Diploma thesis

in Physics

submitted by

Stephan Nicolas Robert Meister

born in Mannheim, 08/26/1984

2010
A Study on Ground Truth Generation for Optical Flow

This diploma thesis has been carried out by Stephan Meister

at the

Heidelberger Collaboratory for Image Processing (IIC)

of the

Interdisciplinary Center for Scientific Computing (IWR)

under the supervision of

Herrn Prof. Dr. Bernd Jähne
Alternative Ansätze zur Erzeugung von Ground Truth für optische Flussverfahren:


A Study on Ground Truth Generation for Optical Flow:

This thesis describes two methods for generating reference or evaluation data for optical flow algorithms. The first method deals with the fact that synthetic image sequences are often considered to be too unrealistic for algorithm evaluation. For that matter the effects of physically plausible renderings on optical flow fields are evaluated. This was done by recording an image sequence and reconstructing it using modern 3D computer graphic and raytracing methods. The results indicate that further studies regarding the illumination and material properties in synthetic scenes are necessary.

The second method uses stereo images and monoscopic camera tracking to calculate the optical flow of an image sequence. The method is tested on one synthetic and two recorded scenes from the automotive sector. The results show that the performance of the method is not yet sufficient for the creation of reference data but allows further investigation regarding the common constraints of classical optical flow methods.

For obtaining the images used in those methods, a stereo camera system, specialized for the acquisition of images with a high resolution in the temporal, spatial and intensity domain was constructed. It is combined with an high precision GPS/IMU system for use in the automotive sector.
Special thanks to Dr. Daniel Kondermann for supervision and support.
Contents

1 Introduction .................................................. 9
  1.1 Introduction ................................................. 9
  1.2 Related Work ................................................ 10
  1.3 Outline ...................................................... 11

2 Experimental Equipment ...................................... 12
  2.1 Stereo Camera System ...................................... 12
  2.2 Noise Handling and Radiometric Camera Calibration ........ 14

3 Improved Synthetic Scenes ................................... 18
  3.1 Method Description ....................................... 18
  3.2 Theory: Image Generation and Camera Calibration ........ 18
  3.3 Theory: Optical Flow ...................................... 20
    3.3.1 Error Measures ...................................... 20
  3.4 Experimental Setup ....................................... 21
    3.4.1 Real-World Scene ................................... 21
    3.4.2 Synthetic Scene .................................... 22
    3.4.3 Reconstruction Evaluation .......................... 24
  3.5 Results ................................................... 26
    3.5.1 Algorithm Evaluation ................................ 26
    3.5.2 Error Evaluation .................................... 27
    3.5.3 Conclusions and Future Work ......................... 29

4 Stereo Augmented Flow Estimation .......................... 33
  4.1 Theory: Stereo ........................................... 34
    4.1.1 Stereo Vision and Epipolar Geometry ............... 34
    4.1.2 Depth Estimation from Disparity .................... 34
  4.2 Detailed Method Description .............................. 36
  4.3 Theoretical Error Evaluation ............................ 38
  4.4 Experiments ............................................... 43
    4.4.1 Synthetic Test Scene ................................ 43
    4.4.2 Car Scenes ......................................... 47
  4.5 Possible Improvements and Future Work .................. 57

5 Conclusions .................................................. 58
  5.1 Summary of Experimental Results ......................... 58
  5.2 Method Evaluations and Outlook .......................... 58
  5.3 Final Comments .......................................... 60

List of Figures ................................................. 61

List of Tables ................................................ 62

Bibliography .................................................... 63
1 Introduction

1.1 Introduction

Subject of this study is the generation and analysis of reference data (also known as ground truth) for optical flow estimation.

Optical flow is the vector field describing the apparent motion between two consecutive frames in an image sequence, and is usually given in pixels per frame. It is itself an approximation of the 3D-motion in a given scene, projected onto a camera.

This motion field has a wide field of application, ranging from motion detection over image segmentation to image compression. Comparably numerous are the practical applications like driver-assistance-systems, estimation of fluid or gas motions in industrial applications or in the entertainment industry.

There are various algorithms for calculating the optical flow; all with different strengths, weaknesses and fields of application. Actually the number of algorithms or algorithm and method related publications is extremely high (well over 1000 papers), while at the same time the number of publications evaluating these methods or showing methods on how to perform such evaluations are thin on the ground. Because any possible user of optical flow methods and algorithms may have difficulties choosing the right solutions for his/her application, it seems crucial that the optical flow community is in need of more evaluations or evaluation techniques.

The best evaluation of any numerical approximation is of course the quantitative comparison with the expected real values. In cases where this true motion is known (therefore ground truth; name derived from aerial or satellite image analysis), algorithms can be evaluated by means of certain error measurements.

Evaluation sequences can either be synthetic (aka computer-generated), in which case the ground truth is exact, or be taken under controlled lab conditions, in which case the ground truth is just an approximation. We therefore prefer the term reference data for flow fields which are used to evaluate optical flow algorithms. This poses a dilemma as test sequences which are both realistic and provide exact ground truth are difficult to produce.

An additional shortcoming of the two methods is that both are only feasible for a small set of real-world applications. Fields like the entertainment or automotive industry often deal with complex and difficult image sequences in which fast changing, suboptimal lighting conditions, reflections, motion blur, non-rigid motion etc. occur. Due to these difficult properties such sequences are almost neither synthesized nor evaluated.

In this study, two methods for generating reference data are presented and analyzed. The first method uses 3D-rendering techniques to generate synthetic scenes which resemble real world images as close as possible (in terms of optical flow). This way the problem of synthetic scenes being to unrealistic is addressed.

The second method uses stereo images to reconstruct the geometry of a given scene. In combination with camera tracking methods (or camera calibration) a point cloud in 3D-space is reconstructed, from which the optical flow can be calculated. This makes it possible to provide
an alternative source of reference data for complex outdoor scenes. Furthermore the resulting
data can be used to determine if the computation of optical flow in any given scene or scene
parts is actually feasible.

Ultimately these two methods could be the starting point for the creation of several more robust
and more accurate methods for the evaluation of optical flow methods.

In addition, a high-performance stereo camera system has been developed and utilized for this
study, which is able to capture fast high-resolution stereo pairs for the use in the automotive
sector.

1.2 Related Work

There are few generic papers to summarize practices and the state of performance evaluation
in computer vision and computer graphics. Some characterization has been done by Thacker
et. al. [30].

Several datasets consisting of image sequences and according reference data for the evaluation
of optical flow computation are available. They include recorded sequences as well as synthetic
ones. Examples are the older Yosemite [11] and marbled block sequences [24] or the newer
ones from the Middlebury dataset [5] which also describes methods for creating these reference
flow fields. Benchmarks based on synthetic scenes and their creation have been described by
McCane et. al. [21]. All these sets can be considered to be very accurate but usually show only
indoor or synthetic scenes.

Similar datasets regarding stereo (mentioned here because of a high number of papers utilizing
both stereo and optical flow methods as well as our own use of stereo data) are also available
in the Middlebury datasets [27]. Evaluations have been performed by Hirschmüller, Scharstein
[13] or Szeliski [28].

Especially worth mentioning are the works by Vaudrey et. al. [35] or Liu and Klette [17] which
inspired some of the methods and test sequences used in this work, as they also used stereo data
and real world scenes taken from a driving car. Badino [3] uses the opposite of our approach
by calculating the camera motion using optical flow and stereo data. Wedel et. al. [38] use the
same data sources to compute the 3D sceneflow in traffic scenes.

Usually optical flow algorithms can be grouped into two classes: The first are gradient based
methods like the original algorithm by Horn & Schunck [14] respectively modern methods like
the algorithm by Zach et. al [40] which are used several times in this work. The second class
are correlation based methods which try to find correlating patches in image pairs. As image
correlations occur in various sub steps of the method described in chapter 4, one could describe
the algorithm as a distant relative of those correlation based methods.

In addition, several methods for producing reference data have been proposed. Examples are
mounting the camera on an robot arm [29], structured light or hidden textures [5].

This thesis relies heavily on works regarding camera calibration. Methods for the calibration
of cameras using already known 3D points are described by Tsai et. al. [34] or zhang [41]. If
these points are unknown, e.g. in case of so called uncalibrated images, the method of Thorsten
Thormählen [31] is used.
1.3 Outline

The first part of this thesis will focus on the hardware used to capture the various used images and stereo-image sequences. The hardware consists of special high-performance components which are in common use in the industrial sector and is especially tailored for use in the automotive sector. For this case it combines two fast high-resolution stereo cameras with a military grade GPS/IMU system (Global positioning system / Inertial measurement unit).

Chapter 3 focuses on the idea that modern 3D-rendering techniques, which are in common use in the entertainment industry can be used to create photorealistic test sequences for optical flow evaluation with physically plausibly effects usually missing in regular synthetic test sequences. Additionally, like any other synthetic sequence, they would provide accurate ground truth for quantitative evaluation. For that matter a real sequences was constructed under laboratory conditions and then synthetically reconstructed using the open-source render and modeling tool blender 1. Then the apparent motions in those scenes were calculated using an up-to-date optical flow algorithm [40] and a thorough examination of the encountered errors and problems was performed.

Chapter 4 of the study describes a method to create reference data especially tailored for the use in the automotive industry but which could also be adapted to other fields of application.

Stereo image pairs allow to estimate the distance of points from the camera, thereby creating a 3D-point cloud which is an actual representation of the 3D scene from which the images are projections of. Secondly camera calibration techniques (commonly called match moving or camera tracking) can estimate the camera motion from a set of consecutive frames. With these two sets of data one can calculate the motion of each projected pixel and thereby the optical flow it creates. Section 4.4 describes how the method performs on one synthetic and two real-world car scenes and compares the results to a classical optical flow algorithm.

---

1Blender Foundation, http://www.blender.org/
2 Experimental Equipment

A major goal of this project is to acquire data under many different and difficult conditions such as rapidly changing lighting conditions, occlusion, reflections and non-linear motion. The automotive sector for example provides us with such a challenging field of application, where the capabilities which modern hardware has to offer can be used up to its full potential.

What follows are some requirements of automotive image sequences and possibly other industrial applications.

Most consumer cameras, but also many industrial or research systems operate at rates of 25 or 30 frames per second. In car scenes, where objects can move at several dozen meters per second, those cameras can produce severe motion blur (depending on exposure time) or temporal aliasing effects. Take for example wheels which appear to rotate backwards if their angular velocity is above a certain value. Another point is that optical flow estimation of mostly carried out on only two consecutive images. Most sequences for evaluation purposes are therefore only a couple of frames long. Algorithm design could benefit from new test sequences several hundred frames long.

A high spatial resolution is of utter importance, as it allows e.g. the reliable detection of feature points or the detection of small motions. In case of stereo imaging it allows for better depth resolution, especially when algorithms without sub-pixel accuracy are deployed. Additionally this allows the use of subsampling schemes (spatial as well as temporal).

Furthermore most evaluations of optical flow (or any other computer vision task) are used on only 8 bit grayscale or color images. This is but merely a historic leftover as earlier cameras and image formats were restricted to those bit depth. But nowadays, even cheap consumer cameras have internal bit depths of 10, 12 or even more bits. Therefore there are few excuses, not to use cameras with a high bit-depth.

2.1 Stereo Camera System

With the mentioned possibilities in mind the capture system has been composed out of the following components: ¹

Hardware

Two Photonfocus MV1-D1312-160-CL CMOS cameras, which deliver 100 frames per second with a resolution of 1312x1082 pixels at 12 bit depth, are used to acquire the images. Additionally they use a global shutter to reduce artifacts due to fast motion and are practically bloom-loss, which is an advantage in fast changing lighting conditions.

¹Special thanks to Martin Schmidt for selecting and assembling the hardware components and for initial works on the capture drivers and compression scheme.
For the lenses Linos MeVis-C high precision lenses with a focal length of 25mm and a maximum radial distortion of less than 0.3 % are used.

Apart from the raw image capturing components, a high precision NAV440 Navigation System by Crossbow ² is also part of the system. The NAV440 combines an global positioning system (GPS) receiver with an inertial measurement unit (IMU).

For regular consumer grade GPS receivers the main source of error (apart from a too small number of visible satellites) are the varying signal delays due to ionization in the upper atmosphere. The NAV440 employs a satellite based augmentation system (SBAS), more specifically the European Geostationary Navigation Overlay Service (EGNOS), to enhance the accuracy of the position estimates to under 5m if the additional satellites are visible. ³ Heading and acceleration data are provided with an accuracy of $< 1.0\text{o}$ and $< 9.8 \cdot 10^{-3} \text{m/s}^2$ while the system operates at frequencies of up to 100Hz so that position information is made available for every frame of a sequence.

The GPS data makes it possible to start the image capture process at precisely determined points, so that real-world image sequences taken at the same position but under different lighting (or e.g. weather) conditions can be compared.

The cameras are connected to a standard desktop pc, which is optimized for low power consumption of $\approx 120 \text{ W}$, to make it suitable for mobile use. Except for the storage system the pc has no particularly high processing power, although the software makes heavy use of processing threads so a processor with a least 2 cores is necessary.

At full speed the cameras reach a raw data rate of 541 MB/s. At the time of first construction the maximum data rates for harddisks reached values of 80-130 MB/s, depending strongly on interface, individual model and the actual region of the disk plater where write-operations occur. This last point is especially problematic as this represents a contraint on the minimum transfer-rate, not it's average.

Write operations to a hard-disk are usually slower if sectors on the inner regions of the drive plater are accessed. Data fragmentation and space allocation policies of the file systems in use make this matter even worse.

Modern Solid-State-Disk (SSDs) have other shortcomings, as their initially high write performance can drop significantly over time once the number of free blocks decreases. This is caused by the fact that in order to write a section of the drive, the whole corresponding block (several kByte) need to be erased first. Advanced write-amplification-algorithms can counter this drop in performance but only the newest generation supports this Trim-command in an RAID-configuration. Furthermore it is not sure that those optimization strategies work in case of a full sequential disk write from beginning to end as the drive controller needs a certain amount of time and free space to restore the write performance.

So, as no single storage medium is capable of the data rates needed, the system uses four of the fastest available hard disk drives in an RAID 0 (redundant array of independent disks) configuration to spread write operations equally over all drives. This allows for maximum writing speed, but makes the system susceptible to data loss when one of the disks is damaged.

²More information on the GPS and EGNOS are available at United States National Executive Committee for Space-Based Positioning, Navigation, and Timing (PNT), http://www.gps.gov/technical/ps/ or the European Space Agency (EAS), http://www.esa.int/esaNA/egnos.html
Software

In any case a simple form of data compression is necessary to make continuous writing of image data possible. For this matter, one can exploit the fact that the four most significant bit of each 16bit word are zero, as the cameras provide only a pixel depth of 12 bit. By filling the gaps with values from other pixels the size of the data can be reduced to three fourth of the original amount.

Furthermore all file operations have to circumvent operating system write buffers to achieve maximum throughput. This is necessary because writing a file with standard library functions may induce unpredictable latency, depending on the number of bytes, the position of the memory buffer etc. So instead, the program uses the operating systems low-level API to write directly to the filesystem. This imposes the additional constraint that write buffers need to be page-, as well as sector-aligned, resulting in the fact that write operations must always target multiples of a certain hardware-dependant number of bytes.

Another option would have been to ignore the filesystem completely and directly access the hard disks as block devices. However the performance impact was negligible in contrast to the increased burden of managing raw block data.

The acquisition application is written in C++ using the Qt Development Framework ⁴ in order to provide an graphical user interface, while the image processing part of the application depends heavily on the CImg library ⁵. The current list of additional features which are usually not found in regular capture software reads as follows⁶:

- Single Frame mode allows acquisition of single images or accumulated images over an arbitrary number of frames
- automatic fixed-pattern-noise reduction in single frame mode
- capture trigger over TCP/IP
- automatic start of sequence acquisition at predefined GPS-coordinates⁷
- automatic pausing of acquisition if vehicle speed drops below a certain threshold⁷
- custom file format for capturing simultaneously stereo images, camera parameters, gps position, inertia data and additional meta data⁷
- circular memory buffer allows capturing of up to 12 seconds before the acquisition command was issued.

2.2 Noise Handling and Radiometric Camera Calibration

Handling of sensor noise is an important aspect of any non-purely synthetic image processing task.

Image sensors (both CCD and CMOS sensors) have different noise sources. Most obvious in cameras is the dark response non uniformity (DRNU), which is caused by (mostly

⁴Nokia, Qt Development Frameworks, http://qt.nokia.com/
⁶Additional thanks to Christoph Koke and Julian Coords for the NAV440 implementation, GPS trigger code, extensive code refactoring and bugfixes.
⁷Implemented by Christoph Koke
thermal) creation of electron/hole pairs in the semiconductors in use. Its intensity depends on the temperature and set exposure time.

By taking additional images with a closed aperture (or closed lens cap) one can sample over this noise and subtract it’s mean from all captured images. Of course this does not take into account the temporal variation of this noise source, but this is usually below 0.5% of the maximum intensity.

In a second postprocessing step the photo response non-uniformity (PRNU) and the sensor non-linearity are dealt with. Different sensor elements will report different digital values for the same amount of light (photon current) falling onto it. CMOS sensors are more prone to this effect as each pixel has its own amplifier with its own amplification curve. For CCD sensors there can be quantitative differences between alternating rows. Also, the resulting gray-values deviate from an optimal linear response to the incident light. Figure 2.1 shows the response curve of two pixels. Superimposed is a second order polynomial curve fitted to the data points.

By mounting the cameras into an integrating-sphere the light intensity to gray-value output curves were measured for 3 different wavelengths at every sensor pixel. For each incident photon current, the (temporal) arithmetic mean over 400 images was computed to get the response for each pixel. This was done for 200 different light intensities ranging from full darkness to an intensity where practically all pixels were saturated. By performing an second order polynomial fit on the data the inverse function of the pixel response can be determined. This allows for calculation of the photon intensity for a given pixel gray-value. To compensate for defective pixels, a local 3x3 median filter was used to replace the values of those irregular pixels.

Defect pixels were detected by two methods: First all pixels whose gray-value is below a certain threshold in the integrating sphere image with maximum light intensity are marked. This involves complete dark or weak pixels, but not hot ones.

Second all pixels which always differ significantly from their neighbours are marked. This is done by taking about 700 sample images from different scenes and checking for each pixel how its gray-value differs from the mean of its 4-neighbourhood. Those pixels where the difference is above a certain threshold (100 in this case) get marked. If the pixel was marked at least 30 times in those 700 images it is finally marked as a dead/hot pixel.

Keep in mind that those pixels are not necessarily defective but may only have response curves which are slightly off the mean. Even for low quality sensors the number of completely defective pixels is usually below 0.1 %. In this case the number of picture elements where this applies is
1402 for camera A and 2453 for camera B. The results of the radiometric rectification can be seen in Figure 2.2, which shows a part of the sky.

Figure 2.1: photo response curve for a regular and a defective pixel
Figure 2.2: Results of radiometric image rectification
3 Improved Synthetic Scenes

3.1 Method Description

This chapter investigates the fact that the results of optical flow algorithms often show conceptual differences when being applied to real-world and synthetic sequences. Synthetic images range from simple patterns like sinusoid waves\cite{9} to complex scenes rendered with raytracing algorithms (Grove or Urban sequences in the Middlebury Datasets\cite{5}).

Vaudrey et al.\cite{35} concluded that the major problems which make synthetic sequences inferior to real-world sequences consist in the fact that object and texture boundaries in synthetic images are much more distinguished and the brightness consistency between two frames is violated more often in real-world images.

Most optical flow algorithms either assume a brightness constancy constraint (BCC) or linear brightness variations. Furthermore, regularization techniques often involve the assumption of piecewise constant flow vectors or so-called smooth flow fields which usually means quadratic penalization of the first order flow derivatives. This essentially means that flow divergences and rotations are suppressed in the solution. Usually this is poorly motivated as divergences are anything but seldom and often mark valid regions of interest. These assumptions are one reason why synthetic ground truth is often criticized by researchers as well as industry partners.

Additionally Rokita\cite{26} and Kaneda et al.\cite{16} argue that artificial noise, like gaussian or salt-and-pepper noise are insufficient reproductions of real-world effects. The Intensity of thermal noise (or dark-current) is dependent on the exposure time and the chip temperature and can vary over the length of a sequence. Simulation of other noise sources like shot noise or fixed pattern noise are even more seldom and their effect on optical flow estimation is not well understood.

This thesis tries to address the point regarding the violated brightness consistency constraint, and at least show up a method to handle the other ones.

For that purpose a real-world scene with effects which are usually missing in synthetic ones was created and then a reproduction of those effects using modern raytracing techniques was performed. By reducing the realism of the synthetic scene one can investigate which effects are responsible for the discrepancies between the computed optical flow of the real and the synthetic scene.

By using more advanced rendering algorithms, noise sampling methods or other entertainment industry based techniques for creating photorealistic images one can select the definite parameter space to analyze which effects have the strongest influence and need to be dealt with separately.

3.2 Theory: Image Generation and Camera Calibration

Before any actual scenes or images are created, it is advantageous to have a look on how a camera can be described in a mathematical sense. The principles described here are applicable for real-world cameras, as well as for virtual ones.
Generally a camera describes a mapping from a point in $\mathbb{R}^3$ space to a Plane. (More generic concepts also deal with line cameras etc. but for the sake of brevity we will limit ourselves to the most common subset of cameras)

The simplest camera model which performs such a mapping is the pinhole camera. It can be characterized by a point $C$ called the *camera center* or center of projection and an image plane at a given distance $f$ (focal length) to the camera center. We place the point $C$ at the origin of an Euclidean coordinate system and the image plane perpendicular to the $z$-axis.

A given point $X = (x, y, z)$ is projected onto $X' = (x', y', z')$, the point of intersection between the image plane and the line $CX$. Using the similarity of triangles we can derive the coordinates of the projected point to be

\[ x' = \frac{fx}{z} \quad , \quad y' = \frac{fy}{z} \]

By introducing homogeneous coordinates we can express this as a linear mapping represented by a simple matrix multiplication.

\[
\begin{pmatrix}
  x'' \\
  y'' \\
  z'
\end{pmatrix}
= 
\begin{pmatrix}
  f & 0 & 0 & 0 \\
  0 & f & 0 & 0 \\
  0 & 0 & 1 & 0
\end{pmatrix}
\begin{pmatrix}
  x \\
  y \\
  z \\
  1
\end{pmatrix}
\]

with $x' = x''/z'$ and $y' = y''/z'$. Here $C$ is known at the camera matrix. In a more general case we use:

\[
C = 
\begin{pmatrix}
  f_1 & s & a_1 & 0 \\
  0 & f_2 & a_2 & 0 \\
  0 & 0 & 1 & 0
\end{pmatrix}
\]

where $f_1$, $f_2$ denote different focal lengths for the $x$ and $y$ directions (a prominent example are CCD-cameras with non-square pixels), $s$ the skew and $a_1$ and $a_2$ denote a shift of the camera's principal point. (From here on we assume that our cameras use square-pixels with a skew of zero)
The matrix $C$ holds all so-called \textit{internal} camera parameters. To fully describe the camera system one needs an additional 4x4 matrix which parameterizes the rotation and translation of the camera center. This transformation matrix holds the \textit{external} camera parameters, representing the position and orientation of the camera. The process of estimating these camera parameters is called camera calibration. This is usually performed by finding point correspondences between several images and then estimating the world coordinates of these points. Examples are the methods by Tsai [34] or Abraham and Hau [2].

In classical camera calibration, a test-target marked with easily detectable points (for example a checkerboard pattern with known tile size) to ease the process of finding point correspondences. We use this principle twice in the following sections. When no test target is available or it's use is not feasible, so called \textit{auto-calibration} can be used. Although in this case the estimated camera matrices usually have a linear scale ambiguity. (We will deal with this problem when using the voodoo tracker [32] in section 4.2).

Estimating the change of the external camera parameters over time, aka the camera movement is also called \textit{camera tracking} or \textit{match moving}. This methods are often used to insert artificial 3D objects into real film footage.

In case of real-world cameras one usually has to include non-linear effects like lens distortions, chromatic aberration or distortions caused by a bayer filter etc.

### 3.3 Theory: Optical Flow

#### 3.3.1 Error Measures

To quantize the error or the accuracy of a given optical flow field one has to define certain error measures. Jähne, Hausecker and Geißler ([15], Section 13.8) describe some of the most used measures. We want to emphasize on the \textit{direction error} and the \textit{endpoint error}.

The \textit{direction error} is defined as the angle between the reference flow vector $\vec{g}$ and the computed flow vector $\vec{f}$.

$$E_d = \arccos \frac{\vec{f} \cdot \vec{g}}{||\vec{f}|| ||\vec{g}||}$$

The \textit{endpoint error} is defined as the length of the difference vector between the reference and the computed flow.

$$E_e = ||\vec{g} - \vec{f}||$$

Alternatively the error relative to the local flow can be given:

$$E_{e,e} = \frac{||\vec{g} - \vec{f}||}{||\vec{g}||}$$

The author considers the endpoint error to be more significant as large directional differences occur rather seldom in most applications (with the exception of occlusions). The directional error can also give the wrong impression of large flow errors when the corresponding flow amplitudes are small.

The mean of the endpoint error and standard deviation is practical to give a single value estimate on the quality of an optical flow estimate. Bainbridge-Smith and Lane [4] pointed out
that the mean errors are susceptible to outliers. Therefore the median of the endpoint error is also regularly given, as it is more robust against those outliers. If median and mean differ (too much) the error can neither be normal nor uniformly distributed which indicates an bias in the estimate.

Furthermore Jähne et. al. [15] describe the root mean square error between the original image \( i \) and the image warped according to the computed flow \( \hat{i} \) as:

\[
E_{\text{rms}} = \sqrt{\left\langle (i - \hat{i})^2 \right\rangle}
\]

or in the normalized form

\[
E_{\text{rms}} = \sqrt{\left\langle \frac{(i - \hat{i})^2}{\|\nabla i\|^2} \right\rangle}
\]

Occasionally we also use the mean-of-absolute values, called **residual error** in the following sections:

\[
E_{\text{res}} = \left\langle |i - \hat{i}| \right\rangle
\]

In this thesis, for all averaging operations confidence maps are used to calculate the values only for image regions where the flow computation was successful or where other constrains are met (example: correct disparity estimates in section 4).

### 3.4 Experimental Setup

To provide a proof of concept the author deliberately resigned from creating a scene with effects, which are known to be difficult to reproduce, or which depend on too much parameters. This includes reflective or transparent surfaces, complicated occlusions or complex light setups with multiple interreflections. Even if modern render algorithms can simulate those and even more complicated properties (e.g. dielectric surface properties, complex **Bidirectional reflectance distribution functions (BRDF)** etc.) would taking those into consideration bring an enormous inflation of the parameter space. So a relative simple scene which does not differ very much from the synthetic or engineered laboratory scenes already in use by the optical flow community was created.

The scene consists of a block of unfinished wood which stands on a wooden disk rotating at a speed of 0.5 degrees per frame.

#### 3.4.1 Real-World Scene

Reflections and indirect illumination get minimized by surrounding the scene with black packaging foam which has very low diffuse and direct reflectance values. It also provides an image background which is static even if objects in the scene move.

Illumination was provided by 6 collinear arranged LEDs. Due to their small surface they are a much better approximation of point light sources than light bulbs etc. This makes for an easier simulation with virtual point or spot lights. Additionally the shadow edges produced by them are crisper than those of spatially extended light sources like fluorescent lamps.
The objects themselves are made out of unfinished wood which provides a distinctive texture for flow estimation with only minor reflections or specular highlights. To ease the later step of camera calibration surfaces markers in the form of pencil lines are present on the wooden disk.

The block and plate are driven by a stepper motor whose rotation is reduced by a high gear ratio. This way a very low angular position error of 0.02 degrees for the rotating plate is achieved. Additionally the plate was at rest in each frame, so no motion blur occurs in the images.

Image capturing was performed with the camera system described in Section 2.1

3.4.2 Synthetic Scene

Reconstructing the scene was done using classical and image-based modelling and rendering. Most of the work was done using Blender 1, an open-source 3d modelling and render software. One important point is the programs ability to directly export the scenes optical flow into the OpenEXR format 2, currently the only commonly used image format which supports negative float-precision values.

The reconstruction process can be broken down into four parts: Meshes, surfaces/materials, scene geometry and lighting.

Meshes

Object meshes are based on measurements done by hand, or in case of the rotating plate, by scanning it with a professional flatbed-scanner. The resulting meshes resemble the original objects down to an accuracy of ≈ 1mm; even less for the XY-Dimension of the rotating plate, due to the high resolution of the flatbed scanner. The block’s mesh has a resolution of ≈ 4000 triangles and the plate has ≈ 1300 triangles. Although those were created by triangulating n-point polygons, many of the triangles are coplanar and don’t add to the surface detail. The number of polygons could probably be reduced by a factor of 2-4 without any greater loss of reconstruction accuracy. But as explained further below, a fine mesh is needed to accurately interpolate the optical flow between vertices.

UV-Maps were created for the meshes to apply surface textures. This process is commonly known as UV-unwrapping. Here small texture distortions were introduced, especially at the rounded corners of the block.

Surfaces/Materials

The textures were created from images taken with the same camera system which was used to create the real-world scene. Hence, this guarantees that the textures would have the same dynamic range as the real materials. The final texture resolutions were at least two times higher than the size of the objects in the final renderings, so interpolation errors were minimal. Also during texture creation, resizing of the base images was kept to a minimum and usually conducted using nearest-neighbour methods. As the original images and the final renderings were grayscale images, all textures can be applied as grayscale values to the object’s diffuse color channel.

1Blender Foundation, http://www.blender.org/
The next part consisted of reconstructing the physical surface properties by using different reflection models (usually called shaders, or more specific pixel shaders in the computer graphics community).

Real opaque physical surfaces either reflect or absorb incoming light depending on its wavelength and and incidence direction. As real surfaces are usually not perfectly planar, but can show a very rough microstructure at decreasing dimensions, the usual law of reflection (where the incidence angle is the same as the reflection angle) does not hold. Instead the four-dimensional Bidirectional reflectance distribution function (BRDF) is used, which gives the amount of reflected light for every incidence and reflection angle. Although several methods for measuring the BRDF response up to a certain accuracy have been proposed, for example [37], [23] or [1], it still remains a tedious and time-consuming process. Additionally several simplification models are in wide use in computer graphics, as they can simulate the BRDF of most real-world materials with high physical accuracy. Some of them even account for reflections, subsurface scattering etc., usually in combination with raytracing methods to model the behaviour of light inside of objects or surfaces.

For the diffuse reflections of the wood a Oren-Nayar shader [22] with a low roughness value was used. This shader is known to simulate natural materials like sand or wood quite well, as it takes surfaces roughness into account. This way a more natural light distribution than for the simple lambertian model can be achieved.

For the specular reflections a basic blinn-phong-shader [8] was used. While this is not the most advanced specular shader, we think that errors introduced by this model are minimal as the amount of specular reflection was rather low, due to the general low reflectivity of wood.

Scene geometry

Finally, the objects were positioned inside a virtual environment by using the previously captured real-world scene as a reference. The initial camera calibration is done using the camera calibration method by Tsai, R. Y. [34].

From there the geometry was refined by aligning points drawn on the surface to reference points on the meshes. The calibration attempts were limited by blender's camera model, the accuracy of the meshes and the accuracy of the feature/reference points. The relative positions of the block and the plate to each other were fixed, which leaves us with 6 degrees of freedom (more specific the external calibration parameters of the camera).

In the final renderings the objects are aligned with an approximate reproduction error of 2-3 pixels or more at points close to the camera (e.g. the front of the plate). This is again caused by errors in the meshes themselves (the plate for example is no perfect disk). Also the actual displacements can seem to be larger due to errors in the texture mapping.

Lighting

The scene was illuminated with 6 spotlights which are decent approximations of the LEDs used in the original scene, at least regarding the lights geometric properties. As the spotlights are not spatially extended like the real light sources, we tried to simulate that property by defining a rather wide spotlight angle with a linear intensity falloff towards the edge of the light cone. Additionally light is reflected from a plane on the left of the camera's field-of-view. (This plane

was also present in the real-world scene as a piece of cardboard). The light is simulated by means of radiosity rendering.

The light color of the LED's was not part of the simulation, as in the final render step all color information was discarded to acquire grayscale images.

**Final considerations**

Reference data for the optical flow is generated in blender by calculating the screen coordinates for each vertex of the objects meshes. Values for intermediate pixels are bilinear interpolated from the three values of each triangle. This is equivalent to the polygon coloring method known as Gouraud shading. This method can lead to large errors, especially along the edges of extended polygons which are not parallel to the camera plane, where flow discontinuities can occur. But the so introduced error can be severely reduced by applying surface subdivision. This means that each polygon (in case of blender only 3-Gons and 4-Gons are supported) is broken down into several smaller ones by adding vertices at the middle of each edge. That way the number of points between which interpolations take place is increased. As the number of polygons increases with the subdivision levels, the computational cost can rise significantly as different render algorithms can have a worse than linear complexity.

Based on this final setup the realism of the sequences was reduced to evaluate which effects have an influence on the optical flow. First the specularity value of all materials was set to zero to eliminate specular highlights. Second, the wood's diffuse shader was changed from the oren-nayar to a more simple lambertian one. Finally, to simulate a very simple synthetic scene the global illumination, raytracing and shadows was deactivated. This last scene bears the most resemblance to the previously mentioned synthetic sequences.

As direct grayscale rendering is not possible in blender, all scenes were rendered with standard RGB colors of 8bit depth per color channel and then converted to grayscale. The renderer uses a standard color conversion scheme with $I = 0.11 \cdot B + 0.59 \cdot 0.3 \cdot R$ as the real color Intensity response curve of our system was not known at that point in time.

As final experiment consisted of sampling pixel noise from the camera used to capture the real sequences and adding it to the rendered scenes. For this case the arithmetic mean over 400 images was computed to get the fixed pattern noise (FPN). Usually this noise is subtracted from all real images taken so it does not present the real noise present in our images. So by taking several black-level images and subtracting the FPN (which in principal is the noise of the noise) one can get noise samples to add to the synthetic images. As a resulting noise values only reached an average of $0.11 \pm 0.19$ so its influence in this case was minimal.

### 3.4.3 Reconstruction Evaluation

Under optical inspection the real and rendered sequences are already very similar to each other (Figure 3.2).

But at the first look the more simple scenes(C,D,E) can seem to be closer to the real scene(A) than the realistic(B) one. There are no distinct specular highlights visible, but if the specularity is reduced to zero the plate has a uniform brightness, while with specularity the left side is brighter. This effect is actually more realistic as the light cone produced by a given real-world lamp is not necessarily homogeneous. Without the more simple shader model(D) edges with a tight curvature appear too dark as they reflect light into an too small angle. This effect can be
Figure 3.2: Rendered and real scenes
seen at the left front side of the wood block. This leads to the conclusion that the realism does indeed increase from scene E (which neglects all these effects) to scene B.

However, used models and rendering methods have some shortcomings. First, the assumption that the material properties are constant for the whole surface of an object is obviously not met. Specular and diffuse reflection vary based on the wood grain. This can be observed at the top of the block where a rather strong specular reflection occurs, even without any obvious change in the surface normal. Another example is the side of the disk which is composed of various strains of different woods.

Another one is the assumption that the ground truth for the synthetic scenes corresponds to the actual ground truth for the real scenes. But obviously this was only true for certain areas of the images and only within the tolerance of pure optical inspection, meaning 1-2 pixels. This most definitely caused by an error in the camera calibration method and errors in the mesh as many automatic methods reach reconstruction errors in the sub-pixel range. On the other hand, evaluation of the residuals of the warped images (see below) showed that the assumption was not too far from reality.

3.5 Results

![Image](image1.png)  ![Image](image2.png)

Figure 3.3: Difference between second and first frame of real sequence, Difference of second and first frame of realistic sequence

3.5.1 Algorithm Evaluation

We the optical flow in the sequences was computed using the algorithm by Zach et. al. [40] which performs well on the Middlebury dataset and is available as Matlab source code.4

With the exception of the regularization strength all algorithm parameters were kept constant. The regularization was varied to find a sufficient good estimate to perform the analysis.

Those were in detail $\theta = 0.25$, $\epsilon = 0.01$, pyramid scale factor = 0.5, 4 pyramid levels, edge weighted TV, 20 iterations per warp with 5 warps per level and active structure texture pre-processing.

4Implementation was provided by the Institute for Computer Graphics and Vision, Graz University of Technology under the GNU General Public License
The flow between frames two and three for each sequence and the corresponding endpoint errors were computed and are shown in Figure 3.5. Figure 3.4 shows the warping residuals as absolute difference between frame three and the warped frame two.

### 3.5.2 Error Evaluation

Although the mean endpoint error for $\lambda = 400$ is slightly lower than the one for $\lambda = 200$, the later one seems to be better, as the higher regularization strength produces small regions with unusual high errors. The high smoothing causes the flow to “leak” into the darker or shadowed regions which causes the mean endpoint error to decrease. The slightly lower median of the errors (Table 3.2) supports this assessment.

Figure 3.5 shows the endpoint errors computed for $\lambda = 200$. The images are normalized to show the same gray value for a given value of the endpoint error. The Errors are capped at a value of one pixel so that all errors bigger than that are pure white.
<table>
<thead>
<tr>
<th></th>
<th>( \lambda = 100 )</th>
<th>( \lambda = 200 )</th>
<th>( \lambda = 400 )</th>
<th>( \lambda = 700 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>real(A)</td>
<td>0.52 ± 0.83</td>
<td>0.48 ± 0.75</td>
<td>0.47 ± 0.72</td>
<td>0.51 ± 0.72</td>
</tr>
<tr>
<td>realistic(B)</td>
<td>0.43 ± 0.87</td>
<td>0.40 ± 0.83</td>
<td>0.39 ± 0.83</td>
<td>0.41 ± 0.93</td>
</tr>
<tr>
<td>realistic with noise</td>
<td>0.44 ± 0.87</td>
<td>0.40 ± 0.83</td>
<td>0.39 ± 0.83</td>
<td>0.40 ± 0.92</td>
</tr>
<tr>
<td>no specularity(C)</td>
<td>0.43 ± 0.84</td>
<td>0.38 ± 0.76</td>
<td>0.37 ± 0.75</td>
<td>0.39 ± 0.89</td>
</tr>
<tr>
<td>simple shader(D)</td>
<td>0.41 ± 0.82</td>
<td>0.37 ± 0.75</td>
<td>0.35 ± 0.72</td>
<td>0.36 ± 0.84</td>
</tr>
<tr>
<td>minimum realism(E)</td>
<td>0.36 ± 0.76</td>
<td>\textbf{0.29 ± 0.63}</td>
<td>0.31 ± 0.76</td>
<td>0.37 ± 1.11</td>
</tr>
</tbody>
</table>

Table 3.1: Average endpoint errors and standard deviations

<table>
<thead>
<tr>
<th></th>
<th>( \lambda = 100 )</th>
<th>( \lambda = 200 )</th>
<th>( \lambda = 400 )</th>
<th>( \lambda = 700 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>real(A)</td>
<td>0.24</td>
<td>\textbf{0.24}</td>
<td>0.26</td>
<td>0.31</td>
</tr>
<tr>
<td>realistic(B)</td>
<td>0.11</td>
<td>0.11</td>
<td>0.12</td>
<td>0.14</td>
</tr>
<tr>
<td>realistic with noise</td>
<td>0.11</td>
<td>\textbf{0.11}</td>
<td>0.12</td>
<td>0.14</td>
</tr>
<tr>
<td>no specularity(C)</td>
<td>0.11</td>
<td>\textbf{0.10}</td>
<td>0.11</td>
<td>0.13</td>
</tr>
<tr>
<td>simple shader(D)</td>
<td>0.10</td>
<td>\textbf{0.10}</td>
<td>0.11</td>
<td>0.13</td>
</tr>
<tr>
<td>minimum realism(E)</td>
<td>0.08</td>
<td>\textbf{0.07}</td>
<td>0.09</td>
<td>0.11</td>
</tr>
</tbody>
</table>

Table 3.2: Median of endpoint errors

The mean endpoint errors as shown in table 3.1 were measured only in the regions of interest (flow evaluation result on the black background did not contribute to the mean) The error of the real scene is the largest which can either be attributed to errors in the alignment or to deviations caused by real-world effects. One of those would be the fact that the back of the plate is poorly lit which makes flow evaluation difficult.

The smallest endpoint error was reached for the most simple rendered scene with a mean of 0.29 ± 0.63 pixels (last images in Figures 3.2 and 3.5).

In shadowed regions the endpoint error is quite large, as the image contrast is rather low. This effect is observable in the back of the turntable and it's left front. Naturally, switching off shadows during rendering improves the optical flow evaluation in these regions and therefore reduced the endpoint error.

One can see that the specularity plays only a minor role in this scenario as switching it off leads only to a marginal decrease of the mean endpoint error. On the other hand local effects can be observed at the corners of the wooden block, where a small region with huge errors (> 10 pixels) shrinks in size significantly once the specularities are reduced. For scenes or objects where distinct highlights are visible the influence should be stronger.

The endpoint error drops significantly once the reflectance model is changed to a pure lambertian one. This should mostly be caused by the increase in image contrast, as the former used oren-nayar shader produces a more homogeneous brightness distribution on a curved surface.

While the calculated endpoint errors for the real and rendered scenes are quite similar with respect to their mean and standard deviation, this is not the case if we consider the spatial structure of the error. The strongest deviations occur at the front edge of the wooden block and the side of the turntable. As a similarity measure we calculated the normalized cross correlation between the endpoint errors of the real and the realistically rendered scene (calculated with a window size of 15 pixels, background was masked out). Across all surfaces there are areas with low (negative) correlations which shows that there are big dissimilarities in the local structure of the errors. On the contrary, the cross correlation between the realistic and simple scenes shows quite a few similarities, especially on the side of the plate and the right front of the block.
Adding noise to the rendered scenes resulted in no significant change of the computed flows or errors (mean endpoint error $\Delta < 0.01$ pixels). The spatial structure remained also the same so we abstained from including the corresponding images.

Figure 3.7 shows scatter plots of the individual errors in x and y direction.

In some regions the flow evaluation is very bad as several outliers with an error of up to 15 pixels appear. These regions can be located in the lower left front of the plate where the UV-texture is slightly misaligned and a black stripe appears. Except for this error the spatial distribution of the central cluster is quite similar for all sequences. For the realistic rendering (B) the scattering is a bit more dense in the upper left quadrant than for the simple rendering (E), and it therefore resembles the real scene more.

In the opinion of the author these discrepancies are caused by a suboptimal reproduction of the surface properties and textures. Neither lens distortions nor noise are large enough factors to have any significant influence. Figure 3.3 shows the (normalized) difference between frames two and three and therefore corresponds to the temporal derivative of the sequences. One can use this as a similarity measure between consecutive frames. The real sequence shows more and finer details than the rendered one, possibly because of the lack of texture noise in the synthetic scene.

### 3.5.3 Conclusions and Future Work

The following methods can generally improve the realism of synthetic sequences and bring them closer to real sequences. If this approach can also bring them closer together in terms of the optical flow remains an open question, as one needs to evaluate the effects of each one in detail.

First, the camera calibration technique was limited by the precision of the available 3D-meshes. Those meshes can be produced with very high precision using various types of 3D-scanners, or by using objects which were created from mesh data; e.g. by means of stereo lithography.

This is an important aspect as errors introduced due to incorrect meshes can have significant influences at object discontinuities (visible at the edge of the rotating plate, Figure 3.5), a region where traditional optical flow algorithms already have problems.

For an accurate reconstruction of surfaces, materials and textures the bidirectional reflectance distribution function needs to be known. In addition to classical gonioreflectometers, several advanced techniques have emerged lately, so there is a lot of space for improvement.

Once the influence of the BRDF is better understood, it may be sufficient to use generic or parametric BRDFs for certain surface types (wood, metal, etc.) to create sufficient scenes. That way one does not need to know the BRDF for a certain object (and hence does not need to measure it) but can use a already known one to speed up the production process without loosing accuracy. An Alternative would be take the design of modern rendering engines more into consideration and create multiple textures for color, diffuse and specularity values or normal maps. An example would be the method of Ma et. al. [19]. Here it is unclear if the drop in physical accuracy in favor of easier to achieve photorealism would be of significance.

Physically correct, or at least photorealistic light can be simulated using global illumination models. Examples are different tracing methods like metropolis light transport as well as image based lightning [36, 25]. As the quality of reflections, highlights and transparency effects are highly dependent on these methods we deem this part important to bridge the gap between the easy synthetic sequences and the often suboptimal real ones. Additionally the render pipelines in which the above methods are implemented would allow the easy creation of corresponding
confidence maps. It will also allow to use more sophisticated camera or image sensor models, an important part to simulate effects like lens distortions or noise.

The scene was limited to rigid motion with is a rather strict constraint. On the other hand this method could easily be modified to simulate non-rigid motions, as long as they can be represented by mesh deformations. Motion of fluids, gases etc. can be simulated but the cost for modelling and computing them can be significant. Another problem is that fluid simulations (or more correctly the visualization of fluids) use volumetric rendering or particles and not polygons. In this case the apparent motion is ambiguous as it is not clear what the flow of several overlapping fluid components should be. The disadvantage is that computational cost may rise quickly as the complexity of the scenes increases. Even on dedicated render farms the render time per frame for current rendered movies can approach hours or even days, so there is always a trade-off between effort and achieved accuracy.

The open-source nature of blender makes it suited for special adaptations regarding reference data creation. On point that comes to mind would be a special build which does not perform the bilinear interpolation of the optical flow as it is currently the case. An alternative would be for example the method used for the synthetic scenes in the middlebury dataset by Baker et al. ([5], Section 3.2.) where clustering is used the select to most prominent flow vector out of several possible solutions (In case of overlapping fragments).

Final remarks

The above results indicate that synthetic sequences can in principle be used to create ground truth sequences for optical flow estimation. Most discrepancies between the real and the rendered sequences were due to differences in the geometry, but some differences in the local structure of the computed flows can not be explained by this issues. The warping residuals (figure 3.4) and cross correlations indicate that high resolution properties of the objects surfaces where not reproduced with optimal accuracy.

Future work should therefore focus on modeling the effects which influence the texture or the brightness constancy assumed by many optical flow algorithms. Vaudrey et al. assume that BCC violations occur quite often and results from Chaper 4 and industrial partners support that claim. If further evidence can be acheived, the above method can be used to model local BCC violations in order to classify how optical flow algorithms react on them.

The evaluation showed furthermore that a simple global evaluation by means of certain error measures (mean endpoint error or direction error) are not sufficient to decide if a given result is valid or not. Performance measures which evaluate local structure or mark uncertainties could be of advantage.
Figure 3.5: Normalized endpoint errors
(0.0(black) - 1.0(white))
Figure 3.6: Normalized cross correlation between the endpoint errors of two sequences. Positive values are gray/white, negative values are blue.

Figure 3.7: Differences between ground truth and calculated flow for x and y directions
4 Stereo Augmented Flow Estimation

This section tries to combine stereoscopic image data with modern camera tracking methods to evaluate the optical flow of images taken from a moving car in typical traffic situations. Traffic scenes can be especially challenging for optical flow methods due to the following facts:

- the brightness constancy constraint is often violated as lighting conditions can change quickly because of clouds, trees, tunnels etc.
- Fast movement speeds cause very high flows in the range of several dozens of pixels per frame, especially in regions with low texture like the street’s surface
- Reflections can occur in windshields, windows and other glass surfaces
- Occlusions due to moving objects are quite common

Many optical flow algorithms claim to be suited for a wide field of sequence types. The question is how one can be sure that these claims are entailed when their evaluation is limited to sequences with good-natured effects. And furthermore, if the performance on real application specific sequences is demonstrated, how can one be sure that the results are accurate when no ground truth is provided?

This thesis describes a method to produce reference data with which those claims could either be confirmed or be rejected, as it is not constraint by the usual shortcomings of optical flow algorithms. Depending on the desired accuracy it can provide reference data, sanity checks for other optical flow methods or help to decide if flow computation is actually feasible in a given sequence. Furthermore the resulting flows can be used to create new ground truth sequences by means of image warping, which would not be subject to possible physical inconsistencies.

The basic idea is the following. By calculating the disparity of an stereo image pair and combining it with camera calibration data one can get an accurate depth map of a scene as shown in Section 4.1.2.

Each pixel of an image reprojects onto a line in real space. With the known distance from the camera plane for each of this lines, one can reconstruct a 3D point cloud representing the original scene which was projected onto the camera. This is a well understood method to create 3D meshes or reconstructions of real world objects.

With a sufficient number of images (more specific, different views) the transformation matrices of a moving monoscopic camera can be calculated. This is a common method used in the entertainment industry, often called match moving or, in this work, camera tracking.

The calculated camera transformation matrix (in this case a rotation and translation matrix) can be applied to the 3D point cloud computed in the previous step. The resulting points can then be projected onto the camera plane to get new pixel positions for each point. For rigid scenes this results in a dense optical flow field.
4.1 Theory: Stereo

4.1.1 Stereo Vision and Epipolar Geometry

Reconstructing the 3D information of a scene from a set of two images is a well understood problem in computer vision. A thorough description on the mathematics involved can for example be found in [10]. We just want to briefly show the basics. For a given set of two cameras, a point in 3D-space will be projected onto two different points on the camera planes according to their respective camera matrices. For one image the depth information (the distance of the 3D-space point from the camera plane) is lost as the point could lie anywhere on the line emerging from the cameras focal point and intersecting the camera plane at the corresponding pixel. By finding the projection of the point in the second image we are provided with a second viewing ray with the intersection of those two lines representing the original point. The point’s 3D-coordinates can then be reconstructed by means of triangulation.

The problem of finding the projection of a point in the second image is called the correspondence problem and in it’s most generic form involves a two-dimensional search in the second image. This search space can be reduced to one dimension by introducing the epipolar constraint.

The viewing ray from the first image will be projected onto a line on the second image plane. This is true for all image points, as long as the focal points of both camera are distinct. To solve the point correspondence one has to search only along this epipolar line. The orientations of the epipolar lines depend only on the camera matrices and their relative position to each other. Furthermore, for cameras whose image planes coincide, all epipolar lines are parallel which makes the search for point correspondences especially easy.

If the relative camera positions are known (if the cameras are calibrated), an image transformation can be applied to make the images fulfill this constraints. This process is known as image rectification. More specific, the transformation can be chosen in such a way, that the lines become parallel to the image border. That way the search is limited to a scanline.

4.1.2 Depth Estimation from Disparity

For two calibrated cameras, the world coordinates of each image point can be reconstructed up to a similarity transformation. ([10], Section 10.2). If the relative distance of the cameras (called Stereo Baseline) is known, this ambiguity can be dropped completely and the world coordinates for each pixel can be calculated by simple geometry.

For this example we consider the world and camera coordinate systems to be identical.

Let \( X = (x, y, z) \) be a point in space projected onto the pixel \((x_1, y_1)\) in camera 1 and \((x_2, y_2)\) in camera 2. The camera center of camera 1 is located in the coordinate system origin. Furthermore let \( f \) be the focal length, \( b \) the stereo baseline and \( p_x \) and \( p_y \) the cameras’ pixel sizes (We assume that both cameras have the same pixel size)

For rectified cameras, which fulfill the epipolar constraints, \( y_1 \) and \( y_2 \) are equal. We will limit ourself to cameras which are not tilted, meaning both camera axes are parallel. The disparity \( d \) is the in-image distance \( x_2 - x_1 \) of a given left-right correspondence in pixel. For camera’s which are tilted, the disparity can become negative, while for non-tilted cameras a disparity of
Figure 4.1: stereo camera setup

Figure 4.2: stereo camera setup in XZ-Projection
zero corresponds to points at infinity.  

Considering the viewing rays from both cameras we get the equality.

\[
X = \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} + r_1 \begin{pmatrix} p_x x_1 \\ p_y y_1 \\ f \end{pmatrix} = \begin{pmatrix} b \\ 0 \\ 0 \end{pmatrix} + r_2 \begin{pmatrix} p_x x_2 \\ p_y y_2 \\ f \end{pmatrix}
\]

center of cam 1  

center of cam 2

with \( r_1, r_2 \in \mathbb{R} \). Solving the equations yields \( r_2 = r_1 \) and \( z = r_1 f \).

With \( p_x = p_y = p \) this results in:

\[
z = \frac{fb}{pn}
\]

4.2 Detailed Method Description

This method differs from classical optical flow algorithms in several points:

First, it does not depend on any brightness constancy constraint: As already discussed in Chapter 3, it is the opinion of the author that this constraint is violated rather often in various applications of optical flow. An example is the degradation of the flow in case of specularities or shadows (See Section 3.5.2). The method here is mostly independent of local or global illumination changes, and generally performs well, even in regions with low image contrast.

Second the demand for the smoothness of the optical flow is replaced by a smoothness constraint for the disparity or image depth. This is not a large difference, as the smoothness terms are in both cases used to deal with the same type of circumstances. These are e.g. textureless regions where both the flow and disparity estimates are ambiguous, and discontinuities e.g. in case of occlusions or object boundaries.

Due to the epipolar constraint (Section 4.1.1) the search space is only one-dimensional. This results in reduced computational cost. This advantage is strengthened more by the fact that the cost is also independent of the flow amplitude.

Algorithm Description

The first step in creating the reference data is to calibrate both cameras of the stereo system. Especially their relative orientation is of importance as this information allows it to rectify the resulting stereo pairs.

This is achieved by using the calibration method described by Abraham and Hau [2]. It uses a physical test-field object covered in reflective points to give a robust and accurate estimation of the camera parameters by means of bundle adjustment. Additionally our calibration application allows to estimate and apply different camera distortion models which parametrize non-linearities in the projection function when the pinhole model is not sufficient.

1 if not described otherwise, only left to right disparities are used in this work.

2 algorithm implementation provided by Robert Bosch GmbH
Then the disparity is calculated using a semi-global matching (SGM) algorithm combined with a rank filter. The SGM method was originally proposed by Hirschmüller [12] with further motivation of the rank filter given in [13] where this method performed well in comparison to other algorithms. Our implementation has currently no sub-pixel accuracy which makes the depth estimates rather coarse due to the inverse dependency on the disparity.

In this step we also calculate a confidence map which marks pixels where the disparity estimate is ambiguous or impossible due to non overlapping image regions (e.g. a n pixel wide border around the image caused by an accumulation window). This map is later used to restrict the optical flow calculation and error averaging operations to sensible areas.

For tracking the camera (or alternative calculating the external calibration parameters) the voodoo camera tracker [32] is used, which is based on a reference tracking method by Thorsten Thormählen [31]. In detail, the monoscopic tracking is only applied on the left images. This tracking consists of five steps:

- Feature Detection: Finding image points which can be tracked between consecutive sequence frames.
- Correspondence Analysis: Identify a feature point from one frame in the successor frame
- Outlier Detection: Drop feature point pairs whose motion is inconsistent with the already estimated one
- Estimation of camera parameters
- Bundle Adjustment: Optimize the found camera parameters by reducing the reprojection error

Most alternative methods like [3] or [20] use the optical flow field for motion estimation. The problem there is the rather high computational cost and that some additional camera motion constraint is needed, as the motion between each frame is at first independent of the previous translations. Instead of this the present approach uses feature tracking, which means a sparse set of points is sufficient and it is usually more exact than optical flow. Furthermore the bundle adjustment step distributes the reprojection error evenly over all observations which leads to a more smooth (and therefore usually more realistic) motion estimate.

As a result the voodoo tracker calculates the camera translation up to an arbitrary scaling factor (metric camera). The 3D-Feature points which are generated by the tracker's internal feature detector can be used to determine this scaling factor. For each feature point the depth is calculated using the disparity at that point and then related to the point's z-coordinate. A simple linear least-squares fit then gives the scaling factor for the translation matrix.

This is an important step as the whole process stand's and falls with the accuracy of the scale estimation. The two parameter (set's) of the voodoo tracker which have the most influence are the feature detector and the correspondence analysis step.

The best calibration results can be achieved by the Kanade-Lucas-Tomasi feature tracker (KLT) [18, 33] which combines feature detection and correspondence analysis into one algorithm. This tracking method works best for motions smaller than 1-2 pixel, e.g. in this case points which are further away in the image background. In traffic sequences, points in the foreground, for example on the street, move to fast to get detected. So only feature points with low disparity (between 1 and 40) are usable, which corresponds to higher errors in the depth estimates and finally also in the scaling factor.

Therefore an alternative feature detector based on the work by Foerstner et. al. [9], which is an option in voodoo, was used. This detector finds feature points which are more evenly spread around the images (See Figure 4.3 for an example).

37
The correspondence analysis is carried out by calculating the cross correlation in a limited search window (usually 50 by 50 pixel). Irregular feature points are detected with random sample consensus (RANSAC). This includes points whose apparent motion is inconsistent with previous motions or the initial estimate for the camera translation.

Figure 4.3 shows an exemplary set of feature points and the corresponding feature tracks. Figure 4.9 shows the result of such and scale factor estimate for 2 different sequences.

The final translation matrix can then be multiplied with the camera matrix, (or to the point cloud for that matter), to get the new point positions in the second frame. Finally it is possible to reproject each 3D point onto the camera plane and calculate the difference between it’s new and old pixel positions. Under the assumption of a rigid scene this yields the optical flow.

### 4.3 Theoretical Error Evaluation

Undoubtedly the optical flow calculated by the described method will have errors. This chapter will investigate if those errors are theoretically small enough to consider the method as an valid alternative for classical optical flow methods. Optical flow endpoint errors of several pixel would severely limit the conclusiveness of the method. Therefore some theoretical lower error bounds based on the camera configuration described in Sections 2.1 and 4.4 are given. For that matter we assume certain gaussian distributed error ranges in the input data and see how this influences the expected flow.

If we consider the errors of the focal length \( f \), stereo baseline \( b \) and sensor size \( s \) / pixel size \( p \) to be negligible, we have 7 degrees of freedom left, namely the disparity, the camera translation and the three angles of the camera rotation.
Generic Error Estimation

Now consider a point \( \mathbf{X} = (X_1, X_2, X_3) \) lying on the ray which intersects the image plane of a camera in the pixel at \( \mathbf{y} = \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \). The distance \( d \) from the image plane is given by the disparity \( n \) (See Section 4.1.2). The point therefore has the coordinates

\[
\mathbf{X} = \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \cdot (d(n) + \Delta d(n))
\]

with \( \Delta d(n) = \frac{f_\mu}{p n^2} \cdot \Delta n \) and \( \Delta n \) being the disparity error.

After transformation with the (erroneous) transformation matrix \( T \) it will be projected by means of the camera matrix \( C \) (with internal camera parameters, as defined in Section 3.2) onto the pixel at position

\[
\mathbf{\hat{y}} = C \cdot (T + \Delta T)(\mathbf{\hat{X}} + \Delta \mathbf{\hat{X}})
\]

Here \( T \) is composed of a translations and a rotation, but only for small rotation angles (where \( \sin \) and \( \cos \) can be considered linear) can the error \( \Delta T \) be handled as an additive error.

The final endpoint error is then

\[
\|\mathbf{\hat{y}} - \mathbf{\hat{y}}\| = \|C((T + \Delta T)(\mathbf{\hat{X}} + \Delta \mathbf{\hat{X}}) - T \cdot \mathbf{\hat{X}})\|
\] (4.1)

Due to the rotations and the inverse depth dependency on the disparity, the error is not linear and calculating it's propagation can get quite complicated.

On the other hand, if we separate the various error sources, their effects can be interpreted more easily.

Errors caused by incorrect camera rotation

First we will discuss the effects which are caused by the camera tracking. We will split this error up in an translational and a rotational part.

The optical flow error due to an erroneously rotated camera can be directly calculated from the view geometry. For a rectilinear (undistorted) camera with the sensor size \( s \) the field of view is \( \alpha = \tan^{-1} \frac{s}{2f} \). With the focal length \( f = 25mm \), the pixel size \( p = 8.0\mu m \) and the resolution of 1312x1082 we get a field of view of 23.7° x 19.6° or a per-pixel-field-of-view of 0.018°. Any pitch or yaw rotation of the camera with this value will introduce an optical flow of 1 pixel. For roll rotations (around the optical axis) the optical flow will be a rotation field with the vector length depending on the actual pixel but not on the focal length.

A given pixel \( \begin{pmatrix} x \\ y \end{pmatrix} \) transforms with the rotational matrix \( \mathbf{R}(\phi) \) to result in the optical flow

\[
\|(1 - \mathbf{R}) \begin{pmatrix} x \\ y \end{pmatrix} \|
\]

For a pixel at the right edge of the sensor \( \begin{pmatrix} \frac{1312}{2} \\ 0 \end{pmatrix} \) in this case, a rotation of 0.088 degrees will result in an optical flow of 1 pixel. This flow will decrease in a linear until reaching zero at the image plane center.
Section 4.2 in [31] (on which the voodoo tracker is based) shows experimental rotation errors in the range of $1.0 \cdot 10^{-2} - 5.0 \cdot 10^{-2}$ degrees for all rotation axes. If we consider the errors to be transferable to our camera setup and movements, then the results should be well below the 1 pixel error threshold.

Additionally, rotational errors should be easy to detect as they will introduce a systematic error or bias, which is the same for all pixels (in case of yaw or pitch rotations). By comparing the resulting optical flow with other methods, e.g. regular flow algorithms, this error can easily be detected as long as the comparing method is known to be bias free. Noteworthy is also that these errors are depth-independent and therefore also independent of the disparity error.

**Errors caused by incorrect camera translation and incorrect disparity**

For the translational part we can separate into motions along or perpendicular to the optical axis.

Given a translation-error in x or y direction of $\Delta z$, the error of the optical flow $\Delta o$ depends only on the distance to the camera:

\[
\Delta o = \frac{\Delta z f}{pd} = \frac{\Delta z(n + \Delta n)}{b}
\]

(The depth $d$ is defined as in Section 4.1.2.) This can be considered as a form of parallax error.

The errors caused by incorrect z-translation and disparity estimate can be handled simultaneously as they both represent displacements along the camera's optical axis.

A drawback is currently that the disparity estimates are only pixel precise, which leads to a quantized depth map where the distance between the different 'levels' increases with the distance to the camera. A disparity error of $\Delta n = 1$ due to this quantization is a good starting assumption. Although Section 4.4 shows that locally the error can be several pixel higher.

The uncertainty in the z-translation is initially based on the error made by the camera tracker and then modified by the disparity scale estimate described in Section 4.2. The results in Section 4.4.2 suggest a scale error of at least 10%.

Furthermore this error is also dependent on which pixel is considered. The projection of a point near the cameras optical axis will not change much when it is displaced along this axis. For points near the edge of the viewing frustum the projection will change significantly when they are displaced along the z-axis.

The resulting endpoint errors can be seen in figures 4.4 and 4.5. The values were calculated by means of equation 4.1 by assuming gaussian-distributed noises and sampling 20000 times for each error value. The base z-translation was 0.1m in all plots. Figure 4.4 shows the endpoint errors for the pixel (200,200). The disparity errors ranging from zero to ten pixels may seem too large, but Section 4.4 will show that this are no unusual values in regions with little texture.

For the given error range the z-translation has a higher influence than the disparity, especially for points close to the camera. But one needs to consider that the z-translation error is a global one valid for the whole image, while disparity errors are distinct for every pixel and may therefore cause local variations.

Figure 4.5 shows how the errors do depend on the pixel position. As the error does only depend on the distance to the camera center, it can be shown for four different base disparities in only a single image. Therefore the image plane with a size of 800x800 pixel is split into four quadrants,
each showing a different base disparity value. The lower left quadrant shows disparities of $10 \pm 1$, the upper left $25 \pm 1$, the lower right $50 \pm 1$ and the upper right $75 \pm 1$. The $z$-translation is $0.1$m per frame which is a good estimate for the translation of the car as described in section 4.4.2. The errors are fixed with $\Delta n = 1$ and $\Delta z = 0.02$.

![Graphs showing endpoint errors for different base disparities.](image)

(a) base disparity = 20

(b) base disparity = 75

**Figure 4.4: Endpoint Errors at pixel (200,200)**

$z$-translation = 0.1m, $z$-error = 0.02, disparity error = 1

All plots indicate that 3D points which are close to the camera or which are near the edge of the viewing frustum will have a higher systematic error when their disparity estimate or the assumed $z$-translation of the camera is incorrect. Generally far away points will have lower endpoint errors than points close to the camera, while the influence of the disparity error will rise with the distance.

**Other error sources**

A different kind of error source are 'holes' in the disparity map. In case of ambiguities (which can occur due to repetitive textures or occlusions) the disparity estimate may fail. Our algorithm implementation is at least capable of flagging these regions. Some optical flow algorithms will
Figure 4.5: Endpoint Errors for disparities 10, 25, 50 and 75 depending on the pixel (x, y) 
(z-translation of 0.1 ± 0.02m and Δd = 1)

also do that, but most of them may introduce discontinuities in the optical flow. The actual behavior depends on the selected smoothing / disparity change penalties. This confidence maps allow us to mark or find regions where optical flow evaluation may be impossible are at least unreliable.

Difficult to evaluate are regions where the scene rigidity constraint is violated (e.g. moving cars etc.). Even the detection of those regions is not trivial as long as no optical flow algorithms are used for comparison or sanity-checks. Possible methods to handle non-rigid scenes are discussed in section 4.5.

Section 6.4 of [31] shows that pixel noise has only a minor influence on the estimated translation matrices. Problems begin to arise at relatively low signal to noise ratios of 28db.

**Conclusion**

Given a sufficiently accurate camera translation the described method should in theory be able to produce flow estimates in the order of one pixel. Rotation or translation errors will either result in an global bias or local variations with a well known structure. The individual per pixel error sources though are harder to predict, so additional investigation in the following sections is needed.
4.4 Experiments

4.4.1 Synthetic Test Scene

First the method was tested on a short synthetic image sequence to show its feasibility. The sequence consists of a forward motion into an textured tunnel as shown in figure 4.6. The renderings were again done in Blender with basically the same method as in Chapter 3, although the rendering is more similar to a simple scanline rendering as it uses uniform ambient lighting and no physical effects like reflections or specularities. Two versions are shown, one with a high and one with a low resolution texture. Both are procedural textures, so high resolution actually means higher frequencies present in the texture, not a pixelwise higher image resolution. Zooming in on the texture will therefore not reveal any artifacts, but distant points may still show aliasing effects.

The lower resolution texture will prove problematic both for the disparity estimation and for the feature detection step of the camera tracking.

The computed disparities range from 10 to 100 pixels. The confidence maps in figure 4.7 show the areas where the disparity estimate was ambiguous. The threshold for the uniqueness of the disparity is set to a rather high value, as this value proved reliable for the car scenes used in the next section. Flagged regions are of course the left side where the images don’t overlap and the sides of the tunnel. In the case of the high resolution texture, problems can arise when the sampling theorem is violated as structures with high frequencies will lead to aliasing. In this case feature tracking and disparity estimate will fail as the texture information is not preserved between images (for both temporal and spatial motion).

For the high resolution texture the disparity map covers 96.1% of the image (ignoring the non overlapping region on the left, and a 13 pixel wide border due to the accumulation window). With the low resolution texture the coverage drops to 79.7%. The flagging threshold is of course a tunable parameter, but the settings which caused this map coverage proved to be reliable.

This means that in some cases reliable flow estimation is only possible in \( \approx 80\% \) of the image area. This sound quite suboptimal, but classical optical flow algorithms seldom perform this kind of evaluation so the size of their confidence regions may be even smaller (although this is unlikely).

For most regions the differences in the disparities are not higher than one or two pixels. But there are some areas where the difference may be significantly higher (in the range of 10 pixels). This indicates that the assumption of an general disparity error of approximately one holds, but can be violated locally. This can be especially the case if a less strict confidence check, which could let some ambiguities pass is used.

As a first test the true known camera transformation is used instead of the tracked one to show how the quantized disparity (and therefore the quantized depth) influences the accuracy of the flow estimates. The resulting flow field was of high accuracy and showed no obvious errors. The final endpoint error (compared against the reference flow given by Blender) is very low, which is of no surprise as the theoretical evaluation showed that the influence of the disparity is minimal at best.

The endpoint errors are shown in figure 4.8; the values are mapped linearly, with black meaning zero and white representing an endpoint error of 0.5 and above for the low resolution image and 0.1 and above for the high resolution image. The mean values of the endpoint error are

43
0.03 ± 0.04 for the high resolution texture and 0.067 ± 0.37 for the low resolution texture, which is comparable to a good optical flow algorithm. Only regions where the disparity calculation was not ambiguous were considered for the mean calculation. A small rectangular region at the end of the tunnel where flow computed by blender is zero was masked out due to an incorrect cutoff value at the meshes vertices.

The second test handles the camera tracking step:

Figure 4.9 shows the scaling between the 3D feature points calculated by the voodoo tracker and the actual depth values computed via the disparity. To get this graph we interpolated the disparity map for each feature point (whose xy-coordinates are computed with subpixel accuracy). The different slopes of the fit curves are not due to errors or fundamental differences between the sequences but caused by the linear ambiguity inherent in the camera tracking process. The relative asymptotic standard error of the slopes is 2% for the low resolution and 2.6% for the high resolution. The true virtual camera had an translation speed of 0.1m. The estimated camera translations where 0.099 ± 0.0026 for the high resolution texture sequence and 0.095 ± 0.002 for the low resolution one (errors are only based on the fit error, as uncertainties for the tracked translations are unavailable).

Linear fitting is usually not very robust regarding outliers, of which are some visible in the plot. To evaluate the goodness of the fit one can check the variance of the residuals (\(\chi^2_{\text{reduced}}\)) which is 1.34 for the low resolution and 0.40 for the high resolution texture. This indicates a rather
good fit, but the error values may still be underestimated.

All evaluations for this sequence have to be taken with a grain of salt as one of the original premises is that synthetic sequences may yield different results than real-world ones. Although the purpose of the sequence was only to test if there are any fundamental problems in the algorithm. Especially the camera tracking needed some validation as it relies on not fully documented third-party code and applications. The results of the scale factor and camera translation estimation show that the voodoo tracker is in theory capable of providing the necessary accuracy.
Figure 4.8: *Synthetic*: endpoint errors

Figure 4.9: *Synthetic*: Scaling between the 3D feature points calculated by the voodoo tracker and the actual depth values computed via the disparity
4.4.2 Car Scenes

For the real-world sequences the camera system described in section 2.1 was used. Again the cameras have quadratic sensor elements with a width of 8 micron and are equipped with lenses of 25 mm focal length. Both are mounted with parallel view axes and a relative distance (stereo baseline) of 30 cm. The cameras are not tilted-in, as Woods et. al. [39] have shown that tilted-in cameras lead to a curvature in the depth map and so called keystone distortions which need to be handled during the image rectification. (This would introduce additional interpolation errors). With this values a disparity of one corresponds to a base distance of \( \approx 937.5m \).

The camera calibration by Abraham and Hau [2], which uses a test target is very exact, as the calculated relative position error between both of the cameras is below 0.1 mm for the x and y axes and around 0.4 mm for the z-axis (along the optical axis). The relative rotational errors are 0.055°, 0.067° and 0.003° for rotations around the x, y and z axes respectively. These errors do not influence the flow as described in the previous section as this are fixed estimates which only influence the image rectification process. Furthermore the root-mean-square of the reprojection error is 0.23 pixels. This is not the optimum for the used algorithm (which is known to cause reprojection errors of below 0.1 pixel) but can be attributed to the relatively high focal length of the cameras in combination with minimal deformations of the test target. Lens distortions were also modeled and evaluated in this step but turned out to be small enough to be ignored. These values influence the quality of the stereo rectification, but as all flow calculations are performed on the rectified images they have only minor influence on the differences between the flow calculation methods.

Two image sequences, one titled Crossing and one called Traffic are available. \(^3\) In those the car was traveling at approximately 40 km/h, respectively 80 km/h. The optical flow in the images roughly ranges from values around 1 pixel/frame to 20 pixel/frame at the street level. Both sequences were taken with the same camera settings and geometry.

**Sequence: Crossing**

In this scene the car is approaching a road intersection (Figure 4.10).

Figure 4.11 shows the confidence map for this scene. Occlusions caused by the traffic lights and street signs, or the textures regions in the sky are the most distinctive areas. The region in the lower left is caused by missing overlap of the image pair. Similar to optical flow algorithms it is possible to increase the smoothing parameters to improve the estimate in these regions but this would also blur disparity discontinuities. The map covers 80.0% of the image. For the following mean calculations we masked out the complete sky reducing the coverage by another 1-2%.

For this sequence the results of the camera tracking are more inaccurate than for the synthetic sequence. Figure 4.13 shows the scale estimate for the process. The number of outliers is significantly higher which leads to problems as the used linear least-squares fit is not very robust regarding outliers. For this set of feature points the curve slope is 97.5 ± 1 (asymptotic standard error), but the removal of one or two points can change this result by a value of \( \approx 3 \). The rather high variance of residuals \( \chi^2_{red} = 832.9 \) also indicates a fit of low quality.

\(^3\)Thanks to Paul-Sebastian Lauer and Robert-Bosch-GmbH for providing us with expertise and vehicle platform to install the camera system into.
Figure 4.11: Crossing disparity map

Figure 4.12: Crossing confidence map (black areas mark possibly unreliable disparity estimate)

Figure 4.13: Crossing: tracking scale estimate

Figure 4.14: Crossing: flow calculated with Zach tv11 method as HSV overlay (small flows are brighter, large flows have full color saturation)
This results in an translation \( \vec{t} = \begin{pmatrix} -0.9 \pm 0.01 \\ -1.75 \pm 0.02 \\ 129.34 \pm 1.33 \end{pmatrix} \cdot 10^{-3} m \) between the two frames. At 100Hz this corresponds to \( \approx 46.5 \text{km/h} \) which is a reasonable speed estimate for driving inside a closed settlement. Unfortunately, speed values from additional sensors (GPS, inertia) where not available for this dataset. The rotations were all below 0.01 degree, although earlier tests showed that the voodoo tracker may underestimate the true rotation and give instead to high xy-translation estimates, especially for small displacements.

This assumption is strengthened by Figure 4.17 which shows the tenth frame of the sequence and the first frame warped according to the flow between the first and tenth frame. Understandably there are heavy image distortions in the regions with suboptimal disparity estimates, but there's also a significant shift between the other regions (For example, the position of the arrow on the street is off by about 15 pixels). As neither occlusions occurred nor the scene rigidity constraint was violated, the only cause left is the camera translation. Therefore the method seems in it's current form only suited for small camera displacements (below 1 meter per frame).

To compare the effects of different scalings, a second camera translation was used to compute the optical flow. This translation was calculated using a slightly modified set of parameters for the forstner feature detector. The second fit resulted in an different z-translation between the frames \( (91 \pm 2\% \cdot 10^{-3} m) \) but otherwise comparable results. The first calculated flow is labeled (A), the second (B).

To compare with a classical algorithm, the optical flow was also computed with the method by Zach et. al. [40], with a parameter set similar to the one used in Chapter 3. (regularization strength \( \lambda = 200 \), 12 pyramid levels at 0.8 each, edge weighted TV and 4 warps per level). A second flow field was computed with the algorithm by Black and Anandan (B&A) [7]. (Parameters: \( \lambda = 1.0 \), 9 pyramid levels at \( \frac{1}{15} \) each)\(^4\) With the exception of smaller artifacts in the B&A results, the flow fields showed no fundamental differences.

The chosen parameters show already some of the problems of classical optical flow estimation. The sequence includes an apparent motion of up to 20 pixel per frame, but many flow algorithms can only detect flows of 1-2 pixel per frame reliably. If the flow amplitudes are bigger, pyramid warping schemes have to be used to reduce the flow in each level. Often used pyramid scaling schemes involve gaussian or laplacian pyramids with scale factors of \( s = [0.5 - 1.0] \). But often those parameters are not well motivated. It's not even clear why schemes of the form \( s^n \) are used. A serious drawback is also the increase in calculation time and memory consumption that comes with an increasing number of pyramid levels.

The here described method works in principle for arbitrary motions and therefore for arbitrary flow speeds. The computational cost is also independent from the flow vectors, as long as the disparity range needs no adaption due to extremely close objects.

Figure 4.15 shows the endpoint errors and direction errors \(^5\) between our method and the classical optical flow. (Remark: We call the difference endpoint error, even if none of the two involved flow fields are true ground truth)

For (A) (first depth scaling) the global difference is lower (regarding the median) than for (B) (second depth scaling), but the spatial distribution is different. For close points (e.g. on the street) the difference is higher but it’s lower for far away points. This would indicate that the

---

\(^4\)Implementation provided by Sun,Computer Science Department, Brown University.http://www.cs.brown.edu/~dquon/research/software.html

\(^5\)direction error and the angle error of the translation matrix are not directly correlated
<table>
<thead>
<tr>
<th></th>
<th>endpoint mean error</th>
<th>endpoint median</th>
<th>direction error</th>
</tr>
</thead>
<tbody>
<tr>
<td>A (Zach)</td>
<td>0.86 ± 0.36</td>
<td>0.73</td>
<td>12.1° ± 10.1°</td>
</tr>
<tr>
<td>B (Zach)</td>
<td>0.87 ± 0.22</td>
<td>0.93</td>
<td>14.6° ± 10.3°</td>
</tr>
<tr>
<td>A (B&amp;A)</td>
<td>0.98 ± 1.12</td>
<td>0.75</td>
<td>12.9° ± 13.0°</td>
</tr>
<tr>
<td>B (B&amp;A)</td>
<td>1.03 ± 1.06</td>
<td>0.98</td>
<td>15.2° ± 12.9°</td>
</tr>
</tbody>
</table>

Table 4.1: Crossing: endpoint and direction errors

<table>
<thead>
<tr>
<th></th>
<th>x mean</th>
<th>x median</th>
<th>y mean</th>
<th>y median</th>
</tr>
</thead>
<tbody>
<tr>
<td>tracked flow (A)</td>
<td>1.56 ± 2.90</td>
<td>1.14</td>
<td>2.79 ± 3.08</td>
<td>1.19</td>
</tr>
<tr>
<td>tracked flow (B)</td>
<td>1.78 ± 2.48</td>
<td>1.43</td>
<td>2.85 ± 2.70</td>
<td>1.48</td>
</tr>
<tr>
<td>zach</td>
<td>1.36 ± 3.05</td>
<td>1.17</td>
<td>2.28 ± 2.98</td>
<td>0.58</td>
</tr>
<tr>
<td>B&amp;A</td>
<td>1.35 ± 2.77</td>
<td>1.25</td>
<td>2.12 ± 2.60</td>
<td>0.68</td>
</tr>
</tbody>
</table>

Table 4.2: Crossing: XY Flow Components

The estimated translation between the frames was too high for (B), while the rotation estimate was closer to reality than (A).

We can confirm the assumption by examining the x and y flow components separately (figure 4.15): The most obvious discrepancy can be found in the median of the y flow component. This indicates either an rotation or y translation bias in the camera transformation matrix.

On object boundaries the methods performs suboptimal as the invalid confidence region around the edges is quite large. This partly caused by the selected smoothing parameters of the stereo algorithm and the size of the algorithms aggregation window. But settings which give sharper edges can create additional depth discontinuities or artifacts on other surfaces, so this is a rather heavy trade-off.
Figure 4.15: *Crossing*: endpoint errors for different depth scalings
(0.0(black) - 2.0(white))

Figure 4.16: *Crossing*: residual of image warped wrt estimated flow
(values between 0 and 500 gray levels)
Figure 4.17: Crossing: tenth frame; first frame warped wrt flow between first and tenth frame

**Image Warping:** For a sequence, warped according to the tracked flow between frame one and two, the errors are shown in figure 4.18 and table 4.3. The errors are quite low, most of the deviations are caused by effects on the image border, the sky or in case of Black & Anandans method by blob-artifacts. This demonstrates how the calculated flow can be used to generate ground truth sequences.

![Image](image1)

(a) Zach  
(b) B&A

Figure 4.18: Crossing endpoint errors for warped sequence  
(0.0(black) - 1.0(white))

<table>
<thead>
<tr>
<th></th>
<th>endpoint error</th>
<th>median</th>
<th>direction error</th>
<th>median</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Zach)</td>
<td>0.17 ± 0.39</td>
<td>0.08</td>
<td>1.46 ± 6.32°</td>
<td>0.36°</td>
</tr>
<tr>
<td>(B&amp;A)</td>
<td>0.40 ± 1.56</td>
<td>0.09</td>
<td>2.73 ± 12.21°</td>
<td>0.39°</td>
</tr>
</tbody>
</table>

Table 4.3: Crossing endpoint and direction errors for warped sequence  
(confidence map was not used, only masked out a 13 pixel wide border for averaging)
<table>
<thead>
<tr>
<th></th>
<th>endpoint error</th>
<th>median</th>
<th>direction error</th>
<th>median</th>
</tr>
</thead>
<tbody>
<tr>
<td>with Car &amp; Shadow (Zach)</td>
<td>1.75 ± 3.32</td>
<td>0.82</td>
<td>24.9 ± 31.56°</td>
<td>14.24°</td>
</tr>
<tr>
<td>without Car &amp; Shadow (Zach)</td>
<td>0.80 ± 0.41</td>
<td>0.79</td>
<td>25.55 ± 33.10°</td>
<td>13.61°</td>
</tr>
<tr>
<td>with Car &amp; Shadow (B&amp;A)</td>
<td>2.02 ± 3.45</td>
<td>0.84</td>
<td>25.32 ± 31.5°</td>
<td>15.4°</td>
</tr>
<tr>
<td>without Car &amp; Shadow (B&amp;A)</td>
<td>1.24 ± 2.37</td>
<td>0.81</td>
<td>25.98 ± 32.94°</td>
<td>14.48°</td>
</tr>
</tbody>
</table>

Table 4.4: Traffic endpoint and direction errors

**Sequence: Traffic**

The second real sequence is an interesting example of both the cases where the method fails and where it can surpass classical flow estimation.

In the case of cars moving in or against the direction of the camera, the rigidity constraint is clearly violated. As expected, the classical and the “tracked” flow field differ strongly in those regions. (Figure 4.23). Although in this case it would be possible to detect the incorrect motion by comparing the disparities for the second frame with the estimated camera translation. If the 3D pointcloud for the second frame does not (roughly) coincide with the translated pointcloud of the first frame, the rigidity constraint was probably violated.

In contrast, the shadow, which is getting cast onto the road by the car, marks a region where regular flow estimation can fail. The shadow itself is moving with the same relative speed as the car, while the underlying street is moving like it’s surroundings. This represents an local violation of the brightness constancy constraint where the results of classical algorithms are hard to define. Classical algorithms will estimate a flow which corresponds to the shadow movement, unless a very high regularization strength is used.

The question is what is the correct flow vector at this point? Our method estimates a flow field which does not differ much from the surrounding street level as the disparity estimate is quite robust against illumination changes. The high discrepancy of the flow fields in this region can be used as a sanity check to test if the evaluated fields make actually sense.

This sequence also shows how the direction error is a rather bad error measure. The difference of flow angles increases significantly with the distance, which leads to the impression that the optical flow estimate is suboptimal. But as the length of the flow vectors also decreases with the distance the actual difference is only marginal.
Figure 4.19: Traffic sequence
Figure 4.20: Traffic disparity map
(confidence coverage: 74.5%)

Figure 4.21: Traffic confidence map

Figure 4.22: Traffic: tracked flow as hsv overlay

Figure 4.23: Traffic: endpoint error
(0.0(black) - 2.0(white))
(Errors can be higher than 2.0 or 45° but are capped at those values for better visualization)

Figure 4.24: Traffic: direction error
(0°(black) - 45°(white))
4.5 Possible Improvements and Future Work

As already mentioned, the assumption of a rigid scene is a rather strict constraint. One possibility to ease this, would be to generate the point cloud for the second frame and register it against the cloud of frame one to detect which pixels violate the constraint. If the depth of a pixel in frame two does not correspond to the depth of the transformed point cloud of frame one, then the constraint was probably violated in that point. This can also be used as a sanity check for the computed camera motion. By computing the convex hull of the points where the rigidity was violated (by means of triangulation), the scene can be separated into individually moving objects.

In the future we want to use the motion data provided by the IMU to enhance the camera tracking step. Speed or steering information from the car could also be incorporated. This would especially ease the calculation of the transformation scale factor (figure 4.9) which is currently quite inaccurate. Another possibility is to use the method in a laboratory environment where the camera motion can be controlled more accurately. An example would be to mount the camera on an industrial robot arm or optical bench.

Additional computation of the right-to-left disparity can provide additional sanity checks for occluded regions. This could improved the suboptimal behaviour at object boundaries.
5 Conclusions

5.1 Summary of Experimental Results

Two new methods for creating reference data for optical flow evaluation have been presented in this thesis. Both still have problems or uncertainties regarding the accuracy of the created optical flow but are none the less promising. They may be especially suited to further investigate the implications of the brightness constancy constraint in optical flow estimation.

Chapter 3 deals with the fact that synthetic sequences are often considered to be too unrealistic to be used for evaluation of optical flow algorithms. The flow differences between a rendered and a recorded scene were investigated. The endpoint error of the rendered scenes started at 0.29 ± 0.63 for the least realistic one and increased to 0.40 ± 0.83 for the most realistic one, thereby approaching the value of the recorded scene (0.48 ± 0.75). The median of the endpoint error increased in a similar manner (from 0.07 to 0.11 in comparison to 0.24). Differences in the local structure of the error, that can be attributed to local brightness and texture variations, were found.

Chapter 4 describes the use of stereo cameras and monoscopic camera tracking to calculate the optical flow in a scene. It was tested on one synthetic and two real world sequences, and the calculated flow fields were compared with those created by two classical optical flow algorithms. The best endpoint difference was 0.03 ± 0.04 for the synthetic scene, 0.83 ± 0.36 for the recorded scene complying to the rigidity constraint and 1.75 ± 3.32 for the recorded scenes where this constraint was violated. Further analysis shows suboptimal flow estimations in case of object borders or occlusions.

Finally a stereo camera system for fast high-resolution image capturing has been developed (see chapter 2). As planned it is capable of capturing stereo images with an resolution of 1312 x 1082 pixel at 100 frames per second and 12bit grayscale resolution per pixel. The system is in active use and has already produced several terabyte of automotive scene footage. In this thesis only the advantage of high image resolution and capture speed has been taken, but there are already plans to use the high bit depth and GPS/IMU data.

5.2 Method Evaluations and Outlook

If we treat the stereo based flow estimation just as an alternative calculation method, we can name several advantages over classical algorithms:

First, it can detect nearly arbitrary large flow vectors. Other flow algorithms usually rely on image pyramids to calculate flows of more than one pixel per frame. This means one has to estimate the maximum flow or gradient in a sequence to select an appropriate pyramid scaling scheme. Higher pyramid levels also add to the computational cost of the algorithm. The alternative method described above does not have these dependencies. Second, the epipolar constraint reduces the correspondence problem to a one-dimensional search, thereby reducing the computational cost.
The monoscopic camera tracking is but one source to estimate the camera translation. Other methods which incorporate the stereo data further, or which use external motion data (e.g. IMU, wheel sensors) can be used as a replacement or to supplement the available motion data. The accuracy can be even more increased when the movement is restricted (e.g. by mounting the system on an robot arm etc.).

Most important is the fact that the method does not depend on any brightness constraints. This implicates a high robustness regarding illumination changes, shadows and reflections, effects where other algorithms may produce unpredictable results. Pixel noise is also a minor problem.

Finally, the used stereo algorithm can produce confidence maps which can be used to detect object boundaries and occlusions. Additional sanity checks for those regions are possible by additionally calculating the right-to-left disparity of the image pair.

On the other hand there are certain disadvantages:

First of all, the scene is currently limited to rigid geometry and motion, although some methods to loosen this constraint have been discussed. Second, the method does not perform very well on object boundaries. As a result of this the computed flow is not completely dense as confidence can only be given for \( \approx 80\% - 90\% \) of the image area.

Furthermore the accuracy of the calculated flow fields is currently not on par with the newest classical flow algorithms. This prevents the method from directly being used as a general source for reference data. Although it may still be useful for that purpose in a small set of specialized applications, and further improvements may cross the gap to state-of-the art algorithms. This seems like a valid assumption as the main limiting factor is currently the accuracy of the camera translation, for which we have already mentioned some possible improvements. Theoretical evaluation and the tests on the synthetic scene have shown that a high accuracy is possible and the disparity has only a minor influence on the expected flow fields.

One advantage regarding further use of the method is that the resulting flow, even if it is not true ground truth, represents a physically plausible flow field. Neither an incorrect camera transformation nor an incorrect depth map can produce effects which can be considered to be physically implausible. The basic assumptions of optical flow algorithms (the brightness constancy constraint and flow field smoothness) are constraints which have barely any root in the physical representation of the world.

One could argue that the rigidity constraint is a similar unrealistic assumption. A given scene or situation where the brightness constancy assumption is true can of course be a valid physical situation, but not necessarily a likely one. A rigid scene on the other hand is, even for highly dynamic content like automotive scenes, a totally valid conjecture.

Image sequences with known ground truth could be generated by warping an image according to a given flow field. The ground truth is then simply the flow used to warp the image and the BCC is automatically valid. This warping method is already in use to compare the results of different flow algorithms against each other. (See [15], Section 9.3.4 and 13.8.2 for details). In case of an artificial flow field or a field calculated by classical means it is not clear if this sequence has any physical relevance for the desired application. On the other hand, if the warping is done using a flow field calculated with the above described method, the resulting image sequence may be more physically plausible even though the BCC is fulfilled. One still needs to determine whether a non-violated BCC is actually a positive outcome for an given application or not.

Chapter 3 results indicate that at least parts of the above mentioned criticisms regarding synthetic scenes can be overcome with the use of sophisticated rendering techniques. Additionally it allows to investigate which real world effects cause optical flow results to deteriorate. Further
study regarding the simulation of the bidirectional reflectance distribution function (BRDF) and global illumination models seem necessary to give a final answer if synthetic sequences can be used for algorithm evaluation. None the less the results so far look promising.

The BCC is a more complicated matter as it is not yet clear, how accurate local illumination changes can be modeled. But the field of computer graphic is making steady progress and most of the more sophisticated methods have not yet been used to evaluate optical flow in the described manner.

**Combination of synthetic scenes and stereo based flow**

The evaluation of both methods has shown that simple means or medians of different error measures are not always sufficient to evaluate optical flow fields. Cross correlation images or warping residual images show differences in the local structure of several flow fields which where not apparent in the mean endpoint error values. For many applications, object boundaries mark valid regions of interest, but are seldom treated in a special manner to address their importance. Performance of the stereo based flow estimation was poor in these regions but at least it provided some detection scheme to flag them. It may be beneficial to investigate the properties of the here used error measures or to develop new ones to address these issues.

It is also possible to combine the above methods for sequence generation. The stereo augmented flow estimation can provide object meshes (by triangulating the 3D point cloud) and realistic textures, while the advanced rendering methods can be used to change or adapt illumination conditions and mark regions of interest. A partial reconstruction of an objects' BRDF out of at least two view angles may also be possible. This could provide a fully automated evaluation system which can not only produce reference data but also give estimates on where optical flow computation is actually feasible.

### 5.3 Final Comments

The experiments in this work have shown that the creation of reference data is no easy task. Optical flow algorithms are already very exact in regions where the brightness constancy and smoothness assumptions are met. Reference data is most valuable in regions where this is not the case, so methods for creating it should not depend on those assumptions. Existing image sequences address this point seldom and outdoor sequences with known ground truth are practically unheard-of, so further studies in this direction seem necessary. So, hopefully the results of this thesis will help to develop more sophisticated methods and sequences, which will allow better evaluations of current and future optical flow algorithms.
List of Figures

2.1 photo response curve for a regular and a defective pixel .......................... 16
2.2 Results of radiometric image rectification ............................................ 17

3.1 camera model ....................................................................................... 19
3.2 Rendered and real scenes ...................................................................... 25
3.3 Difference between second and first frame of real sequence, Difference of second
and first frame of realistic sequence ......................................................... 26
3.4 Warping Residuals ................................................................................ 27
3.5 Normalized endpoint errors .................................................................... 31
3.6 Normalized cross correlation between the endpoint errors of two sequences ... 32
3.7 Differences between ground truth and calculated flow for x and y directions ... 32

4.1 stereo camera setup ............................................................................... 35
4.2 stereo camera setup in XZ-Projection .................................................. 35
4.3 Example Feature Points and Tracks ...................................................... 38
4.4 Endpoint Errors at pixel (200,200) ...................................................... 41
4.5 Endpoint Errors for disparities 10,25,50 and 75 depending on the pixel (x, y) .... 42
4.6 Synthetic Test Sequence ........................................................................ 44
4.7 Synthetic: Disparity and Confidence Maps .......................................... 45
4.8 Synthetic: endpoint errors ..................................................................... 46
4.9 Synthetic: Scaling between the 3D feature points calculated by the voodoo tracker
and the actual depth values computed via the disparity .............................. 46
4.10 Crossing: Sequence ............................................................................ 48
4.11 Crossing disparity map ........................................................................ 49
4.12 Crossing confidence map ...................................................................... 49
4.13 Crossing: tracking scale estimate ......................................................... 49
4.14 Crossing: flow calculated with Zach tv11 method as HSV overlay ............ 49
4.15 Crossing: endpoint errors for different depth scalings ........................... 52
4.16 Crossing: residual of image warped wrt estimated flow ......................... 52
4.17 Crossing: tenth frame; first frame warped wrt flow between first and tenth frame 53
4.18 Crossing endpoint errors for warped sequence ...................................... 53
4.19 Traffic sequence ................................................................................. 55
4.20 Traffic disparity map ............................................................................ 56
4.21 Traffic confidence map ........................................................................ 56
4.22 Traffic: tracked flow as hsv overlay .................................................... 56
4.23 Traffic: endpoint error ........................................................................ 56
4.24 Traffic: direction error ......................................................................... 56
List of Tables

3.1 Average endpoint errors and standard deviations .......................... 28
3.2 Median of endpoint errors ................................................. 28

4.1 Crossing: endpoint and direction errors .................................. 51
4.2 Crossing: XY Flow Components ......................................... 51
4.3 Crossing endpoint and direction errors for warped sequence ........... 53
4.4 Traffic endpoint and direction errors ..................................... 54
Bibliography


Erklärung:

Ich versichere, dass ich diese Arbeit selbstständig verfasst habe und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.

Heidelberg, den 30.09.2010 ..........................