sigma^2} < T \\ T & \frac{r_i^2}{\sigma^2} \geq T \end{cases} \tag{C.14}$$

with

$$T = 2\log\left(\frac{\gamma}{1-\gamma}\right) + (D-d)\log\left(\frac{U^2}{2\pi\sigma^2}\right) \tag{C.15}$$

where γ is the user given mixture parameter as defined in eq. C.12, $D - d$ is the co-dimension of the algebraic manifold (in the case of two view fundamental matrix estimation is $D - d = 2$) and U is some measure of the area in which outliers may be detected, assuming uniform pdf for the outlier distribution (Torr, 2002). With a slight increase in computational cost, γ can be calculated, too.

LMedS and LTS: Replacing the sum in eq. C.1 with a median leads to the least median of squares estimator (LMedS) (Rousseeuw and Leroy, 1987):

$$\min \text{med}_i \; r_i^2 \tag{C.16}$$

Unfortunately the LMedS can neither be solved in close form nor can it be reduced to an iteratively re-weighted least squares problem. Therefore, either a complete search in parameter space must be conducted or an investigation of random subsets of the parameter space must suffice.

To resolve this problem, Rousseeuw and Leroy (1987) introduced the least trimmed squares estimator (LTS):

$$\min \sum_{i=1}^{h} (r^2)_{i:n} \tag{C.17}$$

where $(r^2)_{1:n} \leq \ldots \leq (r^2)_{n:n}$ are the ordered squared residuals and $h = [n/2] + 1$.

Both estimators achieve a breakdown point of 50%. The LTS estimator is fast in comparison to the LMedS estimator, because there is no exhaustive search in the parameter space involved. Often the LMedS estimator is used together with a random search in the parameter space similar to the random search in the RANSAC algorithm.

MINPRAN: The MINimise the Probability of RANdomness (MINPRAN) algorithm (Stewart, 1995) is only based on a known outlier probability density function p_o. Given a set of parameter Φ describing a model, it is possible to calculate the probability $p_{r,k}$ that k outlier fall within $\Phi \pm r$ from the known outlier pdf p_o. This is used as follows: Given a parameter set Φ and a threshold r, the number of inliers with respect to Φ and r is given by j. It is now possible to calculate the possibility $p_{r,j}$ that j data points within $\Phi \pm r$ are only based on the outlier pdf p_o. $p_{r,k}$ is hence closely related to the probability that Φ is not the correct model (i.e. it is solely based on outliers). The "probability of randomness" p_Φ is now defined as the minimum of $p_{r,k}$ over all possible r. Note that k is the number of inliers according to the model Φ and the threshold r. Hence k depends on r and on the model Φ. In summary:

- Randomly select a minimum set of n data and extract the parameters Φ from them.
- Calculate the "probability of randomness" p_Φ by calculating for every data point i:
  - the residuum $r_{\Phi,i}$,

  - the number of data points $k_{\Phi,i}$ which would be inliers to Φ with respect to the threshold $r_{\Phi,i}$,
  - the probability $p_{r_{\Phi,i},k_{\Phi,i}}$ that $k_{\Phi,i}$ points fall within $\Phi \pm r_{\Phi,i}$ from the outlier pdf p_o .

  The "probability of randomness" p_Φ is the minimal $p_{r_{\Phi,i},k_{\Phi,i}}$.

- If the "probability of randomness" p_Φ is smaller than certain exit threshold $p_{\Phi,0}$, exit with success.
- Repeat the above steps L times.
- Either use the parameters Φ with the smallest "probability of randomness", calculate the least squares solution and exit with success or exit with failure.

An exit threshold $p_{\Phi,0}$ is calculated in advance from a user-given hallucinate probability p_h. The calculation of the $p_{\Phi,0}$ is computationally complex and therefore done offline and stored in a look-up table. A similar but slightly more complex expression as eq. C.8 for the number of necessary repetitions is also derived in Stewart (1995).

NAPSAC: The N Adjacent Points Sample Consensus (NAPSAC) (Myatt et al., 2002) differs from the RANSAC algorithm in the selection method for the minimal data set needed to calculate a solution. In the RANSAC algorithm each of the n data points from the whole set of m data points is selected with probability $\frac{1}{m}$. Myatt et al. (2002) stated, that a significantly higher possibility of choosing a set of inliers exists, if the members of this set lie in the vicinity of each other.

IMPSAC: In Torr and Davidson (2003) the basic idea of the MAPSAC algorithm is extended to a multi scale approach. Specifically the fundamental matrix estimation is conducted using an image pyramid and a feature matcher capable of handling rotations. The information gathered from a coarse pyramid level is transferred to the next finer level using a particle system. This allows the propagation of multimodal probability distributions from pyramid level to pyramid level. In this way the chance of ending up in a local extremum is significantly reduced. They dubbed this approach IMPSAC. It is a synthesis of IMPortance sampling and RANdom SAmple Consensus.

C.2 Covariance Approximation of an Estimated Vector

Given a overdetermined system of m equations $f_i(\mathbf{x})$ with n unknowns x_i

$$\mathbf{f}(\mathbf{x}) = \mathbf{c} \tag{C.18}$$

with the parameter vector $\mathbf{x} = (x_1, x_2, \ldots, x_n)^T$, the measurement vector $\mathbf{c}$ and the m-dimensional function $\mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_m(\mathbf{x}))^T$. Solving equation C.18 results in

the solution vector $\hat{\mathbf{x}}$ and the residual error vector $\hat{\mathbf{r}} = \mathbf{f}(\hat{\mathbf{x}}) - \mathbf{c}$. Usually the regression model

$$\mathbf{f}(\hat{\mathbf{x}}) = \mathbf{f}(\mathbf{x}_0) + \mathbf{r}_0 \tag{C.19}$$

with the true solution vector $\mathbf{x}_0$ and the estimation error vector $\mathbf{r}_0$ is used. Note the difference between the residual error vector $\hat{\mathbf{r}} = \mathbf{f}(\hat{\mathbf{x}}) - \mathbf{c}$ and the estimation error $\mathbf{r}_0 = \mathbf{f}(\hat{\mathbf{x}}) - \mathbf{f}(\mathbf{x}_0)$. The estimation error vector $\mathbf{r}_0$ is in particular unknown, because the true solution $\mathbf{x}_0$ is unknown.

The estimation error vector $\mathbf{r}_0 = (r_1, r_2, \ldots, r_m)^T$ is usually assumed to consist of iid (independent identically distributed) entries $r_i \in \mathcal{N}(r|0, \sigma_r^2))$ when using an unbiased estimator. In this case, the expectation of the estimation error vector vanishes $E[\mathbf{r}_0] = \mathbf{0}$ and the expectation of $\mathbf{r}_0\mathbf{r}_0^T$ is the identity matrix scaled by the variance of the estimation error $E[\mathbf{r}_0\mathbf{r}_0^T] = \sigma_e^2\mathbb{I}$.

The covariance of the solution $\mathbf{\Sigma}_{\hat{\mathbf{x}}\hat{\mathbf{x}}}$ is the expectation of the squared distances from the mean

$$\mathbf{\Sigma}_{\hat{\mathbf{x}}\hat{\mathbf{x}}} = E[(\hat{\mathbf{x}} - E(\hat{\mathbf{x}}))(\hat{\mathbf{x}} - E(\hat{\mathbf{x}}))^T] \tag{C.20}$$

When using an unbiased estimator ($E[\hat{\mathbf{x}}] = \mathbf{x}_0$) equation C.20 simplifies to

$$\mathbf{\Sigma}_{\hat{\mathbf{x}}\hat{\mathbf{x}}} = E[\hat{\mathbf{x}}\hat{\mathbf{x}}^T] - \mathbf{x}_0\mathbf{x}_0^T \tag{C.21}$$

Two different methods for the determination of the solution covariance are described next.

Backward Propagation of Covariance

Given the covariance matrix of the measurements $\mathbf{\Sigma}_{\mathbf{cc}}$ and a function $\mathbf{f}(\mathbf{x}) = \mathbf{c}$ mapping the parameters $\mathbf{x}$ to the measurements $\mathbf{c}$. The covariance matrix of the parameters $\mathbf{\Sigma}_{\mathbf{xx}}$ as computed by a maximum likelihood estimator is approximated by (Hartley and Zissermann, 2004)

$$\mathbf{\Sigma}_{\mathbf{xx}} \approx (\mathbf{J}^T\mathbf{\Sigma}_{\mathbf{cc}}^{-1}\mathbf{J})^{-1} \tag{C.22}$$

with the Jacobian $\mathbf{J} = \mathbf{J}(\mathbf{f})|_{\hat{\mathbf{x}}}$ of the function $\mathbf{f}$ at $\hat{\mathbf{x}}$. When the function $\mathbf{f}$ is linear, equation C.22 holds exactly. However, equation C.22 only holds for the non-over-parametrised case (i.e. $m = n$). When the parametrisation is redundant, the matrix $(\mathbf{J}^T\mathbf{\Sigma}_{\mathbf{cc}}^{-1}\mathbf{J})^{-1}$ is not invertible and equation C.22 cannot be used. Instead the pseudo inverse of $(\mathbf{J}^T\mathbf{\Sigma}_{\mathbf{cc}}^{-1}\mathbf{J})$ could be used

$$\mathbf{\Sigma}_{\mathbf{xx}} \approx (\mathbf{J}^T\mathbf{\Sigma}_{\mathbf{cc}}^{-1}\mathbf{J})^{-} \tag{C.23}$$

The pseudo inverse can be computed using the SVD of the matrix $(\mathbf{J}^T\mathbf{\Sigma}_{\mathbf{cc}}^{-1}\mathbf{J})$.

Linear Systems

For linear systems equation C.18 simplifies to

$$\mathbf{F}\mathbf{x} = \mathbf{c} \tag{C.24}$$

with coefficient matrix $\mathbf{F}$ and the measurement vector $\mathbf{c}$. When $(\mathbf{F}^T\mathbf{F})$ is invertible, a solution in a least square sense is given by

$$\hat{\mathbf{x}} = (\mathbf{F}^T\mathbf{F})^{-1}\mathbf{F}^T\mathbf{c} \tag{C.25}$$

and the covariance of the solution can be derived as follows:

$$\begin{aligned}
\mathbf{\Sigma}_{xx} &= E[\hat{\mathbf{x}}\hat{\mathbf{x}}^T] - \mathbf{x}_0\mathbf{x}_0^T \\
&= E\big[\big((\mathbf{F}^T\mathbf{F})^{-1}\mathbf{F}^T\mathbf{c}\big)\big((\mathbf{F}^T\mathbf{F})^{-1}\mathbf{F}^T\mathbf{c}\big)^T\big] - \mathbf{x}_0\mathbf{x}_0^T \\
&= E\big[\big((\mathbf{F}^T\mathbf{F})^{-1}\mathbf{F}^T\mathbf{c}\big)\big(\mathbf{c}^T\mathbf{F}(\mathbf{F}^T\mathbf{F})^{-1}\big)\big] - \mathbf{x}_0\mathbf{x}_0^T \\
&= (\mathbf{F}^T\mathbf{F})^{-1}\mathbf{F}^T E[\mathbf{c}\mathbf{c}^T]\mathbf{F}(\mathbf{F}^T\mathbf{F})^{-1} - \mathbf{x}_0\mathbf{x}_0^T \\
&= (\mathbf{F}^T\mathbf{F})^{-1}\mathbf{F}^T E\big[\big(\mathbf{F}\mathbf{x}_0 - \mathbf{r}_0\big)\big(\mathbf{F}\mathbf{x}_0 - \mathbf{r}_0\big)^T\big]\mathbf{F}(\mathbf{F}^T\mathbf{F})^{-1} - \mathbf{x}_0\mathbf{x}_0^T \\
&= (\mathbf{F}^T\mathbf{F})^{-1}\mathbf{F}^T E\big[\mathbf{F}\mathbf{x}_0\mathbf{x}_0^T\mathbf{F}^T - \mathbf{r}_0\mathbf{x}_0^T\mathbf{F}^T - \mathbf{F}\mathbf{x}_0\mathbf{r}_0^T + \mathbf{r}_0\mathbf{r}_0^T\big]\mathbf{F}(\mathbf{F}^T\mathbf{F})^{-1} - \mathbf{x}_0\mathbf{x}_0^T \\
&= \mathbf{x}_0\mathbf{x}_0^T - (\mathbf{F}^T\mathbf{F})^{-1}\mathbf{F}^T E[\mathbf{r}_0\mathbf{r}_0^T]\mathbf{F}(\mathbf{F}^T\mathbf{F})^{-1} - \mathbf{x}_0\mathbf{x}_0^T \\
&= \sigma_e^2(\mathbf{F}^T\mathbf{F})^{-1}
\end{aligned} \tag{C.26}$$

The variance of the estimation error vector σ_e is however generally unknown and particularly not equal to the variance of the residual error vector. For example, in case a minimum number of measurements is made, the residual error vector is always the zero vector even though the estimation error does not vanish when the measurements are corrupted by noise.

With redundant measurements, the variance of the estimation error σ_e^2 can, however, be estimated using the residual error vector $\hat{\mathbf{r}}$ (McGlone, 2004)

$$\hat{\sigma}_e^2 = \frac{\hat{\mathbf{r}}^T\Sigma_{\mathbf{cc}}^{-1}\hat{\mathbf{r}}}{R} \qquad R = m - n \tag{C.27}$$

with the covariance matrix of the observations $\Sigma_{\mathbf{cc}}$ and the redundancy R. The redundancy is given by the number of observations m minus the number of unknows (parameters) n.

C.3 Jacobians for Essential Matrix Estimation

Parameter estimation often relies on Jacobians of the function $\mathbf{f}$. The specific case of essential matrix estimation is investigated here. The essential matrix is parametrised by a unit vector and a orientation quaternion. If the parametrisation of the essential matrix is given by a unit vector $\boldsymbol{e} = (e_x, e_y, e_z)^T$ describing the epipole and a unit quaternion

$\boldsymbol{q} = (q_w, q_x, q_y, q_z)^T$ describing the relative orientation and if $\boldsymbol{x}_1 = (x_1, y_1, w_1)^T$ and $\boldsymbol{x}_2 = (x_2, y_2, w_2)^T$ is a pair of corresponding 2D image points, the Jacobians of the essential constraint $g(\boldsymbol{e}, \boldsymbol{q}, \boldsymbol{x}_1, \boldsymbol{x}_2) = \boldsymbol{x}_2^T [\boldsymbol{e}]_\times \mathbf{R}(\boldsymbol{q}) \boldsymbol{x}_1 = 0$ are given by:

$$\frac{\partial g}{\partial \boldsymbol{x}_1} = \boldsymbol{x}_2^T [\boldsymbol{e}]_\times \mathbf{R}(\boldsymbol{q}) \qquad \frac{\partial g}{\partial \boldsymbol{x}_2} = ([\boldsymbol{e}]_\times \mathbf{R}(\boldsymbol{q}) \boldsymbol{x}_1)^T \tag{C.28}$$

$$\frac{\partial g}{\partial \boldsymbol{e}} = \begin{pmatrix} \boldsymbol{x}_2^T [(1,0,0)^T]_\times \mathbf{R}(\boldsymbol{q}) \boldsymbol{x}_1 & \boldsymbol{x}_2^T [(0,1,0)^T]_\times \mathbf{R}(\boldsymbol{q}) \boldsymbol{x}_1 & \boldsymbol{x}_2^T [(0,0,1)^T]_\times \mathbf{R}(\boldsymbol{q}) \boldsymbol{x}_1 \end{pmatrix} \tag{C.29}$$

$$\frac{\partial g}{\partial \boldsymbol{q}} = 2 \boldsymbol{x}_2^T [\boldsymbol{e}]_\times \mathbf{B} \tag{C.30}$$

with the matrix $\mathbf{B}$

$$\mathbf{B} = \begin{pmatrix} \mathbf{b}_1 & \mathbf{b}_2 & \mathbf{b}_3 & \mathbf{b}_4 \end{pmatrix} \tag{C.31}$$

consisting of the column vectors

$$\begin{aligned} \mathbf{b}_1 &= \begin{pmatrix} q_w x_1 - q_z y_1 + q_y z_1 \\ q_z x_1 + q_w y_1 - q_x z_1 \\ q_x y_1 - q_y x_1 + q_w z_1 \end{pmatrix} & \mathbf{b}_2 &= \begin{pmatrix} q_x x_1 + q_y y_1 + q_z z_1 \\ q_y x_1 - q_x y_1 - q_w z_1 \\ q_z x_1 + q_w y_1 - q_x z_1 \end{pmatrix} \\ \mathbf{b}_3 &= \begin{pmatrix} q_x y_1 - q_y x_1 + q_w z_1 \\ q_x x_1 + q_y y_1 + q_z z_1 \\ q_z y_1 - q_w x_1 - q_y z_1 \end{pmatrix} & \mathbf{b}_4 &= \begin{pmatrix} q_x z_1 - q_w y_1 - q_z x_1 \\ q_w x_1 - q_z y_1 + q_y z_1 \\ q_x x_1 + q_y y_1 + q_z z_1 \end{pmatrix} \end{aligned} \tag{C.32}$$

D Probability Theory

Only continuous random variables are considered in this section.

D.1 Basics

Cumulative Distribution Function

For a continuous random variable x, the *cumulative distribution function* (cdf) $P_x(\nu)$ represents the probability that x takes on a value less or equal to ν. The cdf is continuous, monotonically increasing and the limits at $\pm\infty$ are given by

$$\lim_{\nu\to-\infty} P_x(\nu) = 0.0 \qquad \text{and} \qquad \lim_{\nu\to\infty} P_x(\nu) = 1.0 \tag{D.1}$$

When the cdf is continuously differentiable, its derivative is the probability density function. However, not every cdf has an associated probability density function. For example, the infinite decimal number whose digits are generated by a dice with six sides has a cumulative distribution function, but no associated probability density function exists, because of the missing digits $7, 8, 9$ and 0.

Probability Density Function

A *probability density function* (pdf) $p(x)$ describes the probability distribution of a random variable x. The connection between a continuously differentiable cdf $P(X)$ and the associated pdf $p(x)$ is given by

$$\int_{-\infty}^{x} p(\nu)d\nu = P(x) \qquad \text{or} \qquad p(x) = \frac{dP(x)}{dx} \tag{D.2}$$

The integral of any pdf must be one

$$\int_{-\infty}^{\infty} p(x)dx = 1.0 \tag{D.3}$$

Expectation Value

The *expectation value* of a random variable x with pdf $p(x)$ is given by

$$E[x] = \int_{\infty}^{\infty} xp(x)\mathrm{d}x \tag{D.4}$$

The expectation value of a function $g(x)$ is given by

$$E[g(x)] = \int_{-\infty}^{\infty} p(x)g(x)\mathrm{d}x \tag{D.5}$$

The expectation operator is a linear operator

$$E[ax + by] = aE[x] + bE[y]$$

with constants $a, b \in \mathbb{R}$ and random variables x, y. The expectation of a constant c is the constant c itself

$$E[c] = c$$

and hence

$$E[E[x]] = E[x]$$

Momentum:

The n^{th} *momentum* of a random variable x with pdf $p(x)$ is given as

$$E[x^n] = \int_{-\infty}^{\infty} p(x)x^n\mathrm{d}x \tag{D.6}$$

Therefore the first momentum of a random variable is its mean.

$$\overline{x} = E[x] \tag{D.7}$$

Central Momentum:

The n^{th} *central momentum* of a random variable x with pdf $p(x)$ is given by

$$E[(x - \overline{x})^n] == \int_{-\infty}^{\infty} p(x)(x - \overline{x})^n\mathrm{d}x \tag{D.8}$$

Therefore the second central momentum of a random variable is its covariance matrix

$$\begin{aligned}
\mathbf{\Sigma}_{xx} &= E[(x - \overline{x})^T(x - \overline{x})] \\
&= E[x^T x - x^T\overline{x} - \overline{x}^T x + \overline{x}^T\overline{x}] \\
&= E[x^T x - x^T E[x] - E[x]^T x + E[x]^T E[x]] \\
&= E[x^T x] - E[x]^T E[x] - E[x]^T E[x] + E[x]^T E[x] \\
&= E[x^T x] - E[x]^T E[x]
\end{aligned} \tag{D.9}$$

Conditional Probability

The *conditional probability* $p(x|y)$ is the probability of x under the condition that y holds or the probability of x given y. It can be calculated as follows:

$$p(x|y) = \frac{p(x \wedge y)}{p(y)} \tag{D.10}$$

where $p(x \wedge y)$ or $p(x, y)$ is the joint probability of x and y and $p(x \wedge y) = p(x, y) = p(x \text{ and } y)$.

Margin

The *margin* $p(x)$ of a joint distribution $p(x, y)$ describes the probability of the outcome of x independent of y and can be calculated as follows:

$$p(x) = \int p(x, y)\mathrm{d}y = \int p(x|y)p(y)\mathrm{d}y \tag{D.11}$$

The variables x and y can also be vectors. In this case the integral must be replaced by a multidimensional integral. In the discrete case the integrals turn into sums.

Bayes Law

Bayes' Law states the following:

$$p(x|y) = \frac{p(y|x)p(x)}{p(y)} \tag{D.12}$$

It can be derived as follows:

$$\begin{aligned} & p(x \wedge y) && = p(y \wedge x) \\ \Longleftrightarrow \quad & p(x|y)p(y) && = p(y|x)p(x) \\ \Rightarrow \quad & p(x|y) && = \frac{p(y|x)p(x)}{p(y)} \\ \Rightarrow \quad & p(x|y) && = \frac{p(y|x)p(x)}{\int p(y|x)p(x)\mathrm{d}x} \end{aligned} \tag{D.13}$$

Quantile

The α-*quantile* q of a probability distribution $p(x)$ has the following property: If a value is chosen randomly according to the distribution, the probability to get a value lower than q is $\alpha\%$. Figure D.1 illustrates this relation.

$$\alpha = \int_{-\infty}^{q} p(x)dx \tag{D.14}$$

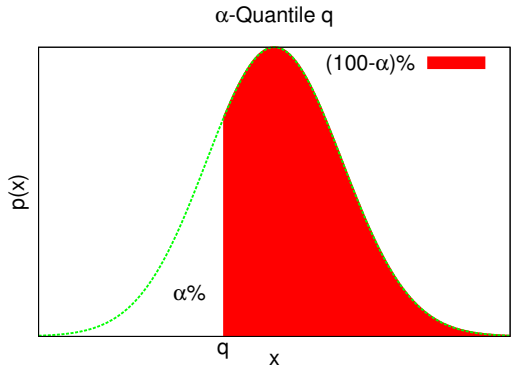


Figure D.1: Visualisation of the α-quantile q of a normal distribution. The area of the probability distribution function $p(x)$ between minus infinity and q is given by α.

and hence the α-quantile q is given by the inverse the cumulative distribution function $P(x)$ at α

$$q = P^{-1}(\alpha) \qquad \alpha = P(q) \tag{D.15}$$

The 0.5-quantile (50%-quantile) is also called *median*.

D.2 Important Probability Distributions

Uniform Distribution

The uniform distribution assigns the same probability density to each point. Because of the norm-one characteristic of each pdf, a uniform distribution can only be defined over a restricted interval. Hence, the uniform distribution is mainly used for random variables describing physical states which are naturally bound to an interval, e.g. angles. Figure D.2 illustrates the pdf of uniformly distributed random variable describing an angle.

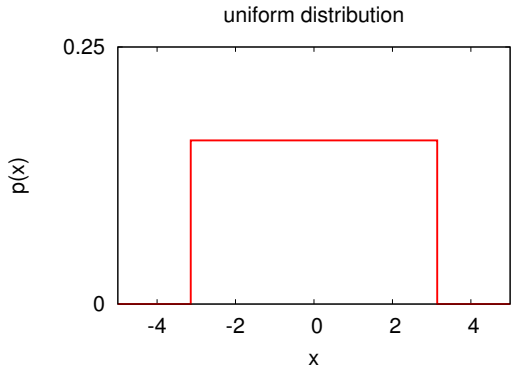


Figure D.2: Probability density function of a uniformly distributed random variable used to describe an angle between two vectors.

Normal Distribution

The probability density function (pdf) of the normal distribution is given by (Figueiredo, 2004)

$$p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{D.16}$$

The normal distribution is completely determined by its first two moments, namely the mean μ and the variance σ^2. The pdf $p(x)$ of the one-dimensional normal distribution with mean μ and variance σ^2 is abbreviated by $\mathcal{N}(x|\mu, \sigma^2)$.

The pdf of the n-dimensional normally distributed vector $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)^T$ is given by (Figueiredo, 2004)

$$p(\boldsymbol{x}) = \frac{1}{\sqrt{(2\pi)^n \det(\boldsymbol{\Sigma_{xx}})}} e^{-\frac{1}{2}(\boldsymbol{x}-\overline{\boldsymbol{x}})^T \boldsymbol{\Sigma}_{\boldsymbol{xx}}^{-1}(\boldsymbol{x}-\overline{\boldsymbol{x}})} \tag{D.17}$$

It is also determined by its mean $\overline{\boldsymbol{x}} = (\overline{x}_1, \overline{x}_2, \ldots, \overline{x}_n)^T$ and covariance matrix $\boldsymbol{\Sigma_{xx}}$. The pdf is abbreviated by $\mathcal{N}(\boldsymbol{x}\,|\overline{\boldsymbol{x}}, \boldsymbol{\Sigma_{xx}})$. The covariance matrix $\boldsymbol{\Sigma_{xx}}$ consists of the variances $\sigma^2_{x_i x_i}$ and covariances $\sigma^2_{x_i x_j}$ of the vector entries x_i

$$\boldsymbol{\Sigma_{xx}} = \begin{pmatrix} \sigma^2_{x_1 x_1} & \cdots & \sigma^2_{x_1 x_n} \\ \sigma^2_{x_2 x_1} & \cdots & \sigma^2_{x_2 x_n} \\ \vdots & \ddots & \vdots \\ \sigma^2_{x_n x_1} & \cdots & \sigma^2_{x_n x_n} \end{pmatrix} \tag{D.18}$$

Each covariance matrix has real entries, is symmetric and positive definite and thus has positive eigenvalues. When the vector entries x_i are statistically independent (i.e. $\sigma^2_{x_i x_j} = 0 \;\forall\; i \neq j$), the covariance matrix is a diagonal matrix.

Beta Distribution

The beta distribution $B(x|\alpha, \beta)$ with parameters α and β is defined as (Figueiredo, 2004)

$$B(x|\alpha, \beta) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} x^{\alpha-1}(1-x)^{\beta-1} \tag{D.19}$$

Its mean $\overline{x}$ and variance σ^2 are given by (Figueiredo, 2004)

$$\overline{x} = \frac{\alpha}{\alpha+\beta} \tag{D.20}$$

and

$$\sigma^2 = \frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)} \tag{D.21}$$

The beta distribution can be used to model the probabilities in bounded intervals and is a conjugate prior for Bernoulli distributions in Bayesian estimation theory (Figueiredo, 2004). It is hence also a conjugate prior for itself. Figure D.3 shows the beta distribution for different parameters α and β.

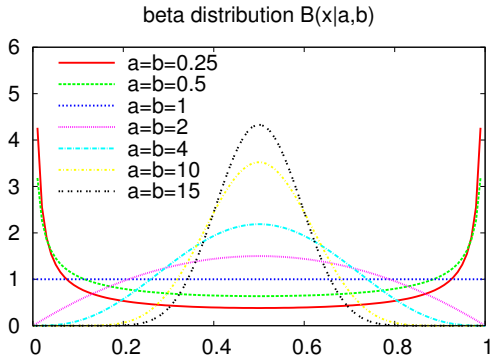


Figure D.3: Beta distribution for different parameters α and β

χ^2 Distribution

Given k independent normally distributed random variables $Y_i \in \mathcal{N}(Y_i|0,1)$, the χ^2 distribution is the probability density function of the sum of the squared Y_i

$$x = \sum_{i=1}^{k} Y_i^2 \tag{D.22}$$

The χ^2 (chi square) distribution (Kreyszig, 1965) with k degrees of freedom is defined as

$$p_{\chi^2}(x|k) = \frac{x^{k/2-1}e^{-x/2}}{\Gamma(k/2)2^{k/2}} \tag{D.23}$$

The mean of the χ^2 distribution is k and the variance is given by $2k$. The density takes its maximum at $k-2$ (Kanatani, 2005). Figure D.4 shows the χ^2 distribution for different degrees of freedom k.

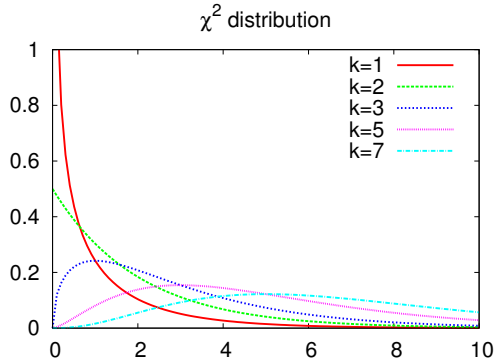


Figure D.4: Chi square distribution for different degrees of freedom k

D.3 Error Propagation

Given a possibly nonlinear and multidimensional function $\mathbf{y} = \mathbf{f}(\mathbf{x})$ and an input vector $\mathbf{x}$ with known probability distribution $\mathbf{p}(\mathbf{x})$, the question about the probability distribution of $\mathbf{y}$ arises immediately. Three different approaches for the propagation of the probability distribution $\mathbf{p}(\mathbf{x})$ through $\mathbf{f}(\mathbf{x})$ are introduced next.

Gauss The Gaussian error propagation holds rigorously for linear functions $\mathbf{f}(\mathbf{x}) = \mathbf{F}\mathbf{x}$. It propagates the first two moments of a distribution (i.e. the mean and the variance) and thus holds rigorously for normal distributions $\mathbf{p}(\mathbf{x}) = \mathcal{N}(\mathbf{x}|\overline{\mathbf{x}}, \mathbf{\Sigma_{xx}})$. It is, however, commonly used as an approximation for nonlinear functions and/or non-normal distributions.

Unscented Transform The unscented transform (UT) describes the distribution using a set of *sigma points* such that the first two moments of the distribution are preserved. It imposes no restrictions on the functions, but only propagates the first two moments of the distribution while some flavours of the UT at the same time minimise third order moments (skew).

Monte Carlo Monte Carlo propagation represents the pdf by a large number of samples drawn from it. It imposes neither restrictions on the distribution nor on the function. The computational burden of this method is very high due to the large number of samples required for accurate propagation.

Gaussian Error Propagation:

The function $\mathbf{f}(\mathbf{x})$ can be approximated by the linear parts of a Taylor series developed around the mean $\overline{\mathbf{x}}$

$$\mathbf{f}(\overline{\mathbf{x}} + \mathbf{e}) = \mathbf{f}(\overline{\mathbf{x}}) + \mathbf{J}(\mathbf{f})|_{\overline{\mathbf{x}}}\, \mathbf{e} + O(\mathbf{e}^2) \approx \mathbf{f}(\overline{\mathbf{x}}) + \mathbf{J}(\mathbf{f})|_{\overline{\mathbf{x}}}\, \mathbf{e} \tag{D.24}$$

with the Jacobian $\mathbf{J}(\mathbf{f})|_{\overline{\mathbf{x}}}$ of the function $\mathbf{f}$ developed around the mean $\overline{\mathbf{x}}$ and a small error vector $\mathbf{e}$. If $\mathbf{e}$ is drawn from a zero mean normal distribution $\mathbf{e} \in \mathcal{N}(\mathbf{e}|\mathbf{0}, \mathbf{\Sigma_{xx}})$, the expectation $E[\mathbf{e}]$ vanishes and the expectation $E[\mathbf{e}\mathbf{e}^T]$ is given by $\mathbf{\Sigma_{xx}}$. Computing the mean of the function $\overline{\mathbf{y}} = \overline{\mathbf{f}}(\mathbf{x})$ using the approximation of eq. D.24 results in

$$\begin{aligned} \overline{\mathbf{y}} = \overline{\mathbf{f}}(\mathbf{x}) &= E\big[\mathbf{f}(\mathbf{x})\big] \\ &\approx E\big[\mathbf{f}(\overline{\mathbf{x}}) + \mathbf{J}(\mathbf{f})|_{\overline{\mathbf{x}}}\, \mathbf{e}\big] \\ &= E\big[\mathbf{f}(\overline{\mathbf{x}})\big] + \mathbf{J}(\mathbf{f})|_{\overline{\mathbf{x}}}\, E\big[\mathbf{e}\big] \\ &= E\big[\mathbf{f}(\overline{\mathbf{x}})\big] \\ &= \mathbf{f}(\overline{\mathbf{x}}) \end{aligned} \tag{D.25}$$

The covariance matrix of the result $\mathbf{\Sigma_{yy}}$ can be approximated by

$$\begin{aligned}
\mathbf{\Sigma_{yy}} &= E\big[\big(\mathbf{f}(\mathbf{x}) - \overline{\mathbf{y}}\big)\big(\mathbf{f}(\mathbf{x}) - \overline{\mathbf{y}}\big)^T\big] \\
&\approx E\big[\big(\mathbf{f}(\mathbf{x}) - \mathbf{f}(\overline{\mathbf{x}})\big)\big(\mathbf{f}(\mathbf{x}) - \mathbf{f}(\overline{\mathbf{x}})\big)^T\big] \\
&= E\big[\big(\mathbf{J}(\mathbf{f})|_{\overline{\mathbf{x}}}\ \mathbf{e}\big)\big(\mathbf{J}(\mathbf{f})|_{\overline{\mathbf{x}}}\ \mathbf{e}\big)^T\big] \\
&= \mathbf{J}(\mathbf{f})|_{\overline{\mathbf{x}}}\ E\big[\mathbf{e}\mathbf{e}^T\big]\ \mathbf{J}(\mathbf{f})|_{\overline{\mathbf{x}}}^T \\
&= \mathbf{J}(\mathbf{f})|_{\overline{\mathbf{x}}}\ \mathbf{\Sigma_{xx}}\ \mathbf{J}(\mathbf{f})|_{\overline{\mathbf{x}}}^T
\end{aligned} \tag{D.26}$$

When the function $\mathbf{f}$ is linear, the error in the approximation of the function by the Taylor series vanishes and the equations D.25 and D.26 hold rigorously.

Unscented Transform:

The *unscented transform* (UT) is based on the following intuition: "With a fixed number of parameters it should be easier to approximate a Gaussian distribution than it is to approximate an arbitrary nonlinear function" (Julier et al., 1995). The source distribution is represented by a set of *sigma points*. Each sigma point is propagated through the nonlinear system function $\mathbf{f}$, and characteristics of the target distribution can be computed using these transformed sigma points. The main advantages of the unscented transform are: (i) Easy implementation, because the nontrivial derivation of the Jacobians, which are necessary for linear error propagation, is circumvented, and (ii) no linearisation of the system function $\mathbf{f}$.

The *symmetric unscented transform* (Julier et al., 1995) uses a set of $2n + 1$ *sigma points* $\mathbf{x}_i$ with associated weights w_i to represent the n-dimensional distribution $\mathbf{p}(\mathbf{x})$. The sigma points are chosen such that they share the mean and the second central moment (i.e. the covariance matrix) $\mathbf{\Sigma_{xx}}$ with $\mathbf{p}(\mathbf{x})$. The sigma points can be computed using the positive and negative columns (or rows) $\mathbf{s}_i$ of the square roots of the input covariance matrix $\sqrt{(n+\kappa)\mathbf{\Sigma_{xx}}} = (\mathbf{s}_1, \mathbf{s}_2, \ldots)$.

$$\mathbf{x}_i = \begin{cases} \overline{\mathbf{x}} & \text{if } i = 0 \\ \overline{\mathbf{x}} + \mathbf{s}_i & \text{if } i \neq 0 \wedge i \leq n \\ \overline{\mathbf{x}} - \mathbf{s}_{i-n} & \text{if } i \neq 0 \wedge i > n \end{cases} \qquad \text{with weights} \qquad w_i = \begin{cases} \frac{\kappa}{n+\kappa} & \text{if } i = 0 \\ \frac{1}{2(n+\kappa)} & \text{otherwise} \end{cases} \tag{D.27}$$

where κ copies of the mean $\overline{\mathbf{x}}$ can be included in the sigma points. Any of the infinite number of the matrix square roots can be chosen. If the orthogonal matrix square root is chosen, the sigma points lie in the direction of the eigenvectors of $\mathbf{\Sigma_{xx}}$ from $\overline{\mathbf{x}}$. Each sigma point is transformed separately $\mathbf{y}_i = \mathbf{f}(\mathbf{x}_i)$, and the approximation of mean $\overline{\mathbf{y}}$ and

covariance matrix $\mathbf{\Sigma_{yy}}$ of $\mathbf{y}$ are given by

$$\overline{\mathbf{y}} = \sum_{i=0}^{2n} w_i \mathbf{y}_i \tag{D.28}$$

$$\mathbf{\Sigma_{yy}} = \sum_{i=0}^{2n} w_i (\mathbf{y}_i - \overline{\mathbf{y}})(\mathbf{y}_i - \overline{\mathbf{y}})^T \tag{D.29}$$

If $\mathbf{f}$ can be expressed using a Taylor series, it can be shown that the propagation is correct up to the fourth term of the Taylor series (Julier et al., 1995).

Example: Triangulation Using the Symmetric Unscented Transform: Let two 2D-points $\boldsymbol{x}_1 \in \mathbb{R}^2$ and $\boldsymbol{x}_2 \in \mathbb{R}^2$ and the associated camera poses $\mathbf{p}_1$ and $\mathbf{p}_2$ be given. The camera poses are given as parameter vectors, e.g. by the concatenation of the orientation quaternion $\mathbf{q}$ and the Euclidean centre of projection $\boldsymbol{C}$, and are hence from $\mathbb{R}^7$. Let the uncertainties of the points be given by $\mathbf{\Sigma}_{\boldsymbol{x}_1\boldsymbol{x}_1}$ and $\mathbf{\Sigma}_{\boldsymbol{x}_2\boldsymbol{x}_2}$ and the uncertainties of the camera poses be given by $\mathbf{\Sigma}_{\mathbf{p}_1\mathbf{p}_1}$ and $\mathbf{\Sigma}_{\mathbf{p}_2\mathbf{p}_2}$. Let the concatenation of $\boldsymbol{x}_1$, $\boldsymbol{x}_2$, $\mathbf{p}_1$ and $\mathbf{p}_2$ be denoted by $\mathbf{a}_{\text{in}} \in \mathbb{R}^{18}$

$$\mathbf{a}_{\text{in}} = \begin{pmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \\ \mathbf{p}_1 \\ \mathbf{p}_2 \end{pmatrix} \tag{D.30}$$

Under the assumption that the points and camera poses are independently distributed, the covariance matrix $\mathbf{\Sigma}_{\mathbf{a}_{\text{in}}\mathbf{a}_{\text{in}}}$ of $\mathbf{a}_{\text{in}}$ is given by

$$\mathbf{\Sigma}_{\mathbf{a}_{\text{in}}\mathbf{a}_{\text{in}}} = \begin{pmatrix} \mathbf{\Sigma}_{\boldsymbol{x}_1\boldsymbol{x}_1} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{\Sigma}_{\boldsymbol{x}_2\boldsymbol{x}_2} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{\Sigma}_{\mathbf{p}_1\mathbf{p}_1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{\Sigma}_{\mathbf{p}_2\mathbf{p}_2} \end{pmatrix} \tag{D.31}$$

37 sigma points $\mathbf{a}_i$ are computed using the square roots of $\mathbf{\Sigma}_{\mathbf{a}_{\text{in}}\mathbf{a}_{\text{in}}}$ as described by equation D.27. Each of these sigma points can be decomposed into two 2D-points (using the first 4 entries) and two camera matrices (using the last 14 entries). The two 2D-points and the two camera poses are used to triangulate a 3D-point $\boldsymbol{X}_i$, resulting in an overall number of 37 3D-points. A weight is assigned to each $\boldsymbol{X}_i$ according to equation D.27. Mean and covariance matrix are computed from the 3D-points $\boldsymbol{X}_i$ and their weight, resulting in the final 3D-point and the associated covariance matrix (equations D.28 and D.29). □

The *simplex unscented transform* reduces the number of sigma points to a minimum of $n + 1$ to match mean and covariance of the input distribution while at the same time minimising higher order moments (Julier and Uhlmann, 2002a). The choice of the simplex sigma points guarantees, that the skew of the distribution which they represent, is zero.

This leads to the fact, that the simplex sigma points are distributed non-symmetrically. The minimal skew set of simplex sigma points is recursively defined: The set of simplex sigma points $\mathbf{x}_i^n$ in dimension n is based on the set of simplex sigma points in dimension $n-1$. In one dimension, a simplex set of sigma points $\mathbf{x}_i^1$ for the standard (i.e. $\overline{\mathbf{x}} = 0$ and $\sigma_{\mathbf{xx}} = 1$) normal distribution[2] is given by

$$\mathbf{x}_0^1 = 0 \qquad \mathbf{x}_1^1 = -\frac{1}{2\sqrt{w_1}} \qquad \mathbf{x}_2^1 = +\frac{1}{2\sqrt{w_1}} \tag{D.32}$$

with the weights

$$w_1 = \frac{1 - w_0}{2} \qquad w_2 = \frac{1 - w_0}{2} \tag{D.33}$$

where w_0 is a free parameter which can be exploited in the scaled unscented transform. The set of simplex sigma points $\mathbf{x}_i^{n+1}$ in $(n+1)$-dimensional space can be computed as follows:

1. Choose w_0
2. Compute weight sequence

$$w_i = \begin{cases} \frac{1-w_0}{2^n} & \text{if } i = 1 \\ w_1 & \text{if } i = 2 \\ 2^{i-1} w_1 & \text{for } i = 3, \ldots, n+1 \end{cases} \tag{D.34}$$

3. Initialise vector sequence (see equation D.32)
4. Expand vector sequence for $j = 2, \ldots, n$ according to

$$\mathbf{x}_i^{j+1} = \begin{cases} \begin{bmatrix} \mathbf{x}_0^j \\ 0 \end{bmatrix} & \text{if } i = 0 \\ \begin{bmatrix} \mathbf{x}_i^j \\ -\frac{1}{\sqrt{2w_j}} \end{bmatrix} & \text{for } i = 1, \ldots, j \\ \begin{bmatrix} \mathbf{0} \\ \frac{1}{\sqrt{2w_j}} \end{bmatrix} & \text{if } i = j+1 \end{cases} \tag{D.35}$$

The *scaled unscented transform* deals with non-continuous functions using the simplex unscented transform. As can be seen from equation D.35, the radius of the bounding sphere of all sigma points scales linearly with the input space dimension n, leading to

[2] A random variable $\mathbf{z}'$ with mean $\mathbf{0}$ and covariance $\mathbf{I}$ can be transformed to a random variable $\mathbf{z}$ with mean $\overline{\mathbf{x}}$ and covariance $\mathbf{\Sigma_{xx}}$ using the linear transformation $\mathbf{z} = \overline{\mathbf{x}} + \sqrt{\mathbf{\Sigma_{xx}}}\,\mathbf{z}'$ with the matrix square root $\sqrt{\mathbf{\Sigma_{xx}}}$.

difficulties for many kind of nonlinearities. Julier and Uhlmann (2002b) suggested the scaled unscented transform which enables the restriction of the bounding sphere of the simplex sigma points by using an arbitrary scale factor α. The scaled unscented transform can, however, also be used based on the symmetric set of sigma points. The computational complexity of the unscented transform is not affected by the scaling. The scaled sigma points $\mathbf{x}_i'$ can be computed using the original sigma points $\mathbf{x}_i$ and the scale factor α

$$\mathbf{x}_i' = \mathbf{x}_0 + \alpha\,(\mathbf{x}_i - \mathbf{x}_0) \tag{D.36}$$

The weights w_i' of the scaled sigma points must be adopted using the original weights w_i

$$w_i' = \begin{cases} \frac{w_0}{\alpha^2} + \left(1 - \frac{1}{\alpha^2}\right) & \text{if } i = 0 \\ \frac{w_i}{\alpha^2} & \text{otherwise} \end{cases} \tag{D.37}$$

resulting in a slightly adopted computation of the moments of the target distribution from the p sigma points

$$\overline{\mathbf{y}} = \sum_{i=0}^{p} w_i \mathbf{y}_i \tag{D.38}$$

$$\mathbf{\Sigma_{yy}} = (1 + \beta - \alpha^2)(\mathbf{y}_0 - \overline{\mathbf{y}})(\mathbf{y}_0 - \overline{\mathbf{y}})^T + \sum_{i=0}^{p} w_i (\mathbf{y}_i - \overline{\mathbf{y}})(\mathbf{y}_i - \overline{\mathbf{y}})^T \tag{D.39}$$

where β is an additional factor, which is used to incorporate higher order information. When the pdfs are Gaussian, $\beta = 2$ minimises the error in higher order terms.

Monte Carlo Propagation:

Monte Carlo techniques represent the uncertainty of the input vector by a very large number N of samples $\mathbf{x}_i$ drawn from the input distribution. Each of these samples is propagated through the function $\mathbf{y}_i = \mathbf{f}(\mathbf{x}_i)$, and the target distribution is represented by these propagated samples. The moments of the target distribution can be approximated using the target samples $\mathbf{y}_i$. The approximation of the mean and the covariance of the target distribution are given exemplarily

$$\overline{\mathbf{y}} = \sum_{i=1}^{N} \mathbf{y}_i \tag{D.40}$$

$$\mathbf{\Sigma_{yy}} = \sum_{i=1}^{N} (\mathbf{y}_i - \overline{\mathbf{y}})(\mathbf{y}_i - \overline{\mathbf{y}})^T \tag{D.41}$$

The accuracy of the distribution representation increases with the number of samples used for representation. The large number of samples required for accurate representation results in a poor computational performance.

Interestingly, it is not strictly necessary to use random numbers for Monte Carlo uncertainty propagation. Sequences of so called quasi-random numbers are sufficient for Monte Carlo sampling methods. These sequences are also called *low discrepancy sequences*. They do not consist of random numbers but have useful properties similar to random numbers, i.e. a sequence of quasi-random numbers $x_1, \ldots, x_n$ is almost uniformly distributed. They have the property that each sub-sequence $x_1, \ldots, x_m$ with $m < n$ is also almost uniformly distributed and the expanded sequence $x_1, \ldots, x_{n+1}$ is also almost uniformly distributed. Using sequences of quasi-random numbers, for example for Monte Carlo integration, even yields a better convergence performance (Morokoff and Caflisch, 1995) than using true random numbers.

D.4 Statistical Testing

In this section uncertainty is always represented by the first two moments of the underlying distributions. An important question is the problem of incidence, e.g. the incidence of a point and a line.

Mahalanobis Distance

The *Mahalanobis distance* of a random vector $\mathbf{x}$ with covariance matrix $\mathbf{\Sigma_{xx}}$ from the origin is given by (Hartley and Zissermann, 2004; Kanatani, 2005)

$$||\mathbf{x}||_{\mathbf{\Sigma}} = \mathbf{x}^T \Sigma^{-1} \mathbf{x} \tag{D.42}$$

If the covariance matrix is singular, the pseudo norm can be defined

$$||\mathbf{x}||_{\mathbf{\Sigma}} = \mathbf{x}^T \Sigma^{-} \mathbf{x} \tag{D.43}$$

with the (Moore-Penrose) generalised (or pseudo) inverse Σ^-. The pseudo inverse equals the inverse when Σ has full rank.

Given two normally distributed vectors $\mathbf{x}_1 \in \mathcal{N}(\mathbf{x}_1|\overline{\mathbf{x}}_1, \mathbf{\Sigma}_{\mathbf{x}_1\mathbf{x}_1})$ and $\mathbf{x}_2 \in \mathcal{N}(\mathbf{x}_2|\overline{\mathbf{x}}_2, \mathbf{\Sigma}_{\mathbf{x}_2\mathbf{x}_2})$, their difference $\mathbf{d}$ is also normally distributed with mean $\overline{\mathbf{d}} = \overline{\mathbf{x}}_1 - \overline{\mathbf{x}}_2$ and covariance matrix $\mathbf{\Sigma_{dd}} = \mathbf{\Sigma}_{\mathbf{x}_1\mathbf{x}_1} + \mathbf{\Sigma}_{\mathbf{x}_2\mathbf{x}_2}$. The Mahalanobis distance between two vectors $\mathbf{x}_1$ and $\mathbf{x}_2$ with associated covariance matrices $\mathbf{\Sigma}_{\mathbf{x}_1\mathbf{x}_1}$ and $\mathbf{\Sigma}_{\mathbf{x}_2\mathbf{x}_2}$ can hence be defined as

$$||\mathbf{x}_1 - \mathbf{x}_2||_{\mathbf{\Sigma}} = (\mathbf{x}_1 - \mathbf{x}_2)^T (\mathbf{\Sigma}_{\mathbf{x}_1\mathbf{x}_1} + \mathbf{\Sigma}_{\mathbf{x}_2\mathbf{x}_2})^{-} (\mathbf{x}_1 - \mathbf{x}_2) \tag{D.44}$$

The Mahalanobis distance is invariant to scale changes and translations. It is the same as the squared Euclidean distance when the sum of the covariance matrices $\mathbf{\Sigma}_{x_1\mathbf{x}_1} + \mathbf{\Sigma}_{\mathbf{x}_2\mathbf{x}_2}$ is the identity matrix. The Mahalanobis distance can be used as common distance measure for points with individual covariance matrices.

Incident Test or χ^2 Test

If $\mathbf{x}$ is a random variable with pdf $\mathcal{N}(\mathbf{x}|\mathbf{0}, \mathbf{\Sigma_{xx}})$, the quadratic form

$$r = \mathbf{x}^T \mathbf{\Sigma}_{\mathbf{xx}}^{-} \mathbf{x} \tag{D.45}$$

is a χ^2 distributed variable with k degrees of freedom (Kanatani, 2005), where k equals the rank of the covariance matrix $\mathbf{\Sigma_{xx}}$. Hence the Mahalanobis distance between two vectors (equation D.44) is χ^2 distributed providing a simple method for hypothesis testing. The probability p that a random variable vector $\mathbf{x}$ with zero mean and covariance $\mathbf{\Sigma_{xx}}$ satisfies

$$\mathbf{x}^T \mathbf{\Sigma}_{\mathbf{xx}}^{-} \mathbf{x} < t \tag{D.46}$$

for $t > 0$ is given by

$$p = \int_0^t p_{\chi^2}(s|k) \mathrm{d}s \tag{D.47}$$

An important task is testing the incidence hypothesis, in other words the decision if a random variable vector $\mathbf{x}_1 - \mathbf{x}_2$ has zero mean based on a single observation of Mahalanobis distance from the origin.

Fixing a significance level α, a threshold $t(\alpha, k)$ for the Mahalanobis distance can be defined such that

$$\alpha = \int_{t(\alpha,k)}^{\infty} p_{\chi^2,k}(s) \mathrm{d}s \tag{D.48}$$

The threshold t is given by the $(1-\alpha)$-quantile of the distribution and can hence be computed using the inverse of the cumulative distribution function $P_{\chi^2,k}$ of the χ^2 distribution

$$t(\alpha, k) = P_{\chi^2,k}^{-1}(1-\alpha) \tag{D.49}$$

The hypothesis is rejected when the Mahalanobis distance is greater than $t(\alpha, k)$ and accepted otherwise. The significance α provides the probability of false negatives, i.e. the probability that two vectors are declared non incident even though they are incident.

In the nomenclature of statistical testing theory: The H_0 *hypothesis* (vectors are incident: $\mathbf{x}_1 - \mathbf{x}_2 = \mathbf{0}$) is tested against the *alternate hypothesis* H_a (points are not incident: $\mathbf{x}_1 - \mathbf{x}_2 \neq \mathbf{0}$) where the threshold t is given by the *(1-α)-quantile* of the distribution.

Bibliography

P. Anadan and M. Irani. Factorization with uncertainty. *IJCV*, 49(2/3):101–116, 2002.

A. Argyros and S. Orphanoudakis. Independent 3D motion detection based on depth elimination in normal flow fields. In *Proc. CVPR*, pages 672–677, 1997.

S. Avidan and A. Shashua. Trajectory triangulation: 3D reconstruction of moving points from a monocular image sequence. *TPAMI*, 22(4):348–357, April 2000.

J. Barron, D. Fleet, and S. Beauchemin. Performance of optical flow techniques. Technical report, Dept. of Computer Science, University of Western Ontario, London, Ontario, N6A 5B7, 1994.

C. Beder. *Grouping Uncertain Oriented Projective Geometric Entities with Application to Automatic Building Reconstruction*. PhD thesis, Photogrammetry Department, Institute for Geodesy and Geoinformation, Bonn University, Germany, Nussallee 15, 53115 Bonn, Germany, Jan 2007.

J. Bezdek, J. Keller, R. Krisnapuram, and N. R. Pal. *Fuzzy Models and Algorithms for Pattern Recognition and Image Processing*. Kluwer Academic Press, 1999.

M. Black and D. Fleet. Probabilistic detection and tracking of motion discontinuities. In *Proc. ICCV*, 1999.

J. Bouguet. Camera calibration toolbox for matlab, 1998. URL `http://www.vision.caltech.edu/bouguetj/calib\_doc/index.html`.

S. Carlsson and J. Eklundh. Object detection using model based prediction and motion parallax. In *Proc. ECCV*, pages 297–306, 1990.

S. Chatterjee and A. Hadi. *Sensitivity analysis in linear regression*. Wiley, 1988.

S. Christy and R. Horaud. Euclidean shape and motion from multiple perspective views by affine iterations. *TPAMI*, 18(11):1098–1104, November 1996.

R. Cipolla and A. Blake. Surface orientation and time to contact from image divergence and deformation. In *ECCV*, pages 187–202, 1992.

J. Clarke and A. Zisserman. Detection and tracking of independent motion. *IVC*, 14: 565–572, 1996.

C. Colombo and A. del Bimbo. Generalized bounds for time to collision from first-order image motion. In *Proc. ICCV*, pages 220–226, 1999.

J. Costeira and T. Kanade. A multibody factorization method for independently moving objects. *IJCV*, 29(4):159–179, 1998.

D. Cremers and S. Soatto. Variational space-time motion segmentation. In *Proc. ICCV*, pages 886–893, 2003.

K. Daniilidis and I. Thomas. Decoupling the 3D motion space by fixation. In *Proc ECCV*, volume 1, pages 685–696, Cambridge, UK, 1996.

F. Dellaert. The expectation maximization algorithm. Technical Report GIT-GVU-02-20, College of Computing, Georgia Institute of Technology, 2002.

A. Doucet, N. D. Freitas, and N. Gordon, editors. *Sequential Monte Carlo Methods in Practice*. Springer, 2001.

C. Elkan. Naive bayesian learning. Technical Report CS97-557, Department of Computer Science, Havard University, San Diego, September 1997.

W. Enkelmann. Obstacle detection by evaluation of optical flow fields. In *Proc. ECCV*, pages 134–138, 1990.

J.-F. Evers-Senne, J.-M. Frahm, J. Woetzel, F. Woelk, and R. Koch. Distributed realtime interaction and visualisation system. In *Proc. VMV*, Erlangen, Germany, Nov. 2002.

S. Fejes and L. S. Davis. Detection of independent motion using directional motion estimation. Technical Report CAR-TR-866, CS-TR 3815, Center for Automation Research, University of Maryland, August 1997a.

S. Fejes and L. S. Davis. Exploring visual motion suing projections of motion fields. In *Proceedings of the DARPA Image Understanding Workshop*, New Orleans, LA, 1997b.

S. Fejes and L. S. Davis. What can projections of flow fields tell us about visual motion. In *Proc. ICCV*, pages 979–986, Bombay, India, 1998. URL `citeseer.nj.nec.com/fejes98what.html`.

C. Fermüller and J. Aloimonos. Direct perception of three-dimensional motion from patterns of visual motion. *Science*, 270(5244):1973–1976, 1995.

Fetzer. *Technische Kundenunterlage Mikromechanischer Drehratensensor DRS-MM 1.0 mit Dämpferplatte*. Bosch, 1998.

M. A. T. Figueiredo. Lecture notes on bayesian estimation and classification, 2004.

M. Fischler and R. Bolles. *RANdom SAmpling Consensus: a paradigm for model fitting with application to image analysis and automated cartography*, pages 381–395. Commun. Assoc. Comp. Mach. 24, 1981.

A. W. Fitzgibbon and A. Zimmerman. Multibody structure and motion: 3D reconsruction of independently moving objects. In *Proc ECCV*, 2000.

W. Förstner. A feature based correspondence algorithm for image matching. In *International Archives of Photogrammetry and Remote Sensing*, volume 26 - 3/3, pages 150–166, Rovaniemi, 1986.

W. Förstner. Uncertainty and projective geometry. In E. Bayro-Corrochano, editor, *Handbook of Geometric Computing: Applications in Pattern Recognition, Computer Vision, Neuralcomputing, and Robotics*. Springer, 2005.

U. Franke, C. Rabe, H. Badino, and S. Gehrig. 6D-vision: Fusion of stereo and motion for robust environment perception. In W. Kropatsch, R. Sablatnig, and A. Hanbury, editors, *Proc. DAGM*, pages 216–223, Berlin, Germany, September 2006. Springer-Verlag Berlin Heidelberg.

C. Gear. Multibody grouping from motion images. *IJCV*, 29(2):133–150, 1998.

S. Gehrig, S. Wagner, and U. Franke. System architecture for an intersection assistant fusing image, map and gps information. In *Proc. IV Symposium*, pages 144–149. IEEE, June 2003.

T. Gern. Abschlußbericht zum Auftrag Nr. 4900044226-F04 (CarPilot). internal, September 2000.

N. Gordon, D. Salmon, and A. Smith. Novel approach to nonlinear/non-gaussian bayesian state estimation. *IEE-Proceedings-F*, pages 107–113, 1993.

M. Han and T. Kanade. The factorization method with linear motions. Technical Report CMU-RI-TR-99-23, The Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213, 1999a.

M. Han and T. Kanade. Perspective factorization methods for euclidean reconstruction. Technical Report CMU-RI-TR-99-22, The Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213, August 1999b.

M. Han and T. Kanade. Reconstruction of a scene with multiple linearly moving objects. In *Proc. CVPR*, 2000.

M. Han and T. Kanade. Multiple motion scene reconstruction from uncalibrated views. In *Proc. ICCV*, 2001.

M. Han and T. Kanade. Multiple motion scene reconstruction with uncalibrated cameras. *TPAMI*, 25(7):884–894, July 2003.

C. Harris and M. Stephens. A combined corner and edge detector. In *4th Alvey Vision Conference, Manchester*, pages 147–151, 1988.

R. Hartley and A. Zissermann. *Multiple View Geometry in Computer Vision*. Cambridge university press, second edition, 2004.

E. Hayman and J.-O. Eklundh. Statistical background subtraction for a mobile observer. In *Proc. ICCV*, 2003.

D. Heeger. Signal Detection Theory Handout, 1998. URL `http://www-psych.stanford.edu/~lera/psych115s/notes/signal/`.

S. Heinrich. Real time fusion of motion and stereo using flow/depth constraint for fast obstacle detection. In L. V. Gool, editor, *Proc. DAGM*, pages 75–82, Zürich, September 2002. Springer-Verlag Berlin Heidelberg.

B. K. P. Horn. Tsai's camera calibration method revisited, 2000. URL `http://people.csail.mit.edu/bkph/articles/Tsai\_Revisited.pdf`.

C. Hue, J.-P. le Cadre, and P. Pérez. Tracking multiple objects with particle filtering. *Trans. AES*, 38(3):791–812, 2002.

M. Irani. Multi-frame correspondence estimation using subspace constraints. *IJCV*, 48(3): 173–194, 2002.

M. Isard and A. Blake. CONDENSATION – conditional density propagation for visual tracking. *IJCV*, 29(1):5–28, 1998a.

M. Isard and A. Blake. ICONDENSATION: Unifying low-level and high-level tracking in a stochastic framework. In *Proc. ECCV*, pages 893–908, 1998b.

M. Isard and J. McCormick. Bramble: A bayesian multiple-blob tracker. In *Proc. ICCV*, 2001.

O. S. J. Heikkilae. A four-step camera calibration procedure with implicit image correction. In *Pro. CVPR*, 1997.

M. I. Jordan. Why the logistic function? A tutorial discussion on probabilities and neural networks. Technical Report 9503, Computational Cognitive Science, MIT, MA, USA, August 1995.

S. J. Julier and J. K. Uhlmann. Reduced sigma point filters for the propagation of means and covariances through nonlinear transformations. In *Proceedings of the IEEE American Control Conference*, pages 887–892, Anchorage AK, USA, 8–10 May 2002a. IEEE.

S. J. Julier and J. K. Uhlmann. The scaled unscented transformation. In *Proceedings of the IEEE American Control Conference*, pages 4555–4559, Anchorage AK, USA, 8–10 May 2002b. IEEE.

S. J. Julier, J. K. Uhlmann, and H. F. Durrant-Whyte. New approach for filtering nonlinear systems. In *Proc. American Control Conference (ACC)*, pages 1628–1632, Seattle WA, USA, 1995.

R. E. Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME - Journal of Basic Engineering*, 82(Series D):35–45, March 1960.

K. Kanatani. Motion segmentation by subspace separation and model selection. In *Proc. ICCV*, volume 2, pages 586–591, Vancouver, Canada, July 2001.

K. Kanatani. *Statistical Optimization for Geometric Computation: Theory and Practice*. Dover, 2005.

Z. Khan, T. Balch, and F. Dellaert. An MCMC-based particle filter for tracking multiple interacting targets. In *Proc. ECCV*, 2004.

U. Köthe. Edge and junction detection with an improved structure tensor. In G. K. B. Michaelis, editor, *Pattern Recognition, Proc. of 25th DAGM Symposium*, volume 2781 of *Lecture Notes in Computer Science*, pages 25–32, Heidelberg, 2003. Springer.

E. Kreyszig. *Statistische Methoden und ihre Anwendung*. Vandenhoeck & Ruprecht, 1965.

W. Langdon. Receiver operating characteristics (ROC), 2003. URL `http://www.cs.ucl.ac.uk/staff/W.Langdon/roc/`.

B. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proc. IJCAI*, pages 674–679, 1981.

Y. Ma, J. Košecká, and S. Sastry. Optimization criteria and geomeric algorithms for motion and structure estimation. *IJCV*, 3(44):219–249, 2001.

Y. Ma, S. Soatto, J. Košecká, and S. Sastry. *An Invitation to 3-D Vision*. Springer, New York, 2004.

W. J. MacLean. *Recovery of Egomotion and Segmentation of Independent Object Motion Using the EM-Algorithm*. PhD thesis, Graduate Depratment of Electrical & Computer Engeneering, University of Toronto, 1996.

L. Marchetti, G. Grisetti, and L. Iocchi. A comparative analysis of particle filter based localization methods. In *Proc. RoboCup Symposium*, 2006.

J. C. McGlone, editor. *Manual of Photogrammetry*. American Society for Photogrammetry and Remote Sensing, Maryland, USA, 2004.

F. Meyer and P. Bouthemy. Estimation of time-to-collision maps from first order motion models and normal flows. In *Proc. Conference on Pattern Recognition*, volume 1, pages 78–82, Aug 1992.

L. S. Monteiro, T. Moore, and C. Hill. What is the accuracy of DGPS? In *European Navigation Conference GNSS*, Rotterdam, The Netherlands, May 2004.

T. Morita and T. Kanade. A sequential factorization method for recovering shape and motion from image streams. Technical Report CMU-CS-94-158, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213-3890, June 1994.

W. J. Morokoff and R. E. Caflisch. Quasi-Monte Carlo integration. *J. Comp. Phys.*, 122: 218–230, 1995. URL `citeseer.ist.psu.edu/morokoff95quasimonte.html`.

D. Myatt, P. Torr, S. Nasuto, R. Craddock, and J. Bishop. NAPSAC: High noise, high dimensional model parameterisation - its in the bag. In *Proc. BMVC*, pages 458–467, 2002.

R. Nelson. Qualitative detection of motion by a moving observer. *IJCV*, 7(1):33–46, 1991.

D. Nistér. An efficient solution to the five-point relative pose problem. In *Proc. CVPR*, volume 2, pages 195–202, 2003a.

D. Nistér. Preemptive ransac for live structure and motion estimation. In *Proc. ICCV*, pages 199–206, 2003b.

D. Nistér. An efficient solution to the five-point relative pose problem. *TPAMI*, 2004.

A. Ogale, C. Fermüller, and Y. Aloimonos. Motion segmentation using occlusions. *TPAMI*, 27(6):988–992, June 2005.

J. Philip. A non-iterative algorithm for determining all essential matrices corresponding to five point pairs. *Photogrammetric Record*, 15(88):589–599, October 1996.

J. Philip. Critical point configurations of the 5-, 6-, 7-, and 8-point algorithms for relative orientation. Technical report, Royal Institute of Technology, Stockholm, February 1998.

C. J. Poelman and T. Kanade. A paraperspective factorization method for shape and motion recovery. Technical Report CMU-CS-93-219, School of Computer Science, Carnegie Mellon University, Pittsburgh, Pennsylvania 15213-3890, Dezember 1993.

E. Rosten and T. Drummond. Fusing points and lines for high performance tracking. In *Proc. ICCV*, volume 2, pages 1508–1511, October 2005. doi: 10.1109/ICCV.2005.104. URL `http://mi.eng.cam.ac.uk/~er258/work/rosten\_2005\_tracking.pdf`.

E. Rosten and T. Drummond. Machine learning for high-speed corner detection. In *European Conference on Computer Vision*, volume 1, pages 430–443, May 2006. doi: 10.1007/11744023_34. URL `http://mi.eng.cam.ac.uk/~er258/work/rosten\_2006\_machine.pdf`.

P. J. Rousseeuw and A. M. Leroy. *Robust Regression and Outlier Detection*. John Wiley and Sons, 1987.

K. Schindler and D. Suter. Two-view multibody structure-and-motion with outliers. In *Proc. CVPR*, 2005.

K. Schindler, J. U, and H. Wang. Perspective n-view multibody structure-and-motion through model selection. In *Proc. ECCV*, 2006.

J. Schmidt and H. Niemann. Using quaternions for parametrizing 3D rotations in unconstrained nonlinear optimization. In T. Ertl, B. Girod, G. Greiner, H. Niemann, and H.-P. Seidel, editors, *Proc. VMV*, pages 399–406, Stuttgart, Germany, 2001.

R. Sekuler, S. N. J. Watamaniuk, and R. Blake. *Stevens' Handbook of Experimental Psychology,*, volume 1, chapter Perception of visual motion. J. Wiley, New York, Jan 2004.

J. Shi and C. Tomasi. Good features to track. In *Proc. CVPR*, pages 593–600, Seattle, June 1994. IEEE.

C. Silva and J. Satos-Victor. Direct egomotion estimation. In *Proc ICPR*, volume 1, pages 702–706, Vienna, Austria, 1996.

D. Sinclair and B. Boufama. Independent motion segmentation and collision prediction for road vehicles. In *Proc. ECCV*, 1994.

P. Smith, T. Drummond, and R. Cipolla. Motion segmentation by tracking edge information over multiple frames. In *Proc. ECCV*, pages 396–410, 2000.

S. M. Smith. ASSET-2: Real-time motion segmentation and object tracking. Technical Report TR95SMS2b, Oxford Centre for functional Magnetic Resonance Imageing of the Brain (FMRIB), Dep. of Clinical Neurology, Oxford University, Oxford, UK, 1995.

S. M. Smith and J. M. Brady. Susan - a new approach to low level image processing. Technical Report TR95SMS1c, Oxford University, 1995.

F. Stein. Efficient computation of optical flow using the census transform. In *Proc. DAGM*, pages 79–86, 2004.

C. Stewart. MINPRAN: A new robust estimator for computer vision. *TPAMI*, 17(10): 925–938, Oct 1995.

H. Stöcker, editor. *Taschenbuch mathematischer Formeln und moderner Verfahren*. Harri Deutsch, 1993.

P. Sturm. Structure and motion for dynamic scenes - the case of points moving in planes. In *Proc. ECCV*, volume 2, pages 876–882, copenhagen, Denmark, May 2002.

P. A. Tippler. *Physik*. Spektrum, 1994.

C. Tomasi and T. Kanade. Detection and tracking of point features. Technical Report CMU-CS-91-132, Carnegie Mellon University, Pittsburg, PA, 1991a.

C. Tomasi and T. Kanade. Shape and motion from image streams: A factorization method 2. point features in 3D motion. Technical Report CMU-CS-91-105, Computer Science Department, Carnegie Mellon University, Pittsburgh, PA, January 1991b.

C. Tomasi and T. Kanade. Shape and motion from image streams under orthography – a factorization method. *IJCV*, 9(2):137–154, 1992.

T. Tommasini, A. Fusiello, E. Trucco, and V. Roberto. Making good features track better. In *Proc. CVPR*, Santa Barbara, CA, USA, 1998. IEEE.

B. Tordoff and D. W. Murray. Guided sampling and consensus for motion estimation. In *Proc. ECCV*, 2002.

P. Torr. Bayesian model estimation and selection for epipolar geometry and generic manifold fitting. *IJCV*, 50(1):27–45, 2002.

P. Torr. *Motion Segmentation and Outlier Detection*. PhD thesis, University of Oxford, UK, 1995.

P. Torr and C. Davidson. IMPSAC: A synthesis of importance sampling and random sample consensus. *TPAMI*, 25(3):354–365, 2003.

P. Torr and D. Murray. Statistical detection of independent movement from a moving camera. *IVC*, 11(4):180–187, 1993.

P. Torr and D. Murray. Development and comparison of robust methods for estimating the fundamental matrix. *IJCV*, 1996.

P. Torr and A. Zisserman. MLESAC: A new robust estimator with application to estimating image geometry. *CVIU*, 1996.

P. H. S. Torr. Geometric motion segmentation and model selection. *Phil. Trans. of Royal Soc. A: Math, Phys. and Eng. Sciences*, 356(1740):1321–1340, May 1998.

P. H. S. Torr, R. Szeliski, and P. Anandan. An integrated bayesian approach to layer extraction from image sequences. *TPAMI*, 23(3):297–303, 2001.

R. Y. Tsai. A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf tv cameras and lenses, an efficient and accurate camera calibration technique. *IEEE Journal of Robotics and Automation*, RA-3(4):323–344, 1987.

M. B. van Leeuwen. *Motion Estimation and Interpretation for In-Car Systems*. PhD thesis, Universiteit van Amsterdam, 2002.

J. Vermaak, A. Doucet, and P. Pérez. Maintaining multi-modality through mixture tracking. In *Proc. ICCV*, 2003.

R. Vidal and S. Sastry. Optimal segmentation of dynamic scenes from two perspective views. In *Proc. CVPR*, volume 1, pages 281–286, 2003.

R. Vidal, S. Soatto, and S. Sastry. Two-view segmentation of dynamic scenes from the multibody fundamental matrix. Technical Report UCB/ERL M02/02, University of California, Berkely, February 2002.

R. Vidal, Y. Ma, S. Soatto, and S. Sastry. Two-view multivbody structure from motion. *IJCV*, 68(1):7–25, June 2006.

H. Wang and M. Brady. Real-time corner detection algorithm for motion estimation. *IVC*, 13(9):695–703, November 1995.

G. Welch and G. Bishop. An introduction to the kalman filter. Technical Report TR 95-041, University of North Carolina, Department of Computer Science, 2001. URL `http://www.cs.unc.edu/~welch`.

F. Woelk and R. Koch. Fast monocular bayesian detection of independently moving objects by a moving observer. In *Proc. DAGM*, pages 27–35, 2004.

F. Woelk, S. Gehrig, and R. Koch. A Monocular Image Based Intersection Assistant. In *Proc. IV Symposium*. IEEE, 2004.

F. Woelk, S. Gehrig, and R. Koch. A monocular collision warning system. In *Proc. CRV*, pages 220–227, 2005.

L. Wolf and A. Shashua. Two-body segmentation from two perspective views. In *Proc. CVPR*, 2001.

K. Y. Wong and M. E. Spetsakis. Motion segmentation by EM clustering of good features. In *Second IEEE Workshop on Image and Video Registration*, Washington DC, USA, 2004.

K. Y. Wong, L. Ye, and M. E. Spetsakis. EM clustering of incomplete data applied to motion segmentation. In *Proc. BMVC*, volume 1, pages 237–246, London, UK, 2004.

J. Yan and M. Pollefeys. A general framework for motion segmentation: Independent, articulated, rigid, non-rigid, degenerate and non-degenerate. In *Proc. ECCV*, 2006.

H. Yu, Q. Chen, G. Xu, and M. Yachida. 3D shape and motion by svd under higher-order approximation of perspective projection. In *Proc. ICPR*, pages 456–460, 1996.

J. Zelek. Bayesian real-time optical flow. In *Proc. VI*, 2002.

L. Zelnik-Manor, M. Amchline, and M. Irani. Multi-body factorization with uncertainty: Revisiting motion consistency. *IJCV, special issue on Vision and Modeling of Dynamic Scenes*, 68(1):27–41, June 2006.

Z. Zhang. Parameter estimation techniques: A tutorial with application to conic fitting, 1996. URL `http://www-sop.inria.fr/robotvis/personnel/zzhang/Publis/Tutorial-Estim/Main.html`.

Z. Zhang. Parameter estimation techniques: A tutorial with application to conic fitting. *IVC*, 15(1):59–76, 1997.

A. Zomotor. *Fahrwerktechnik: Fahrverhalten*. Vogel Verlag, 1991.