Master Thesis

# Comparison of Disparity Algorithms for Stereoscopic Video


Ben John


May 2016


Supervisor: Dr. Stephan Kopf

Department of Computer Science IV
Prof. Dr.-Ing. W. Effelsberg

School of Business Informatics and Mathematics
University of Mannheim

# Abstract

Stereoskopische Videos speichern zwei separate Ansichten einer Szene, die sich üblicherweise nur in geringen horizontalen Verschiebungen der Pixel unterscheiden. Diese Pixelverschiebungen (Disparity) resultieren aus den unterschiedlichen Entfernungen der Objekte in der Szene. Ziel der Master-Abschlussarbeit ist es, bestehende Verfahren zur Berechnung der Disparity zu analysieren, geeignete Verfahren für Videos auszuwählen und diese zu implementieren. Ein spezieller Fokus soll auf Erweiterungen für Videos liegen, indem beispielsweise vorherige oder zukünftige Frames berücksichtigt werden. Zur Messung der Qualität der implementierten Verfahren sollen bestehende stereoskopische Bild- und Videoarchive genutzt werden.

# Contents

# List of Figures

# List of Tables

# 1 Introduction

## 1.1 Motivation

Obtaining depth information as additional data to infer intents from human gestures has arrived in mainstream computing with the release of Kinect at November 4th, 2010[1]. Kinect is a hardware add-on for the Xbox video gaming console which enables users to interact visually with the console without actually using a controller or any other peripheral. The Kinect utilizes two cameras, one capturing colored and the other monochrome images. The monochrome sensor is used in combination with an infrared laser projector to obtain depth information via time of flight (TOF). Time of flight is a method to measure the time light needs to reach objects and then calculate the distance.

With deducing intents from human gestures a step in the field of artificial intelligence was made as the computer is now able to interpret human body language. As this means processing an enormous stream of data (gathering and processing frame by frame) it represents a dataset of large and complex nature, also known as big data. This also implies the need for new data processing techniques in comparison with traditional ones. As a result one could say that computer vision is linked to both, artificial intelligence and big data. New applications which arose from the combination of those topics are for instance:

- robotic,
- medical image analysis,
- automatic surgery,
- 3DTV,
- video compression,
- autonomous driving.

[1] `http://gizmodo.com/5563148/microsoft-xbox-360-kinect-launches-november-4`

Besides the technology of time of flight laser sensors - such as the Kinect[2] - there exists also the possibility to obtain depth information from stereo images by analyzing coherent images with so called disparity algorithms. Thus, it is sufficient to have two calibrated aligned cameras (a stereo camera) to acquire disparity information and calculate the depth at each point. But this leads to another fundamental problem of stereo matching: stereo correspondence, that basically means the labelling of pixels, i.e. which pixel of the left image belongs to the corresponding pixel on the right image as projection of the same three-dimensional point from the captured world projected to the image plane in every image. This problem of stereo correspondence has to be solved in order to actually match those and calculate the disparity. According to Scharstein and Szeliski, stereo correspondence is one of the most heavily investigated topics in computer vision [47]. As there is still a lot of research going on, no algorithm is working without any mistakes and also the runtime is a bit quirky, Microsoft Kinect established itself as a real alternative. This leads us to one of the disadvantages of Kinect sensor: Kinect is sensitive to other infrared sources (like sunlight) due to its nature of utilizing an infrared laser projector to acquire depth information, a stereo camera does not have this issue. Although using two coherent images also have some disadvantages which will be discussed later on, it is an alternative way to receive depth information. Especially thinking about autonomous driving during which at day a lot of sunlight is involved in, other techniques to estimate how far an object is away from one another are necessary to ensure a certain accuracy and fault-tolerance.

Although the topic of this thesis is neither about artificial intelligence nor big data the foundations and algorithms discussed can help machines to sense their environment through cameras, identify objects and estimate the distances towards each other. With the support of neuronal networks computers may also deduce intents, reason about their environment and then execute defined actions. In conclusion computer vision is a research field on its own but other fields may also benefit from the results. Computer vision establishes itself on the consumer market as more research is done. The upcoming iPhone supports this as it will feature a dual camera system [3]. In the year 2011 LG and HTC released the LG Optimus 3D[4] and corresponding the HTC Evo 3D[5]. Both had a stereo camera implemented and an auto-stereoscopic display attached. This enables one to view photos or videos taken in stereographic 3D without the actual need for additional peripheral

[2]Besides the consumer market, for autonomous driving or robotic research Velodyne is a well-known sensor.

[3]`http://9to5mac.com/2016/02/03/sony-dual-cameras-iphone-7-plus/`, 2016-02-22.

[4]`https://en.wikipedia.org/wiki/LG_Optimus_3D`

[5]`https://en.wikipedia.org/wiki/HTC_Evo_3D`

like 3D glasses. Both can be seen as an experiment as there was no big distribution, Apple normally focuses on the mainstream consumer market, opening up the box of possibilities and the need for such algorithms even further. One example application for such a consumer-driven market could be the reconstruction of a face after taking a photo. There exist no method to reconstruct a whole 3D model without having stereo images from all angles of the face, but it is possible to trick the user in having captured a 3D photo. Another concrete example for an application regarding depth estimation in stereo videos: detecting moving people in a stereo video and calculate the distance to the camera of each person[6].

## 1.2 Assignment

The usage and applications of computer vision are huge as seen in the motivational introduction. For understanding how disparity algorithms work it is important to have knowledge of various topics in computer vision. Therefore it was difficult to decide what should be in the thesis and what can be left out. The thesis' main goal is to provide an overview of selected disparity algorithms for stereoscopic videos and evaluated those. Scharstein and Szeliski justified their tiny selection of disparity algorithms with the following: "Compiling a complete survey of existing stereo methods [...] would be a formidable task, as a large number of new methods are published every year." [47]. That said the ones with well documented source code and a research paper, also adaptable within the time scope of this thesis, were integrated.

- The thesis should provide a good fundamental knowledge base for an advanced insight into the area of disparity algorithms for stereoscopic images and videos.
- Existing datasets are examined. Additionally, a novel dataset from the Lehrstuhl für Praktische Informatik IV[7] is presented.
- Based on existing paper and source code, different state of the art algorithms are analyzed and evaluated.
- As there exist a lot of different unaligned code for evaluating / comparing images the decision towards a new implementation of an evaluation suite with OpenCV was made.
- Moreover quality metrics for evaluating those algorithms are defined.

[6] `http://de.mathworks.com/help/vision/examples/depth-estimation-from-stereo-video.html`
[7] `http://ls.fmi.uni-mannheim.de/de/pi4/`

- The algorithms are evaluated on different datasets. The runtime is also measured.
- The impact of image diminishing effects are investigated, like noise from image sensor or artifacts from video compression.
- As a benefit the results can be analyzed visually via a web front-end.
- Finally the results are discussed and future outlook is given.

## 1.3 Outline

The main purpose of chapter 2 is to give an overview of terms and techniques used in this thesis. The following chapter 3 focuses more on disparity algorithms and related work. To give an overview of state of the art algorithms a small summary of current used disparity algorithms is made. This will create the foundations for the later implementation. Chapter 4 describes the implementation and explains reasons for building an evaluation engine. The details of the implementation are explained afterwards. In addition the integration of existing algorithms is illustrated. The evaluation engine was fed with datasets which are introduced in chapter 5. This chapter also explains the used quality metrics and describes the resulting outcome accurately. In the end the results of this thesis are reflected in the concluding chapter. Besides some future work, split in low- and high-level, is pointed out.

# 2 Foundations

In this chapter the foundations for related work and the implementation are built. As a first step, computer vision is introduced with a short explanation how image representation works from a computer's perspective. Human visual perception is put in contrast to how computers perceive and interpret their environment. In addition, the labelling problem regarding stereo correspondence and the disparity between stereo images is illustrated. Furthermore, the depth calculation as well as the taxonomy of disparity algorithms is depicted. Finally, optical flow, a technical method that measures direction and movement of every pixel based on dominant movement in the original scene, is introduced to round this chapter up.

## 2.1 Computer vision

Computer graphics describes the terms and definitions of everything which has to do with basically treating images programmatically on a computer, interpreting and working with them. To give an example, the applications of computer graphics range from image representation, image creation, image transformations to applications of color models. Computer vision shares concepts from the domain of computer graphics, but works in reverse. Instead of modeling a scene and generating an image of it, computer vision optically measures the real world and tries to analyze it by applying models to the captured images. For instance, typical jobs are to get information out of an image, like image segmentation, edge detection, classification, and feature[1] point detection.

A simple example would be to imagine a face of a human being captured by a camera, which may produce errors due to lens distortion, shaky capturing, and sampling of the chip. Image editing would be useful to optimize the image by correction of contrast or brightness, cropping, or further adjustments. The tasks of computer vision are more in analyzing and understanding images for instance (just to name a few):

[1]Geometric shapes or more complex classifiers that are clearly recognizable.

- face localization to know the areas of faces on images,
- feature matching to detect the face on other images,
- feature tracking to track the movement of a person, or
- 3D reconstruction of a facial model.

## Image representation

Two different methods exit of handling images on a computer. On the one hand, a vector image describes its content by representing forms like a circle, line, curve, or rectangle. The properties of these forms and shapes are also included, for instance, coloring, size, and origin. So a vector image basically contains those forms and shapes, their properties and a description of how they are all composed together.

On the other hand, it may become pretty complex using vector images to represent the real-world. In contrast to those there also exist raster images. Raster images (sometimes the term bitmap images is used) are a form to represent natural images, e.g. captured by a CCD[2] image sensor from a digicam. Capturing means sampling information on a matrix of light sensitive sensors to transform received signals into a matrix of color values of the same size.

Both types of images use a coordinate system to describe either the placement of elements (like written above with the properties size and origin of each element) or to describe how each point looks like. The coordinate system most widely used working with images starts in the upper left at the point $(0, 0)$, with the x-axis extending to the right and y-axis extending to the bottom. This can be seen as a grid system with the size of the image $width \times height$ representing $columns \times rows$. By describing how each point looks like the exact description of a pixel is meant [49].

One pixel in a grayscale image can range from $0 - 255$ describing the intensity of this pixel. $0$ means black and $255$ is fully white. In colored images a pixel can have more than one intensity value. More concrete, in a typical RGB[3] raster image each pixel contains three color channels, also called the RGB tuple. Thus

$$3 \cdot 1\, bytes = 3 \cdot 8\, bits = 3 \cdot 8 = 24\, bits$$

are stored per pixel utilizing RGB tuples. In C or C++ such pixel values are normally described as unsigned chars. A char represents eight bit and unsigned means

[2]CCD: charge-coupled device
[3]RGB: red-green-blue color channels

that it ranges from $0-255$ instead of $-128$ to $127$. Sometimes RGB is used with an additional alpha channel specifying the degree of opacity, named RGBA[4]. The composition of these color channels orchestrate the final pixel value as it is obtained by, for example, an image sensor. Figure 2.1 depicts an example of a RGB raster image and shows the values of three pixels. The first marked pixel in figure 2.1 describes the RGB tuple with the following values $(237, 237, 237)$. Utilizing three color channels the final raster image then needs up to 24 bits per pixel, meaning an image the size $width = 300\ px$ and $height = 400\ px$ needs

$$300 \cdot 400 \cdot 24 bits \cdot \frac{1\ byte}{8\ bits} = 360.000\ bytes$$

in memory. Images can be compressed with, for example, the JPEG algorithm but as the later implementation works only with pure raster images, as the unaltered values are examined, the amount of bytes as explained above is to be held in memory during the execution of the implementation.

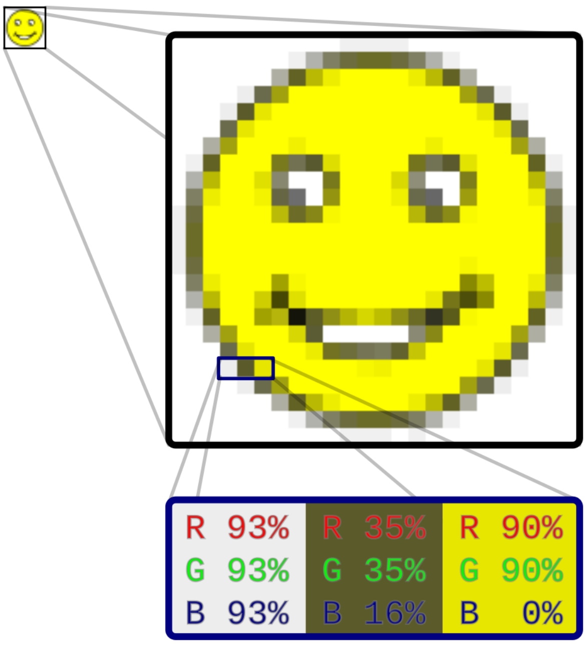



Figure 2.1: Example of a RGB raster image[5]

[4]RGBA: red-green-blue-alpha color channels
[5]Source (accessed 02/2016): `https://en.wikipedia.org`.

## Human visual perception

In his manuscript 'Astronomia Pars Optica' from 1604, Kepler explains the use of both eyes for depth perception. He defined the term binocular as the composition of two latin words, 'bini' for double and 'oculus' for eye. With uniocular as 'uni' for one the sight with only one eye is meant. Binocular vision is then the vision creatures having two eyes obtain while using them together according to Kepler. According to Fahle [17] and Henson [22] creatures with binocular vision have several advantages over creatures with only uniocular vision. Not to mention all but three, the most important ones which affect the depth perception:

1. Considering human beings, the second eye increases the field of view [22]. About 120 degrees are the binocular field of view (projected on both eyes) and two uniocular fields of view with about 40 degrees.

2. This also leads to another advantage with occluded, half-occluded or non-occluded objects [17]. Looking at Figure 2.2, the point $P$ is in focus of the human being. Something directly behind this point may be fully occluded by the object in point $P$. Most of the things besides are non-occluded. Something behind this point $P$ may be half-occluded if it can be seen by either the left or the right eye.

3. An advantage of having two eyes is the third-dimension human beings perceive, which leads to the binocular disparity or retinal disparity. Both terms are used in the literature and both mean the same: extracting depth information out of two coherent retinal images (obtained by the human eyes) [13, 17].

Figure 2.2 depicts the mapping of the three points $R$, $P$ and $Q$ on the retina of each eye. The letter $F$ stands for *foveae* in which the visual axis ends. The eye is constructed out of photoreceptor cells, mainly rods and cones. The rods are necessary for seeing at night while the cones are responsible for humans being able to see the world sharp. In the foveae is the peak of cones and it contains very few rods. This means that the human visual system works the way that the visual axis joins the point of fixation with the foveae. This can be seen in Figure 2.2 as the lines between $F$ of each eye to the point of fixation $P$. Both eyes should be brought into convergence that the point of interest is projected onto the foveae of each eye. Everything on the horopter (the circle) is corresponding (e.g. $P$ and $Q$), all points other than being on the horopter are non-corresponding ($R$) in terms of retinal disparity.

In the later described disparity algorithms which act like a tool for computers to be able to see the shift of pixels from the left image to the right image, the human being does somehow the same. Humans experience the depth which is sensed unconsciously by the eyes and calculated by the brain in real-time. With two eyes basically two slightly different images are obtained. The brain acts as the computer which puts both coherent images together and extends the two-dimensional space into a three-dimensional space and calculates the position of the objects in the z-axis.

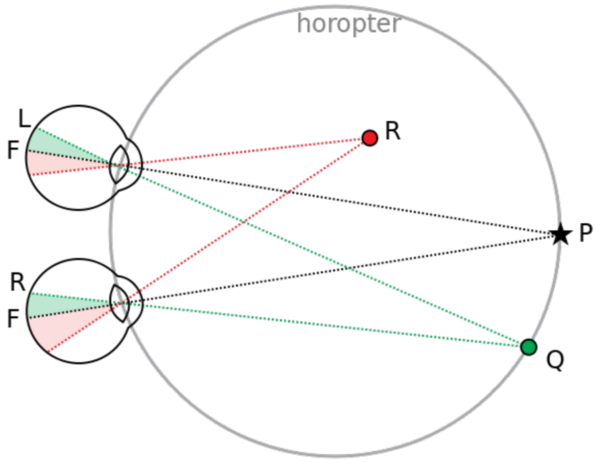



Figure 2.2: Binocular vision with horopter principle[6]

In contrast to the visual perception human beings perceive, computers need to do several steps to obtain disparity and calculate depth information:

- identify objects,
- identify layers,
- match objects / pixels in both images,
- calculate the shift of the pixels from left to right, and
- obtain final depth values.

The human brain enables human beings to see and experience the real-world three-dimensional. A computer has to be programmatically instructed to identify and group objects in both images [4, 13]. This is not an easy task as can be seen in the next sections.

[6]Source (accessed 02/2016): `https://en.wikipedia.org`.

## Stereoscopy

To paint the bigger picture, stereoscopy and the illusion of depth are introduced. Stereoscopy is sometimes linked to the phrase 'the illusion of depth' as it is a technique used to add a third dimension to a flat image to simulate depth [4, 39]. Specifically, the goal is to show each eye a slightly different image and thus achieve depth perception in our brain. With so called stereoscope or special glasses depth perception can be transferred to the consumer in a cinema or at home via showing each eye a different image which then is composed to the final spatial perception.

There exist several techniques to create the stereoscopic effect. One of these glasses is the shutter system. The concept of a shutter glass is that it cycles a block (meaning only one eye is dispatched to the screen) with a certain frequency (usually about 120 fps[7], resulting in 60 fps per eye) synchronously with the 3DTV. This means that only one specific image is passed to exactly one of the consumers eyes. So each eye is shown about 60 fps which naturally is experienced as flicker-free. The older anaglyph 3D technique uses multiplied images tinted with red/cyan to filter out the respective image by the glasses filter foil, thus only one image is dispatched to one specific eye at a time. Nowadays the anaglyph 3D technique is sometimes used in magazines to show 3D graphics. As all techniques are not representing the real-world and the depth perception can be adjusted with for instance camera positioning (image one would reposition his eyes to perceive the real-world differently) they can be summarized as the illusion of depth.

## Epipolar geometry

The geometry of stereo images, called epipolar geometry, plays an important role in understanding the mathematical equations in the upcoming section. The most important terms of epipolar geometry are:

- image plane,
- baseline,
- epipole,
- epipolar line, and
- epipolar plane.

[7] frames per second

The *image planes* in Figure 2.3 and Figure 2.4 are the blue surfaces which represent the captured image through the cameras $O_L$ and $O_R$. The *baseline* is the line joining both camera centers with the image plane. Focusing on the figures, the baseline is the line going from $O_L$ to $O_R$, as $O$ reflects the origin (camera center). An *epipole* is the joint of the baseline with the image plane, referring to the symbols $e_L$ and $e_R$. The *epipolar plane*, visualized as green triangle in Figures 2.3 and 2.4, is determined by point $X$ and both origins $O_L$ and $O_R$. It is the surface reflecting the z-axis, the depth. An *epipolar line* then is the intersection between the origin to the point of interest, in this particular case $X$, which lies on the epipolar plane and intersects the image plane.

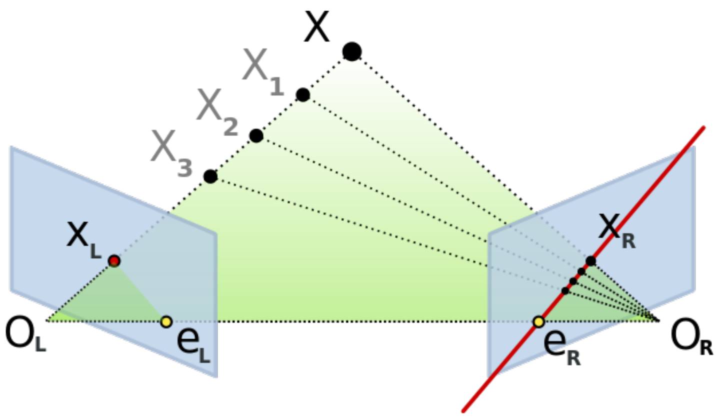



Figure 2.3: Epipolar geometry[8]

This results in an epipolar constraint [13]: Each image point $X_i$ of a space point in the image plane, e.g. consider point $X$ in Figure 2.3 must lie on the corresponding epipolar line $\vec{O_L X}$. More concrete focusing on Figure 2.3: this constraint states that the correspondence for a point on the epipolar line $\vec{O_L X}$ must lie on the line $\vec{e_r X_r}$. As seen above Figure 2.3 depicts the left and right view of an object in point $X$.

The Figures 2.3 and 2.4 both illustrate the epipolar geometry on a pair of un-rectified images and the result after the rectification was done. Rectification[9] is necessary to reduce the search-space from two-dimensional to one-dimensional. For determining the exact position of $X$ (possible positions $X_i$ with $i = [1 \dots 3]$)

[8]Source (accessed 02/2016): `https://en.wikipedia.org`.

[9]Affine transformation (rotation and translation) neglecting geometric distortion to rectify the images.

the diagonal has to be scanned in the unrectified image. In the rectified image only the horizontal needs to be investigated. In the further proceeding this line is called the scanline which most of the algorithms operate on [10, 13]. After the rectification process the following two statements come true:

- Epipolar lines are parallel to the x-axis (horizontal).
- Corresponding points are on the same y-axis (vertical).

Implicitly the following two assumptions were made:

- the focal length $f$ of both cameras which captured the images are the same,
- the origin of one camera is the so called camera principal point (the joint of the optical axis with the image plane and the fovea counterpart) [13].

In conclusion, corresponding points are constrained to be on the same line and thus depth can be inferred by using triangulation and camera parameters. Based on this, the investigations of stereo correspondence and the actual depth calculation using triangulation is discussed in more detail in the next sections.

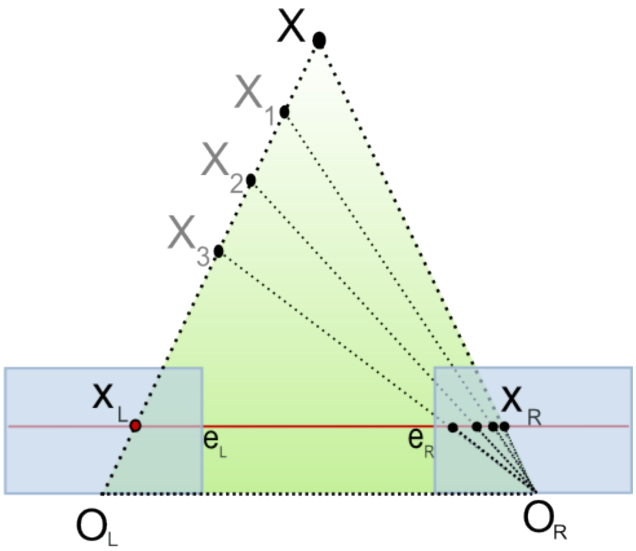



Figure 2.4: Epipolar geometry after image rectification[10]

[10]Source (accessed 02/2016): `https://en.wikipedia.org`.

## 2.2 Stereo correspondence

Stereo correspondence can be seen as a pixel labelling problem [13, 58] between two (or more) stereo images. The essential problem is to find corresponding pixels in images of different cameras and needs to be solved. Without knowing which points belong to each other in two separate stereo images, no conclusions can be drawn for instance calculating the disparity. Henceforth, while talking about images for stereo matching, rectified images are implicitly meant. If images are existing in an unrectified unaligned version then as a preprocessing step the images will be rectified.

### Constraints

Stereo matching algorithms rely on several assumptions about the real-world. From the pioneer work of Marr and Poggio [40] the following constraints can be reasoned which are important for the development of such algorithms (also cf. [13, 30, 58]):

*As a short remark: $X_i$ is a point on a scanline in image $i$ ($i$ can be replaced with either $L$ for left or $R$ for right side) and thus only the x-position is mentioned.*

**Uniqueness**
: As each pixel on a surface has one unique physical position in space, each pixel from each image has at most one disparity value.

**Continuity**
: The term smoothness constraint is also mentioned in the literature. Disparity can vary but smoothly almost everywhere in an image except at object boundaries which represent a discontinuity in depth, i.e. the difference of adjacent points should be small $||X_{L_1} - X_{R_1}| - |X_{L_2} - X_{R_2}|| < \varepsilon$.

**Epipolar**
: Recapture of the epipolar constraint from the section before: corresponding image points have to lie on the corresponding epipolar lines. If the epipolar lines are known to be parallel to the x-axis, the search space can be reduced to a 1D search space along the epipolar lines.

**Ordering**
: Following up the epipolar constraint: if the epipolar lines run in parallel to the x-axis, multiple consecutive image points have to lie on the same corresponding epipolar line in the same ordering.

**Limit**

There is a defined disparity maximum (limit) $d_{max}$ holding $|X_L - X_R| < d_{max}$, defining the maximum disparity value which can be found in a stereo image. Hence, $d(x, y)$ is in the range $[0 \ldots d_{max} - 1]$.

**Lambertian**

Algorithms for stereo matching also rely on the assumption of opaque lambertian surfaces, meaning a surface that reflects light equally into all directions and thus appears equally bright independent from where light is coming and where the camera is placed. Thus the algorithms can expect the intensities and colors of corresponding points to be almost the same.

Besides those constraints there also exist some common pitfalls which can disturb the result of algorithm.

## Common pitfalls

Algorithms are using different metrics to analyze similarities in images along scanlines, in whole areas, or at a global view to then estimate the disparity. This can be challenging especially considering the upcoming traps. On the one hand, potential issues from the camera setup can be challenging, such as:

- photometric distortions,
- noise,
- calibration error of the cameras.

On the other hand, the scenery can be tricky:

- specularities and reflections,
- transparent objects,
- matching ambiguity,
- occlusions (missing data) and discontinuities.

These issues also challenge the algorithms to stereo match the pixels correctly. With matching ambiguities, constant or low-contrast regions are meant. A good example for that are textureless regions or repetitive structures. Textureless regions could contain a small set of matching pairs of pixels, other pixels of that region could be erroneously assumed the same. The presented constraints support the algorithms regarding those pitfalls.

### Simplified stereo matching

Figure 2.5 depicts a simplified example of how stereo matching works on a one-dimensional search space: there exist two arrays with $length = 5$, one in the left and one in the right image. Assuming the top row $[p \ldots t]$ reflects one row in the left image. The bottom row $[u \ldots y]$ accordingly the same row in the right image. The pixel $p$, $q$, $w$ and $y$ are unmatched, e.g. occluded. Having a function $d(z)$ which returns the disparity for a given element $z$ in those arrays, $d(r) = -2$ means the shift two to the left. Accordingly $d(s) = -2$ and $d(t) = -1$.

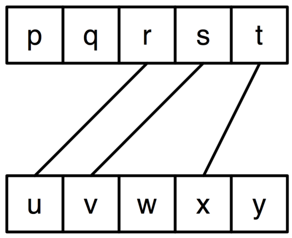



Figure 2.5: Two arrays illustrating stereo matching on a 1D search space [30].

Up to this point the epipolar geometry and the challenge with stereo matching were introduced. The upcoming section defines disparity, illustrates the disparity map, and the depth calculation.

## 2.3 Disparity map between stereo images

In the last two sections epipolar geometry and the problem with stereo correspondence were introduced. In this section the focus is on the term disparity, how disparity can be visualized via disparity maps, and how the depth can be calculated out of those disparity values.

### Disparity

The disparity is the shift of a pixel / object (feature) between two or more images. An object may appear at position $(x_1, y_1)$ in the left one and at position $(x_2, y_2)$ in the right one. The disparity is the shift from the left position to the right one. With $P_i$ declaring a point, left or right side, the following represents the disparity for two points in a two-dimensional space utilizing the pythagorean theorem:

$$D(P_L, P_R) = \sqrt{D_X^2(P_L, P_R) + D_Y^2(P_L, P_R)} \tag{2.1}$$

Henceforth, as the assumption of rectified images was made, only the horizontal disparity $D_X$ is meant by the term 'disparity'.

$$D_X = |X_1 - X_2| \tag{2.2}$$

In other words, having a pixel $(x_1, y_1)$ in a reference image (left) $l$ and a pixel $(x_2, y_2)$ in our matching image (right) $r$ the correspondence is given by:

$$x_2 = x + |d(x_1, x_2)| \quad \text{with} \quad y_1 = y_2, \tag{2.3}$$

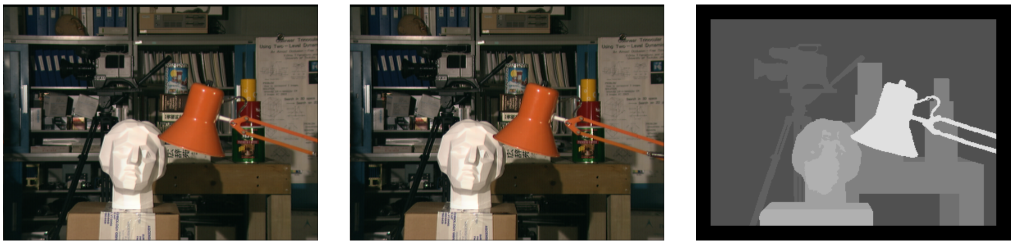
where $d(x, y)$ is the function which delivers values out of the disparity space $(x, y, d)$ computed by the algorithms.

Resulting in matching pixels from one image to another, the disparity for each pixel-wise combination is calculated as seen in the previous subsection (simplified stereo matching) and presented here. Such disparities can also be seen as the inversed distances to observed objects. As a matter of fact, at the border of each image some pixels can not be calculated caused by the non-existing counterpart for matching. Those pixels with no fellow are called 'occluded' pixels. For example, in some cases pixels are hidden in one image by an object due to the blocking line of sight of this object.

### Disparity map

In order to actually analyze the output of algorithms ground-truth data is necessary. An algorithm normally outputs a disparity map reflecting the disparity space $(x, y, d)$. This disparity map can be seen as matrix having the size of the original image ($m \times n$) and containing values ranging from $0$ to $d_{max} - 1$ utilizing one color channel (grayscale). The maximum disparity can be set via parameter for most of the algorithms and a feasible value which yields to sound results is $64$. For better visual analysis the disparity maps are usually normalized to values ranging from $0 - 255$ [13, 41, 47]. Figure 2.6 c) shows the ground-truth data representing the disparity map. The disparity map depicts grayscale intensities with lighter gray representing pixels / objects closer to the camera.

Tying in with the term ground-truth Martull et al. created the first "highly realistic CG dataset that properly models real-world imperfections, while providing accurate ground truth." [41]. Without such datasets bad evaluation of stereo matching algorithms can be made as there would been no reference to evaluate against. Figure 2.6 shows the previous dataset of the University of Tsukuba, the well-known *Head and lamp scene.*



(a) left input image (b) right input image (c) ground-truth data

Figure 2.6: Tsukuba benchmark stereo image pair of the University of Tsukuba [41].

The input for a perfect algorithm would be the reference image (a) and the matching image (b). After computation the result would be similar to the ground-truth data (c). With evaluation metrics the computed disparity map is then compared to the ground-truth data. Measuring instruments serve as a quality indicator for an algorithm's performance. The above example is given for basic knowledge and understandability of how a disparity map actually looks like. Details of this process are examined in the evaluation Chapter 5.

## Depth calculation

From an obtained disparity map and given camera parameters the depth can be calculated. The mathematical description of the following equations has been introduced by Cyganek and Siebert in Chapter 3.4.9 (Depth Resolution in Stereo Setups) of their book "*An introduction to 3D computer vision techniques and algorithms*" [13]. Assuming the focal length of the camera's lens and the baseline[11] are known, the following holds:

$$Z = \frac{f \cdot B}{d} \quad \text{and} \quad d = \frac{f \cdot B}{Z} \tag{2.4}$$

$$X = \frac{x \cdot Z}{f} \quad \text{and} \quad Y = \frac{y \cdot Z}{f} \tag{2.5}$$

where:

- $Z$ is the distance along the z-axis (camera axis),
- $f$ is the focal length,
- $B$ is the baseline (in meters),

[11]The distance between both image sources.

- $d$ is the disparity of the point.

After $Z$ is determined, $X$ and $Y$ can be calculated using the usual projective camera equations (2.4-2.5) where the point $(x, y)$ is the pixel location in the 2D reference image and $(X, Y, Z)$ describes the real 3D position [4, 13, 41, 47]. Figure 2.7 depicts the depth calculation from disparity with $X$ being the estimated point and $d = x - x'$.

The following subsection describes the more general steps of how disparity algorithms work, known as the taxonomy.

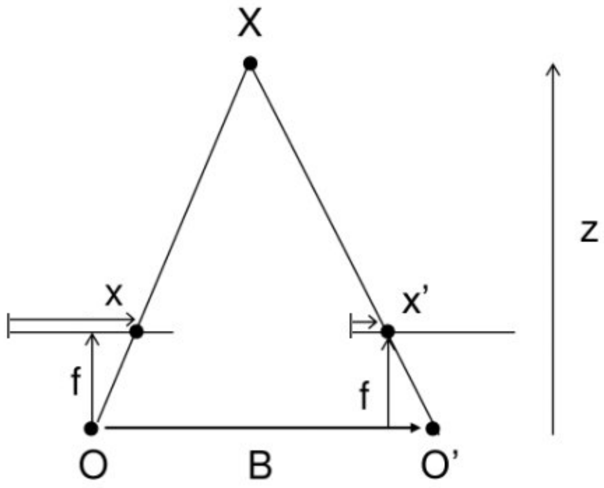



Figure 2.7: Depict of depth calculation from disparity.[12]

[12]Source (accessed 02/2016): `http://docs.opencv.org`.

## 2.4 Disparity algorithms

In the last sections different aspects affecting stereo matching were introduced. To get a better understandability of the algorithm's technique, this section focuses on the diversity and taxonomy of disparity algorithms.

### Diversity of disparity algorithms

A lot of different algorithms exist and their workings differ slightly. According to Cyganek and Siebert [13], Scharstein and Szeliski [47], the following categories to separate disparity algorithms exist. Some of these classifications are discussed in more detail the related work Chapter 3.

First, the output of an algorithm is rated: they can create sparse or dense disparity maps. On the one hand, most of the algorithms produce a dense disparity map meaning that almost every pixel got a corresponding shift value. On the other hand, sparse algorithms only calculate values around, for instance, feature points (cf. feature matching). One advantage of sparse algorithms compared to dense disparity algorithms is that they are normally faster in computation but limited in applications. Approaches to interpolate sparse disparity maps into dense disparity maps exist, but they tend to produce inaccurate results in comparison to dense algorithms.

Second, Cyganek and Siebert [13] categorize direct and indirect methods. Indirect methods are feature based or operate in the transformed image space (cf. Chapter 6.3.7 in [13]). Direct methods use intensity based measures.

Finally, disparity algorithms are classified into local and global methods:

- Local Methods
  - Feature matching
  - Block (area) matching
- Global Methods
  - Belief propagation
  - Graph cuts
  - Dynamic programming
  - Layering (hierarchical scale-space)

## Taxonomy of disparity algorithms

Assumptions need to be made before starting to describe the taxonomy of disparity algorithms:

1. The algorithm is fed with a pair of rectified images as input.
2. The algorithm produces a dense integer disparity map, which means that disparity is estimated at each pixel.
3. Most of the current algorithms works according to the following steps (see Figure 2.8)

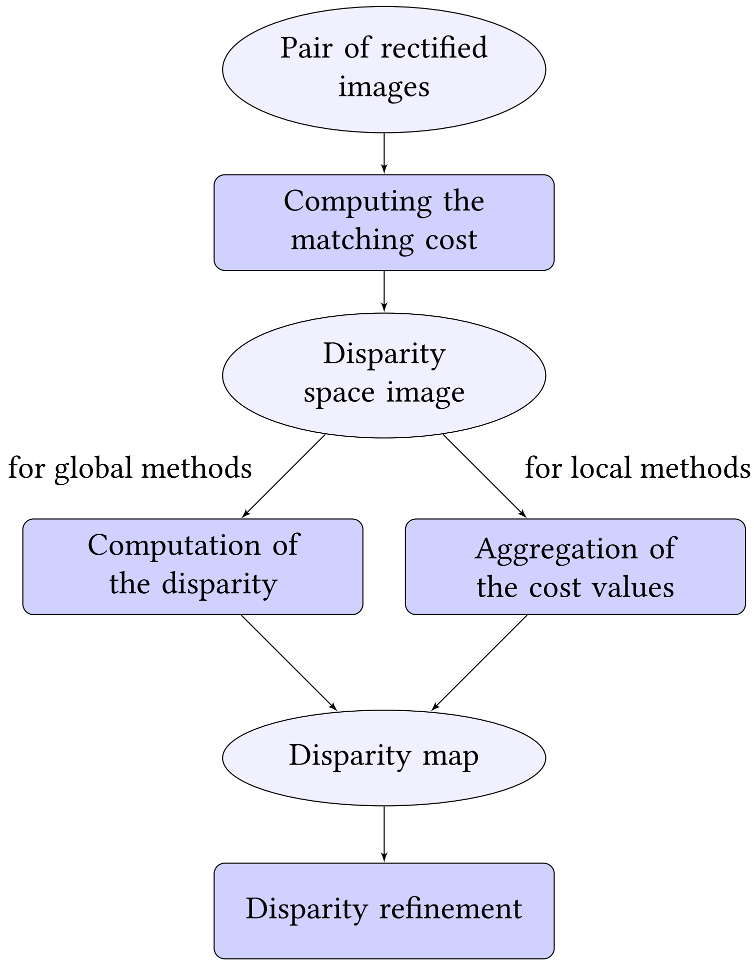



Figure 2.8: Basic processing flow of typical disparity algorithms, cf. [13, 47].

The upcoming subsections discuss the steps in more detail, especially regarding the computation of the matching cost and the subsequent aggregation.

### Matching cost functions

At first, the similarities of pixels in both images are calculated. In general, the literature shows matching cost as the dissimilarities of pixels. The matching cost needs to be computed for the decision which pixel belongs to another. Hence, the cost needs to be low for similar pixels. Some of the matching criteria used for determining the matching cost are described in Table 2.1 cf. [7, 13, 21, 31, 47].

| Method | Formula |
|---|---|
| Sum of absolute differences | $\sum_{i,j \in U} \lvert I_1(x_L + i, y_L + j) - I_2(x_R + i, y_R + j) \rvert$ |
| Sum of squared differences | $\sum_{i,j \in U} (I_1(x_L + i, y_L + j) - I_2(x_R + i, y_R + j))^2$ |
| Normalized cross-correlation | $\frac{1}{n} \sum_{x,y} \frac{(I_1(x_L, y_L) - \overline{I_1})(I_2(x_R, y_R) - \overline{I_2})}{\sigma_{I_1} \sigma_{I_2}}$ |

Table 2.1: Most common similarity measures

$I_k(x, y)$ stands for an intensity value of the image $k$ at the point with given coordinates $(x, y)$. The set $U = U(i, j)$ describes close-by points located around the point $(i, j)$. The sum of absolute differences (SAD) similarity measure is one of the simplest ones and describes the difference between pixel values. The absolute intensity differences of both images $I_1$ and $I_2$ are summed up for all adjacent pixels in the neighborhood (described with $U$). Zero stands for the equality of both regions. In optimal images nearly every pixel in the left image should have a corresponding pixel in the right image, fulfilling the constraints from the section before, and thus the calculated SAD should sum up to zero. The lower the result, the more similar the pixels and the cheaper the matching cost are.

In the sum of squared differences (SSD) similarity measure the pixel differences are squared and summed up. This measurement needs a bit more computational power and is usually chosen to discriminate high differences. It can yield to better results if outliers need to be excluded and the difference is not strong enough while using SAD.

There also exist the normalized cross-correlation (NCC). Cross-correlation measures the correlation between two intensity values in a point $(x, y)$. The normalized cross-correlation subtracts the mean $\overline{I}$ of the intensities and divides by the standard deviation $\sigma_I$ to normalize the intensity values. This may be necessary to balance brightness variations. NCC is excluded in most scientific investigations regarding disparity algorithms as it behaves similar to SSD (cf. [13, 27, 47]).

## Disparity space image

Related to the disparity space introduced in the section before, the disparity space image (DSI) should be defined. The DSI is an image or a function over a continuous or discretized version of the disparity space $(x, y, d)$ and represents the matching cost (i.e. the dissimilarity) of a given $d(x, y)$. It can be imagined as a three-dimensional matrix with the x-axis meaning the column, the y-axis the disparity and each combination the matching cost for that particular value as the z-axis. The disparity space image $C(x, y, d)$ is the result of the matching cost values over all pixels and all disparities, where the function $C$ that denotes the matching cost for the given input parameter. This leads to the aggregation step, during which the matching cost form the final disparity for local methods.

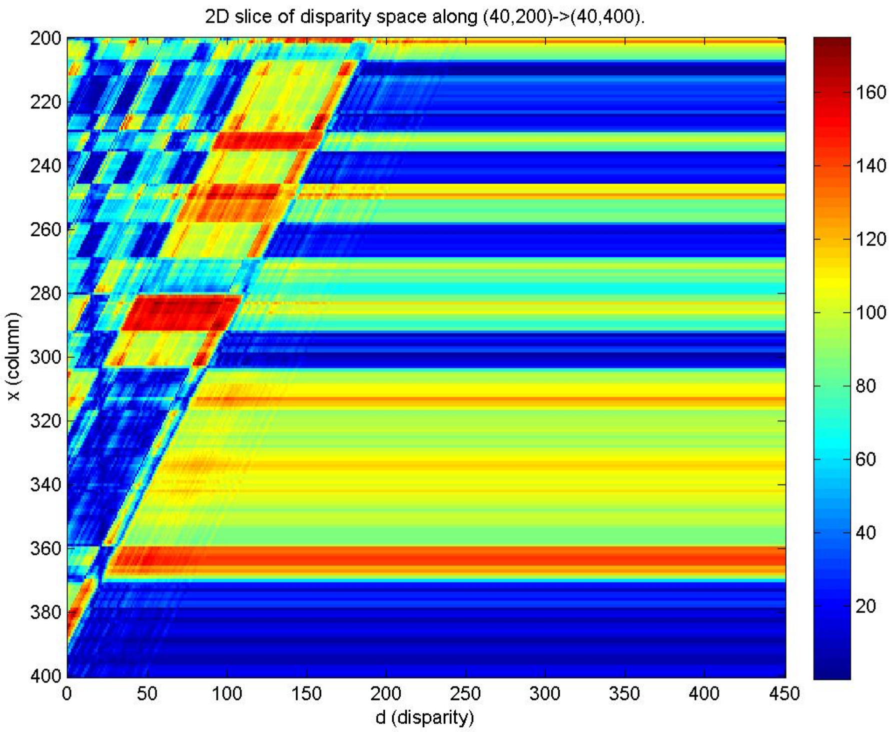



Figure 2.9: Illustration of a disparity space image.[13]

[13]Source (accessed 03/2016): `http://www.cs.virginia.edu/~cab6fh/CV_4/WRITEUP.html`.

### Aggregation

In the aggregation step the decision has to be made, which discrete set of disparities represents the scene best [47]. As the matching cost values over all pixels and all disparities are stored in the DSI the minimum for each row is chosen as the best matching pixel and thus declared as the corresponding pixel. In other words: for every pixel the disparity with the lowest cost is selected. This strategy is known as the winner takes it all (WTA). [13, 47]. As the pixel with the lowest cost is chosen, the following holds:

$$d(x, y) = \arg\min_{d'} C(x, y, d'). \tag{2.6}$$

### Disparity computation

After the aggregation, the actual disparity is computed in this step. It is split up for the two different methods: local and global.

**(i) Local methods.** Local methods focus on the matching cost computation and the cost aggregation steps. The final disparity computation is trivial as the minimum cost value (least dissimilarities) over each row is chosen (WTA).

**(ii) Global methods.** In contrast to local methods, global methods unify the three basic steps into a single one by defining an energy function to be minimized. It ties in with the labelling problem [53]. A row in the DSI can be imagined as the different labels (i.e. disparity values) one pixel can receive. The labelling problem describes the search of the disparity as the choice of the correct label. Each pixel should only have one label assigned in the end.

Let $P$ be a set of pixels and $D$ a set of disparities. The energy function aims to find a disparity $d$ which minimizes some energy:

$$E(d) = E_{data}(d) + \lambda E_{smooth}(d). \tag{2.7}$$

The data term $E_{data}(d)$ defines the matching cost for a given disparity function $d$ and expresses how well the disparity function $d$ matches with the input image pair. $C$ is the matching cost DSI:

$$E_{data}(d) = \sum_{(x,y)} C(x, y, d(x, y)). \tag{2.8}$$

As each pixel should be matched to a good find in the other image but simultaneously the adjacent pixels should be normally piecewise smooth, i.e. about the same value / intensity, the smoothness term $E_{smooth}(d)$ is introduced to reflect that (cf. stereo correspondence constraints). The $\lambda$ is introduced to control how much the smoothness term should influence the overall data term. To make the smoothness term computationally affordable it is, depending on the algorithm, usually restricted on the differences between adjacent pixel disparities [13, 47], i.e. the disparity gradient:

$$E_{smooth}(d) = \sum_{(x,y)} p(d(x,y) - d(x+1,y)) + p(d(x,y) - d(x,y+1)), \quad (2.9)$$

where $p$ is a "monotonically increasing function of disparity difference" [47]. Depending on the used algorithm, other smoothness term functions exist. The optimization problem to solve is defined as the minimization of the energy function, i.e.:

$$D = \arg\min_d E(d), \quad (2.10)$$

where $D$ is the disparity map containing the final values for every $(x, y)$ and $d$ a set of parameters or a disparity function affecting the energy value.

The search space for finding a solution is large, as an $n \times m$ image with $k$ disparities has about $k^{n \times m}$ possible solutions. According to Scharstein and Szeliski [47], Cyganek and Siebert [13] finding the global minimum is *NP-hard*. The related work in Chapter 3 gives an introduction into solving those optimization problems.

#### Disparity refinement

Disparity refinement can be seen as an optional post-processing step some algorithms perform automatically or may be requested manually. Refinement steps can also be implemented independently from the algorithm as they are executed on the final disparity maps. Sometimes the literature mentions those as clean-up steps. Here is a list of some known refinement steps:

- Sub-pixel estimation for higher accuracy.
- Disparity verification with left-to-right and right-to-left disparity map comparison (can also detect occluded areas).
- Filtering of disparity values, for instance using a median filter to remove mismatches.
- Interpolation of missing values: can be necessary when using an algorithm which produces a sparse disparity map.

### Simplified block matching

For demonstration purpose of a working local disparity algorithm, block matching, also known as area matching, is sketched in a simplified version. The following algorithm assumes rectified images. Thus, the algorithm is executed along the scanlines.

1. Divide the images in blocks of the size $m \times n$ (e.g. $8 \times 8$).
2. Find the corresponding block along the scanlines as shown in Figure 2.10, i.e. the block with the lowest matching cost (e.g. sum of absolute differences).
3. Calculate for this block the displacement (the shift from left to right image) which results in the disparity.
4. This yields in the final, ideally *bijective*[14] disparity map after finding the corresponding block from the left to the right image and vice versa. If a block could not be matched the bijective criteria is not fulfilled.

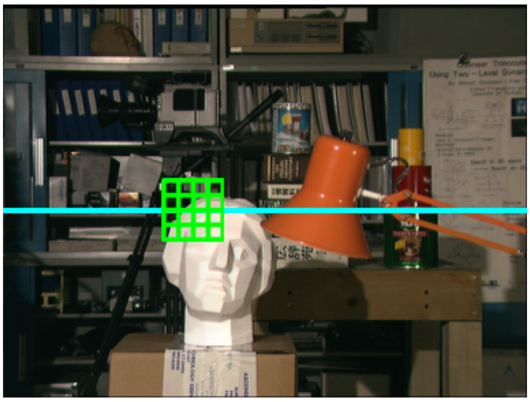


Figure 2.10: Illustration of block matching along a scanline.

In general, block matching leads to more accurate results with smaller window sizes. A bigger window leads to more smoothing which results in lower noise. The window size depends on the image size and its content. Therefore, no general assumption can be made. For each scenery the window size should be adjusted individually.

[14] Considering two sets, for each element of the first set a corresponding element of the second set is found. It also holds that both sets contain the same amount of elements. Thus, it is a one-to-one correspondence which also works inversely.

## 2.5 Sub-pixel accuracy

As seen in the sections before, the disparity algorithms produce a disparity map consisting of integer values only. For most of the imaginable applications integer values should be enough. However, the world is continuous and there are applications which rely on accurate disparity estimations. For instance, having no sub-pixel values, image-based rendering produces an image for visualizing the disparity map, which can appear to be made up of many thin shearing layers [47]. To get an accurate sub-pixel value, the most common technique is to use curve fitting by utilizing an *n-th* polynomial-order function. In this particular case [47], a second-order polynomial function, i.e. a parabola is used. The curve is fitted around three or more values of the matching measure. The point of interest lies in the center of the chosen window (as Figure 2.11 depicts). The minimum of this parabola is the searched value [13, 48].

Curve fitting with a second-order-polynomial in Figure 2.11 works with three data points: $(d_{i-1}, m_{i-1})$, $(d_i, m_i)$ and $(d_{i+1}, m_{i+1})$. $d_i$ is the found integer value for disparity and $m_i$ is a match value for the displacement $d_i$. With the curve fit a new minimum value $d_x$ is found which no longer needs to lie on the integer grid.

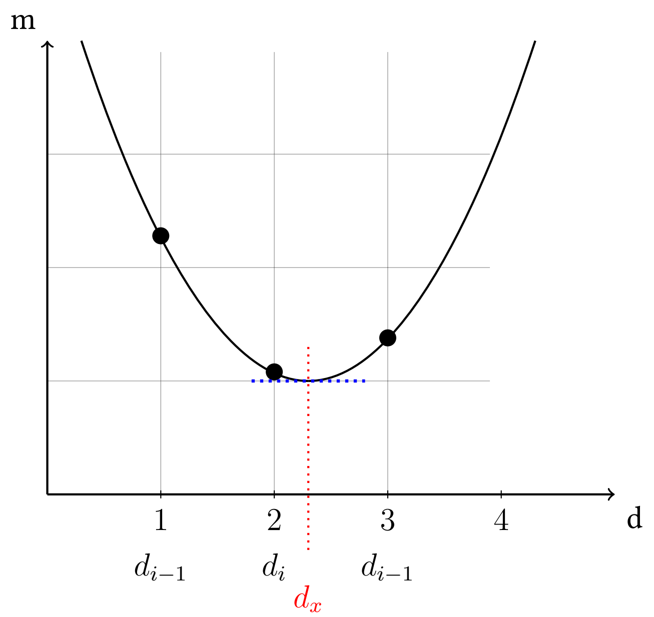



Figure 2.11: Sub pixel estimation of a disparity value around adjacent pixels.

## 2.6 Optical flow

Similar to the problems discussed in the section before, optical flow is also an image matching problem. The optical flow is defined as vectors describing small local displacements like moving objects or camera motion between two consecutive frames [7, 13]. The principle of the matching problem of images is comparable to disparity algorithms. The main difference is that instead of analyzing left and right image, a scene is investigated and the disparity describes small local vectors. To be more precise, the optical flow relies on the assumption that a certain point $(x_1, y_1)$ in a frame at time $t_1$ will be matched to a point $(x_2, y_2)$ in a frame at time $t_2$. Different approaches for estimating the optical flow of pixels exist, like:

- Correlation or block-matching,
- feature tracking,
- energy-based methods, or
- gradient-based methods.

Optical flow is heavily used in autonomous driving, automated traffic surveillance systems and video compression like H.264 [11, 35, 42, 45]. Recently a dataset containing ground-truth data of real-world sceneries regarding optical-flow information was released by Kondermann et al.. Figure 2.12 shows an example of estimating the movement of a vehicle. In the left image of Figure 2.12 most of the vectors are null as no local displacement can be estimated. Only a few vectors (small white dots) near the vehicle illustrate the displacements.

Figure 2.12: Optical flow estimation to obtain motion vectors (left) and pixel velocity magnitudes (right).[15]

[15]Source (accessed 02/2016): `http://de.mathworks.com/discovery/optical-flow.html`.

# 3 Related work

In this chapter the related work regarding disparity algorithms is treated. As integration of some disparity algorithms for the later evaluation was part of this thesis, the ones which were actually implemented are examined in more detail. The well-known semi-global matcher by Hirschmüller, also implemented in the OpenCV library [7], is introduced. OpenCV[1] is an extensive image processing framework, with the main goal towards real-time computer vision. Geiger et al. introduce an approach that enables fast matching of high-resolution images, which is also outlined in the upcoming section. Both approaches utilize local methods for estimating disparity maps. One candidate adopting global methods is the Middlebury MRF library, which is also introduced. It implies solving optimization problems (i.e. the minimization of a global energy cost function). The library's implemented methods to solve such optimization problems are outlined in greater detail. In the end, an outlook on disparity algorithms on stereoscopic videos is given, which includes an approach towards spatiotemporal consistency and remapping of the disparity range.

## 3.1 Semi-global matching

Hirschmüller combines two different methods, global- and local-matching for determining accurate disparity at a lower runtime as other global algorithms, which are time consuming even on current hardware [24, 25].

The semi-global matching (SGM) method utilizes pixel-wise matching of so called mutual information (MI) via entropy $H$. The joint entropy of two images $I_1$ and $I_2$ results from the sum of their combined entropy and a global two-dimensional smoothness constraint $H_{I_1,I_2}$ which leads to the following cost:

$$MI_{I_1,I_2} = H_{I_1} + H_{I_2} + H_{I_1,I_2}. \tag{3.1}$$

The discussed one-dimensional constraints from Chapter 2 are applied as well. Calculating the matching cost based on mutual information is insensitive to different video recording conditions and illumination changes [24, 55]. The joint

[1] `http://opencv.org`

entropy $H_{I_1,I_2}$ is low (meaning low information content) for rectified images as one image can be predicted by the other. The MI matching cost is defined as the following:

$$mi_{I_1,I_2}(i,k) = h_{I_1}(i) + h_{I_2}(k) - h_{I_1,I_2}(i,k), \tag{3.2}$$

where $h_1$ and $h_2$ are calculated from the probability distribution of corresponding intensities. Thus, $h_{I_1,I_2}(i,k)$ serves as the matching cost for the two intensities $i$ and $k$. The idea then is, that one image needs to be warped[2] such that corresponding pixels are at the same location in both stereo images:

$$I_1 = I_b \quad \text{and} \quad I_2 = f_D(I_m), \tag{3.3}$$

where $I_b$ is the base image, $I_m$ the match image and $f_D(x)$ is a function which outputs the matching corresponding point. As the matching cost represent the information content of two intensities $I_1$ and $I_2$, which should be low (i.e. as equal as possible), the disparity map $D$ needs to be known *a priori* for warping. Hence, the MI matching cost needs to be calculated either iteratively or hierarchically. On the one hand, an iteratively approach utilizes a random disparity image for calculating the MI matching cost, which serves as the base for the next iterations. On the other hand, the MI matching cost can be calculated hierarchically by recursively using an up-scaled disparity image, which has been calculated at half resolution with a common similarity measurement like SAD. For a deeper explanation of how the mutual information are exactly calculated and used in the SGM method compare [24–27].

## OpenCV BM and SGBM

The OpenCV library [7], currently at version 3.1.0, offers two implementations for disparity estimation, block matching and semi-global block matching based on the idea of Hirschmüller. This version also contains a new filter, which was initially introduced with version 3.0.0, named *Disparity WLS Filter*[3]. WLS stands for weighted least squares (in the form of a fast global smoother). This disparity filter smoothes the disparity and also performs a left-right-consistency check to refine the results in especially half-occluded and uniform areas [43]. This yields to better and more accurate results but has the drawback of loosing negative disparity values. Negative disparity appears if the stereo cameras are verged or inclined

[2]In this context warping can be seen as a function which maps pixels from the destination image to pixels in the original image. Then the pixels are copied at the mapped position to the coordinates in the destination image.

[3]`http://docs.opencv.org/3.1.0/d9/d51/classcv_1_1ximgproc_1_1DisparityWLSFilter.html`

towards each other. The WLS filtering results in disparity ranging from 0 to $D_{max}$, which is set beforehand as a parameter. Thus the negative disparity is $-1$.

## 3.2 ELAS: Efficient large-scale stereo matching

Geiger et al. propose a novel approach for estimating the disparity with so called support points [19, 20]. A support point is like a feature, a point which can be robustly matched. For those support points, a sparse disparity map is calculated. For more robustness, only the support points which can be matched left-to-right and right-to-left are retained. To remove ambiguities, the ratio between the best and the second best match of all points is taken into account. If the ratio exceeds a fixed threshold, the points are removed. A support point which has a different disparity value than all its neighbor (adjacent) points, is categorized as an outlier and removed as well. As the found support points may not cover the whole image, additional support points in the image corners are added. They adopt the disparity value of their nearest neighbor. Then, image coordinates of the remaining support points are used to create a 2D mesh via Delaunay triangulation. To obtain a dense disparity map missing disparities are interpolated using mesh of the Delaunay triangulation by using the nearest-neighbor on the same image line. For more information how the support points get calculated and how the interpolation is done exactly, compare [19, 20].

## 3.3 Middlebury MRF library

The Middlebury MRF library [48, 52] utilizes a global energy function consisting of Markov random fields to formulate an energy minimization problem and offers the following methods to solve this optimization problem:

1. iterated conditional modes (ICM),
2. graph cuts expansion approach (cf. [5, 6, 34]),
3. graph cuts swap approach (cf. [5, 6, 34]),
4. sequential tree-reweighted max-product message passing (TRWS) (cf. [33, 56]),
5. sequential belief propagation (BPS) (cf. [6]),
6. max-product belief propagation (BPM) (cf. [6]).

The following subsections give a rough overview on some of those methods. Additionally, a short introduction into MRF-based energy functions is given. Also, the generalized concepts of how the above techniques help to solve such optimization problems are outlined.

### 3.3.1 Solving optimization problems

Many problems in computer vision, for instance image smoothing, can be described in terms of energy minimization. The stereo correspondence problem is formulated in such a way as described in Chapter 2. Thus, solving of optimization problems is a key part in modern stereo matcher algorithms. They solve the labelling problem as described in Chapter 2. Most of the current disparity algorithms use global methods to solve an energy minimization problem. Usually they utilize Markov random fields (MRF) based energy functions. As such MRF based energy functions are *NP-hard* approximation algorithms like the following are typically used [13, 54]:

- dynamic programming,
- belief propagation,
- graph cuts.

All of these methods have in common that they are supposed to solve so called inference problems, or at least provide approximated solutions. Markov random fields, mentioned before, are also known as Markov network. Bayesian networks as well as Markov networks are so called graphical models. Such graphical models help to understand the reasoning behind those formulations and to actually build algorithms which solve those inference problems. Both networks express the dependencies of nodes as the conditional probability. A chain of nodes is called the joint probability[4]. This then is the product over-all probabilities. The goal of algorithms which solve inference problem is to compute certain marginal probabilities[5], i.e. the probability that some pixel reach a specific label node [13]. With inference the computation of these marginal probabilities is meant. Marginal probabilities are defined as the sums over all possible states of all the other nodes in the system. They are also called beliefs [59].

[4]The joint probability $P(A \wedge B)$ is the probability of event A and event B occurring. It is the probability of the intersection of two or more events.

[5]The marginal probability is an unconditional probability as it is not conditioned on another event.

### Markov random fields

Markov random fields (MRF), also called Markov network, are used to formulate problems in a probabilistic way represented as an undirected graph consisting of random variables. For a simple undirected graph compare Figure 3.1.

A MRF is a graph $G = (V, E)$ where $V = 1, 2, ...N$ denotes a set of vertices or nodes. Each node is associated with a random variable $u_j$ for $j = 1...N$. $E$ describes the edges $(i, j) \in E$ between the nodes $i$ and $j$. The neighborhood of a node $i$ is the set of nodes to which the node $i$ is adjacent, i.e. $j \in N$ if and only if $(i, j) \in E$. The neighborhood of a node $i$ is denoted as $N_i$.

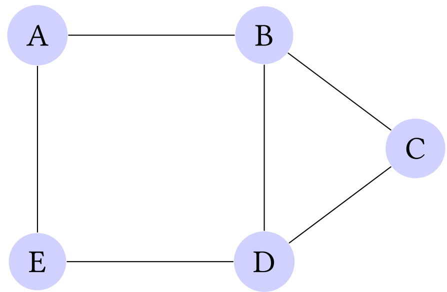



Figure 3.1: Simple undirected unweighted graph

The Markov random field satisfies:

$$P(u_i \mid \{u_j\}_{j \in VN}) = P(u_i \mid \{u_j\}_{j \in N_i}), \tag{3.4}$$

where $N_i$ is the so called Markov blanket of node $i$. It describes that the graph should be conditionally independent of all of the other variables given its neighbors. A hop from one node to another can be seen as a chain of probabilities which have to occur, also called Markov chain. The main idea behind MRF in combination with computer vision problems is to formulate the labelling problem in such a way, that each pixel has a likelihood to belong to a certain label [53]. The core problem is to find exactly one label for each pixel, which is represented as a node in a MRF. This label represents the optimal solution to an underlying problem, in the case of stereo correspondence: the disparity of a pixel regarding a reference pixel [13].

Contrary to MRF, also Bayesian networks exist. A Bayesian network is a directed graph whereby MRF is undirected. This implies an important aspect: the direction of a certain probability to hop from one node to another. Whereby MRF can not represent induced and non-transitive dependencies. Two independent random variables may be connected by an edge because of possible dependencies.

Bayesian networks overcome these limitations.

The underlying stereo model of the Middlebury MRF library is based on the research of Sun et al.. They model stereo matching by three coupled MRF [51]:

- $D$ as the smooth disparity field,
- $L$ for representing depth-discontinuities,
- $O$ is a spatial binary state for handling occlusions.

Figure 3.2 depicts the relationship between $D$, $L$ and $O$. The conditional probability[6] over $D$, $L$ and $O$ given a pair of stereo images $I = I_L, I_R$ is defined as:

$$P(D, L, O|I) = \frac{P(I|D, L, O)P(D, L, O)}{P(I)}. \tag{3.5}$$

They then approximate inference via belief propagation over this equation. For a deeper dive into this topic compare [6, 13, 33, 51, 53, 56, 59].

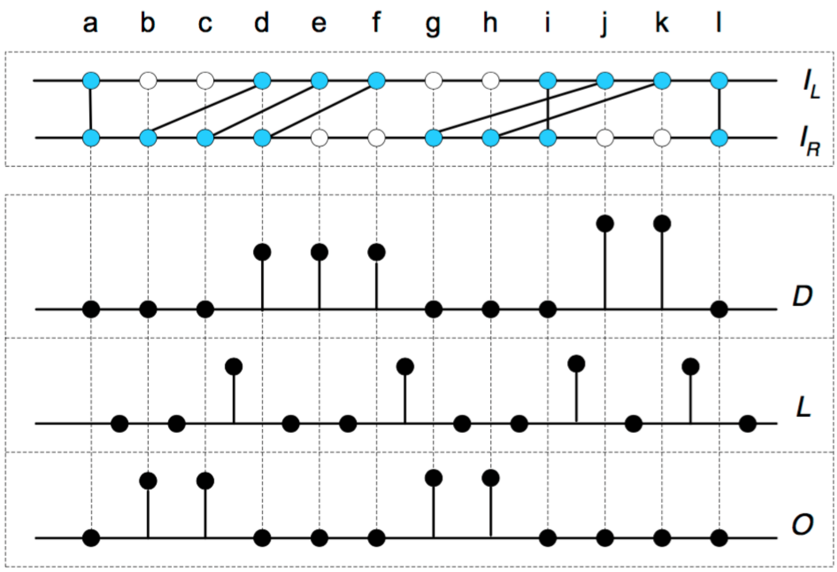



Figure 3.2: Stereo matching model by three coupled MRF's [51].

[6]Bayes' theorem: $P(A|B)$, a conditional probability, is the probability of event A occurring, given that event B occurs. $P(A|B) = \frac{P(B|A)P(A)}{P(B)}$ where $P(A)$ and $P(B)$ are the marginal probabilities of event $A$ and $B$. $P(B|A)$ is the probability of observing event B given that A is true.

### Factor graph

As those problems are *NP-hard*, several approximation algorithms exist which are outlined in the following subsections. All of these approximation algorithms work on factor graphs. A factor graph represents a factorized function of several variables. Usually there exist two types of nodes in factor graphs, squared and circled ones. Circled ones represent variables of a factor and a squared one represents a factor. Factors define the relationship between variables in the graph as they are obtained by the factorization of the function. Such graphs are bipartite, that means that the nodes of a graph can be divided into two disjoint sets, for instance $U$ and $V$, such that every edge connects a node $U$ to one in $V$. These factor graphs help to understand the underlying problem and to imagine the implementation of such algorithms as they are used for breaking down a problem into pieces.

A MRF function is factorized in partial functions and then formulated as a factor graph. The solution to the problem represented by the factor graph is then approximated. An important notion of factor graphs is the message which can be passed from a node to another. So the edges represent communication channels through which messages can be passed. One way to approximate such a factorized function is the use of message passing algorithms (also called belief propagation), which are described later on. There exist one exception: if the factor graph contains no cycles, meaning it can be represented as a tree, the solution can be computed exactly.

### Dynamic programming

In general, dynamic programming means means dividing an optimization problem into smaller chunks. These chunks get solved individually and in the end, they are connected and the optimization problem is minimized [1, 2, 13]. For stereo matching this applies to the partition of a two-dimensional search problem into a series of isolated one-dimensional search problems on each pair of epipolar lines. These problems are then solved independently. With dynamic programming the following energy function (introduced in the foundations Chapter 2) can be solved independently per scanline.

$$E(d) = E_{data}(d) + \lambda E_{smooth}(d) \tag{3.6}$$

Dynamic programming has two benefits, it is solvable efficiently in polynomial time and it enforces the ordering constraint (as it is solved per scanline). But it can lead to streaking effects, meaning that the result image seems to be constructed of many independent layers.

### Belief propagation

Belief propagation (BP) in general is a technique to perform inference on a probabilistic model like Bayesian networks or Markov random fields [13, 54, 59]. As mentioned before, Sun et al. presented a stereo model for belief propagation. BP works with messages which are passed from one node to another. This is the reason why BP is also known as the message-passing algorithm. The nodes exchange information about probabilities. In the case of stereo matching, the message contains the probability that the receiver node (a node in MRF) should hold a disparity which is consistent with all information already passed to it by a sender. The nodes are partitioned into low- and high-confidence ones. The messages also carry a property, the entropy. The entropy is high when sending from low- to high-confidence nodes and vice versa (cf. [13, 54, 59]). The nodes calculate a new state after an iteration as they know more about the other node's properties, i.e. marginal probabilities of distant and not directly connected nodes. Also the outcome of past iterations, which yields in joint- and conditional probabilities, influences the overall state of a node. When talking about disparity algorithms, the algorithm ends in a tree if no state is changed anymore and the exact energy can be inferred. In an acyclic graph the algorithm finishes if the overall energy does not improve anymore.

### Graph cuts

For approximating problems formulated in a Markov random field there also exist graph cuts algorithms [6, 13]. In general, graph cuts assume a graph $G$ with a set of nodes $N$ and connected by a set of edges $E$. The goal is to delete enough edges so that each pixel is connected to exactly one label node. Given a weighted graph $G$ with source $s$ and sink $t$ nodes. The graph should be partitioned into two sets, $S$ and $T$, where $s \in S$ and $t \in T$. This set presents a subgraph such that the sum of edge weights spanning this partition is minimized.

In the case of computer vision, graph cuts are inspired by the combinatorial optimization methods for maximum flow [12, 13]. Two basic variations of the maximum flow problem exist, called $\alpha$-$\beta$-swap and $\alpha$-expansion. Initially, three labels exist: $\alpha$, $\beta$ and $\lambda$. Normally, one step would be to change the label of a pixel, calculate the energy again and then infer if the change was good or not, depending on the delta. For instance one pixel labelled with $\lambda$ would then be $\beta$. The $\alpha$-$\beta$-swap algorithm interchanges whole areas of $\alpha$ with $\beta$ whereby areas of $\lambda$ remain unchanged. In an $\alpha$-expansion a huge number of pixels labelled $\beta$ and $\lambda$

are changed into $\alpha$. But in each of those methods the outcome is then measured. In Chapter 2 the following equation was introduced:

$$D = \arg\min_{d} E(d). \tag{3.7}$$

If the outcome of such a swap or expansion is better, meaning $E(D_{after}) < E(D_{before})$, the algorithm continues. If not, the algorithm stops. Thus, both algorithms are expected to be stopped after the first unsuccessful run (i.e. energy increases). The difficulty is to find the optimal swap move and the initial seed. Both is described in [6, 34, 50, 54].

## 3.4 Disparity algorithms on videos

Although stereo correspondence is a research field which has been heavily investigated for a few decades, no real disparity algorithm for videos yet exists. One reason for that can be the lack of solid ground-truth data as only a few datasets has been introduced lately [8, 48]. Also the computational bottleneck of dealing with multi-dimensional data can be an issue, for instance adding a new dimension to the disparity space image which reflects the relationship between multiple frames. As a video is defined by multiple consecutive frames, every disparity algorithm for images can also be applied on videos. The drawback of this trivial approach is the lack of taking the correlation of the frames into account. However, novel approaches were presented by Khoshabeh et al. [32], Lee et al. [38], Davis et al. [14], Richardt et al. [46], Hosni et al. [28], and are outlined.

### 3.4.1 Spatiotemporal consistency

The following approaches have in common that they want to deal with the occurrence of noise. On the one hand, noise can occur through estimating disparity. The disparity maps may vary from frame-to-frame which can create a flickering effect over time, that disturbs the human visual system [32]. This happens as the frames are observed separately and not as a consecutive entity. On the other hand, image or video sensors always produce little noise, although it may not be visible for humans. As a matter of fact, stereo images from real cameras will produce kindly different images due to a variety of reasons, for instance sensor response differences or luminance [13, 32]. Thus, they will not completely match each other which can also yield to noise in disparity maps. Another important factor for the occurrence of noise, which has not been investigated yet, is video compression. This idea is more explained in the implementation Chapter 4.

Khoshabeh et al. [32] present a two steps approach. First, the disparity maps are computed frame-by-frame. The computed disparity maps are treated as a space-time volume. Then they apply a video restoration algorithms to reduce noise in this space-time volume. This video restoration algorithm is based on the augmented Lagrangian method for total variation (TV) image restoration [9]. Basically, it is an algorithm for denoising images by keeping an eye on the frames before and after. Thus, object edges and depth-discontinuity areas are preserved. By applying this algorithm they benefit from three properties of the algorithm: variation regularization, spatial smoothness and temporal consistency, which are established at the same time. This leads to more accurate and spatiotemporal consistent maps. To simulate real scenery they added gaussian noise to rendered sequences, distributed as $\mathcal{N}(0, 20)$[7]. The outcome are better results when comparing bad pixels (threshold of 1) and visually clearly better disparity maps as depict in Figure 3.3. As this approach works on computed disparity maps, current image-based disparity algorithms can thereby be easily adapted to the video domain.

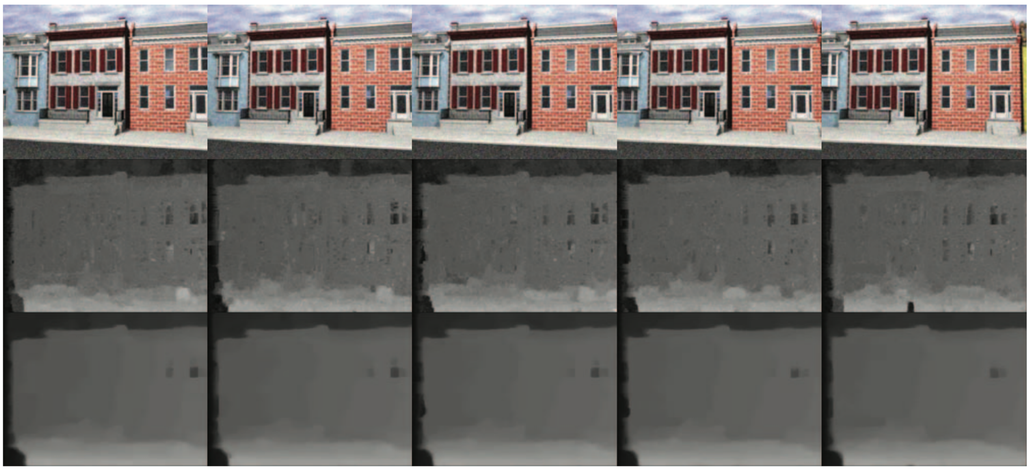


Figure 3.3: Spatiotemporal disparity refinement with the augmented Lagrangian method for TV [32]. Top: Original. Middle: Disparity. Bottom: Processed.

Yet another approach regarding video disparity estimation utilizing the spatiotemporal method above is presented by Lee et al. [38]. They focus on salient regions in the video. Motion is an important factor in video processing. There exist algorithms for estimating salient regions in videos. Generally, moving objects lean towards a higher degree of saliency. As typical disparity algorithms tend to have

[7]Normal (gaussian) distribution is denoted as $\mathcal{N}(\mu, \sigma^2)$, where $\mu$ is the mean and $\sigma^2$ the variance.

difficulties in estimating the disparity along moving edges and textureless areas, this can help to focus on especially those areas. They utilize motion cues[8] in combination with a modified census transform[9] with a noise buffer to obtain disparity maps. These disparity maps are more accurate and robust towards the edges of moving objects and in textureless areas. Finally, they also apply the previously introduced method to derive a spatiotemporal consistency.

The work of Richardt et al. [46], Hosni et al. [28] is build on the same principle, which has its origin in the basic approach of Davis et al. [14]. They define a matching cost function with an additional property $T$ for the time axis. A space-time cost volume is then generated by stacking the cost maps of input frames. A simple approach could be to smooth the disparity over time by applying a box filter after the disparity map is computed. This would imply that the disparities inside this space-time window are constant. As a result object borders may be over smoothed and get lost in a non-static scene. They overcome this issue by assuming that the disparity of an object is approximately constant over a small time window and applying weighted box filter. Therefore they build a 3D filter kernel, which weights the pixels. Pixels which belong to the same object get a high weight and pixels belonging to a different object a lower weight.

Tying in with this approach, Richardt et al. [46] rewrote the filter as a so called dual-cross-bilateral filter. Instead of using a custom weight model to preserve edges they utilize a bilateral filter which is a common edge-preserving smoothing technique. The cross-bilateral filter preserves those edges while smoothing with respect to a different image. A method for especially stereoscopic images is to use adaptive support weights for correspondence search (cf. [60]). This variant smoothes the cost space while preserving edges in both input images. This combined filter is then named the dual-cross-bilateral (DCB) filter. The implementation is called DBC grid. Spatiotemporal consistency is retained with the added temporal dimension $T$. Observing all frames as a whole is computational complex and difficult, because of multi-dimensional data, they consider 5 frames as one temporal entity. The DBC grid with this added temporal constraint is called temportal DBC grid. Their approach clearly visibly reduces errors as can be seen in Figure 3.4. The Figure shows a selected frame of a recorded 'skydiving' stereo video, from left-to-right: video frame (red-cyan anaglyph), DCB Grid, Temporal DBC Grid [46].

[8] Motion cues are responsible for the perception of motion.

[9] Census transform is basically an algorithm, implemented as a filter, for the classification of textures.

They present a real-time GPU-based implementation for competing with current state-of-the-art disparity algorithms regarding runtime. At this point in time their implementation is the fastest technique in the Middlebury[10] benchmark.

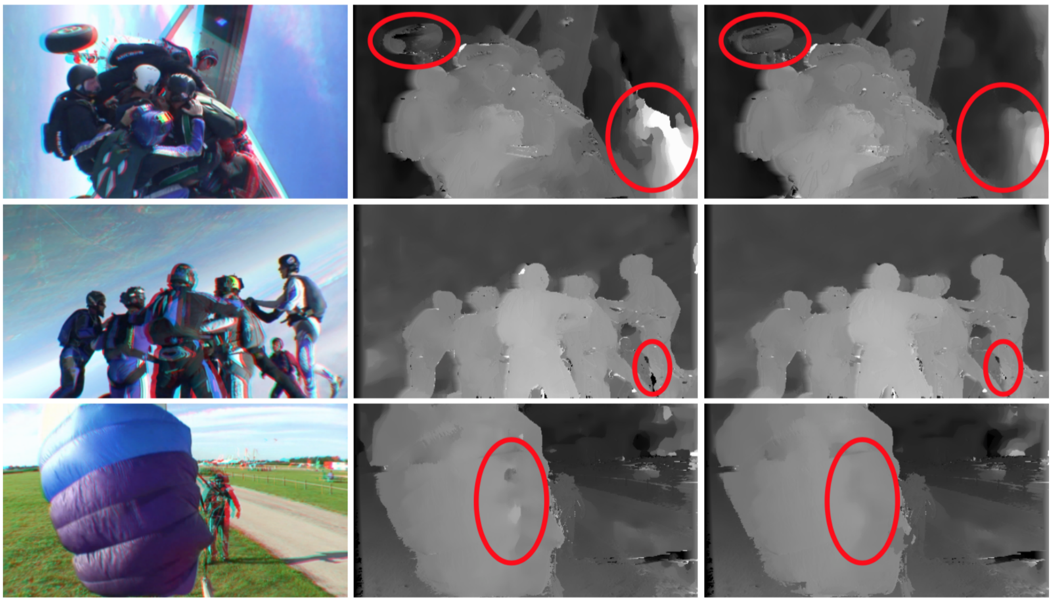


Figure 3.4: Disparity maps of a selected frame of the 'skydiving' stereo video [46].

### 3.4.2 Remapping the disparity range of stereoscopic videos

Lang et al. examine the problem of remapping the disparity range of stereoscopic images and video [37]. Remapping of disparity range can be necessary for various reasons. Humans notice the projected stereo content differently, depending on the screen size and the distance to the screen. Another issue is negative disparity. The fundamental underlying problem is the interplay of the human visual perception and restrictions of displays. For instance, displaying a close object on a distant screen may result in a negative disparity and then, humans may experience the viewing as uncomfortable. This can lead to temporary diplopia. Those issues are a real problem in the film industry regarding 3D movie productions. The disparity for the best human experience should lie in the so called comfort zone, which is the area where eyes feel comfortable. Too high positive disparity can lead to retina rivalry areas, which are muscular intense due to focus issues, whereas negative disparity can even result in painful retina rivalry areas. If 3D content is optimized for a cinema screen it will look differently on a home TV screen or even

[10] http://vision.middlebury.edu/stereo/eval3/

a tablet device, leading to a distinct viewing experience. This entails the need of changing the disparity after a stereoscopic movie was recorded for the adaption to the current viewing situation of the user. For this purpose, they introduce a set of basic disparity mapping operators for the control and the retargeting of the depth of stereoscopic videos. To actually use those operators stereoscopic warping of video streams is also presented. Basically, those disparity mapping operators define editing operations how the disparity can be modified. The goal is to map the disparity to a new range such that the resulting output view fulfills a stereoscopic, a temporal and a saliency constraint (cf. [37]), i.e. provide consistent disparity values according to the new range. These constraints are identical to related work on video retargeting [36]. Stereoscopic warping is image warping with the help of the introduced disparity mapping operators. The outcome of the paper are production-oriented rules and guidelines for editing disparity of stereoscopic content. In a survey, user concluded that the applied techniques, i.e. stereoscopic warping with disparity mapping operators, yield in a better viewing experience due to depth structure changes without distracting visual artifacts.

# 4 Implementation

In this chapter, the thoughts made beforehand are outlined. The following sections describe accurately the implementation, its components and the subsequent evaluation pipeline for disparity maps. Chapter 3 pointed out that no real algorithm for stereoscopic video disparity exist yet. For stereoscopic videos no evaluation engine yet exists. Datasets with high-resolution stereo videos are rarely spread. Source code for existing disparity algorithms are open-sourced and available for the public domain only in a few cases. Additionally, there exist a lot of different unaligned code for evaluation and comparing disparity maps. Thus, the decision towards a new implementation of an evaluation suite, built on top of OpenCV, was made. Found source code of disparity algorithms was refloated and integrated. Different masks for fine-grained evaluation were implemented as well. An image diminisher which alters stereo images by adding noise, to simulate real scenery, or artifacts from video compression. To round the evaluation suite up, a small web viewer to visualize the results was created.

## 4.1 Preliminaries

As development platform a MacBookPro was used with the following specifications: i5-4258U CPU @ 2.40GHz (dual-core), 8 GB RAM, a fast SSD. For the later evaluation phase a desktop computer with an i5-2500k @ 3.30GHz (quad-core) was considered. The programming part was done with Atom[1], a modern text editor, and CLion[2] from JetBrains, a cross-platform IDE especially for C++. CMake as a cross-compiling makefile generator was utilized. Everything except the web result viewer was implemented using C++. As a timer saver and for reducing code duplicates OpenCV as master library was used. The final build-chain consists of some shell scripts and CMake as makefile generator. With CMake it was possible to cross-compile the app for Linux and use a fast server-instance from DigitalOcean[3] for the generation of the disparity maps and to actually evaluate those. To not rely on different environments, a docker[4] image was created, open-sourced and

[1] https://atom.io
[2] https://www.jetbrains.com/clion/
[3] https://www.digitalocean.com
[4] https://github.com/benjohnde/dockerbase-opencv

used. Docker helps developers to build, ship and run distributed applications. A docker image serves as basis for containerized virtual environments. Those docker containers work in a chroot[5] environment and are isolated from other processes. It is not a complete virtualized machine as it runs on the system's kernel.

## 4.2 Overview

Initially, a rapid monolithic prototype was built featuring the execution of different disparity algorithms, the creation of bitmasks and the evaluation of a given scene with different parameters. But as more datasets were found and various bitmasks as an evaluation method were established the need for a leaner process chain arose. Especially as disparity algorithms need some time to compute the disparity map for one frame. Hence, as videos consist of multiple frames (in our datasets about 90 frames in mean) this is a time consuming task. Sometimes metrics change or a new metric is established, the threshold can be adjusted. These are the reasons for an approach towards microservices with which computed disparity maps can be evaluated repeatedly and independently. Thus, the monolith was rewritten partitioned in smaller microservices shaping three different components:

- disparity algorithm executer,
- mask creator,
- evaluation engine.

The figure 4.2 shows the composition and figure XXX the evaluation chain of these microservices. The output which each one of those microservices in the chain can generate or operate on is structured in a simple folder tree:

- /datasets/{dataset_identifier}/{dataset_sequence_identifier}/{appendix}

There {appendix} can be either {disparity_maps_computed}, {disparity_maps_-smoothed} or {bitmasks}.

The computed disparity maps are saved in a binary format since OpenCV is currently not able to use the OpenEXR file format properly (only reading is possible). Hence for visualization they have to be normalized in the range of 0-255 or may be presented as a heat map. The evaluation is done with simple python scripts reflecting the evaluation chain.

[5]Chroot stands for "change root" and helps to change the root directory for a current process.

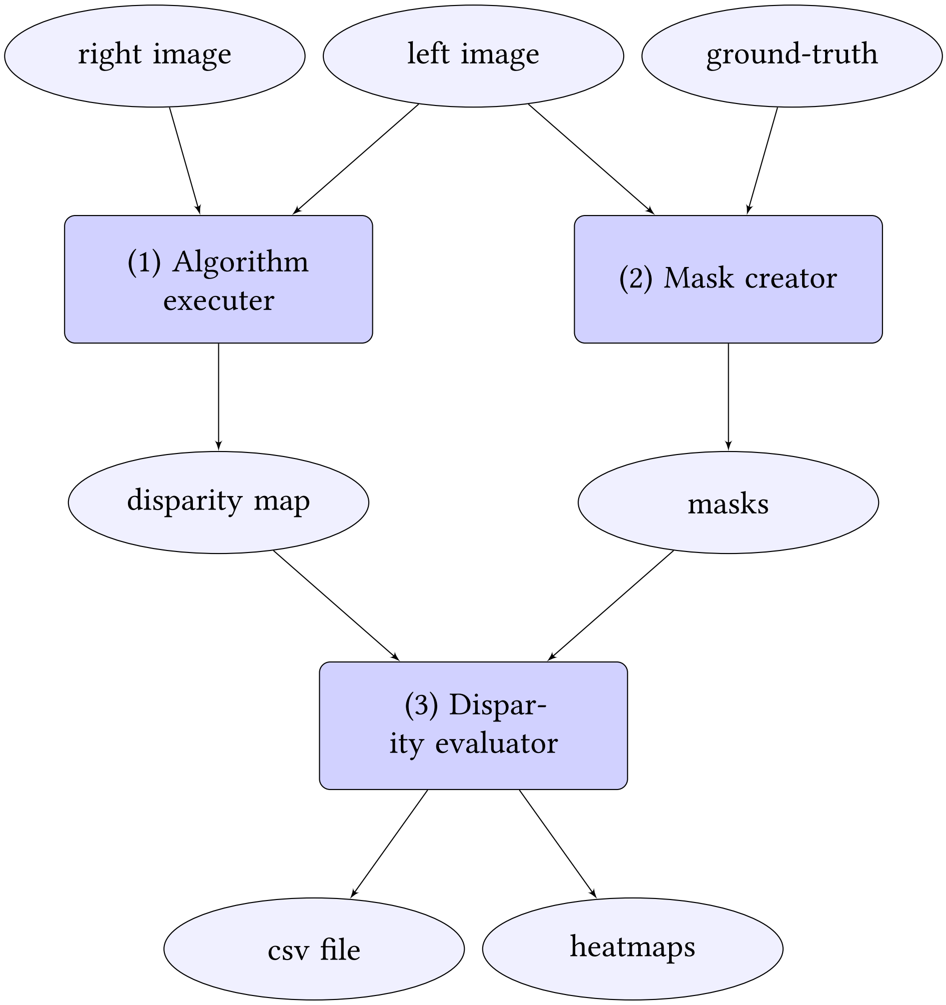



Figure 4.1: Processing pipeline of the implementation.

## 4.3 Evaluation engine for videos

At the current point in time, no real disparity algorithm for especially videos exists yet. As a video is defined by multiple consecutive frames, every disparity algorithm for images can be applied on videos. The drawback of this trivial approach is the lack of taking the correlation of the frames into account. None the less it is possible to focus on some other details, for instance:

- possible outliers in the sense of frames,
- mean performance (error rate) of those algorithms on a complete scene,
- runtime variety in a sequence,

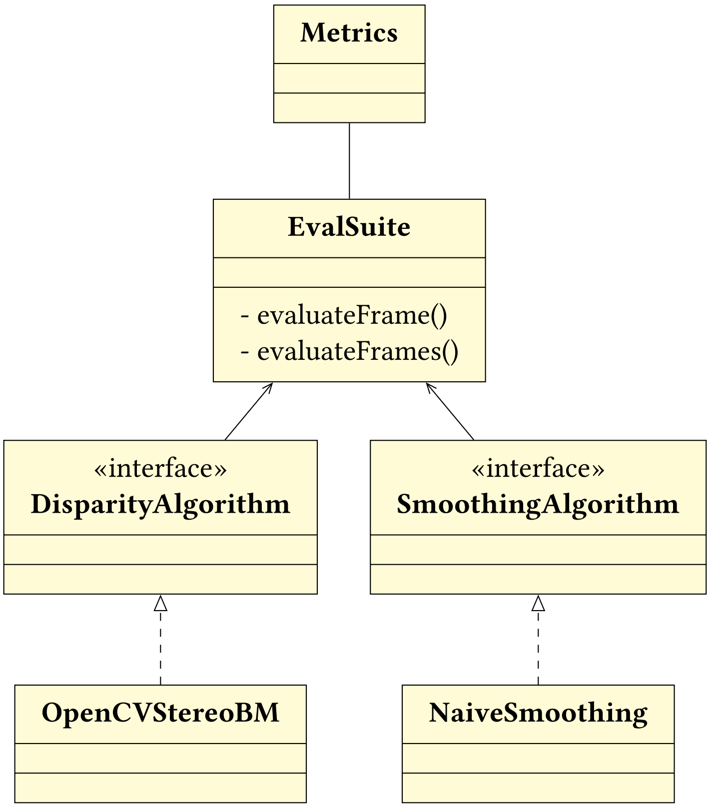



Figure 4.2: Simplified UML diagram of general architecture.

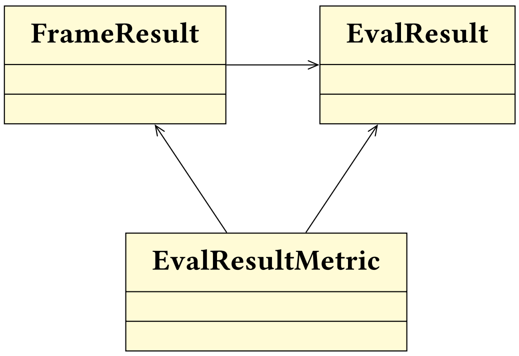



Figure 4.3: Simplified UML diagram of result composition for further processing.

- analyzing the impact of noise, and
- trying to smooth noisy areas in the resulting disparity map with other frames.

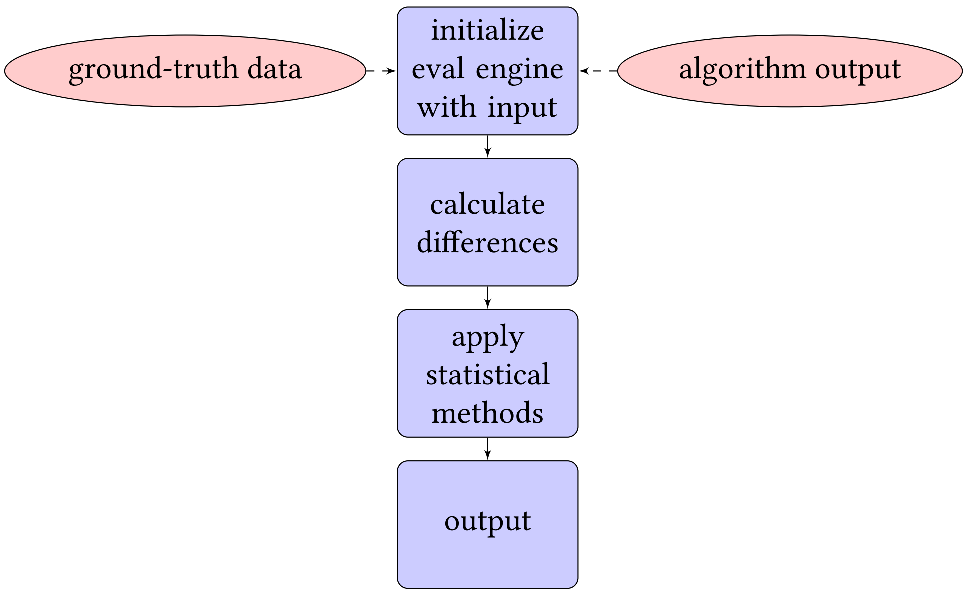
Noise can occur through video compression. As video sensors tend to be noisier than image sensors it also more present in stereoscopic videos, if not rendered by a computer. As noise can be simulated and added onto the rendered video, it is investigated how noise disturb stereo matcher. As these ideas are more explained in the implementation chapter 4 the following subsection a novel approach for videos are examined.

In contrast to other implementations, input and output are clearly defined and thus different techniques can be adapted easily. There exist combined frameworks which fulfill two tasks, disparity calculation (as the algorithm is implemented) and the final evaluation step. This makes it harder to use the evaluation module separately from the rest. None the less the open source community around computer vision also lacks of code for stereo matcher. Due the diversities of algorithms and eval suites the decision was made to go for an OpenCV implementation of an eval suite for disparity algorithms.

Works basically as a wrapper. Can output statistical stuff. Basically works on disparity maps / images.

Huge mistake could be to apply the metrics on the whole disparity map from the algorithm. The output contains areas which are black (value = 0). This can be:

- noise
- mistaken be the algorithm
- expected disparity
- non-occluded areas



The eval engine has two modes to be queried:

- via command-line which also results in a console output,
- with a configuration file ($config.json$) which leads to an output in a given folder for the web result viewer.

### Preprocessing

Different tasks are executed before the actual disparity algorithm are invoked and the evaluation takes place:

- Normalization
- Gaussian noise

### Postprocessing

Postprocessing only consists of one task: normalization of the disparity map. Some algorithms struggle with calculating a larger number than the number of disparities of 32. Some datasets only calculated the disparity to be in a range from $0$ to $63$. Grayscale normally ranges from $0$ to $255$. Thus the disparity map is normalized to range from $0$ to $63$.

Simply only disparity normalization.

## 4.4 Fine-grained evaluation via masks

The evaluation would be trivial by just comparing the computed disparity map with its ground-truth companion. This trivial comparison would be a pitfall, as the results would be erroneous due to for instance unknown disparity or occluded regions. As remedy bitmasks are introduced to simply focus on interesting pixels. In this section the following bitmasks are introduced and how they are calculated:

- Depth-discontinuity
- Textured regions recognition
- Discover occluded pixels
- Saliency detection
- Unknown disparity

First describe the motivational introduction to bitmasks why would one use those?

## Depth-discontinuity

Determining correspondence can fail in textureless or depth-discontinuous regions as mentioned above. Thus it is also interesting how the algorithms handle such regions. For this purpose a depth-discontinuity mask was implemented as well.

Explain:

- Dilate
- Erode
- maybe show short example image (combined dilate/erode)
- explain how the bitmask is calculated

## Textured regions recognition

As stereo matching algorithms act on the assumption that the disparity is smooth, especially if contrast and color intensity do not change drastically, it can be interesting to see how those algorithms treat textured and textureless regions. Thus a bitmask for textured regions recognition was implemented.

Explain (shortly):

- Sobel
- pow
- boxFilter
- how we get the bitmask

## Discover occluded pixels

An occluded pixel is defined as a pixel which is hidden in one of the two images, for instance an object hides it from a different angle. In the case of stereo matching the disparity can not be calculated for such a pixel. Thus occluded pixels have to be handled properly, as they could distort our result. For this purpose a simple mask is introduced to indicate which pixels on the scene are visible for both cameras and which are not.

Explain and cite two papers (taxonomy of disparity algorithms). There it is explained how everything is working. Explain how the get the result (use algorithm).

Thus we need to take care about non-occluded areas. For this purpose we generated bitmasks (the size $w$ * $h$) for each video dataset.

"In addition to disparity maps, for stereo matching method evaluation it is interesting to have a non-occluded area mask. This mask represents in white color the pixels on the scene that are visible from both cameras and in black color the pixels that are visible from only one camera. To obtain the non-occluded area mask, we simply cross-checked the left and right disparity maps. Pixels that are visible in both cameras will have the same value in both disparity maps, but for occluded pixels the left and right disparity value will be different. The performance of the stereo matching algorithm on areas where pixels are occluded is one of the most important quality indicators of the algorithm, as it is very difficult to find the matching point of a pixel in one of the images if it is not visible on the other image." [41].

### Saliency detection

Another criteria for the later evaluation is how the algorithms operate on regions which are salient in a specific scene. There exist some algorithms for saliency detection in either images or videos [7, 16]. OpenCV offers two different saliency categories to be computed:

- $StaticSaliency$ in images, and
- $MotionSaliency$ on videos.

Explain how saliency detection works, implemented via OpenCV. Otsu's algorithms, threshold and K-Means algorithm [29].

## 4.5 Integration of existing algorithms

Deciding which algorithms should be describe was not an easy task. On the one hand, the algorithms which shall be implemented during this thesis are important and thus should be described definitely. On the other hand, there is a huge diversity of used technologies amongst disparity algorithms. For instance various programming languages, the decision between cpu- versus gpu-rendering, different coding styles and used libraries. As a matter of fact, this makes it harder to implement and then evaluate every single disparity algorithm. Thus, the algorithms from Middlebury were integrated in the evaluation suite and streamlined. Additionally, the so called efficient large-scale stereo matcher (ELAS) was also integrated. The parameters used for each algorithm are described in the implementation chapter. This section is for giving an overview on these algorithms as well as their parameters for the later implementation chapter 4.

This does not exist currently, so we use the Middlebury Test Suite with its algorithms for images and apply them on videos.


Figure 4.4: Basic integration of existing algorithms

- Consists of multiple steps
- Wrapper to call different algorithms
- normalizing of output
- actual eval process

Input: ImageLEFT, ExpectedLEFT, ImageRIGHT, ExpectedRIGHT Output: a good metric for showing good/bad disparity, a few ideas.

## 4.6 Image diminisher to simulate real use cases

### Gaussian noise

randn[6]

[7]

As seen in the related work chapter 3, some approaches use restoration algorithms in order to reduce noise which can occur. Hence noise generation was added as a preprocessing step in order to see how noise disrupts disparity algorithms. We use gaussian noise meaning that the noise is gaussian-distributed.

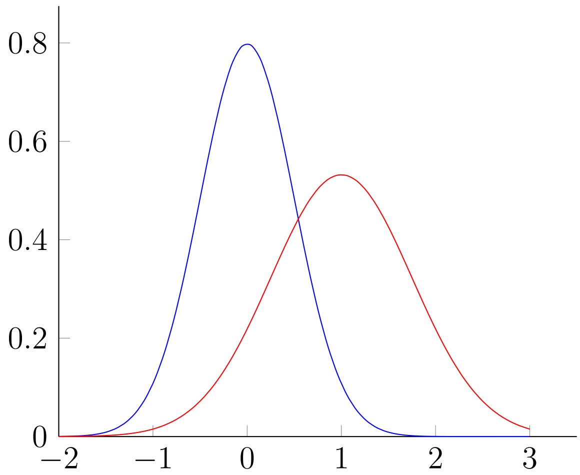
$$f\left(x\right) = ae^{-\frac{(x-b)^2}{2c^2}}$$

$$p_G(z) = \frac{1}{\sigma\sqrt{2\pi}}e^{-\frac{(z-\mu)^2}{2\sigma^2}}$$

In this example, $z$ represents the grey level which is added to the image matrix later on. $\mu$ is the mean value (= 0). $\sigma$ is the standard deviation.

The $\sigma$ can be set in our evaluation suite in order to see how this distracts the image.

[6] http://docs.opencv.org/master/d2/de8/group__core__array.html#gaeff1f61e972d133a04ce3a5f81cf6808




### Video compression

FFMPEG [18].

## 4.7 Web result viewer for evaluation suite

For fine-tuning the algorithm's parameters as well as implementing the bitmasks it was helpful to see the visual output of both. As the resulting bitmasks for each frame with the computed result disparity map were saved on the hard-drive for further investigations a web result viewer was created for visualizing the output. The following features were implemented:

- Starting new computations with different parameters and scene selection.
- Playing frame-by-frame with different speeds.
- Online csv-export of result.

- Insert screenshot (one or two from web result viewer)
- Describe basic features
- Describe what could be done with the evaluation web suite in near future
- But for thesis evaluation a csv exporter was used.

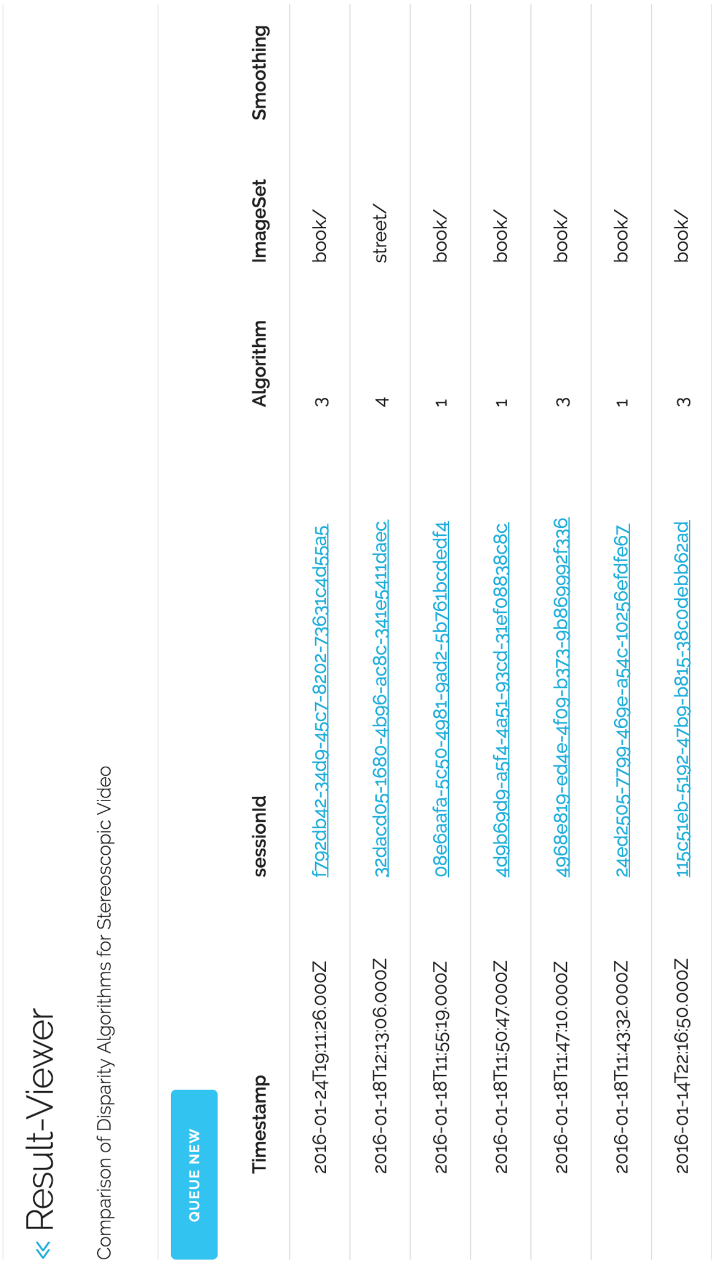



Figure 4.5: Overview page of web result viewer.

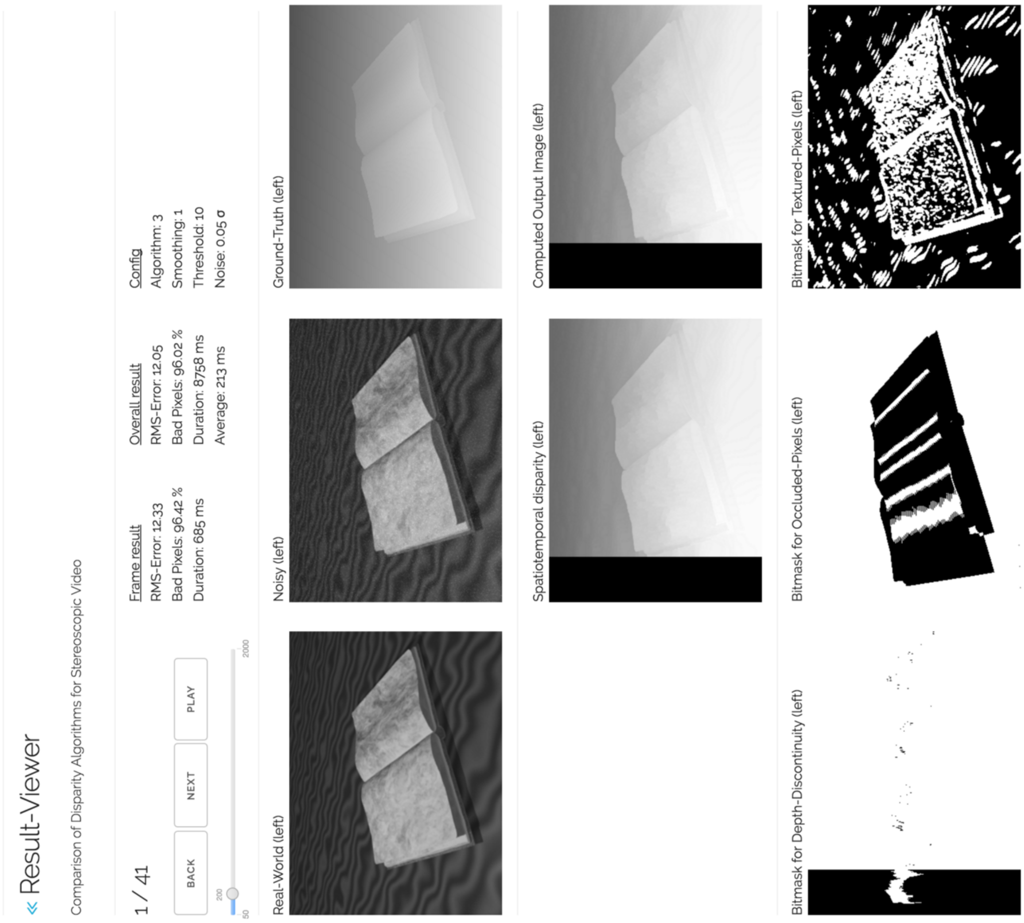



Figure 4.6: Detail of one result in the web result viewer.

## 4.8 Discussion

The following modules were actually implemented:

- Reader for the PFM file format.
- Python scripts for upcoming evaluation.
- Shell scripts for getting the docker containers and thus the work distributed among different instances[7].
- Evaluation processor for

[7]With instances virtual machines from DigitalOcean are meant.

# 5 Evaluation and results

Short outline as usual.

## 5.1 Datasets

- Tsukuba Stereo Dataset (used Tsukuba Map as reference)
- Stereo videos with ground truth disparities from cambridge[1]
- SVDDD - a high-resolution Stereoscopic Video Dataset with precise Depth and Disparity information

### Ground-truth data

In general there exist two ways to gain ground-truth information of pixels:

- in the real-world sense the area,
- in computer-animated scenes let the renderer calculate the disparity.

The former can be achieved via area scanner for instance a radar. This approach is of course more error-prone than the latter one. It can lack of accuracy due to false measurements. The latter one provides real ground-truth information.

Provide real-world ground truth disparity maps [35].

#### Availability

The aim of such datasets is to provide data which we can rely on to evaluate the performance of computer vision algorithms, such as disparity algorithms in this thesis. Without having such datasets it would be crucial to rate the overall quality of for instance stereo matching algorithms.

We use the datasets for stereo correspondences from Middlebury, cf. [48]. They provide us with three examples. Each of those examples include an image from each camera (left, right) and the resulting ground-truth disparity map.

[1] `http://www.cl.cam.ac.uk/research/rainbow/projects/dcbgrid/datasets/`

We also use the small images from [41].
Famous Tsukuba ground-truth dataset:

- 64 levels of disparity, so grayscale image
- generated scenes with Autodesk Maya 2012

[41]
Ground truth data? How to provide such data?

### Tsukuba stereo dataset

This dataset can be seen as the origin of stereoscopic datasets. The first dataset ever released for working with stereoscopic images. [41].

### Middlebury stereo dataset

[48].

Figure 5.1: Middlebury stereo dataset example

### Cambridge stereo dataset

The Cambridge stereo dataset consists of five different rendered scenes in $400x300$ resolution with each about 100 frames [46]. The dataset scales the disparity for each scene from $0$ to $64$.

### SVDDD - a high-resolution Stereoscopic Video Dataset with precise Depth and Disparity information

The Lehrstuhl für Praktische Informatik IV[2] has created a novel dataset on its own containing depth information for stereoscopic videos. Before the evaluation it was

[2] `http://ls.fmi.uni-mannheim.de/de/pi4/`

clear that this dataset is going to be evaluated as well. The difference is that this dataset was not analyzed before. Thus it is possible that the chosen algorithms work not properly on this dataset. If this case occurs there exist two possibilities why this happens:

- On the one hand, the disparity information are not properly calculated, or
- on the other hand, those algorithms have some troubles with the constructed scene.

Focusing on the latter one, the scenes were created with Blender and utilizes the XXX (insert me pls) open-source scenes. A second camera was added to the scene for obtaining depth information. The parameter for each scene can be extracted from the paper. The most important points which can lead to a discrepancy between the computed disparity maps and the ground-truth data are the following:

- rays of light were removed only in the ground-truth-data,
- motion and object blur was reduced only in the ground-truth data,
- fain-grained textures were reduced only in the ground-truth data.

## 5.2 Quality metrics

To evaluate the quality of computed disparity maps we need at first to what we compare the computed disparity maps to and secondly how to compare them. In the section **??** the availability of such datasets were discussed.

Insert table here.

| **Tag** | **Description** |
|---|---|
| $\sigma_{10}$ | Guassian noise calculator. |
| | |
| | |

Table 5.1: Overview on used metrics in results

Typical quality measure instruments for comparing disparity maps against their ground-truth reference data are [13]:

- Percentage of bad matching pixels
- Root-mean-square error

- Parameter-free measures

In addition the following are also considered:

- Mean absolute error in pixels
- Error quantiles
- Precision and accuracy

### Percentage of bad matching pixels

$$\text{PBMP} = \frac{1}{n} \sum_{x,y=0} (|d_a(x,y) - d_e(x,y)| > \delta_t) \tag{5.1}$$

Percentage of bad matching pixels for given threshold.

### Root-mean-squared error

The mean squared error (MSE) as well as the root mean squared error (RMSE) are both the most popular metrics in image and video processing. The MSE is as the name implies the mean of the squared differences between the intensities of pixels in two pictures at the same position. In conclusion the average difference per pixel is then the root of the squared error.

$$\text{RMS-Error} = \sqrt{\frac{1}{n} \sum_{x,y=0} (d_a(x,y) - d_e(x,y))^2} \tag{5.2}$$

It represents the sample standard deviation of the differences between predicted values and observed values. Here $d_a(x,y)$ is the actual disparity value for given $x$ and $y$. $d_e(x,y)$ is our expected disparity value from our ground-truth data. Hence the RMSE is the difference between values on average.

### Energy efficiency

$$EE_{alg} = \frac{clean\ pixels}{runtime\ in\ seconds} \tag{5.3}$$

where $clean\ pixels$ are defined as bad pixels in percent of non-occluded pixels and $runtime\ in\ seconds$ the runtime for this particular frame. $EE_{alg}$ is then calculated in the mean of all frames of a given sequence over all examined sequences and all datasets.

## 5.3 Measurement

- First took normal depth map images. But not that good, only values from 0-255.
- Actual evaluation process consists of comparison of real calculated values of depth map by results of algorithms.
- How does the eval actual look like?

[3], [47].

### Parameter tuning

Our results.

### Against reference dataset

It is also important to have some kind of reference dataset with which the evaluation engine can be calibrated with. Of course the settings (i.e. parameters of an algorithm) is dependent on the input material (size, noise) and on the scene (e.g. textured vs textureless). So it is possible to have good parameters for one scene and not for another. However, in order to evaluate those algorithms the Tsubuka stereo dataset was chosen as reference dataset to see how the eval engine actually works with the same parameters on the same images.

## 5.4 Results

The results are visualized with the HSV color model.


Figure 5.2: Scale of hue of the HSV color model.

## Applying disparity algorithms on videos

As said in the implementation chapter 4 applying disparity algorithms on videos is trivial and straightforward. None the less different anomalies can be further investigated while analyzing videos:

- outliers in the form of a single frames which differs too much from the others,
- impact of noise,
- smooth the unknown disparity in frames (next subsection).

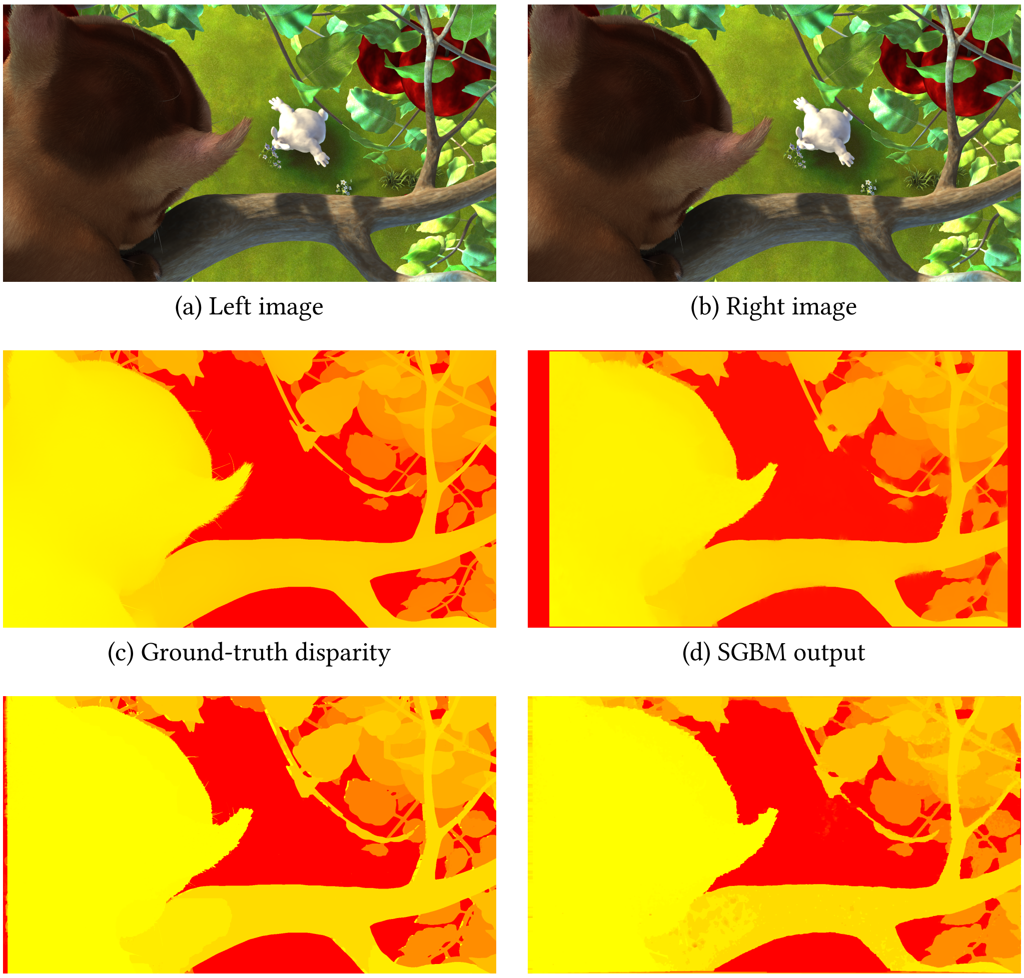
As you can see in the following result matrices the quality metrics regarding the salient regions are a bit esoteric. Of course there exist techniques to estimate salient regions in static images and also in videos. In videos moving objects can be detected as salient, referring to motion saliency[7, 57]. None the less those bitmasks are a bit more esoteric in the sense of there is room for interpretation.

| Rank | Method | Sequence | RMS-All | RMS-Noc | RMS-Sal | RMS-Tex | RMS-DD |
|---|---|---|---|---|---|---|---|
| 1 | StereoSGBM | SVDDD 02_rabbit | 4.35% | 5.43% | 1.0 px | 1.1 px | 300ms |
| 2 | StereoBM | SVDDD 02_rabbit | 4.35% | 5.43% | 1.0 px | 1.1 px | 300ms |

Table 5.2: Result matrix of RMS for videos

| Rank | Method | Sequence | Out-All | Out-Noc | Out-Sal | Out-Tex | Out-DD |
|---|---|---|---|---|---|---|---|
| 1 | StereoSGBM | SVDDD 02_rabbit | 4.35% | 5.43% | 1.0 px | 1.1 px | 300ms |
| 1 | StereoSGBM | SVDDD 02_rabbit | 4.35% | 5.43% | 1.0 px | 1.1 px | 300ms |
| 1 | StereoSGBM | SVDDD 02_rabbit | 4.35% | 5.43% | 1.0 px | 1.1 px | 300ms |
| 1 | StereoSGBM | SVDDD 02_rabbit | 4.35% | 5.43% | 1.0 px | 1.1 px | 300ms |
| 1 | StereoSGBM | SVDDD 02_rabbit | 4.35% | 5.43% | 1.0 px | 1.1 px | 300ms |
| 1 | StereoSGBM | SVDDD 02_rabbit | 4.35% | 5.43% | 1.0 px | 1.1 px | 300ms |
| 1 | StereoSGBM | SVDDD 02_rabbit | 4.35% | 5.43% | 1.0 px | 1.1 px | 300ms |
| 1 | StereoSGBM | SVDDD 02_rabbit | 4.35% | 5.43% | 1.0 px | 1.1 px | 300ms |
| 1 | StereoSGBM | SVDDD 02_rabbit | 4.35% | 5.43% | 1.0 px | 1.1 px | 300ms |
| 1 | StereoSGBM | SVDDD 02_rabbit | 4.35% | 5.43% | 1.0 px | 1.1 px | 300ms |
| 1 | StereoSGBM | SVDDD 02_rabbit | 4.35% | 5.43% | 1.0 px | 1.1 px | 300ms |
| 1 | StereoSGBM | SVDDD 02_rabbit | 4.35% | 5.43% | 1.0 px | 1.1 px | 300ms |
| 1 | StereoSGBM | SVDDD 02_rabbit | 4.35% | 5.43% | 1.0 px | 1.1 px | 300ms |
| 1 | StereoSGBM | SVDDD 02_rabbit | 4.35% | 5.43% | 1.0 px | 1.1 px | 300ms |
| 1 | StereoSGBM | SVDDD 02_rabbit | 4.35% | 5.43% | 1.0 px | 1.1 px | 300ms |
| 1 | StereoSGBM | SVDDD 02_rabbit | 4.35% | 5.43% | 1.0 px | 1.1 px | 300ms |
| 1 | StereoSGBM | SVDDD 02_rabbit | 4.35% | 5.43% | 1.0 px | 1.1 px | 300ms |
| 1 | StereoSGBM | SVDDD 02_rabbit | 4.35% | 5.43% | 1.0 px | 1.1 px | 300ms |
| 1 | StereoSGBM | SVDDD 02_rabbit | 4.35% | 5.43% | 1.0 px | 1.1 px | 300ms |
| 1 | StereoSGBM | SVDDD 02_rabbit | 4.35% | 5.43% | 1.0 px | 1.1 px | 300ms |
| 1 | StereoSGBM | SVDDD 02_rabbit | 4.35% | 5.43% | 1.0 px | 1.1 px | 300ms |
| 2 | StereoBM | SVDDD 02_rabbit | 4.35% | 5.43% | 1.0 px | 1.1 px | 300ms |

Table 5.3: Result matrix of outliers for videos



(a) Left image (b) Right image

(c) Ground-truth disparity (d) SGBM output

(e) MRF ICM output (f) ELAS output

Figure 5.3: Frame 01, scene 03 rabbit of the SVDDD dataset.

#### Result matrix

Insert overview matrix.

### 5.4.1 Runtime

Depict:

- different performance of disparity algorithms,
- down-scaled performance,

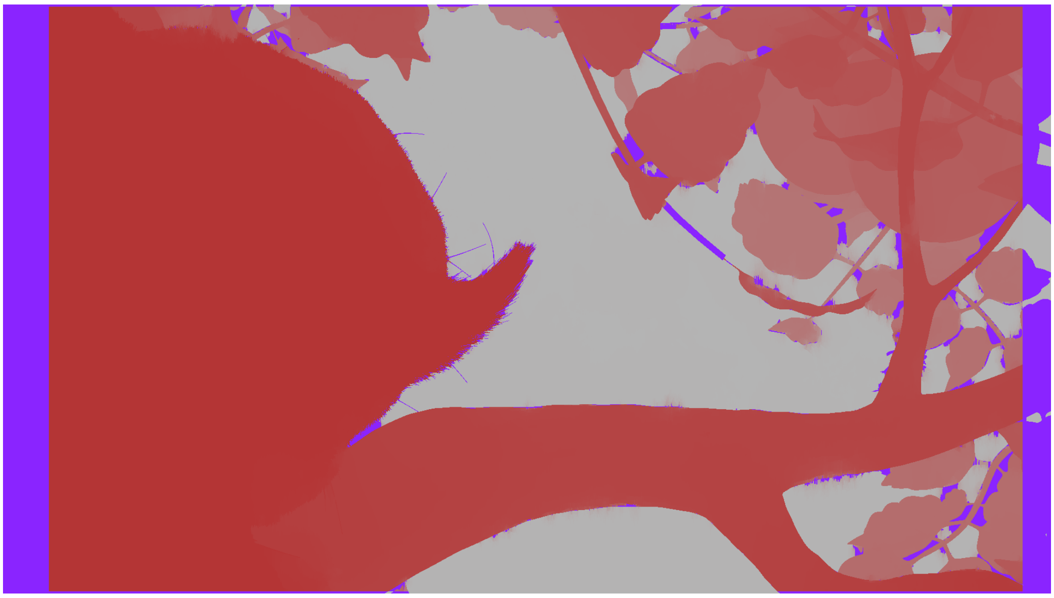


Figure 5.4: Heatmap of outliers with threshold of 4.0 pixels in computed disparity map with OpenCV SGBM, frame 01, scene 03 rabbit, SVDDD dataset.

- first down-scale, then run algorithms, then upscale, runtime,
- smoothing-over-time runtime.

### Result matrix

| Rank | Method | Sequence | Time | Time/MP | Quality index |
|---|---|---|---|---|---|
| 1 | StereoSGBM | SVDDD 02_rabbit | 4.35% | 5.43% | 1.0 px |
| 2 | StereoBM | SVDDD 02_rabbit | 4.35% | 5.43% | 1.1 px |

Table 5.4: Result matrix of runtime for videos

## 5.5 Discussion

This is really important!

# 6 Conclusion

The work described in this thesis has been concerned with the comparison of disparity algorithms.

## 6.1 Thesis summary

- What the thesis brought with each chapter.
- One small paragraph regarding one disparity algorithm.
- One small paragraph regarding dataset of the chair.
- What was implemented.
- What the main result of the evaluation.

## 6.2 Future outlook

A few thoughts on possible future work to improve the presented algorithms, split in low- and high-level from a technical perspective:

### Low-level

- Neuronal networks [44]
- Other matching cost calculation methods [23].
- Focus more on how humans experience depth? [15]

### High-level

- Real-time availability
- higher resolution
- What's on the market, like multi-view?

- Provide more real-world ground truth disparity maps [20, 35].

# Bibliography


[1] E. Angel and R. Bellman. *Dynamic programming and partial differential equations.* Academic Press, Inc., 1972.

[2] R. E. Bellman and S. E. Dreyfus. *Applied dynamic programming*. Princeton university press, 2015.

[3] A. Benoit, P. Le Callet, P. Campisi, and R. Cousseau. Quality assessment of stereoscopic images. *EURASIP journal on image and video processing*, 2008.

[4] B. Block and P. McNally. *3D Storytelling: How stereoscopic 3D works and how to use it.* Taylor & Francis, 2013.

[5] Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 26(9):1124–1137, 2004.

[6] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 23(11):1222–1239, 2001.

[7] G. Bradski. The OpenCV library. *Dr. Dobb's Journal of Software Tools*, 2000.

[8] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black. A naturalistic open source movie for optical flow evaluation. In A. Fitzgibbon et al. (Eds.), editor, *European Conf. on Computer Vision (ECCV)*, Part IV, LNCS 7577, pages 611–625. Springer-Verlag, Oct. 2012.

[9] S. H. Chan, R. Khoshabeh, K. B. Gibson, P. E. Gill, and T. Q. Nguyen. An augmented Lagrangian method for total variation video restoration. *Image Processing, IEEE Transactions on*, 20(11):3097–3111, 2011.

[10] M. M. H. Chowdhury and M. A.-A. Bhuiyan. A new approach for disparity map determination. *Daffodil International University Journal of Science and Technology*, 4(1):9–13, 2009.

[11] I. Cohen and G. Medioni. Detecting and tracking moving objects for video surveillance. In *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.*, volume 2. IEEE, 1999.

[12] T. H. Cormen. *Introduction to algorithms.* MIT press, 2009.

[13] B. Cyganek and J. P. Siebert. *An introduction to 3D computer vision techniques and algorithms.* John Wiley & Sons, 2011.

[14] J. Davis, R. Ramamoorthi, and S. Rusinkiewicz. Spacetime stereo: A unifying framework for depth from triangulation. In *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, volume 2, pages II–359. IEEE, 2003.

[15] G. C. DeAngelis, I. Ohzawa, and R. D. Freeman. Neuronal mechanisms underlying stereopsis: how do simple cells in the visual cortex encode binocular disparity? *Perception*, 24(1):3–31, 1995.

[16] T. Dittrich, S. Kopf, P. Schaber, B. Guthier, and W. Effelsberg. Saliency detection for stereoscopic video. In *Proceedings of the 4th ACM Multimedia Systems Conference*, pages 12–23. ACM, 2013.

[17] M. Fahle. Wozu zwei Augen? *Naturwissenschaften*, 74(8):383–385, 1987.

[18] FFmpeg. FFmpeg, 2010. URL `http://www.ffmpeg.org`.

[19] A. Geiger, M. Roser, and R. Urtasun. Efficient large-scale stereo matching. In *Asian Conference on Computer Vision (ACCV)*, 2010.

[20] A. Geiger, J. Ziegler, and C. Stiller. Stereoscan: Dense 3D reconstruction in real-time. In *Intelligent Vehicles Symposium (IV)*, 2011.

[21] R. A. Hamzah, R. A. Rahim, and Z. M. Noh. Sum of absolute differences algorithm in stereo correspondence problem for stereo matching in computer vision application. In *Computer Science and Information Technology (ICCSIT), 2010 3rd IEEE International Conference on*, volume 1, pages 652–657. IEEE, 2010.

[22] D. B. Henson. *Visual fields.* Butterworth-Heinemann Medical, 2000.

[23] S. Hermann and T. Vaudrey. The gradient - A powerful and robust cost function for stereo matching. In *Image and Vision Computing New Zealand (IVCNZ), 2010 25th International Conference of*, pages 1–8. IEEE, 2010.

[24] H. Hirschmüller. Accurate and efficient stereo processing by semi-global matching and mutual information. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 2, pages 807–814. IEEE, 2005.

[25] H. Hirschmüller. Stereo processing by semiglobal matching and mutual information. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 30(2):328–341, 2008.

[26] H. Hirschmuller. Semi-global matching - Motivation, developments and applications. *Photogrammetric Week*, pages 173–184, 2011.

[27] H. Hirschmüller and D. Scharstein. Evaluation of cost functions for stereo matching. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–8. IEEE, 2007.

[28] A. Hosni, C. Rhemann, M. Bleyer, and M. Gelautz. Temporally consistent disparity and optical flow via efficient spatio-temporal filtering. In *Advances in Image and Video Technology*, pages 165–177. Springer, 2012.

[29] X. Hou and L. Zhang. Saliency detection: A spectral residual approach. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–8. IEEE, 2007.

[30] P.-J. Käck. Robust stereo correspondence using graph cuts. *Master's thesis, Royal Institute of Technology*, 2004.

[31] T. Kanade, H. Kano, S. Kimura, A. Yoshida, and K. Oda. Development of a video-rate stereo machine. In *Intelligent Robots and Systems 95.'Human Robot Interaction and Cooperative Robots', Proceedings. 1995 IEEE/RSJ International Conference on*, volume 3, pages 95–100. IEEE, 1995.

[32] R. Khoshabeh, S. H. Chan, and T. Q. Nguy. Spatio-temporal consistency in video disparity estimation. In *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*, pages 885–888. IEEE, 2011.

[33] V. Kolmogorov. Convergent tree-reweighted message passing for energy minimization. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 28(10):1568–1583, 2006.

[34] V. Kolmogorov and R. Zabin. What energy functions can be minimized via graph cuts? *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 26(2):147–159, 2004.

[35] D. Kondermann, R. Nair, S. Meister, W. Mischler, B. Güssefeld, K. Honauer, S. Hofmann, C. Brenner, and B. Jähne. Stereo ground truth with error bars. In *Computer Vision–ACCV 2014*, pages 595–610. Springer, 2015.

[36] P. Krähenbühl, M. Lang, A. Hornung, and M. Gross. A system for retargeting of streaming video. *ACM Transactions on Graphics (TOG)*, 28(5):126, 2009.

[37] M. Lang, A. Hornung, O. Wang, S. Poulakos, A. Smolic, and M. Gross. Nonlinear disparity mapping for stereoscopic 3D. In *ACM Transactions on Graphics (TOG)*, volume 29, page 75. ACM, 2010.

[38] Z. Lee, R. Khoshabeh, J. Juang, and T. Q. Nguyen. Local stereo matching using motion cue and modified census in video disparity estimation. In *Signal Processing Conference (EUSIPCO), 2012 Proceedings of the 20th European*, pages 1114–1118. IEEE, 2012.

[39] L. Lucas, C. Loscos, and Y. Remion. *3D Video: From Capture to Diffusion*. John Wiley & Sons, 2013.

[40] D. Marr and T. Poggio. Cooperative computation of stereo disparity. *Science*, 194(4262):283–287, 1976.

[41] S. Martull, M. Peris, and K. Fukui. Realistic CG stereo image dataset with ground truth disparity maps. In *ICPR workshop TrakMark2012*, volume 111, pages 117–118, 2012.

[42] M. Menze and A. Geiger. Object scene flow for autonomous vehicles. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3061–3070, 2015.

[43] D. Min, S. Choi, J. Lu, B. Ham, K. Sohn, and M. N. Do. Fast global image smoothing based on weighted least squares. *Image Processing, IEEE Transactions on*, 23(12):5638–5653, 2014.

[44] B. A. Olshausen et al. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381(6583):607–609, 1996.

[45] I. E. Richardson. *H. 264 and MPEG-4 video compression: Video coding for next-generation multimedia*. John Wiley & Sons, 2004.

[46] C. Richardt, D. Orr, I. Davies, A. Criminisi, and N. A. Dodgson. Real-time spatiotemporal stereo matching using the dual-cross-bilateral grid. In *Computer Vision–ECCV 2010*, pages 510–523. Springer, 2010.

[47] D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International journal of computer vision*, 47(1-3):7–42, 2002.

[48] D. Scharstein, H. Hirschmüller, Y. Kitajima, G. Krathwohl, N. Nešić, X. Wang, and P. Westling. High-resolution stereo datasets with subpixel-accurate ground truth. In *Pattern Recognition*, pages 31–42. Springer, 2014.

[49] P. Shirley, M. Ashikhmin, and S. Marschner. *Fundamentals of computer graphics*. CRC Press, 2009.

[50] S. N. Sinha. Graph cut algorithms in vision, graphics and machine learning–an integrative paper. *UNC Chapel Hill*, 2004.

[51] J. Sun, N.-N. Zheng, and H.-Y. Shum. Stereo matching using belief propagation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 25(7): 787–800, 2003.

[52] R. Szeliski, R. Zabih, D. Scharstein, O. Veksler, V. Kolmogorov, A. Agarwala, M. Tappen, and C. Rother. A comparative study of energy minimization methods for markov random fields with smoothness-based priors. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 30(6):1068–1080, 2008.

[53] R. Tamassia. *Handbook of graph drawing and visualization*. CRC press, 2013.

[54] M. F. Tappen and W. T. Freeman. Comparison of graph cuts with belief propagation for stereo, using identical MRF parameters. In *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pages 900–906. IEEE, 2003.

[55] P. Viola and W. M. Wells III. Alignment by maximization of mutual information. *International journal of computer vision*, 24(2):137–154, 1997.

[56] M. J. Wainwright, T. S. Jaakkola, and A. S. Willsky. MAP estimation via agreement on trees: message-passing and linear programming. *Information Theory, IEEE Transactions on*, 51(11):3697–3717, 2005.

[57] B. Wang and P. Dudek. A fast self-tuning background subtraction algorithm. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 395–398, 2014.

[58] S. Wanner and B. Goldluecke. Reconstructing reflective and transparent surfaces from epipolar plane images. In *Pattern Recognition*, pages 1–10. Springer, 2013.

[59] J. S. Yedidia, W. T. Freeman, and Y. Weiss. Understanding belief propagation and its generalizations. *Exploring artificial intelligence in the new millennium*, 8:236–239, 2003.

[60] K.-J. Yoon and I. S. Kweon. Adaptive support-weight approach for correspondence search. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (4):650–656, 2006.

# Declaration of Honour

Hiermit versichere ich, dass diese Arbeit von mir persönlich verfasst wurde und dass ich keinerlei fremde Hilfe in Anspruch genommen habe. Ebenso versichere ich, dass diese Arbeit oder Teile daraus weder von mir selbst noch von anderen als Leistungsnachweise andernorts eingereicht wurden. Wörtliche oder sinngemäße Übernahmen aus anderen Schriften und Veröffentlichungen in gedruckter oder elektronischer Form sind gekennzeichnet. Sämtliche Sekundärliteratur und sonstige Quellen sind nachgewiesen und in der Bibliographie aufgeführt. Das Gleiche gilt für graphische Darstellungen und Bilder sowie für alle Internet-Quellen. Ich bin ferner damit einverstanden, dass meine Arbeit zum Zwecke eines Plagiatsabgleichs in elektronischer Form anonymisiert versendet und gespeichert werden kann.

Mannheim, Mai 2016

Ben John

# Abtretungserklärung

Hinsichtlich meiner Abschlussarbeit mit dem Titel „*Comparison of Disparity Algorithms for Stereoscopic Video*“ räume ich der Universität Mannheim/Lehrstuhl für Praktische Informatik IV, Prof. Dr. Wolfgang Effelsberg, umfassende, ausschließliche unbefristete und unbeschränkte Nutzungsrechte an den entstandenen Arbeitsergebnissen ein. Die Abtretung umfasst das Recht auf Nutzung der Arbeitsergebnisse in Forschung und Lehre, das Recht der Vervielfältigung, Verbreitung und Übersetzung sowie das Recht zur Bearbeitung und Änderung inklusive Nutzung der dabei entstehenden Ergebnisse, sowie das Recht zur Weiterübertragung auf Dritte.

Solange von mir erstellte Ergebnisse in der ursprünglichen oder in überarbeiteter Form verwendet werden, werde ich nach Maßgabe des Urheberrechts als Co-Autor namentlich genannt. Eine gewerbliche Nutzung ist von dieser Abtretung nicht mit umfasst.

Mannheim, Mai 2016

Ben John