Institutionen för systemteknik
Department of Electrical Engineering

Examensarbete

On Vergence Calibration of a Stereo Camera

Examensarbete utfört i Reglerteknik
vid Tekniska högskolan vid Linköpings universitet
av

Sebastian Jansson

LiTH-ISY-EX--12/4627--SE

Linköping 2012

Linköpings universitet
TEKNISKA HÖGSKOLAN

Department of Electrical Engineering
Linköpings universitet
SE-581 83 Linköping, Sweden

Linköpings tekniska högskola
Linköpings universitet
581 83 Linköping

On Vergence Calibration of a Stereo Camera

Examensarbete utfört i Reglerteknik
vid Tekniska högskolan vid Linköpings universitet
av

Sebastian Jansson

LiTH-ISY-EX--12/4627--SE

Handledare: **Jonas Linder**
ISY, Linköpings Universitet
Jon Bjärkefur
Autoliv Electronics
Emil Nilsson
Autoliv Electronics

Examinator: **Thomas Schön**
ISY, Linköpings Universitet

Linköping, 14 september 2012

Avdelning, Institution
Division, Department

Division of Automatic Control
Department of Electrical Engineering
SE-581 83 Linköping

Datum
Date

2012-09-14

Språk
Language

□ Svenska/Swedish
☒ Engelska/English
□ _____

Rapporttyp
Report category

□ Licentiatavhandling
☒ Examensarbete
□ C-uppsats
□ D-uppsats
□ Övrig rapport
□ _____

ISBN
—

ISRN
LiTH-ISY-EX--12/4627--SE

Serietitel och serienummer
Title of series, numbering

ISSN
—

URL för elektronisk version
http://www.ep.liu.se/

Titel
Title
On Vergence Calibration of a Stereo Camera

Författare
Author
Sebastian Jansson

Sammanfattning
Abstract

Modern cars can be bought with camera systems that watch the road ahead. They can be used for many purposes, one use is to alert the driver when other cars are in the path of collision. If the warning system is to be reliable, the input data must be correct. One input can be the depth image from a stereo camera system; one reason for the depth image to be wrong is if the vergence angle between the cameras are erroneously calibrated. Even if the calibration is accurate from production there's a risk that the vergence changes due to temperature variations when the car is started.

This thesis proposes one solution for short-time live calibration of a stereo camera system; where the speedometer data available on the CAN-bus is used as reference. The motion of the car is estimated using visual odometry, which will be affected by any errors in the calibration. The vergence angle is then altered virtually until the estimated speed is equal to the reference speed.

The method is analyzed for noise and tested on real data. It is shown that detection of calibration errors down to 0.01 degrees is possible under certain circumstances using the proposed method.

Nyckelord
Keywords
Ego-motion, Self-calibration, Stereo Calibration, Stereo Camera, Vergence Calibration, Visual Odometry

Abstract

Modern cars can be bought with camera systems that watch the road ahead. They can be used for many purposes, one use is to alert the driver when other cars are in the path of collision. If the warning system is to be reliable, the input data must be correct. One input can be the depth image from a stereo camera system; one reason for the depth image to be wrong is if the vergence angle between the cameras are erroneously calibrated. Even if the calibration is accurate from production there's a risk that the vergence changes due to temperature variations when the car is started.

This thesis proposes one solution for short-time live calibration of a stereo camera system; where the speedometer data available on the CAN-bus is used as reference. The motion of the car is estimated using visual odometry, which will be affected by any errors in the calibration. The vergence angle is then altered virtually until the estimated speed is equal to the reference speed.

The method is analyzed for noise and tested on real data. It is shown that detection of calibration errors down to 0.01 degrees is possible under certain circumstances using the proposed method.

Acknowledgments

Besides the people directly involved with my thesis, mentioned on the title page, I also would like to offer my special thanks to Fredrik Tjärnström at Autoliv for providing me with the opportunity and general idea for the thesis.

In the same spirit I would like to show my appreciation for the help and general expertise provided by other employees of Autoliv.

I also want to acknowledge the help received from by Zoran Sjanic at ISY, Linköping University.

Finally I wish to thank Malin Persson and László Gönczi for help with proof-reading and making this report more accessible.

Linköping, September 2012
Sebastian Jansson

Contents

1 Introduction

This thesis aim to develop a method to estimate the vergence angle for a stereo camera rig mounted within a car. It was done in cooperation with Autoliv Electronics Sweden and with Linköping University.

1.1 Background

Autoliv is a company dedicated to improving automotive safety with the goal of saving lives. In Linköping, the focus is on creating camera systems that can, for instance, alert the driver when a person runs out on the road in front of the car. This can be done thanks to several systems that detect and track objects in the scene ahead.

Figure 1.1: *Autoliv's stereo camera system. The system is installed in front of the rear view mirror.*

Autoliv develops several products, one of which is a stereo camera system that can calculate depth, depicted in Figure 1.1. For a stereo vision system to work properly, the system needs to be calibrated in some manner so that the pose[1] of the cameras are known, otherwise the depth estimate will be inaccurate.

1.1.1 Visual Odometry

Visual odometry estimation, also called visual ego-motion estimation, is the process of estimating how the vehicle is moving based on image data from a camera system.

With high enough frame rates, it can be assumed that the motion will be small in between frames. This is the basis for some ideas where the relation between movement, relative to the camera, of a static point in a scene and the movement of its camera projection is linearized [Longuet-Higgins and Prazdny, 1980, Adiv, 1985]. This concept is investigated in Irani et al. [1994] and further developed using locally planar structures in Irani et al. [1997]. In Golban and Nedevschi [2011] it's shown that the linearization does affect the result, but only to a limited extent.

1.1.2 Vergence Calibration

In a stereo setup as the one used at Autoliv, the cameras are mounted firmly to keep their relative pose fixed. The setup is then calibrated using calibration targets and camera calibration techniques. The calibration technique described in Zhang [1999] is one of many which can do this.

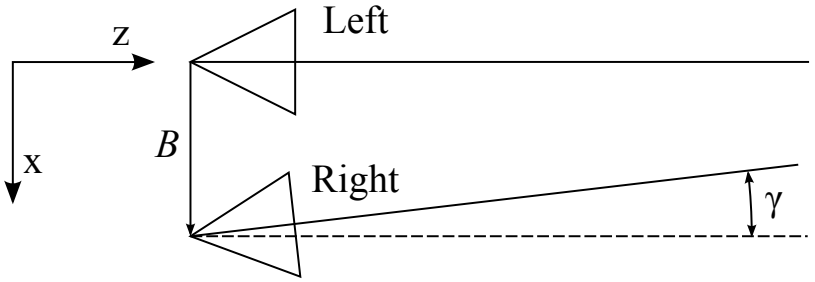


Figure 1.2: *The vergence error γ describes how the right camera is rotated relative to the left camera.*

The vergence angle, i.e. the relative yaw angle between the left and right camera, γ in Figure 1.2, is one of the parameters that needs to be calibrated. If the vergence angle is wrong, the depth estimation will be wrong as well. Due to thermal

[1] Pose is here defined as the combination of position and rotation.

variation, amongst other factors, the vergence angle can change over time. It's difficult to separate vergence error effects from actual differences in depth.

There are several techniques for self-calibration, i.e. calibration based on run-time input. Many such calibration techniques are described in Horaud et al. [2000]. Many of them are based on using a small number of frames and point correspondences in some manner.

Horaud et al. [2000] continues to suggest a method for self-calibration using homographies[2] describing the relationship between different camera views. In Brooks et al. [1996] a method for calibration is described using an analytical form of the fundamental matrix[3] relating the left and right camera.

This thesis presents a different approach to the problem. The ego-motion estimation based on stereo data is compared with the information given by the car's speedometer. If the visual odometry is calculated for other reasons, this method might reduce the extra processing required. It also has potential to give good results in short time.

1.2 Goals

The goal of this master thesis is to design and evaluate a method for estimating the vergence angle between the two cameras in a stereo camera system mounted inside a car.

This has been divided into the following sub goals:

- Finding and implementing a method for visual odometry using stereo vision data.
- Designing a method for calculating correct vergence using a comparison of visual odometry and mechanical odometry.
- Implement and evaluate the method within the development framework used at Autoliv.

1.3 Exclusions

The stereo images from Autoliv are rectified[4] and the disparity is found using code that is available within Autoliv's framework. This means that the thesis work does not involve solving the problem of finding stereo disparity or the problem of rectifying distorted images.

[2] A homography describes a mapping from the coordinates in one plane to coordinates in another plane using perspective transformation.

[3] The *fundamental matrix* for two 3D views describes how a point in one view relates to a line in the other.

[4] Barrel distortion, lens distortion etc. are removed to create a pin-hole camera equivalent image.

If the car owner has changed to tires with a different radius, the speedometer data might be wrong. This has not been accounted for in this thesis, i.e. the speed from the speedometer is assumed to be true.

1.4 Method

At first, a theoretical study was performed on how to do visual odometry. This study was then the basis for the design of a method for finding visual odometry using stereo data. This method was then implemented and tested on sample recordings provided by Autoliv.

The effects of an error in vergence angle on the estimation were calculated. This result was used to create a method of vergence angle extraction using mechanical data as a reference data source.

Finally, the method was implemented within the framework at Autoliv. The complete solution was then tested on recordings of the camera views for a vehicle in motion where artificial calibration error was added.

2

Platform & notation

This thesis was carried out at Autoliv, who have kindly provided development equipment and software as well as test data and systems. This chapter gives a brief description of the system used for development and testing. It also contains a short summary of the notation used in this thesis.

2.1 Stereo Vision System

Autoliv develops three different camera vision systems for cars.

- Night vision system with a FIR (Far Infrared Range, detects heat) camera
- Stereo vision system using normal cameras
- Mono vision system using normal cameras

In this thesis, the focus is on the stereo vision platform. The stereo vision system consists of two physically joined cameras 16 cm apart. The camera setup is placed in front of the rear view mirror.

The internal camera parameters, such as field of view, distance, angle and lens distortion coefficients are estimated in a manually assisted calibration using calibration targets. The images are rectified, that is, they are warped and rotated according to the calibration so they are virtually looking straight forward.

Projections of objects at different distances have different translations when switching between left and right camera. This parallax effect is used to estimate depth.

This is assisted by the calculation of a disparity map which encodes how each point in the picture is translated between the left and right camera. This dis-

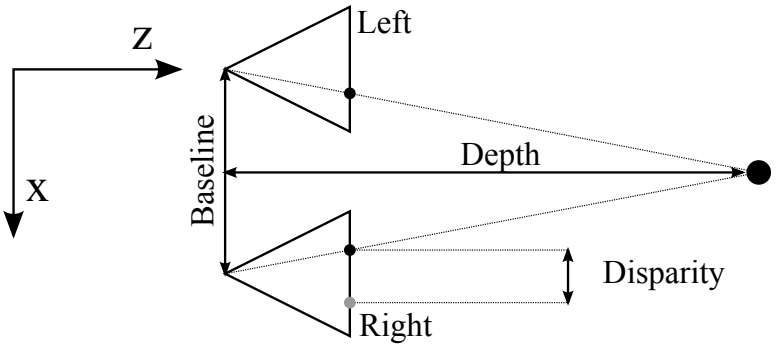


Figure 2.1: *The depth can be calculated using Depth* $= \frac{Baseline}{Disparity}$.

parity map is then used together with information about distance between the cameras to estimate depth. If the calibration is right, the relation is as simple as Depth $= \frac{\text{Baseline}}{\text{Disparity}}$ as shown in Figure 2.1. Note that this assumes that the calibration involves removing the effect of focal length, see Section 2.3.

Nowadays, cars have a data network for all kinds of information relevant to in-car systems. The Controller Area Network (CAN), can be accessed by Autoliv's hardware through the CAN bus. On the CAN bus speedometer information as well as other navigational aids are available.

Depth information and CAN data is provided in Autoliv's Matlab based development platform, which has been used for this thesis. The system gives access to movie clips recorded using the stereo vision setup synced with recorded CAN data.

For this thesis the speedometer information has been the basis of the vergence calibration method. This means that it is required that the error of speedometer data is small. This might not necessarily be the case, for instance if the car owner has changed to tires with a different radius. However, the data in Autoliv's test cars has been shown to be accurate enough for the purposes of this thesis. This has been tested by manual comparison with satellite images. The car used for recording the clips used for testing the ego motion estimation was also fitted with an extra set of sensors which recorded the motion with higher precision.

2.2 Notation

The notation used in this thesis relies on the use of the sub-indices shown in Table 2.2 to represent meaning of a variable. Vectors and matrices are bold, see (2.1) and (2.2) for examples. Global vectors and matrices, for instance the augmented Jacobian, are upright, while point-specific counter-parts are in *italic*. Scalars are in italic, sometimes uppercase. $\hat{\mathbf{e}}_s$ represents the base component for speed s so that $s = \hat{\mathbf{e}}_s \mathbf{m}$. The operator † represents the pseudo inverse, $A^\dagger = (A^T A)^{-1} A^T$.

$$\mathbf{m} = \begin{pmatrix} w_x \\ w_y \\ w_z \\ t_x \\ t_y \\ t_z \end{pmatrix} \tag{2.1}$$

$$\mathbf{I} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{2.2}$$

Table 2.1: *Variables used*

Notation	Explanation
$\mathbf{m}$	Camera (vehicle) motion
$\boldsymbol{d}_i = \begin{pmatrix} \Delta x_i \\ \Delta y_i \end{pmatrix}$	Displacement of the projection for a specific point
$\mathbf{d} = \begin{pmatrix} \boldsymbol{d}_1 \\ \boldsymbol{d}_2 \\ \vdots \end{pmatrix}$	All displacements for all points
$\boldsymbol{J}_i = \frac{\partial \boldsymbol{d}_i}{\partial \mathbf{m}}$	Jacobian for a specific point
$\mathbf{J} = \begin{pmatrix} \boldsymbol{J}_1 \\ \boldsymbol{J}_2 \\ \vdots \end{pmatrix}$	All Jacobians for all points
$\mathbf{T}$	Translation in 3D [m]
$\mathbf{R}$	Rotation matrix
w_x, w_y, w_z	Pitch, Yaw and Roll change of camera [rad/frame]
t_x, t_y, t_z	Translation in 3D [m/frame]
$s = t_z$	Speed [m/frame]
x, y	2D projection coordinates
X, Y	3D world coordinate in left camera system [m]
Z	Distance (depth) from the left camera [m]

Table 2.2: Sub-indices

Notation	Explanation
i	A specific point (with index i)
t	True value
e	Estimated value
c	Correction ($s_t = s_e + s_c$)
p	Predicted/previous value
0	Original point
m	After motion transform has been applied
s	After stereo camera switch
x,y ,z	Corresponding to x, y, z axis

2.3 Coordinates

The calculations use a coordinate system that is relative to the camera. The variable z in Figure 3.2 represents the distance to the camera. The variables x,y, represents the position in, or in a plane parallel to, the projection plane.

In papers about ego motion, pixel coordinates are commonly used. This means that the focal length has to be taken account for in all calculations. In the calculations in this thesis, it's assumed that the image coordinates are centered and normalized so that the focal length is 1, independent of units[1].

$$x_{\text{normalized}} = \frac{x_{\text{pixel}} - x_{\text{center}}}{f_x}, \qquad y_{\text{normalized}} = \frac{y_{\text{pixel}} - y_{\text{center}}}{f_y}, \quad (2.3)$$

where $(x_{\text{center}}, y_{\text{center}})^T$ is the pixel coordinates of the principal point (image midpoint). Note that this also means that the units of the projection and the units in world coordinates coincide if the projection plane is considered to be at 1 unit distance away. Therefore, the coordinate systems in the projection and in world coordinates can be considered the same.

[1] The focal length is 1 meter if the coordinate is given in meters.

3

Ego-motion

Visual odometry estimation, also called visual ego-motion estimation, is the process of estimating how the camera, or the vehicle it's mounted in, is moving based on image data. The data available to do so in this case is two consecutive rectified picture pairs from the stereo camera setup.

Since the method for estimating vergence investigated in this thesis is based on available ego-motion data, a method to acquire such data is required. In this chapter, a method for estimation of ego-motion based on previous work is described.

The stereo camera picture pair can be used to extract depth in the image. If the baseline described in Chapter 2 is known, the depth will be correct in scale. If internal camera parameters are known as well, features in the image can be positioned in 3D relative to the camera.

The idea used in this thesis is to combine information about the displacement of pixels in the 2D picture plane with information about their position in 3D. To do this, a mathematical relation between the two is derived. This relation is then simplified to a linear model to make it usable for an in-car system where computing power is limited.

Two methods of using this relation is then investigated. The first method tracks the displacement for a set of picture coordinates within the image using some tracking algorithm. The inverse of the relation is then used to estimate the vehicle motion that caused this displacement.

The second method works with the intensity images directly without going via tracked points. This is called a *direct method*, since it doesn't use any intermediate data.

3.1 Camera motion and projection displacement

In the time between each frame in the picture stream from the camera setup, the car has made a specific motion, see Figure 3.1. The same event can be seen reversed. In the camera coordinate system, the static scenery has moved with the opposite motion. Here that motion is expressed as that the static scenery point $\boldsymbol{P}_i$ has moved to $\boldsymbol{P}_{i,m}$ while the camera was kept fixed, illustrated in Figure 3.2.

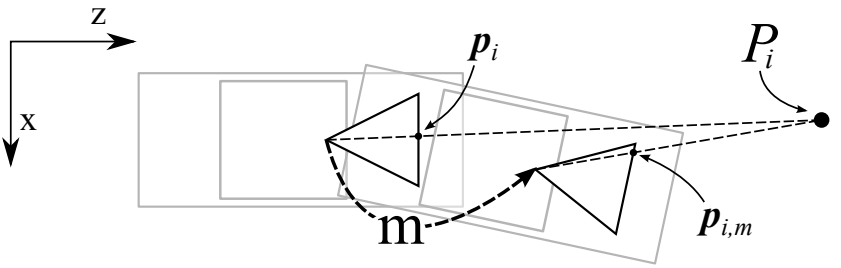


Figure 3.1: *The camera (shown from above) is moved by the motion* $\mathbf{m}$ *(*$\mathbf{R}_m, \mathbf{T}_m$*) between frames while static scenery* $\boldsymbol{P}_i$ *stays put. The projection* $\boldsymbol{p}_i$ *is displaced to* $\boldsymbol{p}_{i,m}$ *as a result.*

This makes it possible to calculate the relation between projection displacement $\boldsymbol{d}_i$ and motion $\mathbf{m}$ for the single point $\boldsymbol{P}_i$. The problem of estimating ego-motion is then formulated as the problem of going from a data set $\mathbf{d}$ describing the displacements of the 2D projections of several unknown points $\boldsymbol{P}_i$, $i = 1, ..., n$, to a motion vector $\mathbf{m}$ describing the motion of the camera in 3D space.

3.1.1 From ego-motion to 2D-motion

The goal is to find out what 2D motion of feature points, i.e. projection of static 3D points, says about the ego-motion of the vehicle. To calculate this, the opposite relation for a single 3D point is calculated.

Consider a point in 3D space, $\boldsymbol{P}_i = (X_i, Y_i, Z_i)^T$. This point is projected on an image plane as seen in Figure 3.2 which gives the picture coordinate

$$\boldsymbol{p}_i = (\frac{X_i}{Z_i}, \frac{Y_i}{Z_i})^T \tag{3.1}$$

(using a pinhole[1] camera model).

[1] A pinhole camera works by having a small hole (modeled as a singular point) where all light goes trough, capturing a projection of the scene with perfect focus

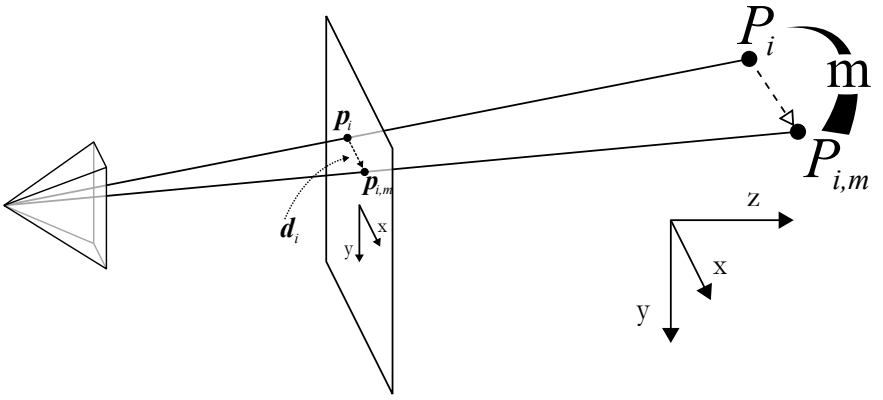


Figure 3.2: *The scenery point $\boldsymbol{P}_i$ is transformed by the motion* $\mathbf{m}$ *to $\boldsymbol{P}_{i,m}$ which causes the projection $\boldsymbol{p}_i$ to be displaced by $\boldsymbol{d}_i$ to $\boldsymbol{p}_{i,m}$.*

The camera is moved according to the translation $\mathbf{T}_m = (t_x, t_y, t_z)^T$ and rotated using the rotation matrix $\mathbf{R}_m$. To mirror the movement of the camera, the coordinates of the points are moved $-\mathbf{T}_m$ and rotated in the opposite direction with

$$\mathbf{R}_m^{-1} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(-w_x) & -\sin(-w_x) \\ 0 & \sin(-w_x) & \cos(-w_x) \end{pmatrix} \cdot \begin{pmatrix} \cos(-w_y) & 0 & \sin(-w_y) \\ 0 & 1 & 0 \\ -\sin(-w_y) & 0 & \cos(-w_y) \end{pmatrix} \cdot \begin{pmatrix} \cos(-w_z) & -\sin(-w_z) & 0 \\ \sin(-w_z) & \cos(-w_z) & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{3.2}$$

The new, transformed, coordinate for $\boldsymbol{P}_i$ is

$$\boldsymbol{P}_{i,m} = \mathbf{R}_m^{-1} \cdot (\boldsymbol{P}_i - \mathbf{T}_m) \tag{3.3}$$

before it's projected into the camera projection coordinate

$$\boldsymbol{p}_{i,m} = \frac{1}{Z_{i,m}} \begin{pmatrix} X_{i,m} \\ Y_{i,m} \end{pmatrix} \tag{3.4}$$

The displacement for a point $\boldsymbol{P}_i$ is defined as $\boldsymbol{d}_i = (\Delta x_i, \Delta y_i)^T = \boldsymbol{p}_{i,m} - \boldsymbol{p}_i$. Since the input data are the pixel coordinates and (estimated) depth, the relation

$$\boldsymbol{p}_i = \begin{pmatrix} x_i \\ y_i \end{pmatrix} = \begin{pmatrix} \frac{X_i}{Z_i} \\ \frac{Y_i}{Z_i} \end{pmatrix} \tag{3.5}$$

is used to motivate the substitutions $[X_i = x_i\, Z_i, Y_i = y_i\, Z_i]$. This gives a complete relation between camera movement and pixel displacement, given pixel coordinates and depth,

$$\boldsymbol{d}_i = f(\mathbf{m}) = f(w_x, w_y, w_z, t_x, t_y, t_z) = \begin{pmatrix} \Delta x_i \\ \Delta y_i \end{pmatrix} =$$
$$= \begin{pmatrix} \frac{\mathrm{C}w_y\,\mathrm{S}w_z\,(y_i\,Z_i - t_y) + \mathrm{C}w_y\,\mathrm{C}w_z\,(x_i\,Z_i - t_x) - \mathrm{S}w_y\,(Z_i - t_z)}{(\mathrm{C}w_x\,\mathrm{S}w_y\,\mathrm{S}w_z - \mathrm{S}w_x\,\mathrm{C}w_z)\,(y_i\,Z_i - t_y) + (\mathrm{S}w_x\,\mathrm{S}w_z + \mathrm{C}w_x\,\mathrm{S}w_y\,\mathrm{C}w_z)\,(x_i\,Z_i - t_x) + \mathrm{C}w_x\,\mathrm{C}w_y\,(Z_i - t_z)} - x_i \\ \frac{(\mathrm{S}w_x\,\mathrm{S}w_y\,\mathrm{S}w_z + \mathrm{C}w_x\,\mathrm{C}w_z)\,(y_i\,Z_i - t_y) + (\mathrm{S}w_x\,\mathrm{S}w_y\,\mathrm{C}w_z - \mathrm{C}w_x\,\mathrm{S}w_z)\,(x_i\,Z_i - t_x) + \mathrm{S}w_x\,\mathrm{C}w_y\,(Z_i - t_z)}{(\mathrm{C}w_x\,\mathrm{S}w_y\,\mathrm{S}w_z - \mathrm{S}w_x\,\mathrm{C}w_z)\,(y_i\,Z_i - t_y) + (\mathrm{S}w_x\,\mathrm{S}w_z + \mathrm{C}w_x\,\mathrm{S}w_y\,\mathrm{C}w_z)\,(x_i\,Z_i - t_x) + \mathrm{C}w_x\,\mathrm{C}w_y\,(Z_i - t_z)} - y_i \end{pmatrix} \tag{3.6}$$

where $\mathrm{S}\cdot = \sin\cdot$ and $\mathrm{C}\cdot = \cos\cdot$.

3.1.2 Jacobian

The Jacobian of (3.6) for a point P_i is

$$\boldsymbol{J}_i = \left.\frac{\partial(\Delta x_i, \Delta y_i)}{\partial(w_x, w_y, w_z, t_x, t_y, t_z)}\right|_{\mathbf{m}=0} = \begin{pmatrix} x_i\,y_i & -x_i^2 - 1 & y_i & -\frac{1}{Z_{i,m}} & 0 & \frac{x_i}{Z_{i,m}} \\ y_i^2 + 1 & -x_i\,y_i & -x_i & 0 & -\frac{1}{Z_{i,m}} & \frac{y_i}{Z_{i,m}} \end{pmatrix} \tag{3.7}$$

This describes the relation between camera motion and pixel displacement for small camera motion, like in Adiv [1985]. That is, for an infinitesimal ego-motion vector $\mathbf{m} = (w_x, w_y, w_z, t_x, t_y, t_z)^T$, the displacement of the projection is $\boldsymbol{d}_i = \boldsymbol{J}_i \cdot \mathbf{m}$. This Jacobian is, explicitly or implicitly, the basis for the methods described in Adiv [1985], Hanna [1991], Irani et al. [1994], Stein et al. [2000], Horn et al. [2007] and many more.

The assumption that motion is infinitesimal also includes the assumption that $Z_i \approx Z_{i,m}$. However, this assumption is often false; the difference $Z_i - Z_{i,m}$ for an object at 20m distance would be 2m when driving in 72km/h (at 20 fps). The quotient $\frac{\text{Distance travelled}}{\text{Distance to object}} = \frac{1}{10}$ isn't small enough to be considered insignificant; the stereo data used in this thesis has been most reliable at distances less than ca. 20m.

To take this error into consideration, $Z_{i,m}$ is treated separately from the depth in the first frame, Z_i. This gives the new Jacobian,

$$\left.\frac{\partial \boldsymbol{d}_i}{\partial \mathbf{m}}\right|_{\substack{t_z = Z_{i,m} - Z_i \\ \mathbf{m} = 0}} = \begin{pmatrix} \frac{x_i\,y_i\,Z_i}{Z_{i,m}} & -\frac{x_i^2\,Z_i + Z_{i,m}}{Z_{i,m}} & \frac{y_i\,Z_i}{Z_{i,m}} & -\frac{1}{Z_{i,m}} & 0 & \frac{x_i}{Z_{i,m}} \\ \frac{y_i^2\,Z_i + Z_{i,m}}{Z_{i,m}} & -\frac{x_i\,y_i\,Z_i}{Z_{i,m}} & -\frac{x_i\,Z_i}{Z_{i,m}} & 0 & -\frac{1}{Z_{i,m}} & \frac{y_i}{Z_{i,m}} \end{pmatrix} \tag{3.8}$$

It is worth noting that for the estimation of translation, the only change is that depth from the second frame is used. Often this is done anyway, to remove the need to store the depth between frames. For this thesis, the second frame depth has been used with the Jacobian from (3.7)

3.2 Displacement field method

A common way to estimate ego motion is to find a number of easily tracked image points and use the displacement of those image points between two frames to get the motion of the camera. The augmented Jacobian matrix for all n points,

$$\mathbf{J} = \begin{pmatrix} \boldsymbol{J}_1 \\ \boldsymbol{J}_2 \\ \vdots \\ \boldsymbol{J}_n \end{pmatrix} \tag{3.9}$$

is here used as a model for how the projection of the same points are displaced based on camera motion. The respective measured displacement for all points,

$$\mathbf{d} = \begin{pmatrix} \boldsymbol{d}_1 \\ \boldsymbol{d}_2 \\ \vdots \\ \boldsymbol{d}_n \end{pmatrix} \tag{3.10}$$

is used as reference data. Since some data points might be wrong, a weighted least square is used; the weights w_i represents the probability that the tracked point is on a static object. The weighting is here based on a comparison of the motion of a point to the expected motion based on previous motion. This is motivated by the fact that the car can't change motion very much between two frames. Given the previous motion $\mathbf{m}_p$, the Jacobian $\boldsymbol{J}_i$ and the measured $\boldsymbol{d}_i$, the weight of the specific displacement is calculated as

$$w_i = \exp\left(-\frac{\left\|\boldsymbol{d}_i - \boldsymbol{J}_i \mathbf{m}_p\right\|^2}{2\sigma^2}\right) \tag{3.11}$$

where σ is the expected standard deviation in displacement error for inliers.

Finally the weighted least square solution is sought, where the squared difference between modeled displacement and reference displacement is minimized over all possible vehicle motions,

$$\min_{\mathbf{m}} \sum_i w_i \left\|\boldsymbol{J}_i \mathbf{m} - \boldsymbol{d}_i\right\|^2 \tag{3.12}$$

There are several ways to select and track points in the image. Usually points that are easy to track are chosen based on contrast and shape. However, as Stein et al. [2000] suggest, such points tend to more frequently occur on other moving objects like cars. Therefore, in this thesis, the features are chosen in a constant grid. When they are tracked the result is a kind of low-resolution optical flow like in Figure 3.3, further referred to as a *displacement field*.

The tracking of displacement of the chosen features has been done with the implementation of the Lucas and Kanade [1981] algorithm in the OpenCV graphics library [Bradski, 2000]. The result using this tracker is quite good (see Sec-

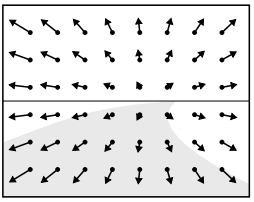


Figure 3.3: *When the features, chosen in a grid, are tracked, the result is a low resolution optical flow or displacement field.*

tion 3.5.1). Solving the tracking problem is not within the objectives of this thesis so this is not investigated further. In the end everything is put together with a data flow like in Figure 3.4 to create Algorithm 3.1.

Algorithm 3.1 Displacement field method

1. Calculate the model matrix (3.7) for how the displacement field relates to the rigid motion of the camera based on depth and pixel position. See Section 3.1.2 for more details on this model.
2. Predict the displacement field based on previous motion and the model matrix. $\mathbf{d}_p = \mathbf{J}\mathbf{m}_p$
3. Use the prediction as a starting point and find the displacement field using a tracking algorithm.
4. Create a weighting vector using (3.11) based on how well the actual displacement matches the prediction.
5. Use the Weighted Least Square (WLS) method to calculate the rigid camera motion based on the displacement field, the model and weights.

3.3 Direct method based on Lucas-Kanade tracker

A direct method [Hanna, 1991] in this context is a method that solves for the ego-motion parameters directly rather than going via a displacement field. In this case it is done using gradient descent in the motion space. The goal is to find the motion $\mathbf{m}_e$ which minimizes the intensity difference squared between the first image and the second image warped according to the inverse of motion $\mathbf{m}_e$.

A similar problem is to find the displacement $\boldsymbol{d} = (\Delta x, \Delta y)^T$ which minimizes the squared intensity difference in the region R between the two intensity pictures $P_1((x, y)^T)$, $P_2((x - \Delta x, y - \Delta y)^T)$,

$$\epsilon = \iint_{\boldsymbol{r} \in R} \left(P_2(\boldsymbol{r} + \boldsymbol{d}) - P_1(\boldsymbol{r})\right)^2 d\mathbf{r} \tag{3.13}$$

where R is a region covering an object or the close surroundings of a point.

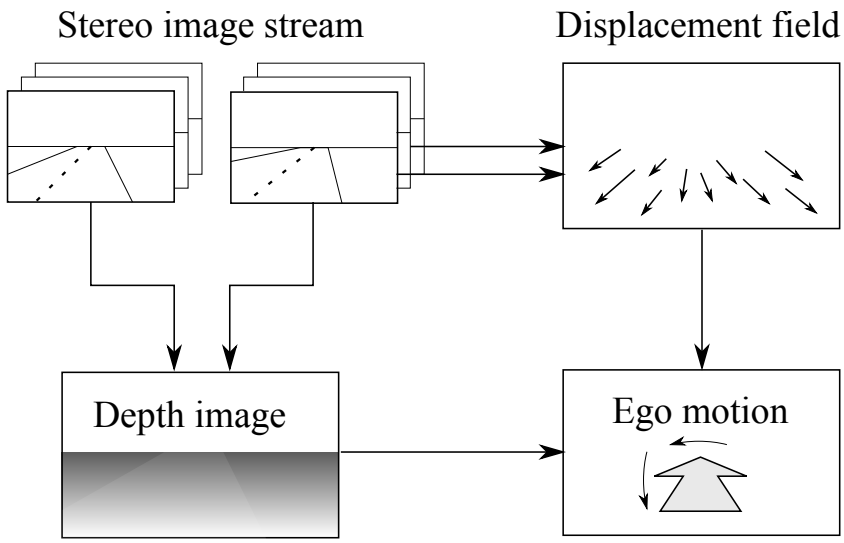


Figure 3.4: *The input is a series of image pairs. The images are used to create a depth image using the stereo core. They are also used to create a displacement field. The result of both those operations are combined to estimate ego-motion.*

The Lucas and Kanade [1981] algorithm is a commonly used method to solve this minimization problem. Johansson [2007] derives this algorithm with the estimation

$$\boldsymbol{d} = \sum_{k=0}^{\infty} \boldsymbol{d}_k \tag{3.14}$$

by iteratively solving

$$\mathbf{Z}\boldsymbol{d}_k = \mathbf{e}_k \tag{3.15}$$

where

$$\mathbf{Z} = \iint_{\mathbf{r}\in R_i} \nabla P_2(\mathbf{r}) \nabla P_2^T(\mathbf{r}) d\mathbf{r} \tag{3.16}$$

$$\mathbf{e}_k = \iint_{\mathbf{r}\in R_i} P_{1,k}(\mathbf{r}) - P_2(\mathbf{r}) \nabla P_2^T(\mathbf{r}) d\mathbf{r} \tag{3.17}$$

$$P_{1,k}(\mathbf{r}) = P_{1,k-1}(\mathbf{r} - \boldsymbol{d}_k) \tag{3.18}$$

However, the displacement $\boldsymbol{d}$, could just as well be a function of the motion $\mathbf{m}$, with the region R covering the full image except for any moving objects. The only

change that has to be made to this solution is that the picture gradient

$$\nabla P_2 = \frac{\partial P_2}{\partial \boldsymbol{d}} = \frac{\partial P_2}{\partial (x, y)} \tag{3.19}$$

is extended with the picture gradient with respect to $\mathbf{m}$,

$$\frac{\partial P_2}{\partial \mathbf{m}} = \frac{\partial \boldsymbol{d}}{\partial \mathbf{m}} \frac{\partial P_2}{\partial \boldsymbol{d}} = \boldsymbol{J} \nabla P_2 \tag{3.20}$$

The new error function

$$\epsilon_m = \iint\limits_{\mathbf{r} \in R} \left(P_2(\mathbf{r} + \boldsymbol{d}(\mathbf{r}, \mathbf{m})) - P_1(\mathbf{r}) \right)^2 d\mathbf{r} \tag{3.21}$$

can now be minimized using

$$\mathbf{Z}_m \mathbf{m}_k = \mathbf{e}_{m,k} \tag{3.22}$$

where

$$\mathbf{Z}_m = \iint\limits_{\mathbf{r} \in R} \xi \left(\boldsymbol{J}(\mathbf{r}) \cdot \nabla P_2(\mathbf{r}) \right) d\mathbf{r}, \quad \xi(\mathbf{v}) = \mathbf{v} \cdot \mathbf{v}^T \tag{3.23}$$

$$\mathbf{e}_{m,k} = \iint\limits_{\mathbf{r} \in R} P_{1,k}(\mathbf{r}) - P_2(\mathbf{r}) \cdot \boldsymbol{J}(\mathbf{r}) \cdot \nabla P_2^T(\mathbf{r}) d\mathbf{r} \tag{3.24}$$

This result is used in combination with (3.7), augmented for each pixel like (3.9), to produce Algorithm 3.2.

Algorithm 3.2 Direct method

1. Calculate a model matrix for how the displacement of each pixel relates to the rigid motion of the camera based on depth and pixel position, (3.7).
2. Calculate weights using (3.11) indicating inlier likelihood for each pixel in the image.
3. Predict the motion for each pixel using previous camera motion and current depth per pixel.
4. Warp the current image towards the previous image based on the prediction.
   (a) Apply calculated weights.
   (b) Use gradient descent to find what correction of the motion minimizes the pictures intensity differences.
   (c) Iterate.

3.4 Chosen method: Displacement field

To be able to compare these methods they were applied to recorded clips with available speedometer and yaw rate data. The car used for recording the clips was also fitted with an extra set of sensors which recorded more precise motion. The results of each method and the commonly available car-data was then compared to the output of those extra sensors. To visualize the result, the output of each method was integrated and fitted to a satellite image of the recording place, shown in Figure 3.5.

The direct method gives similar result to the displacement field method in large. However, since it operates on the full image, it is more expensive to use. Since the displacement field method is less computationally demanding and easier to analyze mathematically it has been chosen for further use in the vergence calibration.

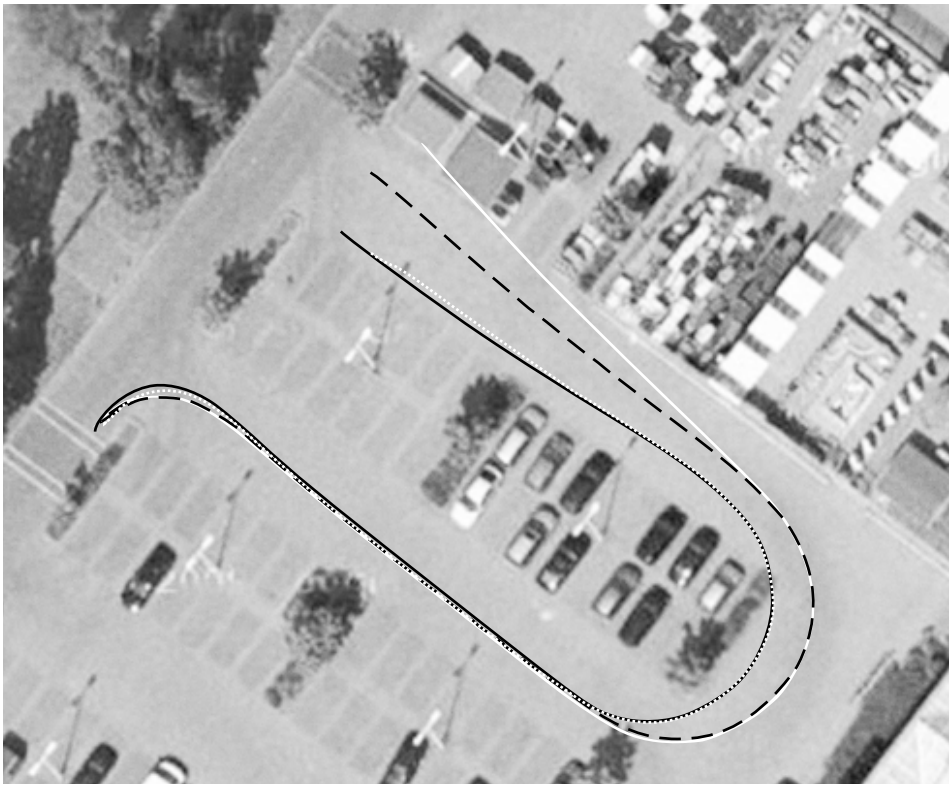


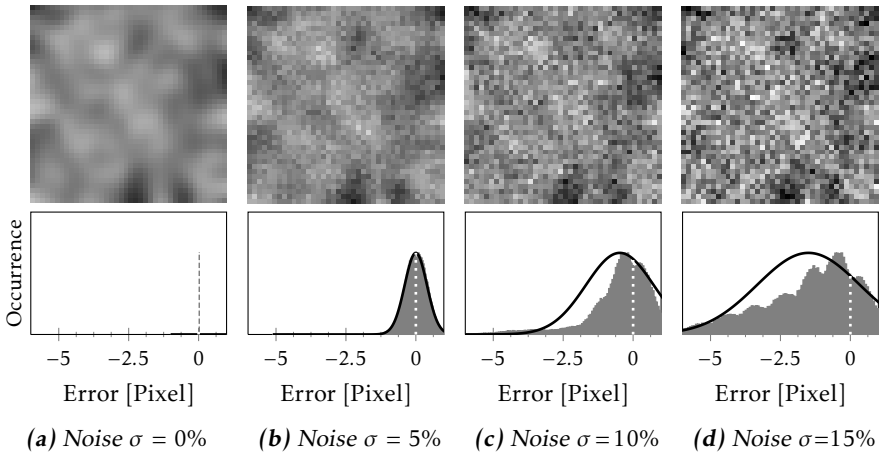
Figure 3.5: *The direct method (white dotted line) gives similar results to the displacement field method (black line). CAN data (white line) and accurate sensor data (dashed black line) are shown for comparison.*

3.5 Speed estimation noise sensitivity

For the vergence calibration method described in Chapter 4 to be feasible, the noise in speed estimation need to be sufficiently bias free, see Section 4.4. In this section, the noise sensitivity of the chosen method is investigated.

3.5.1 Displacement tracking error

The method for finding the displacement field is based on a minimization of the square error between image patches. When the image has added noise, there's a risk that a displacement of the noisy first frame matches the noisy second frame better than for the true displacement. To test this, the base image in Figure 3.6a has been translated five pixels. Bias free Gaussian noise with standard deviation σ given as a percentage of the maximum possible intensity has then been added separately for the first and second image. The image center of the first noisy image was then tracked in the second noisy image using the LK implementation in OpenCV [Bradski, 2000]. In Figure 3.6b, it can be observed that the image affected by less than 5% Gaussian noise has close to zero mean Gaussian distributed error.



(a) *Noise* σ = 0% ***(b)*** *Noise* σ = 5% ***(c)*** *Noise* σ = 10% ***(d)*** *Noise* σ = 15%

Figure 3.6: *The distribution of errors in pixel offset (gray) is close to Gaussian (black line) up to 5%. Above that, the mean error deviates from 0 (dotted line). True translation was five pixels.*

The estimation is no longer correct in mean (white dotted line) when high amounts of noise are applied, shown in Figure 3.6d. In Figure 3.7, the relation between noise level and error is shown. And as can be seen is close to linearly dependent of noise level up to noise with 5% standard deviation.

These tests has not been done with scale and perspective changes. For a camera the motion includes some perspective changes that might be harder to track and introduce bias in motion.

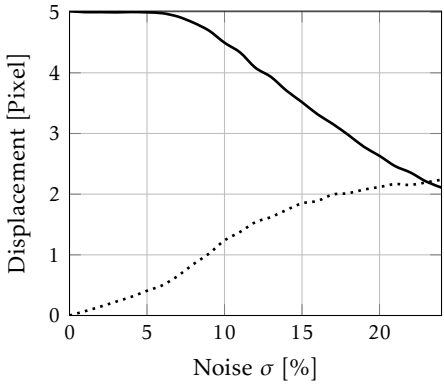


Figure 3.7: *The standard deviation (dotted line) in the error of the estimated displacement increases linearly up until a noise level of ca 6%. The mean displacement (black line) is close to the correct value of 5 pixels up to the same noise level.*

The displacement field method uses a linear mapping from the displacement field $\mathbf{d}_e$ to the estimated motion,

$$\mathbf{m}_e = \mathbf{J}^\dagger \mathbf{d}_e \tag{3.25}$$

The displacement $\mathbf{d}_e = \mathbf{d}_t + \widetilde{\mathbf{d}}$ is assumed to be normally distributed like in Section 3.5.1.

$$\mathbf{d}_e \sim \mathcal{N}(\mathbf{d}_t, \mathbf{Q}_\mathbf{d}), \quad \mathbf{Q}_\mathbf{d} = \sigma \cdot \mathbf{I} \tag{3.26}$$

Since $\mathbf{m}_e$ is a linear combination of normally distributed variables, it is normally distributed like

$$\mathbf{m}_e \sim \mathcal{N}(\mathbf{m}_t, \mathbf{Q}_\mathbf{m}) \text{ where } \mathbf{m}_t = \mathbf{J}^\dagger \mathbf{d}_t \text{ and } \mathbf{Q}_\mathbf{m} = \mathbf{J}^\dagger \mathbf{Q}_\mathbf{d} \mathbf{J}^{\dagger T} \tag{3.27}$$

Consequently:

$$\widetilde{\mathbf{m}} = \mathbf{m}_e - \mathbf{m}_t = \mathbf{m}_e - \mathbf{J}^\dagger \mathbf{d}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_\mathbf{m}) = \mathcal{N}(\mathbf{0}, \mathbf{J}^\dagger \mathbf{Q}_\mathbf{d} \mathbf{J}^{\dagger T}) \tag{3.28}$$

This means that if the input noise of the displacement is bias free, the resulting output error of the method will be bias free. In other words, if a mean over time (given fixed speed) is taken, the error will tend towards zero.

However, this assumes that input noise is Gaussian and bias free. In the real world there might be other kinds of noise than the one tested. There might also be feature points on cars and other moving objects that does not get detected and filtered properly.

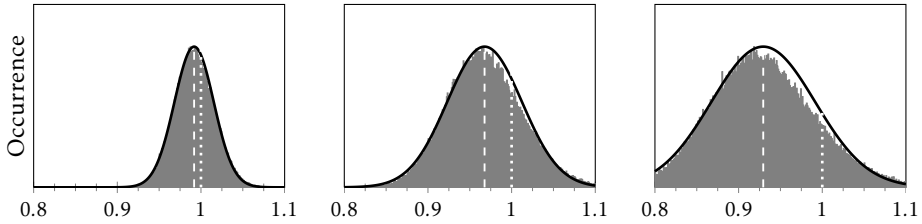
3.5.2 Depth/disparity estimation error

The depth image is acquired by finding the horizontal displacement (disparity) between the left and the right image for each point. Errors have similar characteristics as for the displacement field calculations. At least for a subset of all points in the image, the assumption is that disparity is bias free for low amounts of image noise.

It isn't as simple though, to find out how this affect the ego-motion estimation; this is because the depth is used in the construction of the Jacobian matrix $\mathbf{J}$. The Jacobian matrix is pseudo inverted in the solution $\mathbf{m} = \mathbf{J}^{\dagger}\boldsymbol{d}$, which means that depth isn't linearly affecting $\mathbf{m}$.

The method used to analytically determine the effects of displacement noise can't be used. Therefore, the problem is analyzed with Monte-Carlo sampling using a specific test scene. The testing is done using a simulation in Matlab. A point cloud is made representing a ground surface sampled using a grid pattern like the one used in Section 3.2.

The point cloud is then projected using the projection formula (3.1). The calculated disparity of each point is then altered by zero mean Gaussian noise with standard deviation σ. Then the ego-motion is estimated using the displacement field method for a forward motion of one meter.



(a) *Disparity σ = 2 Pixel* ***(b)*** *Disparity σ = 4 Pixel* ***(c)*** *Disparity σ = 6 Pixel*

Figure 3.8: *Increasing disparity noise results in increasing bias (dashed white line) in speed estimation. The distribution (gray area) is close to gaussian (black line) when the disparity noise is low. The true speed (dotted line) is the same for all trials.*

In Figure 3.8 it can be seen that the error is close to normally distributed for this test situation. However large amounts of noise causes a bias in the speed estimation, as shown in Figure 3.9. This means that even over time, a good estimation can't be achieved if the disparity image has too much noise. This has to be taken into consideration when evaluating the feasibility of the vergence calibration method described in Chapter 4.

The result of this simulation is only truly valid for this specific test case. Real world data isn't planar and feature points will be detected on buildings and other

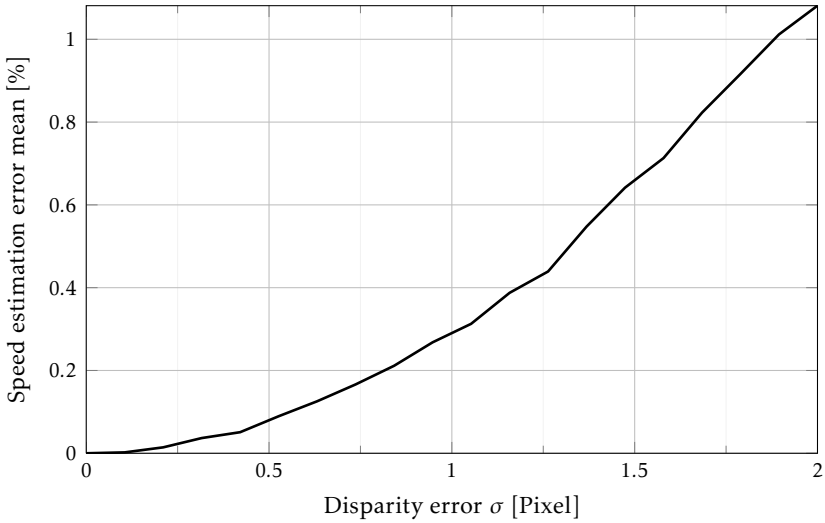


Figure 3.9: *The effect on the estimation error mean is low for sub-pixel standard deviation in disparity error. The true motion was one meter forward.*

static structures. However, a subset of real data will be on the road surface which is planar like in the simulation. Therefore this result can be seen as a rough estimate of the upper bound for the error caused by a Gaussian disparity error given a dense enough grid.

3.5.3 Yaw deviations

The chosen method is based on the assumption that all motion is infinitesimal between frames. A deviation from this assumption will cause deviations in speed estimation. The most common deviation, that the forward motion is larger than infinitesimal is treated in Section 3.1.2. The second most common deviation from the assumptions that causes large errors is that the yaw rate can be large between frames when the car is turning; sometimes it's more than one degree.

The camera is mounted in the front of the car and the rotation axis for a car in motion is in the rear axis. This means that any yaw change for the car will also result in a sideways translation of the camera, assuming no skidding. This relation is included in the calculations below, but it doesn't change the result in any significant way.

To see how big this effect is, a simulation was run in Matlab with the same test scene as in Section 3.5.2. If only frames where the yaw rate is low are used, the speed estimation error caused by high yaw rate can be kept below 0.2% (See Figure 3.10).

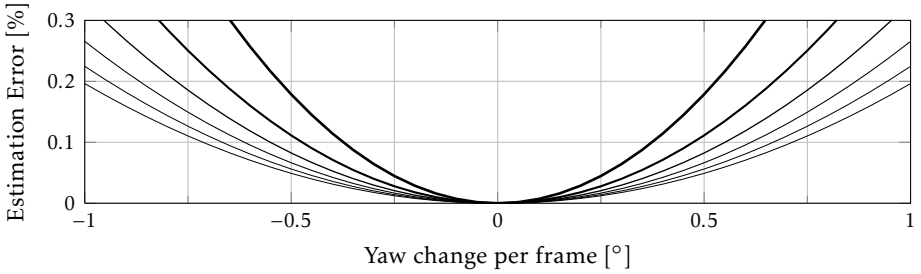


Figure 3.10: *For a yaw rate of less than 0.5° per frame, the speed estimation error is less than 0.2% for a speed of 10 km/h (thickest line). If the speed increases to 110 km/h (thinnest line), the error decreases to less than 0.1%*

The simulation has not included noise. The effects of yaw deviations might be amplified when the data is noisy.

4

Vergence Calibration

This chapter first introduces a potential method for calibrating vergence in short time using normal video input, without any special calibration targets. Then the required components for the method are derived. The noise sensitivity of the method is analyzed. Finally, the method is evaluated on video recordings with calibration error introduced manually.

The output of a stereo camera system is dependent on the knowledge of the relative pose between the cameras. There are six degrees of freedom for the relative camera pose: yaw, pitch, roll, horizontal, vertical and depth axis position. Since the cameras in the setup are mounted in a fixed frame, the relative position does not change enough to cause significant errors.

However, even a tiny variation in the vergence will cause a significant error in depth estimation and therefore in speed estimation as well (see Figure 4.3b). The vergence can be calibrated using other methods used at Autoliv, but short term (less than a few minutes) calibration is a problem that isn't entirely solved. This chapter introduces one potential solution and tries to determine whether it is feasible in a real situation.

4.1 Problem definition

The method is based on the fact that an error in vergence angle, shown as γ in Figure 4.1, will cause an error in depth. This will then result in an error in speed estimation. The speedometer information provided by the car is used as reference, and different amounts of vergence compensation are tried in turn. The vergence angle for which the speed estimation is closest to the true speed is used as the resulting calibration output. Theoretically, the process can be done with

only two consecutive stereo frame pairs. To reduce noise, it is done for many frames.

The estimated forward speed, s_e can be described as a function of vergence error γ, given two consecutive image pairs,

$$s_e = h\left(\gamma | P_{\mathrm{l}1}, P_{\mathrm{r}1}, P_{\mathrm{l}2}, P_{\mathrm{r}2}\right) \tag{4.1}$$

Given the true speed s_t, the vergence error is found according to

$$\hat{\gamma} = \arg\min_{\gamma} \left| s_t - h(\gamma) \right|^2 \tag{4.2}$$

To solve this equation, the function h is sought. The displacement field method is described using $\mathbf{m}_e = \mathbf{J}_e^{\dagger}\mathbf{d}_t$.[1] The only part of $\mathbf{J}_e(\gamma)$ that is a function of γ is the depth $Z_i(\gamma)$ for each point used in the estimation. In the next section, $Z_i(\gamma)$ is derived.

4.2 Estimating depth

As mentioned in Chapter 2, the relation for calculating depth for a calibrated setup is as simple as Depth $= \frac{\text{Baseline}}{\text{Disparity}}$. However, in the scenario of this thesis, the vergence angle is non-zero, so the calibration is incorrect. Therefore, a new expression is derived.

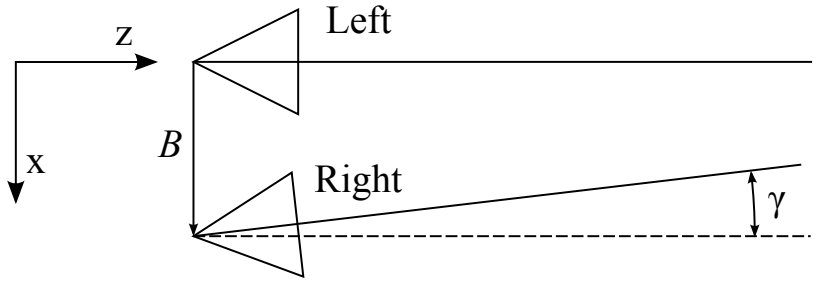


Figure 4.1: *The vergence error γ represents how the right camera is rotated relative to the left camera. The baseline B is the distance between the two cameras.*

Figure 4.1 illustrates a stereo setup with vergence. The left camera is placed in $(0, 0, 0)$ and looking in the Z direction. The right camera is moved with the baseline distance B to end up in $(B, 0, 0)$, then rotated with the vergence angle γ.

[1] For the pseudoinverse to be usable, $\mathbf{J}_e$ has to be of full rank. This is usually the case for actual data with more than a few data points.

Note: In other related works, left and right vergence are sometimes modeled separately [Brooks et al., 1996]. Modeling the angles separately makes calculations longer and the difference can just as well be described as a difference in z-position between the cameras. For small angles, the z-component would be insignificant compared to the x-component. To make calculations more compact, the vergence angle is modeled only for the right camera in this thesis.

Since stereo separation can be seen as a horizontal camera motion, the motion relation from Section 3.1.1 is reused. So the relation between vergence and stereo estimated depth $\mathbf{Z}(\gamma)$ is found using (3.6) with Δx_i seen as the disparity. Eliminating all rotation but vergence $w_y = \gamma$ and all translation but baseline $t_x = B$ gives the expression

$$\Delta x_i(\gamma) = x_i - \frac{\cos\gamma\ (x_i\, Z_i - B) + \sin\gamma\, Z_i}{\cos\gamma\, Z_i - \sin\gamma\ (x_i\, Z_i - B)} \tag{4.3}$$

This equation can be solved to extract depth as a function of vergence error

$$Z_i(\gamma) = \frac{(\sin\gamma\, x_i - \Delta x_i\ \sin\gamma + \cos\gamma)\ B}{\sin\gamma\, x_i^2 - \Delta x_i\ \sin\gamma\, x_i + \sin\gamma + \Delta x_i\ \cos\gamma} \tag{4.4}$$

4.3 Vergence estimation algorithm

The equation (4.2) is numerically solved in Algorithm 4.1 using a grid search. The estimated speed function $h(\gamma)$ is calculated using the displacement field method with the depth replaced with (4.4).

Algorithm 4.1 Vergence calibration algorithm

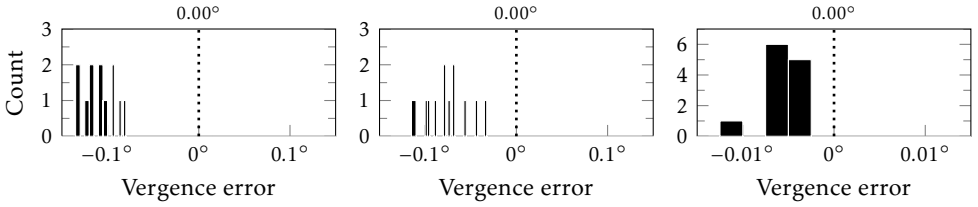
- For each frame in a clip:
  - Estimate motion $\mathbf{m}_e$ given a series of vergences, $\gamma_1, \gamma_2, ..., \gamma_m$.
  - If the motion is forward, (Yaw < Yaw_0) (see Section 3.5.3)
    - Find for which vergences the difference between true and estimated speed changes sign.
    - Add those vergences to the set Γ of estimated vergences.
- Return the mode of Γ as the estimated vergence.

If there is more than one minima of the error function, all are used to be sure to not miss the correct alternative. The basis of the error function, $s_t - h(\gamma)$, should cross zero if the true error is within the range and no errors occurs. This means that the minima of the error function will be when $s_t - h(\gamma)$ crosses zero, i.e. where it changes sign between two consecutive γ_i.

In Section 3.5 the effects of bias free Gaussian displacement noise are found to be zero mean in speed estimation. As long as the errors are reasonably low, the long time mean value of estimations of a given static speed will be accurate. This would mean that the mean value of the estimated vergence error per frame pair should be correct.

However, this is based on an assumption of no outliers and trials with the mean does not perform well as seen in Figure 4.2a. Using the median is often a good way to reduce the effect of outliers. However, the median only works if the outliers aren't too many and biased in any way. As Figure 4.2b shows, using the median does give a slight improvement; the estimation is still quite off.

To improve this, a histogram of estimated vergences is created, and the peaking value, the mode, is used. The results shown in Figure 4.2c are close to the true value. If the vergences are estimated to arbitrary precision the histogram needs to be filtered in some way. In the trials done here the histogram is quantized to 0.025°precision which is enough to create a clear peak in the data for the data set used in these trials.



(a) Method: mean *(b) Method: median* *(c) Method: mode*

Figure 4.2: *The resulting vergence estimations for a number of 30-second clips with* $0.00 \pm 0.005°$ *error are shown. Three statistical methods are used to estimate vergence based on all per-frame estimations in the clip. Using the mode of the estimations yields results closest to the true value*

4.4 Speed estimation vergence sensitivity

For this method to be usable in real traffic situations, the effects of vergence error has to be greater than the effects of other error sources like noise. The speed estimation error caused by a change in vergence of the same size as the desired precision in the vergence estimation needs to be greater than the speed estimation error caused by other factors.

In Chapter 3 the effects of noise and yaw error on speed estimation are investigated. If the results shown in Figure 3.7 and Figure 3.9 are combined, it can be inferred that for image noise with standard deviation $\sigma = 5\%$ the speed estimation error is around 0.1%. If the image noise is increased to 15% the speed estimation error increases to around 1%.

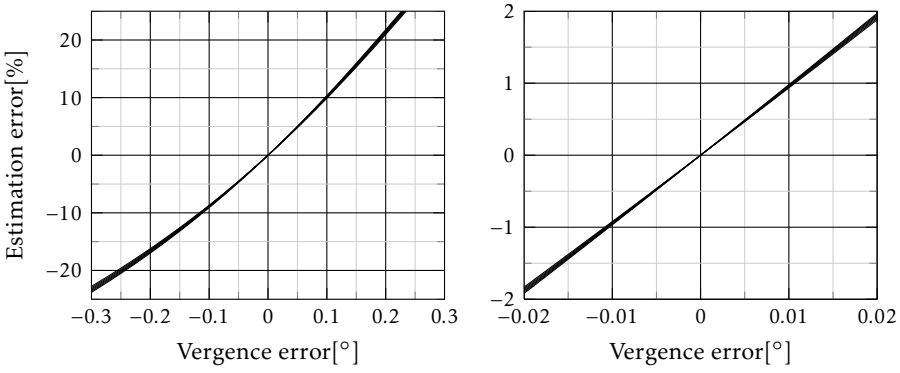
Section 3.5.3 continues to investigate the effects of the car steering on the speed estimation error due to approximation. As long as yaw is kept low, the error caused by steering should be less than 0.2%.

If the error caused by noise and yaw rate are added, an error of between 0.3% and 0.12% could be expected. Since 15% image noise is more than seen in actual data,

an expected error in the lower end of the scale, 0.5% has been used in further analysis.

Since the relation between vergence and speed estimation error is depending on the specific feature points detected in the actual scene, it can't be found for the general case. The scene described in Section 3.5.2 is used here, to analyze the error caused by a vergence error. Figure 4.3 shows that, with the expected error of 0.5% from speed estimation, the vergence should be detectable to within 0.01°for low deviations.

If this conclusion is going to be applicable to a clip in real life; the actual scene should be reasonably similar to the test scene. As mentioned in Section 3.5.2 it might not always be the case. In fact, during testing, an upside down curve has sometimes been observed. This has not been further investigated.



(a) *0.3° range* ***(b)*** *0.02°range*

Figure 4.3: *Even a small vergence error of 0.01°causes an 1% error in speed estimation. For speeds in the range 10 km/h to 110 km/h the slope is similar. This range is fully represented within the thickness of the line.*

4.5 Vergence error calibration method

Given the analysis of noise and errors in Section 4.3 and Section 4.4, it is reasonable to hope for a method of estimating vergence error that is accurate to within 0.01°. This was tested using real recorded clips where the calibration was purposefully incorrect. The clips were divided into three sets, as shown in Figure 4.4. Set 1 contains clips recorded on a german highway during daylight. Set 2 contains mainly german highway during different times of the day. Set 3 contains mixed driving situations in Sweden under varied conditions.

They all had an added offset in vergence calibration of ±0.1°, 0.03°, −0.02° and 0.01° respectively. For brevity, only a subset of the results are presented here. Before adding error, the vergence was calibrated to within 0.01° accuracy.

(a) Set 1: German highway during a foggy day

(b) Set 2: German highway during evening and morning

(c) Set 3: Mixed driving in sweden

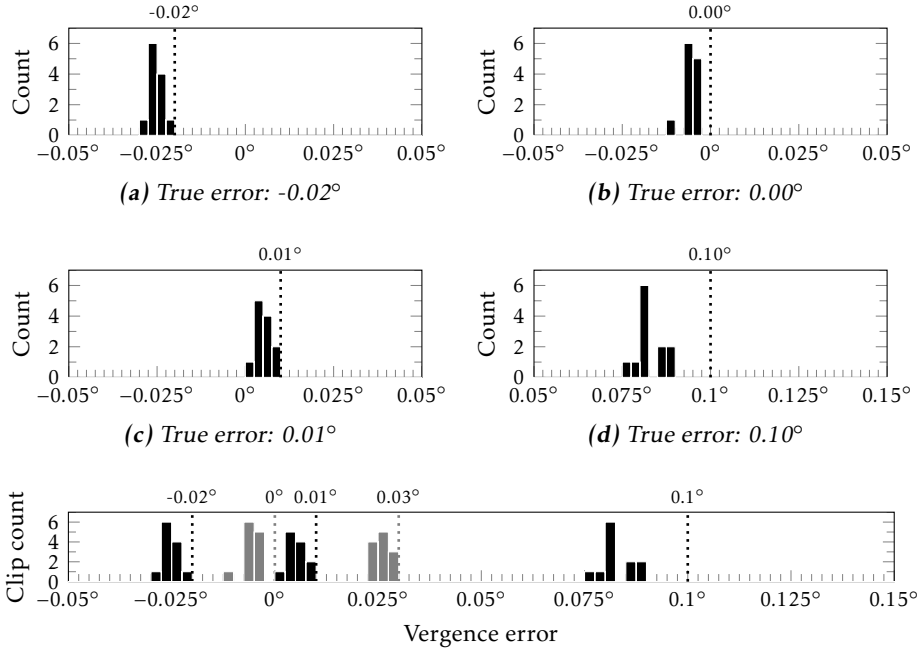
Figure 4.4: *The method was tested on three different clip sets recorded with Autoliv's equipment.*

The clips were run for their full length of 30 seconds. The vergence was estimated for each frame with lower yaw rate than 0.5°/frame. For each clip, the mode of all per-frame vergence estimations was used as the final vergence estimation.

For the clips in the first set (see Figure 4.5), the 0.01° added error is clearly separable from no added error; it can be concluded that, at least for the clips in Set 1, it is possible to detect the vergence with 0.01°precision. The original clips are calibrated to within 0.01°precision; the offset can be explained within that variance.

In the second set, where the clips are more varying, there is also more variance in the estimations; there are some overlap of the estimations, as seen in Figure 4.6. The third set is the one with highest variation in lighting as well as scenery. The initial calibration isn't as precise as in the other sets, causing an offset for all clips. Figure 4.7 shows that these concerns have an adverse effect on the end result. The mean offset between added calibration errors is, however, still clearly separable.

At least for simple situations and scenes, it is clearly possible to detect changes in calibration down to 0.01° using this method. There is a larger error for the 0.1°clips. This error could possibly be averted by using the previous estimate as the initial estimation or by using iteration.



(a) True error: -0.02°

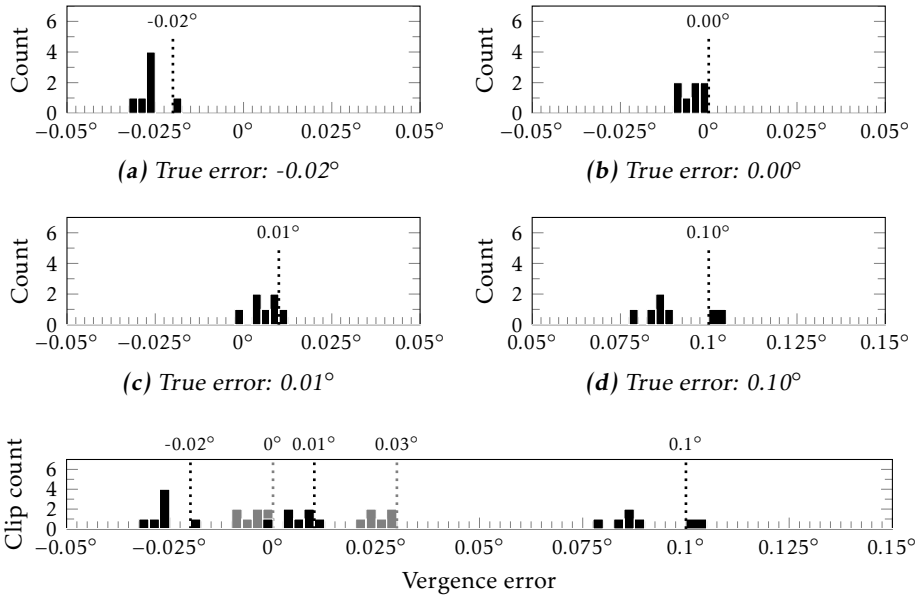
(b) True error: 0.00°

(c) True error: 0.01°

(d) True error: 0.10°

(e) All errors combined

Figure 4.5: *The estimated vergence error for the first set with added calibration error (dotted lines) has a bias. However each error is clearly distinguishable from other errors as seen in the combined subfigure. In the combined figure, the result for a 0.03°error is added. Each bar shows the number of 30-second length clips in the test set with the vergence estimation given on the horizontal axis using the proposed method.*



(a) *True error: -0.02°*

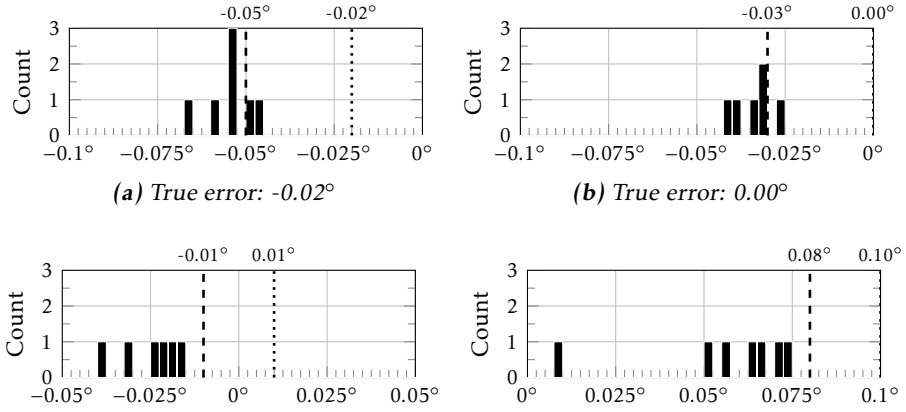
(b) *True error: 0.00°*

(c) *True error: 0.01°*

(d) *True error: 0.10°*

(e) *All errors combined*

Figure 4.6: *In the second set with more varied scenery, the method does not perform as well as in the first set. The ranges are now overlapping for some ranges. Each bar shows the number of 30-second length clips in the test set with the vergence estimation given on the horizontal axis using the proposed method.*



(a) *True error: -0.02°*

(b) *True error: 0.00°*

(c) *True error: 0.01°*

(d) *True error: 0.10°*

Figure 4.7: *The reference calibration for the third set seems off by approximately* 0.02°*. Compared to the offset true value (dashed line) the results are more often too low than for the other sets. Since the ranges are highly overlapping, they are not presented together. Each bar shows the number of 30-second length clips in the test set with the vergence estimation given on the horizontal axis using the proposed method.*

5
Concluding remarks

In this chapter a concluding summary of the thesis is given and ideas for future work on the method are presented.

5.1 Conclusion

In Chapter 3, a method for estimating visual odometry is sought. Two methods are derived, the displacement field method and the gradient descent method. The displacement field method is chosen due to lower computational demand and the fact that it has adequate precision compared to the more costly gradient descent method.

Chapter 4 introduces a method for calibration of vergence angle. It uses a combination of visual odometry, from Chapter 3, and mechanical odometry, from the speedometer in the car. According to estimations in Chapter 4, it is feasible to get accurate results at least to within $0.01°$ precision.

The method is implemented in Autoliv's framework and tested on clips provided by Autoliv. The results from two of the sets have a bias error of approximately $0.005°$ that could be explained by an error in the provided ground truth calibration. It is found that, at least for simple situations and scenes, it is clearly possible to detect changes in calibration down to $0.01°$ using the evaluated method.

5.2 Future work

The tests has a bias in estimation, this is most likely due to calibration errors in the source material. The method could be tested on clips with an error of less

than 0.001° for a more accurate estimation of the precision in mean. In the 0.1° clip the error is larger than the bias, this might be due to approximation errors. The approximation errors due to linearization should be analyzed and a solution for large error calibration should be sought. An iterative solution would probably solve the problem, albeit with a computational cost.

In Section 4.4, a discrepancy between the model and real data is briefly mentioned. This should be investigated further and the reason should be identified.

The described method estimates the speed given all possible vergence errors. This is computationally expensive; it might be possible to reduce the vergence estimation step to a single expression rather than an iteration procedure. This could either be done by restricting the movement for a frame to only forward motion, or by analytically finding the relation between speed error and vergence error based on a given set of feature points.

The accuracy of the solution is dependent on the accuracy of the speed estimation. The direct method described in Section 3.3 could possibly be extended to directly give the vergence error. This could reduce the number of intermediate steps and improve the estimation result.

For more accurate reliability data, a much larger data set should be used. Problems caused by peculiar situations should then be solved as they are found.

Bibliography

Gilad Adiv. Determining three-dimensional motion and structure from optical flow generated by several moving objects. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, PAMI-7(4):384 –401, july 1985. ISSN 0162-8828. doi: 10.1109/TPAMI.1985.4767678. Cited on pages 2 and 12.

G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000. Cited on pages 13 and 18.

M. J. Brooks, L. Agapito, D. Q. Huynh, and L. Baumela. Direct methods for self-calibration of a moving stereo head. In *Proc. European Conference on Computer Vision*, pages 415–426. Springer, 1996. Cited on pages 3 and 25.

C. Golban and S. Nedevschi. Linear vs. non linear minimization in stereo visual odometry. In *Intelligent Vehicles Symposium (IV), 2011 IEEE*, pages 888 –894, june 2011. doi: 10.1109/IVS.2011.5940537. Cited on page 2.

K.J. Hanna. Direct multi-resolution estimation of ego-motion and structure from motion. In *Visual Motion, 1991., Proceedings of the IEEE Workshop on*, pages 156 –162, oct 1991. doi: 10.1109/WVM.1991.212812. Cited on pages 12 and 14.

R. Horaud, G. Csurka, and D. Demirdijian. Stereo calibration from rigid motions. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(12):1446 –1452, dec 2000. ISSN 0162-8828. doi: 10.1109/34.895977. Cited on page 3.

J. Horn, A. Bachmann, and Thao Dang. Stereo vision based ego-motion estimation with sensor supported subset validation. In *Intelligent Vehicles Symposium, 2007 IEEE*, pages 741 –748, june 2007. doi: 10.1109/IVS.2007.4290205. Cited on page 12.

M. Irani, B. Rousso, and S. Peleg. Recovery of ego-motion using image stabilization. In *Computer Vision and Pattern Recognition, 1994. Proceedings CVPR '94., 1994 IEEE Computer Society Conference on*, pages 454 –460, jun 1994. doi: 10.1109/CVPR.1994.323866. Cited on pages 2 and 12.

M. Irani, B. Rousso, and S. Peleg. Recovery of ego-motion using region alignment.

Pattern Analysis and Machine Intelligence, IEEE Transactions on, 19(3):268 – 272, mar 1997. ISSN 0162-8828. doi: 10.1109/34.584105. Cited on page 2.

B Johansson. Derivation of lucas-kanade tracker, November 2007. URL `http://www.cvl.isy.liu.se/education/graduate/undergraduate/tsbb12/reading-material/LK-derivation.pdf`. Cited on page 15.

H. C. Longuet-Higgins and K. Prazdny. The interpretation of a moving retinal image. *Proceedings of the Royal Society of London. Series B. Biological Sciences*, 208(1173):385–397, 1980. doi: 10.1098/rspb.1980.0057. Cited on page 2.

B.D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the 7th international joint conference on Artificial intelligence*, 1981. Cited on pages 13 and 15.

G.P. Stein, O. Mano, and A. Shashua. A robust method for computing vehicle egomotion. In *Intelligent Vehicles Symposium, 2000. IV 2000. Proceedings of the IEEE*, pages 362 –368, 2000. doi: 10.1109/IVS.2000.898370. Cited on pages 12 and 13.

Zhengyou Zhang. Flexible camera calibration by viewing a plane from unknown orientations. In *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*, volume 1, pages 666 –673 vol.1, 1999. doi: 10.1109/ICCV.1999.791289. Cited on page 2.