FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

Reconstructing 3D Data of Frontal Human Faces by a Passive Stereo System with a Correlation-Based Correspondence Method

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieurin

im Rahmen des Studiums

Medieninformatik

eingereicht von

Agnieszka Wojdecka
Matrikelnummer 0501682

an der
Fakultät für Informatik der Universität Wien

Betreuung: Univ.-Prof. Dipl.-Ing. Dr. Helmut Hlavacs

Wien, 10.12.2012

(Unterschrift Verfasserin) (Unterschrift Betreuung)

FAKULTÄT
FÜR !NFORMATIK
Faculty of Informatics

Reconstructing 3D Data of Frontal Human Faces by a Passive Stereo System with a Correlation-Based Correspondence Method

MASTER'S THESIS

submitted in partial fulfillment of the requirements for the degree of

Master of Science

in

Media Informatics

by

Agnieszka Wojdecka
Registration Number 0501682

to the Faculty of Informatics
at the University of Vienna

Advisor: Univ.-Prof. Dipl.-Ing. Dr. Helmut Hlavacs

Vienna, 10.12.2012

(Signature of Author) (Signature of Advisor)

Erklärung zur Verfassung der Arbeit

Agnieszka Wojdecka
Theodor-Kramer Strasse 10/2/7, 1220 Wien

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit - einschließlich Tabellen, Karten und Abbildungen -, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

(Ort, Datum) (Unterschrift Verfasserin)

Abstract

Seeing *frontal human faces* - a topic of computer vision - requires three dimensional information. The perception of a face is natural to us, whereas for a computer, a noisy array of numbers needs to be processed. For stereo vision, that is part of computer vision, this task is challenging, as the skin of a face is low-textured and therefore general approaches are prone to errors. Some applications, that may use such depth information, are computer games or animated movies.

In this thesis, we will focus on finding overall satisfying depth maps or disparity maps, which encode the 3D data, for such applications. We will describe our ideas of a wholly passive approach with consumer video cameras of a correlation-based, local stereo matching method. Despite the fact, that active methods are currently the best methods to recover face geometry, some work [BHPS10, BBB$^+$10b] proved that passive methods can be on the same level, which serves as motivation for our work. We developed a simple approach with minimal user input, that accepts images in high definition (HD) resolution, in order to overcome the problems of low-texturing.

We will evaluate our approach under different lighting conditions, baselines and structural information of the face. Further, we will show where the main difficult areas of the face are. Therefore, an own lighting environment was built and, for the evaluation, synthetic ground truth data were modeled.

The quantitative evaluation demonstrates for a baseline of 0.5 in Blender units a percentage of 43% of erroneous pixels at depth discontinuities. In textured areas, not at depth discontinuities, 9% was measured. In textureless regions a percentage of 13% was reached. These results are retrieved by comparing a sparse disparity map with a synthetic ground truth. Experiments revealed that the most inaccurate areas of the face are the eyes and the nose. Further, one has to pay attention to the baseline's width, because it shoudn't be too big.

Kurzfassung

Das Sehen von menschlichen Gesichtern - ein Thema des Maschinellen Sehens - benötigt dreidimensionale Daten. Die Wahrnehmung eines Gesichtes ist für die meisten Lebewesen natürlich, wohingegen ein Computer ein verrauschtes Zahlenfeld verarbeiten muss. Im Falle des Stereoskopen Sehens, welches Teil des Maschinellen Sehens ist, ist diese Aufgabe eine Herausforderung, da die Haut eines Gesichtes wenig Textur hat. Aus diesem Grund sind Versuche, welche diese Probleme angehen, fehleranfällig. Einige Beispiele für Anwendungen, welche Tiefeninformationen nützen, wären Computerspiele und animierte Filme.

Diese Diplomarbeit konzentriert sich auf das Auffinden von Tiefenkarten oder Disparitätskarten. Sie enthalten dreidimensionale Information für die oben genannten Anwendungen. In meiner Arbeit wird ein völlig passiver Ansatz mit der Verwendung von gewöhnlichen Videokameras beschrieben. Er ist korrelationsbasiert und arbeitet mit lokalen Vergleichsmehoden. Trotz der Tatsache, dass aktive Verfahren zur Zeit am besten geeignet sind für das Wiedergewinnen von 3D Informationen eines Gesichtes, existieren Arbeiten [BHPS10, BBB$^+$10b], die beweisen, dass auch passive Methoden hochqualitative Resultate liefern. Dies dient als Motivation für die eigene Arbeit. Ich habe einen einfachen Ansatz entwickelt, welcher minimale Benutzereingaben verlangt. Er arbeitet mit Bildern in HD-Auflösung, um das Problem der wenigen Textur zu lösen.

Das Verfahren wird unter verschiedenen Lichtbedingungen, Basislängen und strukturellen Informationen des Gesichtes evaluiert. Weiters wird gezeigt, wo die schwer zu berechnenden Bereiche des Gesichtes liegen. Es wurde eine eigene Beleuchtungsumgebung gebaut und eine synthetische *Ground Truth* wurde durch Modellierung erzeugt.

Die quantitative Evaluierung zeigt, für eine Basislänge von 0.5 in Blender-Einheiten, dass der Anteil an fehlerhaften Pixel bei 43% liegt, wenn Tiefen-Unstetigkeitsstellen betrachtet werden. In texturierten Regionen, fern von Tiefen-Unstetigkeitsstellen, wurde ein Fehleranteil von 9% gemessen. In weniger texturierten Regionen wurden 13% an fehlerhaften Pixeln erreicht. Diese Resultate basieren auf schwach besetzten (*engl. sparse*) Disparistätenkarten im Vergleich mit der synthetischen Ground Truth. Weitere Versuche zeigen, dass die schwer zu berechenbaren Bereiche des Gesichtes in den Augen- und in der Nasenregion vorzufinden sind. Die Basislänge sollte möglichst eng gewählt werden.

Contents

CHAPTER 1

Introduction

1.1 Problem Description

Computer vision transforms input - some kind of data, for instance from a camcorder - into either a decision or a new representation [BKSA]. The decision might be “there is a face in this scene“, whereas the representation might be receiving a depthmap [BKSA]. In fact, seeing is natural to a human being. Our brain automatically identifies, in a task-dependent way [BKSA], faces or other objects in an image, while ignoring other irrelevant parts. This ability builds upon years of experiences during a human lifetime. Unfortunately, little is understood from this natural process till now. However, when a computer system aims to see as a human being, only an array of numbers from a sensor is given. Often noise disturbs digital data. The experiences and all the other process involved in the human vision system lack when imitating the act of seeing with a computer [BKSA].

Our task is to transform this noisy array of numbers into a perception: “face“. In fact, till now, there is no consistent solution found to this problem [BKSA].

The act of seeing faces requires three dimensional information. Working with images, as in computer vision, leads to the need of dealing with the loss of the third dimension due to the image formation process.

Facial three dimensional information is of interest for several applications, like human computer interaction, in which the computer should be able to understand the facial expression of the user. In computer surveillance, this information can be used to recognize a person responsible for a criminal act. Further, in teleconferencing, the presentation of a remote person can be improved using three dimensional information. Finally, development is done in the fields of entertainment like computer games, realistic-looking face modeling like plastic surgery and animation like the movie industry [LZSA].

The use of 3D data for face recognition and face modeling is an active research topic and there are many approaches recovering the face geometry. The choice depends mainly on the application. For model-based video coding, simpler methods are the choice, whereas for recognition and identification, accurate methods are required [OTRT05].

In this thesis, we will focus on overall satisfying 3D data, like used for simpler face models. Some applications, that may use our depth information, are computer games or animated movies.

In general, one distinguishes between active and passive methods. Often active methods for 3D measurement with structured illumination or laser scanning are employed, as they tend to give accurate 3D facial information. Obviously, expensive equipment for active methods is a major problem. Using passive methods implicates working with passive sensors, such as ordinary video or CCD cameras.

There are several cues to recover the third dimension of a single image. Some examples are the point of view, shading or texture. Using these cues results in methods, known as "Shape from X" techniques [SW06]. Taking more than one image from the same face results in techniques that yield 3D data driven by different views, like multi view stereo, different instants of time, like structure from motion, or different focal points, like depth from focus [SW06].

Stereo vision is a popular passive approach. The major drawback is its low quality and accuracy of captured 3D information in the past. Therefore, few research work is done in the fields of reconstructing facial 3D information, because it is challenging to work with low-textured surfaces. As a consequence, conventional stereo matching techniques with intensity correlation produce noisy 3D information due to ambiguity.

1.2 Goals

The focus of my master thesis lies on the development of a method for computing 3D depths of a frontal human face, which is the core step in reconstructing the face geometry. The matching of pixels in different images to find correspondences is the key problem. Therefore, the main object is obtaining good correspondence results with a good image acquisition and matching technique. This thesis concentrates on the use of two views at an instant, whereby two pairs are used.

The goal is to develop a simple method for automatic face reconstruction, therefore a completely passive approach with passive sensor, such as consumer high definition videos will be used. For the stereo vision system two cameras will be placed horizontally and calibrated carefully.

First, the correspondence-problem for two images of scene points is solved - also known as the image matching step [CM01] - iteratively with a correlation-based method. Then, the two rays passing through the cameras' projection centers and the corresponding pixels are intersected [CM01], resulting in the 3D scene point - also known as the reconstruction step. Extra knowledge of the head's shape is minimized as easy user input.

In this work, it is assumed that the lighting conditions are controlled, therefore we will propose a simple image acquisition technique.

For the development the framework OpenCV [The00] in C++ was used as it has several routines already implemented and offers a good framework to work with cameras and to perform mathematical operations. Further, we parallelize the algorithm by using the Boost C++ library [Boo12].

The motivation behind the proposed method is to investigate how the combination of several matching approaches affects the result when compared with a synthetic ground truth.The modeling of the face mesh was done with Blender 2.34 and 2.63. Further, experiments with real-world datasets were realized. Besides developing a simple method with minimal user-input, our main goals are to show

- where the most inaccurate regions of the face are to be found and
- how the proposed correlation-based method behaves with varying parameters and under the following difficult circumstances
  - different baselines
  - different lighting
  - more structure/ less structure in the face

1.3 Structure of the Thesis

The thesis is divided into seven chapters. In the first chapter, we give an introduction to the topic by a problem description and by describing the motivation and the goals of the thesis. In the second chapter, we present some of the most important approaches for solving the correspondence-problem and the reconstruction step. Some related work, that mainly deals with correlation-based methods, is outlined. Further, the fundamental concepts of stereo vision are presented in the third chapter. On the basis of the third chapter, the fourth chapter introduces the own decisions and approaches. The fifth chapter outlines the image data and calibration data acquisition, the lighting setup, and describes the setup configuration, as for instance the used libraries and the optimization. The sixth chapter presents an evaluation of the results, whereas the acquisition of the ground truth is also described. The seventh chapters gives a conclusion of the work.

CHAPTER 2

Overview of Approaches

Approaches can be subdivided into active and passive methods. Active sensors hold this name as they actively project rays, e.g. light, on the face, as opposed to passive methods, that use existing light. The focus of this work lies on a passive method, stereo vision, that is classified in local and global methods. Related work to our approach will be presented at the end of the chapter.

2.1 Active and Passive Methods

For facial geometry reconstruction, active methods are preferred, as they tend to produce clean data. Such methods yield 3D information using active light, like structured illumination (e.g. [Mar96]) or laser scanner (e.g. [Yue95]).

Obviously, the expensive equipment is a major drawback. Further, they lack accuracy in high-textured regions. But they are not that sensitive to different lighting conditions and produce very accurate point measurements in low-textured regions [3dM, YTCA10].

Using structured light may be irritating to the person in front of the camera, because the pattern is projected onto the face. Moreover, trying to reconstruct fine-scaled details, like facial hair, fails, because the pattern is not visible [Dim].

When using laser scanners, the model needs to hold still for a few seconds, which may be unnatural. Moreover, they fail on shiny surfaces like oily skin or dark surfaces like dark skin and hair. Resulting 3D data of face scans do not tend to be smooth, due to holes and spikes [LZSA].

Using passive methods implicates working with passive sensors, such as ordinary video or CCD cameras. Some passive techniques, yielding 3D information, use multi stereo vision (e.g. [BHPS10]) or just stereo vision (e.g. [CM01]), two orthogonal views (e.g. [IY96]) or a single view (e.g. [KSB11]). Further, one distinguishes between shape from shading (e.g. [KSB11]), shape from silhouettes (e.g. [LMPM03]) and shape from motion (e.g. [LZJC00]).

A popular passive approach is the employment of a stereo vision system. Few research work is done in the fields of face geometry reconstruction with this method, as it is challenging

to reconstruct the low-textured surface of a face. As a consequence, conventional intensity correlation-based methods (e.g. [FM98]) of stereo images produce noisy 3D data. However, one can work with high resolution images to profit from textural information like skin pores, freckles, scars, wrinkles etc. Moreover, they are sensitive to bad lighting and occlusion. Therefore, one needs to control the acquisition setup. Some techniques require manual input (e.g. [LZJC00]), marker (e.g. [GGW+98]) or special make-up (e.g. [Wil90]). But usually these approaches lack accuracy. Often, extensive databases of faces are used (e.g. [KSB11]) to build a generic face.

Passive stereo vision is advantageous as it just needs a single shot to capture the underlying image data for 3D reconstruction with a low-cost system. The nature of passive methods is that there is a correspondence for every pixel in the first camera image with another pixel in the second camera image [Dim], except for ambiguities and occlusion. Moreover, it is possible to develop a fully automatic system.

2.2 Local and Global Methods

The matching process of stereo vision is categoriezd into local and global methods. Traditional local methods constrain the matching process in the local neighborhood, whereas global methods constrains the process globally through the whole image or on scan-lines [BBH03]. As a consequence, global methods are less sensitive to ambiguities, in contrast to local methods, but they are computationally costly. Local stereo methods usually use aggregated intensity differences over a small area of the image, a so-called window, while global methods work pixelwisely [HS07]. At discontinuities, window-based local methods have problems to be accurate, because the pixels in a window are constraint to have the same depth. The foreground fattening problem occurs as the smoothness assumption over a window is violated. Conversely, global methods can produce good results, when their smoothness assumption is formulated cleverly.

Some popular approaches for local methods are block matching, gradient-based techniques or feature matching and for global methods dynamic programming, graph cuts or belief propagation [BBH03]. Global methods appear to be very rarely employed for face reconstruction, probably because they are computationally expensive. Local methods can find correspondences accurately, as they are globally independent.

Currently, popular local methods are seeds-growing algorithms. At first step, points of interest are matched and then, picking the ones with best similarity measure, initial seed points are defined. Next, they are sorted by the similarity measure. At every iteration, the best matches from the sorted list of seed points are chosen, while new seed points are collected from the adjacent areas of the current seed point. The algorithm terminates, as only points that have not been collected before are selected for the list of seed points. This method is extended in (e.g. [DS11]) by a threshold, that rejects best matches with low similarity measure. False matches are excluded through the left/right consistency check. In a second step, piecewise dynamic programming is applied, such that scan-lines are divided, according to the set of high confidential matches, into smaller sections. The result is a smooth and dense point cloud of the face.

2.3 Related Work

In our approach, we require equipment, that is low-cost and provides simple capturing. No active light, like structured light or laser-scanned model, should be used. Semi-automatic reconstruction should be possible without the need of geometry scans or extensive user-input. The method should work fully without markers, face paint or fluorescent makeup. For that purpose, a simple, passive and local method is applied. Further, the own approach works with block-matching, therefore it is window- and correlation-based.

Despite the fact, that active methods are recently the best methods for recovering face geometry, some research work [BHPS10, BBB+10b] proved that passive methods are on the same level. In the following, we describe approaches that implement window-based, correlation-based passive stereo vision matching methods, because they are most relevant for the own topic.

In 1998 and 1999, Fua and Miccio [FM98] developed a fast and simple system to fit an animated face model to noisy stereo data. Their acquisition system consisted of a simple video. They used successive, gray-scale images from the video sequence as a stereo pair and assumed that intrinsic and extrinsic parameters were known beforehand. To reconstruct the face geometry, including hair and ear, they used the stereo method described in [Fua93]. In detail, they proposed a correlation-based approach that produces depth maps with holes. A block-matching for every pixel with normalized mean-squared differences along an "epipolar band" was used. They stated that the final result, using the normalized mean-squared difference, was similar to using a normalized cross correlation. Further, many false matches were rejected by the left/right consistency check. The holes were closed with interpolation that preserves image features. They proved that their method is suitable for faces but admitted that computational time is not optimal. In fact, they used a hierarchical approach, but only to increase the density of the disparity map and not to reduce the search area from coarser to finer images.

In 2001, Chen et al. [CM01] built a stereo system with two consumer digital cameras for reconstructing the geometry of frontal human faces. The calibration process was skipped, because the fundamental matrix was computed automatically. The images were rectified, thus, the vertical scale was distorted. In a second step, the horizontal scale was handled. The correspondence problem was solved by translating it into a global optimization problem, where a 3D correlation volume was built, depending on the pixel position and the disparity. The disparity values were gained through extracting the maximal disparity surface, that was formed by seed voxels, which are likely to be correct matches in the volume. For similarity measure a normalized cross-correlation was applied. Outliers were removed by the left/right consistency check and disparities, with a difference higher than one pixel in the left and right neighborhood, were removed. Holes were closed by diffusion from the nearest neighbors. Their method makes use of local and global advantages. However, manual correction for the hair and intensive computational work was necessary.

In 2010, state-of-the-art work is described by Beeler et al. [BBB+10b, BBB+10a] and Bradley et al. [BHPS10]. Both described a passive multi-view stereo vision system for recovering the

3D geometry of faces using correlation-based methods. Their research work demonstrates that passive stereo systems, using correlation methods, are suitable for accurate reconstruction of faces. Impressive work was done by Beeler et al., as they captured mesoscopic geometry like pores of static faces, whereas Bradley et al. lack these details and also do not deal with "expressive motions" [BHB+11]. Reversely, Bradley et al. implemented a method, that provides fully-automatic reconstruction for 30 frames per second.

In [BHPS10], Bradley et al. described an acquisition system with seven stereo pair HD video cameras. For capturing details, the cameras were zoomed-in, thus, they captured a patch of the face, such that each pair had an overlap of the face surface. Illumination was controlled by nine LED light fixtures, resulting in evenly bright illumination. The binocular camera setup was calibrated by a method that used the rectification error instead of the reprojection error [BH10]. The correspondence matching step was solved by rectification and block-matching based on normalized cross correlation (NCC), as described in [BBH08]. This was done iteratively, whereas for every iteration a depth constraint was used to restrict the matching process further. Therefore, an over-smoothed depth image of the current one was used, resulting in per-pixel constraints. The process needed only three iterations to gain sufficient reconstruction. A scaled-window matching method was applied that reduces horizontal distortions. Finally, the resulting 3D data were merged. In the post-processing, the face mesh was smoothed, but this is not part of the reconstruction step any longer.

Beeler et al. used, among others, an acquisition system of SLR cameras with indirect illumination. They developed a calibration method with a calibration sphere, that had randomly distributed fiducials on its surface, as opposed to the traditional method with a calibration checkerboard plane. The method is suitable for faces, as it is similar sized to human heads and one can place it at a position where the person in front of the camera is to be expected. Pictures were taken in a single-shot with standard light, in contrast to other state-of-the-art methods that work with active light. The captured images were in a 12 bit RAW format, they were grayscaled and rectified. The correspondence problem was solved by a pairwise stereo matching under a pyramidal approach by two rounds for each layer. A block matching with normalized cross-correlation (NCC) was applied. In the first round, matches were computed for all pixels accordingly to the preceding layer. In the second step, the primarily computed pixels were checked accordingly to the smoothness, uniqueness and ordering constraints and, if the constraints were violated, they were re-matched accordingly to the valid pixels in the neighborhood. For the final result, they performed a surface refinement in "continuous" 3D and model skin pores and wrinkles in an extra step to obtain realism. One problem that had to be coped with was specularity, for instance, on the nose tip, as it disturbed the results. The work bases on [FP10], that describes a new method for MVS reconstruction with a "match, expand and filter procedure".

Both research groups demonstrated that passive stereo vision systems can reach quality alike as active systems. In both approaches the eye and eyelash geometry as well as the eye-brows and other facial hair are not fully realistically reconstructed. They argued that the brightness constancy assumptions was often violated in these areas.

In [BHB+11], research work was done by contributions of both parties. However, the correspondence problem was solved by the method of Beeler et al., as described in [BBB+10a]. They proposed to improve the method by zooming-in, like in [BHPS10], to gain more detail.

A commercial solution, that reconstructs 3D information with a multi-view passive stereo system, is Dimension Imageing 3D [Dim], which works with high-resolution images. The reconstruction process involves correlation-based methods.

CHAPTER 3

Background

3.1 Preliminaries

To relate world, camera and image some conceptual definitions need to be done.

For the reconstruction of 3D data just a small segment of the world is relevant, the so called scene. In the following the term scene will be used throughout. In a scene several scene points are considered for the reconstruction and they are described by coordinates in several coordinate systems.

Coordinate Systems

A coordinate system is used for the determination of positions in space. Thus, the position of a point is uniquely defined by specifying coordinates in space. The projection of coordinates of a system to a different coordinate system is done by a coordinate transformation [Tönny], whereas the positions of the scene points remain the same. Therefore, through the definition of different coordinate systems for scene, camera and image, these systems can be related to each other.

The image coordinate system (ICS) [Moy00] determines the position of an image point $P_i = (x_i, y_i)$ in the image plane. It has two axes X_i and Y_i, as the image has two dimensions, with the origin O_i positioned at the top left corner of the image. The coordinates along the axes are specified as real numbers, so called image coordinates, and when using integers, they are rounded off to so called pixel coordinates $p_i = (u_i, v_i)$ with the axes u and v.

The camera coordinate system (CCS) [Moy00] determines the position of a camera point $P_c = (x_c, y_c, z_c)$. It is an orthogonal coordinate system with three axes X_c, Y_c and Z_c. The CCS depends on the camera's position and orientation and, thus, X_c and Y_c are parallel to the axes of the ICS. The origin O_c is the optical center of the camera and Z_c is the optical axis, orthogonal to the image plane. The intersection of the optical axis and the image plane is referred to as the principal point, with its coordinates x_0 and y_0, typically at the center of the image plane.

The description of the 3D scene and its scene point $P_s = (x_s, y_s, z_s)$ is done in the world coordinate system (WCS) [Moy00]. It is also an orthogonal coordinate system with three axes X_s, Y_s and Z_s as well as with the origin O_s.

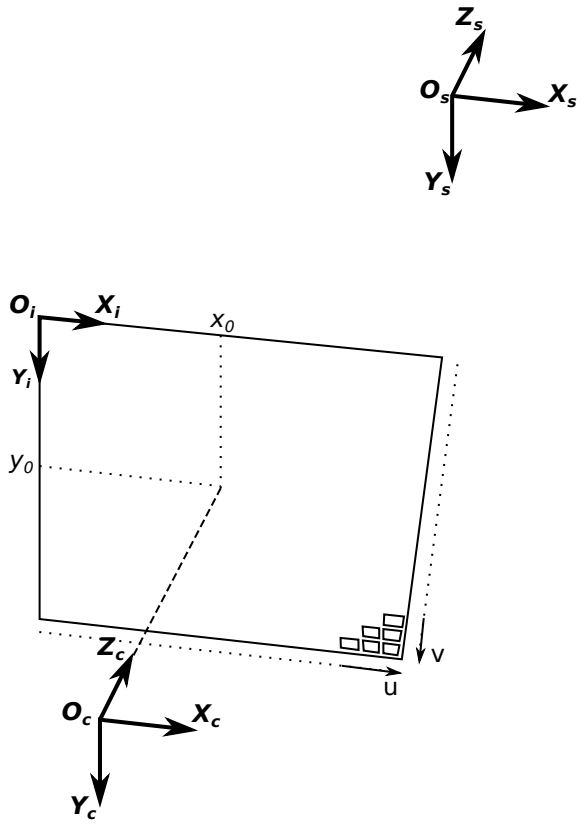


Figure 3.1: The three coordinate systems, ICS, CCS and WCS, are shown in relation to each other. Further, the principal point and the rounding off of the pixel coordinates is illustrated.

The subscripts of x, y and z, used throughout this thesis, will define the relation to the respective axis.

A coordinate transformation can be achieved through a projective transformation. A projective transformation transforms points in n-space to points in m-space, whereas m is inferior to n [JDFHSA]. In our scenario, this means that scene points are projected to image points through a projective transformation.

Projective Transformation

First of all, we need to understand the concept of the projective space and projective geometry, which describes the invariant properties under projective transformation. In fact, we perceive our world in Euclidean 3D space and its geometry is a subset of projective geometry. Therefore, Euclidean geometry is more structured than projective geometry. In between projective and Euclidean geometry we have affine geometry [Bar08] as follows

$$Projective \rightarrow Affine \rightarrow Euclidean.$$

Euclidean transformations preserve lengths and angles of objects. An Euclidean transformation is equivalent to a coordinate transformation from an orthogonal coordinate system to another orthogonal coordinate system, mostly done by a rotation and a translation [Schny].

Affine geometry introduces the concept of parallelism and the ratio of distance that are invariant under affine transformations. Affine transformations change the form of an object, therefore scaling and shearing is possible. Lengths and angles may not be preserved after such an affine transformation [Schny].

Projective geometry is more general than affine and Euclidean geometry. Therefore, more transformation is possible. Lengths, the ratio of areas and parallelism are not necessarily preserved. However, projective transformations preserve the cross ratio of distances, the incidence, whether a points lies on a line or plane, and the type, e.g. a line remains a line [Bar08].

The camera's viewing process is similar to a human eye's viewing process as both use similar geometric properties. As our space is in 3D Euclidean space and the image plane is in 2D Euclidean space, the image formation is done by projective transformation from 3D to 2D.

Essential to the projective space is the introduction of homogeneous coordinates. Homogeneous points will be noted by a tilde. An Euclidean point, for instance in 2D space, may be $p = (x, y)$. The same point in projective space is represented by homogeneous coordinates, which means that it is extended by a further coordinate $\tilde{p} = (x, y, 1)$, usually by 1. Consequently, points in the Euclidean space are translated to lines in projective space. As $\tilde{p}$ is equivalent to $\lambda\tilde{p}$ $\forall\ \lambda \neq 0$, the direction and not the length of a vector is relevant in projective space. If we want to transform $\tilde{p}$ back to the Euclidean space, a division through the last coordinate is necessary. When the last coordinate of $\tilde{p}$ is 0, this point does not exist in the Euclidean space as the division by 0 is forbidden. This results in a subspace of points lying at infinity, which only exists in projective space. The idea of homogeneous coordinates may be also expanded to higher dimensions of vectors [Schny].

The projective space allows the mathematical description of the perspective projection, which is a type of projective transformation [JDFHSA] from 3D to 2D. We will make the assumption that the mapping of the scene on the image plane is done by the perspective projection.

Perspective Projection

The projection of a space point on the image plane is the intersection of the straight line, that is the connection from space point to optical center, with the image plane. All the points lying on this straight line are projected on the same image point. Thus, the projection is a non invertible transformation, as the information of the depth is lost. Moreover, the original appearance of angles is also discarded. The distance from the optical center to the image plane is finite. After the perspective projection, parallel lines in Euclidean space are no longer parallel in projective space. Their intersection is the so called vanishing point. This point lies at infinity and does not exists in Euclidean space [Schny].

When using homogeneous coordinates for the perspective projection, the projection is translated from a non-linear to a linear problem [Bar08].

A general perspective projection can be expressed by the following matrix

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ a & b & c & d \end{pmatrix},$$

which projects on the plane $ax + by + cz + d = 1$ [JDFHSA].

A simple camera model, that may use the perspective projection for its image formation process, is the pinhole camera.

Pinhole camera

In digital cameras, the image is formed on a sensor (CCD or CMOS). The light is translated to a number and the array of numbers is equivalent to the digital image [Gal11]. In general, using the pinhole camera a scene points $P_s = (x_s, y_s, z_s)$ traverse through an infinitesimal aperture, the so called pinhole or the optical center. The light traverses through the pinhole and its intensity is formed on the image plane as an image point $P_i = (x_i, y_i)$. As the optical center is located between image plane and scene, the image of a scene object is reversed and the size of the object decreases, when its depth increases. The distance between the image plane and the pinhole is called the focal length f, which is an important camera parameter. Figure 3.2 demonstrates the projection of a point with a pinhole camera model.

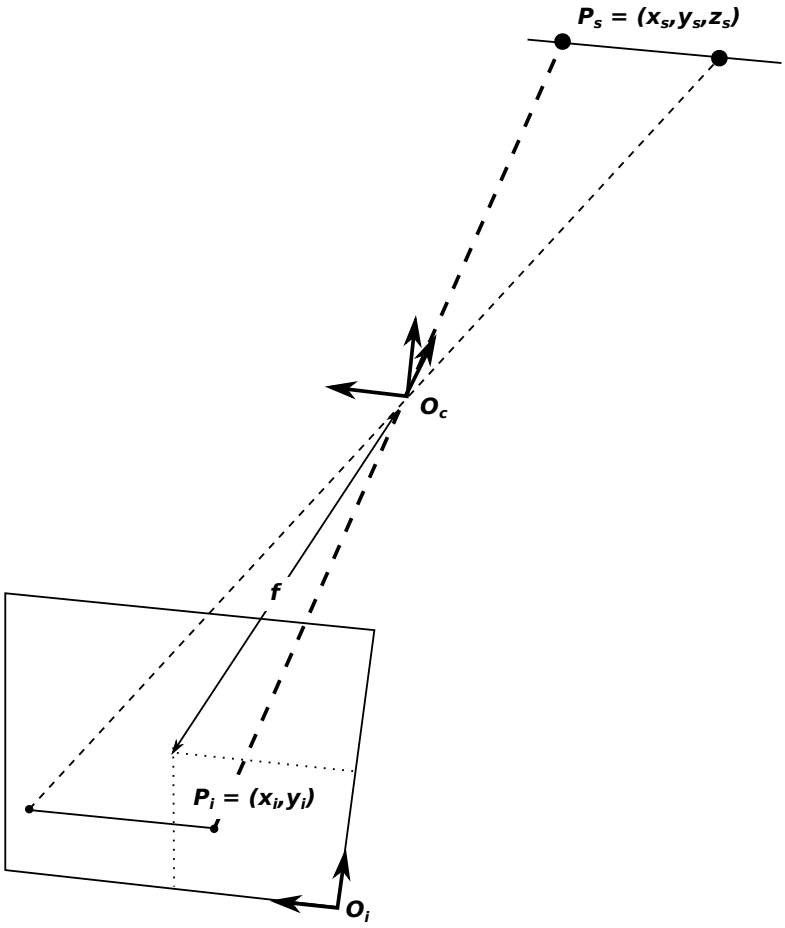


Figure 3.2: A scene point is projected on an image plane, using the pinhole camera model, whereas the pinhole lies in front of the image plane. Further, the focal length is illustrated.

Using an ideal, perspective projection of the scene on the image plane, the optical center is located behind the image plane, which differs from a real camera or the eye [Schny]. This is shown in figure 3.3. Again, the size of scene objects in the image varies inversely with their depths. However, when using the perspective projection, we need to pay attention to the reversion of the x-axis and y-axis, which we have already done in the ICS.

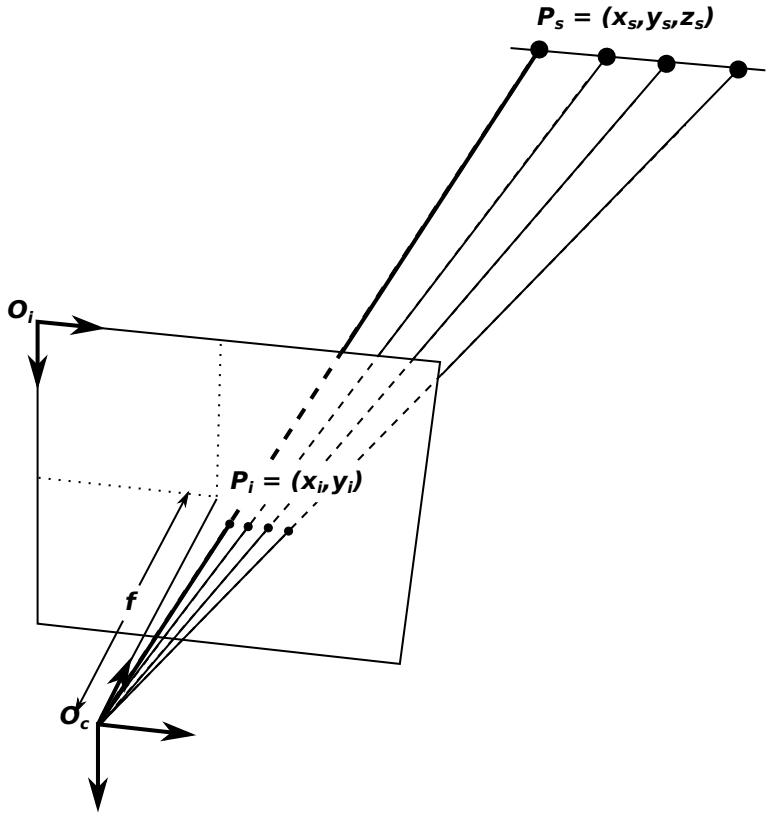


Figure 3.3: A scene point is projected on an image plane, using a perspective projection. Therefore, the pinhole lies behind the image plane. Further, the focal length is illustrated.

The image coordinates are transformed, using the fact that $x_i/f = x_s/z_s$ and $y_i/f = y_s/z_s$, by

$$x_i = fx_s/z_s \text{ and } y_i = fy_s/z_s,$$

whereas x_i and y_i are inversely related to z_s, so, when, for instance, z_s increases, x_i and y_i respectively decrease.

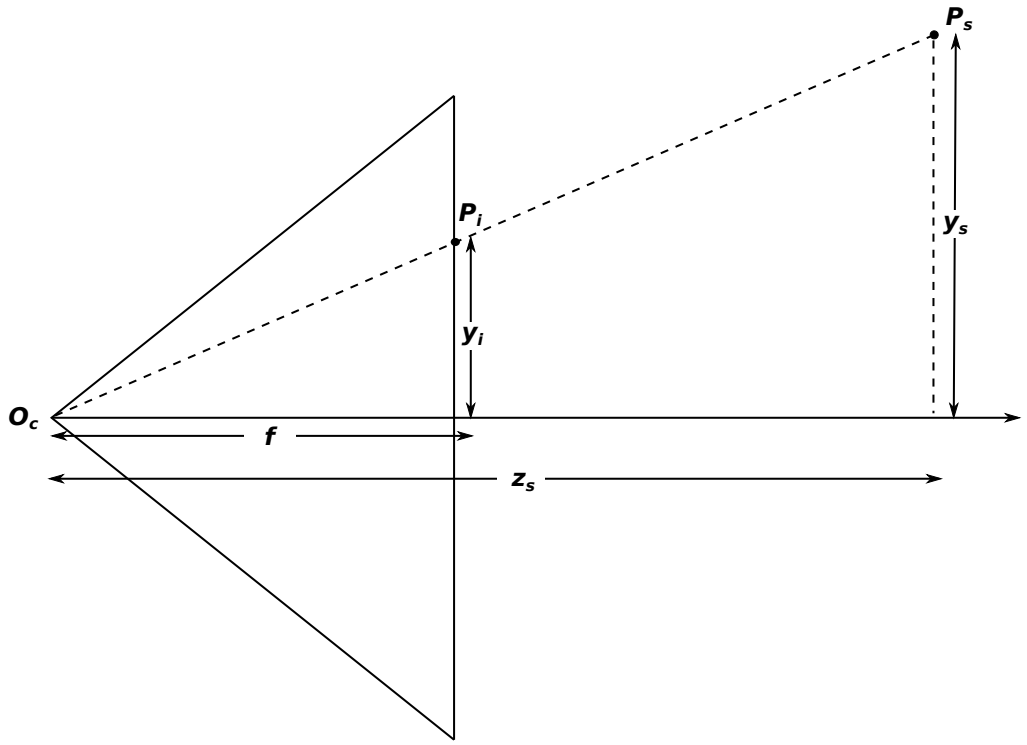


Figure 3.4: The fact that $y_i/f = y_s/z_s$ is illustrated by a pinhole camera model.

As the depth information is lost under perspective projection, the recovery of depth, the 3D reconstruction, needs further information. For reconstruction, a second image of the same scene, with a slightly different perspective, needs to be introduced. This image can be captured by the same camera or by a second pinhole camera, both having a different position and/or orientation. We will make the second assumption, introducing a second pinhole camera. Such a setup is called a stereo vision setup.

Stereo Vision Setup

Having two perspective projections of the pinhole cameras, we will introduce the superscripts l and r, respectively for the left and the right camera. Or, in a more general case, we will use the subscripts 1 and 2 for the first and the second camera.

Both cameras view the same scene point P_s. We assume that the cameras are horizontally aligned to each other. The projection lines go through P_s and the two optical centers, O_c^l and

O_c^r. The collisions of the lines with the image planes are P_i^l and P_i^r. The line connecting O_c^l and O_c^r is the so called baseline.

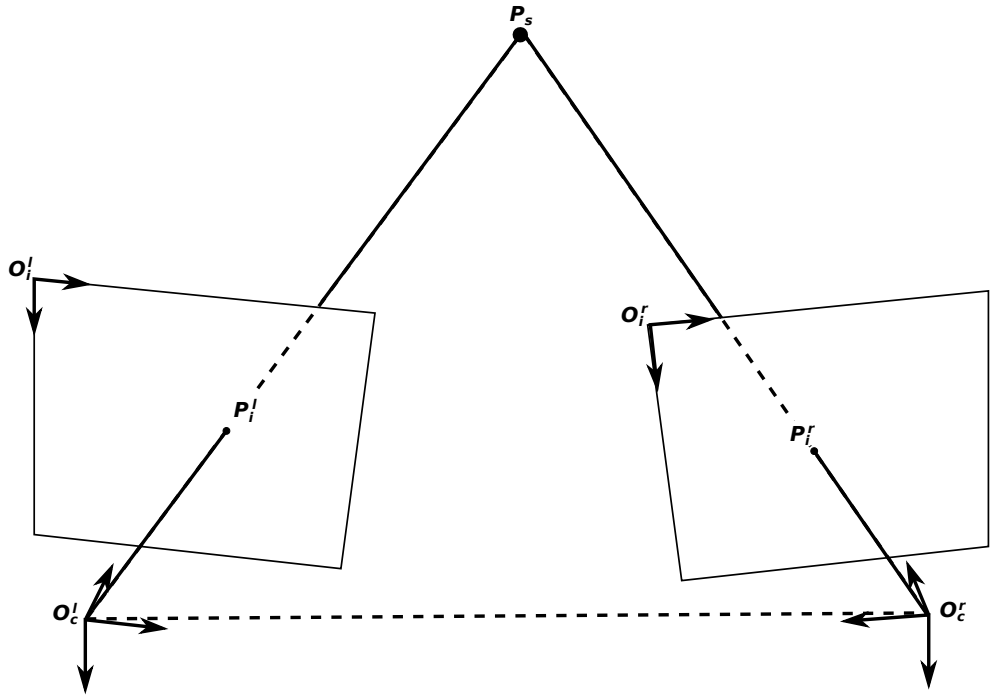


Figure 3.5: Stereo vision setup.

After traversing the projection line of the left camera from P_i^l, to recover depth, we need further guidance. The projection line represents potential scene points at increasing depth values. In contrast to a single perspective projection, the depth of P_s can be recovered. For the reconstruction, the two projection lines need to be intersected. However, to intersect the lines, the corresponding points P_i^l and P_i^r need to be known beforehand. Therefore, we need to relate the two perspective projections with the so called epipolar geometry.

Epipolar Geometry

The epipolar plane is determined by the scene point and by the optical centers. The collisions of the baseline with the image planes are the so called epipoles, e^l and e^r. The epipoles are the projections of O^l in the right image plane and O^r in the left image plane. They may also lie outside of the image or lie at infinity, when the image planes are parallel to the baseline. The intersecting lines of the epipolar plane with the image planes are called epipolar lines, l^l and l^r. An epipolar line is the image of the projection line, for instance l^r is the projection of the left projection line onto the right image plane. If viewing another scene point, the plane will traverse

differently. Thus, the epipolar lines will traverse differently in the image planes. However, every epipolar line passes the epipole, as every projection line goes through the optical center.

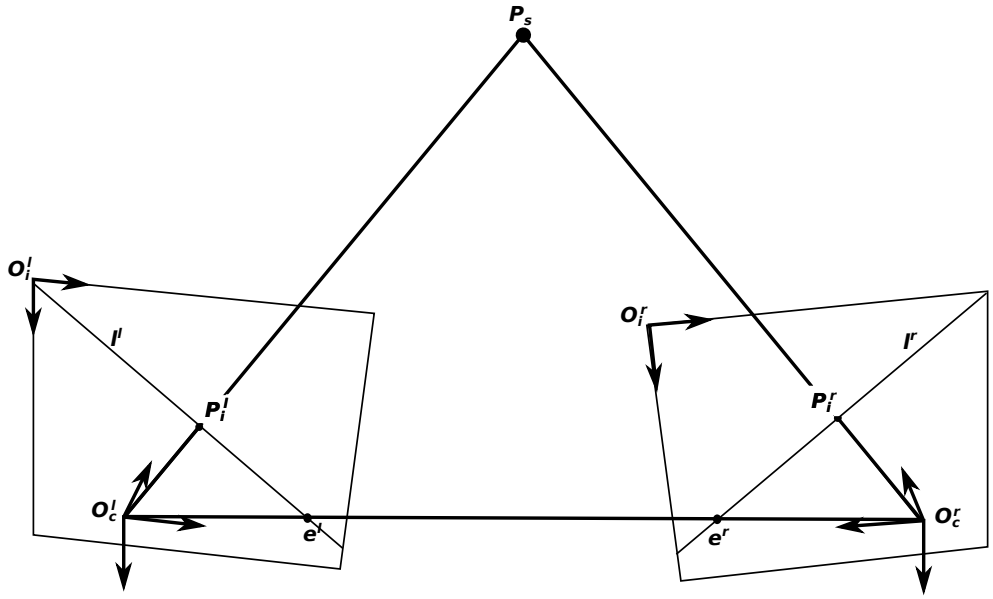


Figure 3.6: Epipolar geometry.

In fact, the epipolar geometry describes the property, that the right correspondence of the left image point of a scene point lies on right epipolar line, respectively on the left epipolar line for the right image point.

The process of finding two corresponding points is called the correspondence problem.

Correspondence and Reconstruction Problem

The correspondence problem is a complex problem in computer vision as it is still an active research topic, and solving it is the main focus of this thesis.

Given P_i^l in the left camera image, the correspondence problem is associated with finding the point in the right image P_i^r, that represents the same scene point. This is a 2D search problem along l^r, which is the projection of all possible scene points. Reversely, if P_i^r is given, the correspondence problem has to be solved along l^l.

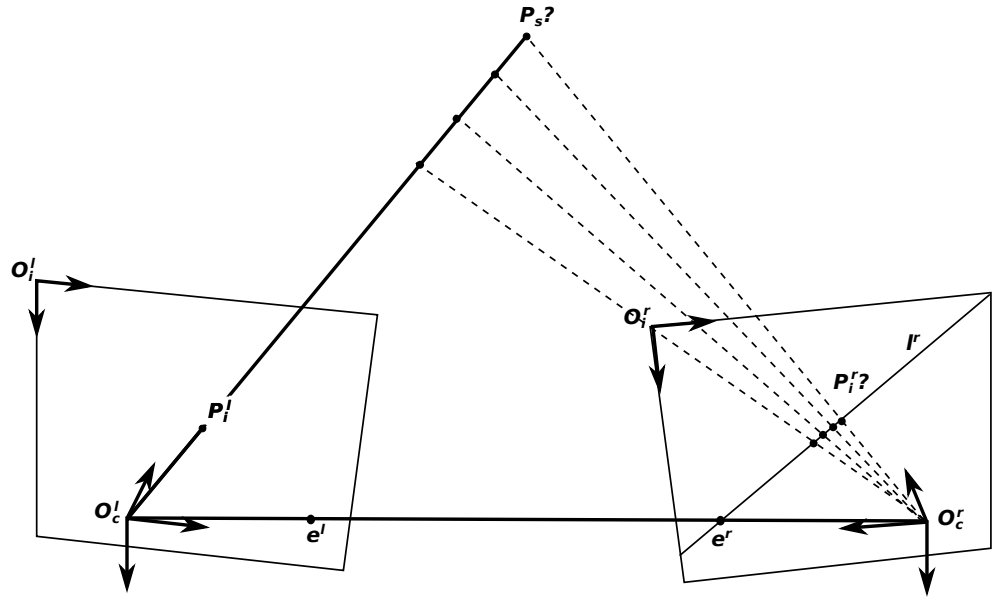


Figure 3.7: Correspondence problem.

Given a pair of P_i^l and P_i^r, the reconstruction problem is associated with finding the depth of the scene point. This may be solved by intersecting the projection lines.

With a point in the left and right image, after solving the correspondence problem, we can compute the so called disparity. The disparity is the difference in x- and y-coordinates between the left and right point and results in two disparity maps. Given the correspondence pair, one can directly compute the 3D depth, which will be the result of solving the reconstruction problem. The reconstruction problem may also be solved by taking the disparity value and by similar triangles, since the disparity value is inversely proportional to depth.

Stereo matching methods

Stereo matching methods intend to find corresponding image points in a stereo image pair. Thus, methods that solve the correspondence problem have a high computational complexity.

There are several classifications of stereo matching methods. Based on the strategy to match the points, we can divide them into global and local methods. Local methods compare a small template around a pixel in the first image, the so called matching window, with same sized windows for every pixel on the epipolar line in the second image. The best match results in a correspondence pair. It is assumed that the templates for the best matches have the same

appearance. Global methods compute a cost function for the whole image or for scan-lines. Global methods yield better matching results, but are computationally costly.

Local methods can be further subdivided into sparse and dense methods. Sparse methods match only so called feature points, e.g. edges or corners, that have been found in an image. Therefore, the saliency of a point is measured. Feature points are salient points. Sparse methods produce a sparse disparity map or depth map. In contrast to them, dense methods aim to find a corresponding point in the second image for every point in the first image, resulting in a dense disparity map or depth map.

In this thesis, local methods, which result in a sparse depth map, will be applied due to their simplicity and lower computational complexity. For these matching methods several assumptions, so called constraints, have to be made. Stereo matching methods differ by the constraints that they are using and how they implement these constraints.

Constraints

In the following the constraints [Men97, BBB+10a], that are relevant for this thesis, are described.

1. Photometric constraint: It assumes that corresponding pixels have the same intensity (or color). This constraint holds when the Lambertian model for the surface of the scene object is assumed. A surface is Lambertian, when the luminance is the same independent of the viewing direction [Hal10]. This constraint may be violated due to noise or non uniform illumination that may cause highlights on a surface.

2. Similarity constraint: Given, for instance, a line, the projections of the line should have similar properties in all images. Therefore, a horizontal line in one image cannot be matched with a vertical line in the second image. This constraint is violated, if, for instance, the line is split into two parts or is occluded in the second image.

3. Smoothness constraint: Within a matching window, the pixels are supposed to have the same disparity values. If a matching window overlaps a depth discontinuity or a surface that is slanted, then this constraint is violated.

4. Epipolar constraint: The corresponding point of an image point has to lie on the same horizontal scan-line in the other image. This is demonstrated in figure 3.8. Therefore, the image planes are assumed to be parallel to the baseline. This is the fundamental geometric constraint between the images. It may be violated when the camera's parameters are not known precisely.

5. Uniqueness constraint: Every point in the first image can have at most one point from the second image. This constraint is violated when given an image with slanted or transparent objects, as shown in figure 3.9.

6. Ordering constraint: If a point is on the left side of another point in the same image, then the ordering will be preserved on a second image. This constraint may be violated when the points appear in the forbidden zone [Huq01], which is demonstrated in figure 3.10. It does not hold, for instance, for thin foreground objects.

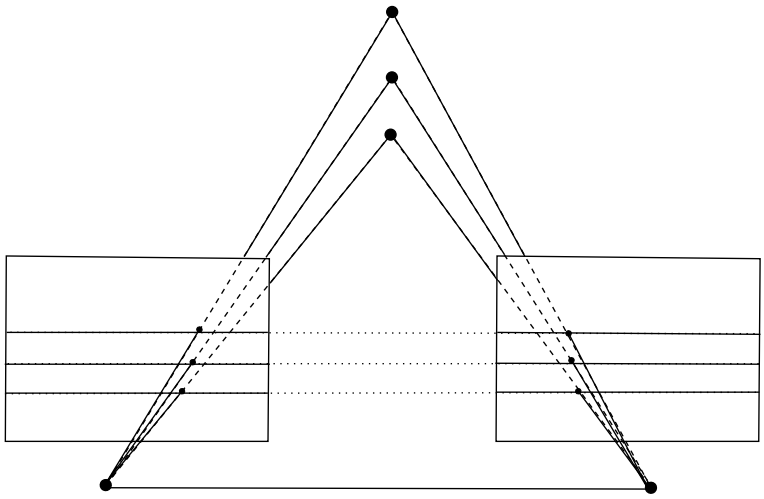


Figure 3.8: Epipolar constraint.

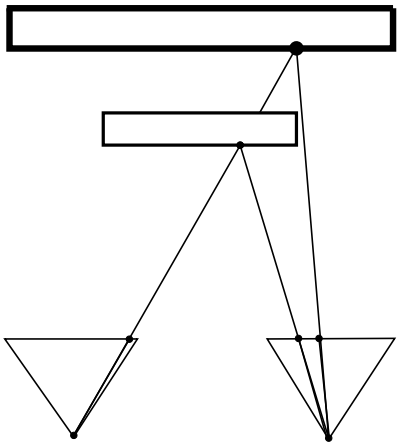


Figure 3.9: Violation of uniqueness constraint through a transparent object.

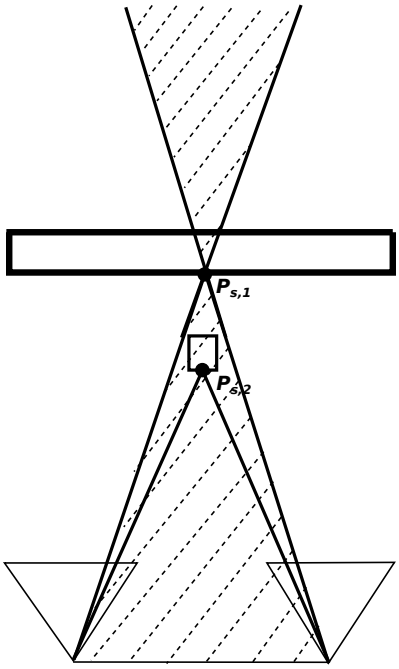


Figure 3.10: $P_{s,1}$ and $P_{s,2}$ belong to two different objects in the scene. The forbidden zones attached to $P_{s,1}$ are illustrated by the dashed lines. $P_{s,2}$ violates the forbidden zone constraint and therefore the ordering of the two projection of the scene points is reversed when comparing the left and right image plane.

Several matching problems appear when constraining the matching methods, due to violations.

Stereo Matching Problems

Local stereo matching methods generally fail when matching at homogeneous or low-textured regions [Men97]. It is impossible to find a distinct, best match in the second image. The intensity variation in the image pair is too low. Due to this problem, the uniqueness constraint may be violated. Repetitive patterns also violate the uniqueness constraint as several best matches are found along the epipolar line. Further, some points, that are visible in one image, may not be visible in the other image due to occlusion. This leads again to a violation of the uniqueness constraint and of the similarity constraint.

Perspective distortion due to perspective projection is a problem, when the baseline between the cameras is high. This violates the similarity constraint. Illumination variation leads to a situation where the similarity constraint and the photometric constraint are violated again. This may occur when the light is reflected differently, due to different views, and causes different intensity values for the projection of the same scene points [Men97]. In case of highlights, this problem is given.

If the matching window overlaps depth discontinuities, the smoothness constraint is violated. This problem worsens when the matching window increases. In this case, the object/surface in the background is measured to have the same disparity as the object/surface in the foreground. This is the so called foreground fattening problem.

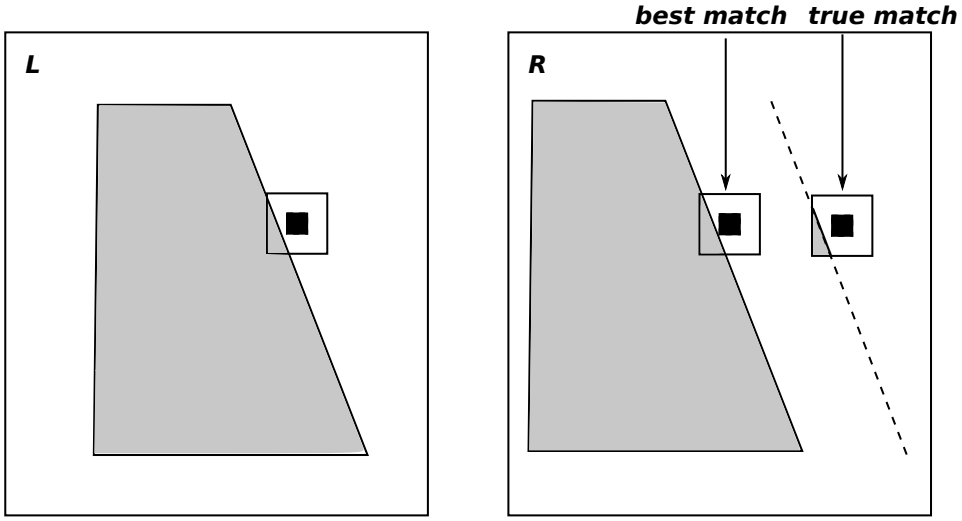


Figure 3.11: Foreground fattening problem. The gray area represents the foreground and the white area represents the background. Background pixels close to discontinuities may be matched to the foreground.

Often the surfaces in the image are assumed to be fronto-parallel to the cameras. However, given slanted surfaces, the matching may result in a n to m pixles-matching of the surfaces because, given different views of the surfaces, the surfaces are sampled differently. Slanted surfaces violate the uniqueness constraint, the smoothness assumption and the similarity constraint.

The size of the matching window is critical [Men97], as there is no optimal solution. If the window is too small the matches are unreliable, whereas, if the window is too big the foreground fattening problem tends to occur. However, low-textured regions yield for big window size, whereas reconstruction of fine details yields small window size. If the size of the matching window is big, the window will easily overlap different surfaces, which violates the smoothness assumption.

Finally, if the camera parameters are not accurately enough known, the epipolar constraint is violated, as the epipolar lines do not traverse exactly through the corresponding points.

3.2 Depth from Stereo Vision

Stereo Camera Calibration

Throughout the process of stereo camera calibration, camera parameters, that mathematically describe the relation from WCS to ICS, are gained. The relation between the scene points and the image points, through the perspective projection, can be found. This is necessary for 3D reconstruction.

For camera calibration, a known 3D structure of the scene is necessary. This can be achieved if a special calibration pattern and known control points are captured by both cameras. With an optimization process the camera parameters can be estimated [Schny].

For that purpose of stereo camera calibration, so called extrinsic and intrinsic parameters need to be calculated.

The extrinsic camera parameters need to translate the first CCS to the second CCS. In detail, they describe the external properties of the camera, the 3D position and orientation of the cameras to each other. They consist of a single 3×3 rotation matrix R and a 3×1 translation vector t, that relate the two cameras from the first to the second camera.

From this calibration data, we want to compute the rotation and translation for each camera separately. We assume that the first camera system has always the orientation and position of the WCS. This means that the rotation matrix of the first camera R_1 is equal to the 3×3 identity matrix and the translation vector t_1 is set to zero for all coordinates. The second rotation matrix R_2 is equal to the rotation matrix of R, as the rotation goes from the WCS to the second CCS. Similarly, the translation t_2 goes from the WCS to the second CCS, therefore t_2 is equal to t. This can be written as

$$R_1 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \text{ and } t_1 = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

or as

$$R_2 = R \text{ and } t_2 = t.$$

Finally, a scene point needs to be rotated and then translated to be transformed in a specific CCS with R_1 and t_1 or with R_2 and t_2. A scene point is transformed to a camera point by the following equations:

$$\begin{pmatrix} x_c^1 \\ y_c^1 \\ z_c^1 \end{pmatrix} = R_1 \begin{pmatrix} x_s \\ y_s \\ z_s \end{pmatrix} + t_1 \text{ or}$$

$$\begin{pmatrix} x_c^2 \\ y_c^2 \\ z_c^2 \end{pmatrix} = R_2 \begin{pmatrix} x_s \\ y_s \\ z_s \end{pmatrix} + t_2.$$

Intrinsic parameters are necessary to carry out the transformation from CCS to ICS. In detail, they describe the internal properties of a camera, how points of the camera system are mapped to image coordinates. They consist of the already mentioned focal lengths f_x and f_y expressed in pixels and the principal point with x_0 and y_0. Sometimes the skew is also considered but, as we assume that the axes of a pixel are rectangular, it is set to 0. Further, the intrinsic parameters consist of four distortion coefficients (k_1, k_2, k_3, k_4) to model the lens distortion of a real-world camera [The00]. k_1 and k_2 are radial distortion coefficients and k_3 and k_4 are tangential distortion coefficients. This is different to the pinhole camera, which does not consider lens distortion, as it describes an ideal perspective projection. The transformation [The00] goes as follows, whereas it is assumed that $z_c \neq 0$,

$$x_i = f_x(x_c/z_c) + x_0 \text{ and } y_i = f_y(y_c/z_c) + y_0.$$

We can also rewrite the mapping [The00] in so called camera matrices K_1 and K_2, that consists of the following intrinsic parameters:

$$K_1 = \begin{pmatrix} f_{x,1} & 0 & x_{0,1} \\ 0 & f_{y,1} & y_{0,1} \\ 0 & 0 & 1 \end{pmatrix} \text{ and } K_2 = \begin{pmatrix} f_{x,2} & 0 & x_{0,2} \\ 0 & f_{y,2} & y_{0,2} \\ 0 & 0 & 1 \end{pmatrix}.$$

The rigid motions, rotation and translation in the Euclidean space, can be expressed in so called matrices of extrinsic parameters E_1 and E_2:

$$E_1 = [R_1|t_1] \text{ and } E_2 = [R_2|t_2].$$

Finally, the ideal perspective projection can be expressed as

$$s\tilde{P}_{i,1} = K_1 E_1 \tilde{P}_s \text{ and } s\tilde{P}_{i,2} = K_2 E_2 \tilde{P}_s.$$

The transformation of CCS to ICS [The00] can be extended by the distortion coefficients as follows:

$$x' = x_c/z_c,$$

$$y' = y_c/z_c,$$

$$r^2 = x'^2 + y'^2,$$

$$x'' = x'(1 + k_1 r^2 + k_2 r^4) + 2k_3 x' y' + k_4(r^2 + 2x'^2),$$

$$y'' = y'(1 + k_1 r^2 + k_2 r^4) + k_3(r^2 + 2y'^2) + 2k_4 x' y',$$

$$x_i = f_x * x'' + x_0 \text{ and}$$

$$y_i = f_y * y'' + y_0.$$

With this camera parameters the epipolar geometry can be expressed mathematically by the fundamental matrix.

Fundamental Matrix

The relation between epipolar lines and epipolar plane is formulated by the fundamental matrix expressed in pixels. Given R_1, R_2, t_1, t_2, K_1 and K_2, the fundamental matrix is computed as follows.

First of all, the translation vector between first and second camera [MAP04], can be calculated by

$$t = R_2(-R_1^T t_1 + R_2^T t_2).$$

This translation vector is estimated by the stereo camera calibration. Further, the vector product matrix T [Schny] of the translation is built as

$$T = \begin{pmatrix} 0 & -t_z & t_y \\ t_z & 0 & -t_x \\ -t_y & t_x & 0 \end{pmatrix}.$$

The rotation R between the first and the second camera [MAP04] is defined as

$$R = R_2 R_1^T.$$

This rotation matrix is estimated by the stereo camera calibration. Finally, the fundamental matrix F [MAP04], a 3×3 matrix, is defined as

$$F = K_2^{-T} T R K_1^{-1}.$$

This matrix depends on the intrinsic and extrinsic parameters. The equation for the epipolar lines [Schny] are as follows:

$$l_2 = F\tilde{P}_{i,1} \text{ and } l_1 = F^T \tilde{P}_{i,2}.$$

For the epipolar constraint, we want to assume that the epipolar lines are parallel to each other as the image planes are parallel to the baseline. In such a case, the fundamental matrix [Schny] is defined by

$$F = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{pmatrix}$$

such that

$$l_2 = F\tilde{P}_{i,1} = \begin{pmatrix} 0 \\ -1 \\ y_{i,1} \end{pmatrix} \text{ and } l_1 = F^T\tilde{P}_{i,2} = \begin{pmatrix} 0 \\ 1 \\ -y_{i,2} \end{pmatrix}.$$

Given the definition of a straight line as $ax + by + c = 0$ with $m = -a/b$ and $d = -c/b$ of the slope-intercept form $y = mx + d$, the gradient is $m = 0$, as $a = 0$, and the intercept is $d = y_{i,2}$ and $d = y_{i,1}$ in both images. Therefore, both epipolar lines are horizontal with the same y-coordinate in the image planes [Schny].

To fulfill the epipolar constraint the image planes need to be projected in a common plane. This process is called rectification.

Rectification

This transformation is based on the camera parameters of the calibration process. After the rectification, epipolar lines of first and second image are collinear and parallel to the X_i-axis. Corresponding points lie on the same line, that goes from first to second image. Thus, it reduces the search space for stereo matching from 2D to 1D, as we only need to find the x-coordinate in the second image. Further, it reduces perspective distortion (in y-direction), that is caused by the perspective projection of different camera views. Figure 3.12 illustrates two stereo images with its corresponding rectified images. The epipolar lines no longer intersect in an epipole, as the epipole is at infinity. An important property of rectification is that the optical center of both cameras remain unchanged after rectification [Schny].

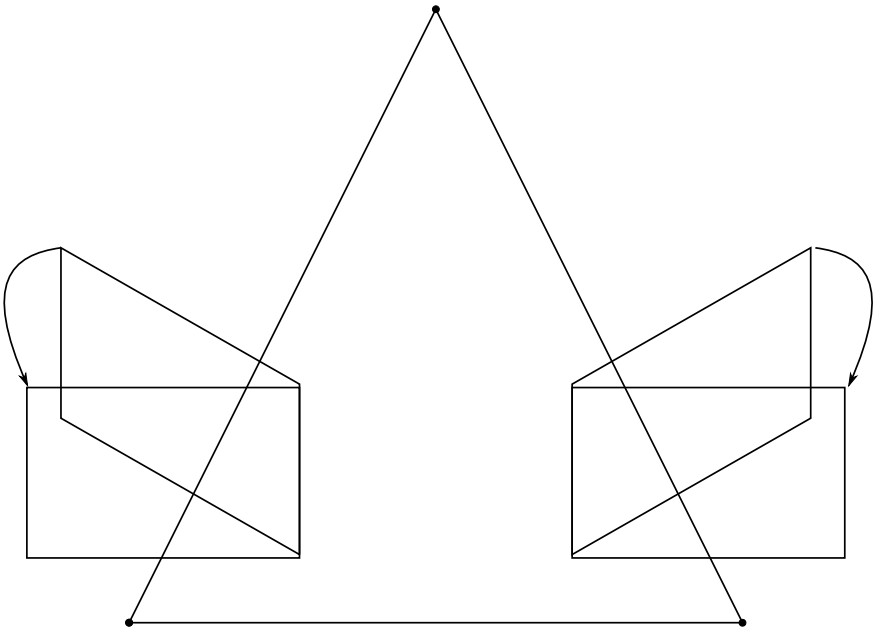


Figure 3.12: Rectification.

Given a rectified pair of corresponding points, the correspondence problem can now be formulated as

$$y_{i,2} = y_{i,1} \text{ and } x_{i,2} = x_{i,1} - d,$$

whereas d is the disparity value along the X_i-axis. We can now apply a stereo matching method on the stereo image pair.

Local Stereo Matching Methods

Local stereo matching methods use a technique known as block matching method. The similarity of matching windows is compared. To measure similarity between left and right template several metrics can be used. Two examples are intensity differences and correlation. For the purpose of correlation, color or intensity of an image can be used. We will calculate the correlation value on the basis of intensity values. The matching window is assumed to be an odd $n \times n$ window. Further, with local stereo matching methods, we will assume that the neighborhood of corresponding pixels remains constant, as stated by the smoothness constraints. Some popular metrics [BBH03] are

- sum of squared differences (SSD)
- normalized cross correlation (NCC)

With the SSD, the so called matching costs will be high, if the pixels differ a lot, or low, if the pixels are similar. The next step is to aggregate the costs in the chosen matching window, as defined for the first image I_1 and the second image I_2. $SSD(y_{i,1}, x_{i,1})$ can be formulated as follows

$$\sum_{p=-n/2}^{n/2} \sum_{q=-n/2}^{n/2} (I_1(y_{i,1}+q, x_{i,1}+p) - I_2(y_{i,1}+q, x_{i,1}+p+d))^2.$$

In contrast to SSD, with NCC, we have low cost if the pixels differ a lot and high cost if the pixels are similar. $NCC(y_{i,1}, x_{i,1})$ can be formulated as follows

$$\frac{\sum_{p=-n/2}^{n/2} \sum_{q=-n/2}^{n/2} (I_1(y_{i,1}+q, x_{i,1}+p) - \bar{I_1}) * (I_2(y_{i,1}+q, x_{i,1}+p+d) - \bar{I_2})}{\sqrt{\sum_{p=-n/2}^{n/2} \sum_{q=-n/2}^{n/2} (I_1(y_{i,1}+q, x_{i,1}+p) - \bar{I_1})^2 * \sum_{p=-n/2}^{n/2} \sum_{q=-n/2}^{n/2} (I_2(y_{i,1}+q, x_{i,1}+p+d) - \bar{I_2})^2}},$$

whereas $\bar{I_1}$ and $\bar{I_2}$ are, respectively, the means of pixel values in the corresponding regions.

SSD is simpler and therefore, computationally less expensive than NCC. However, NCC is insensitive to radiometric gain and bias [BBH03] due to its normalization. There exist several variation of SSD and NCC. An example is sum of absolute differences (SAD), that is computationally even more efficient than SSD. NCC may vary due to different normalization techniques. An example is zero mean normalized cross correlation (ZNCC) [MR10].

With these measures, the cost aggregation for every point on the epipolar line is computed. A winner-takes-all principle (WTA) is applied to find the best match of the matching cost. Often, a threshold for the correlation value is set manually to reduce the risk of false matches.

However, this method relies on images having textured regions and fails in low textured regions. The choice of size of the matching window is critical as the performance of the outcome and the computational time depend on it. Moreover, depth discontinuities cause problems due to the square window. Therefore, several alternatives have been proposed. An example is the method of Hirschmüller [Hir01] that uses instead of one matching window for each pixel, several sub-windows, and computes the correlation value by adding the values of the best sub-windows.

To improve the matching quality, the search space can be reduced by assuming an upper and lower bound of depth or disparity. Sometimes, the epipole is taken to compute the lower bound. However, having rectified images, this is not an option. Often, depending on the baseline and on the distance of the scene objects, the search space is restricted manually. Thus, the possible disparity value d will range from d_{min} to d_{max}.

Reconstruction of Depth

Given the position and orientation of the cameras as well as their parameters, we can reconstruct the projection lines. For each line and camera a projection of the pixel into the scene, $P_{s,1}$ and $P_{s,2}$, can be computed. Getting the center of $P_{s,1}$ and $P_{s,2}$, we will receive the scene point P_s and can take its z_s as result.

First of all, we need to define the inverse projection from ICS to CCS. Therefore, we need the inverse K_I of a camera matrix as

$$K_I = \begin{pmatrix} 1/f_x & 0 & -x_0/f_x \\ 0 & 1/f_y & -y_0/f_y \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{pmatrix}.$$

K_I can be applied for the first and second camera matrix, resulting in $K_{I,1}$ and $K_{I,2}$. Then, we need to transform back into the WCS with an affine transformation

$$W = \begin{pmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \\ 0 & 0 & 0 & 1 \end{pmatrix},$$

whereas, the rotation from camera to scene, is defined as

$$R^T = \begin{pmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{pmatrix}.$$

Further, the translation $-R^T t$ from camera to scene, is defined as

$$-R^T t = \begin{pmatrix} t_1 \\ t_2 \\ t_3 \end{pmatrix}.$$

W, R^T and $-R^T t$ can be applied for the first and second camera, resulting in W_1 and W_2. With $K_{I,1}$, $K_{I,2}$, W_1 and W_2 a pixel can be projected into WCS as

$$\tilde{P}'_{s,1} = W_1 K_{I,1} \tilde{P}_i \text{ and } \tilde{P}'_{s,2} = W_2 K_{I,2} \tilde{P}_i,$$

which are scene points computed with lost depth information. However, as the projection lines traverse through these points, we can define the directions of the projection lines with $l_1 = P'_{s,1} - t_1$ and $l_2 = P'_{s,2} - t_2$, with t_1 and t_2 being equal to the projection center of the cameras.

$P_{s,1}$ and $P_{s,2}$ can be computed by

$$P_{s,1} = t_1 + s_1 l_1 \text{ and } P_{s,2} = t_2 + s_2 l_2,$$

whereas s_1 and s_2 are defined as

$$d = t_1 - t_2,$$

$$a = l_1 l_2 / (l_1)^2,$$

$$b = d l_1 / (l_1)^2,$$

$$vec = a l_1 - l_2,$$

$$s_1 = (b l_1 - d) * vec / (vec)^2 \text{ and}$$

$$s_2 = a s_1 - b.$$

Finally, P_s can be computed by

$$P_s = \begin{pmatrix} (x_{s,1} + x_{s,2})/2 \\ (y_{s,1} + y_{s,2})/2 \\ (z_{s,1} + z_{s,2})/2 \end{pmatrix},$$

whereas z_s of P_s is the resulting depth.

CHAPTER 4

Depth Reconstruction of Faces

4.1 Scene Description and Difficulties

For our scene we assume to see one frontal human face with the first camera. If several faces in the scene appear, the largest will be reconstructed, if the images are not marked beforehand. The expression of the face is assumed to be neutral but some expressions like smiling do not make a difference for computing the depthmap. Further, we constrain that there is no rash movement in the scene, such that motion blurring is avoided. The background should differ significantly from the face, in specific from the skin color.

The reconstruction of faces is especially challenging as the the texture-variation is rarely noticable. Therefore, often structured light, special makeup or markers are used to reconstruct the geometry. When using HD resolution, fine-scaled details, like facial hair, wrinkles, blemishes and so on may help to find automatically correspondences as they give a natural surface texture of the face.

Figure 4.1: Natural texture of the skin with pores, wrinkles and other blemishes.

Noise and other environmental aberrations may cause a lack of texture, that makes it difficult to identify and reconstruct feature points. We have to pay attention to bad lighting, as it may reduce textural information in the image pair. Especially, when reflections appear, one loses important details. The tip of the nose and the eyes are prone to reflections when captured under direct lighting.

Figure 4.2: Reflection in the eye, that are difficult to avoid.

One can avoid such occurrences by using make-up, special lighting or post processing. When designing the lighting, one needs not only to cope with reflections, but also with shadows.

The age may be important for the quality of the reconstruction. With the advance of the age, the skin becomes uneven and gains textural information. In fact, the reconstruction of facial hair, eye geometry, including eyelashes and eyebrows, or even the teeth and the tongue would give more realistic 3D results. However, our method is not designed for such a task.

The camera images are shot under special lighting conditions. For further information, see the subsection 5.1.

4.2 Stereo Calibration

The first step, to reconstruct the image, is to apply a stereo calibration. For the stereo calibration the already implemented method of OpenCV was used. Therefore, a checkerboard pattern, with black and white squares attached to a planar surface, is held in front of the camera and recorded.

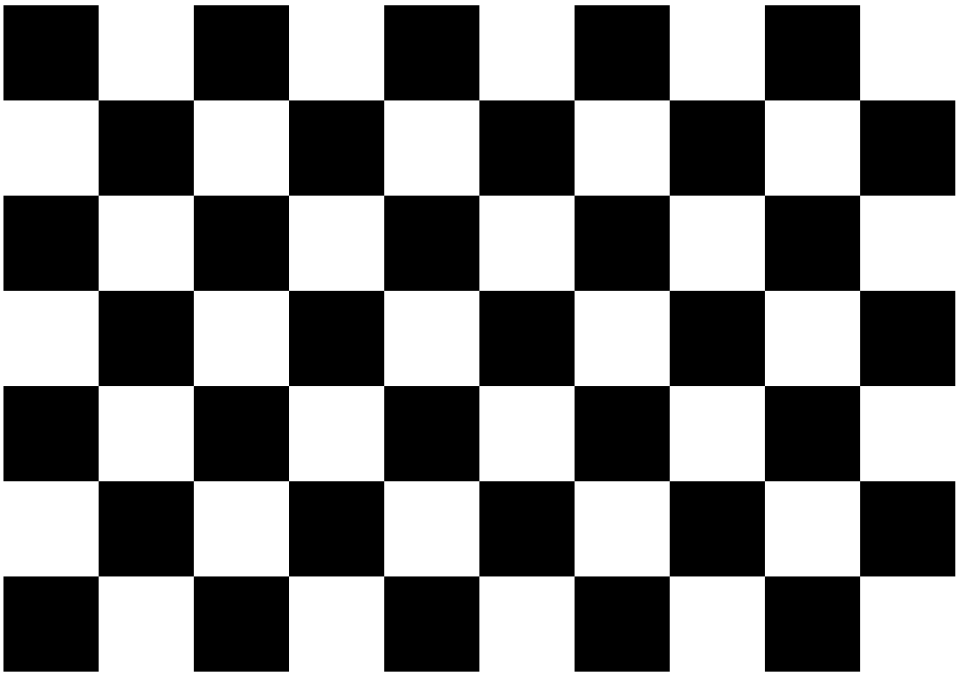


Figure 4.3: Calibration pattern: a checkerboard pattern.

In general, any object with appropriate features can be used as a calibration object [BKSA]. However, it is much easier to deal with planar checkerboard pattern, as for instance with a 3D calibration object.

For the checkerboard corners, we used a checkerboard pattern of 10×7 fields which results into 9×6 corner points (just counting the internal corner points). However, three corner points in a known pattern would be sufficient to solve the problem. Generally, one needs $2NK \geq 6K + 4$ [BKSA], whereas N are the corners and K are the images of the checkerboard pattern. Then we have 6*K extrinsic parameters and 4 intrinsic parameters. Further, even though one image would be, as stated before, sufficient, several images, under different orientation by moving the plane, were taken for this thesis. In practice, considering for noise and numerical stability, we took around 40 pairs of images for robustness, as this yields high-quality results.

First of all, feature points need to be detected in the image. This process involves a corner detection by the Harris interest point operator [BKSA]. A further refinement is done by an iterative subpixel localization with a gradient-based search, whereas the resulting grid corners of the checkerboard pattern are advantageous for the subpixel localization [BKSA].

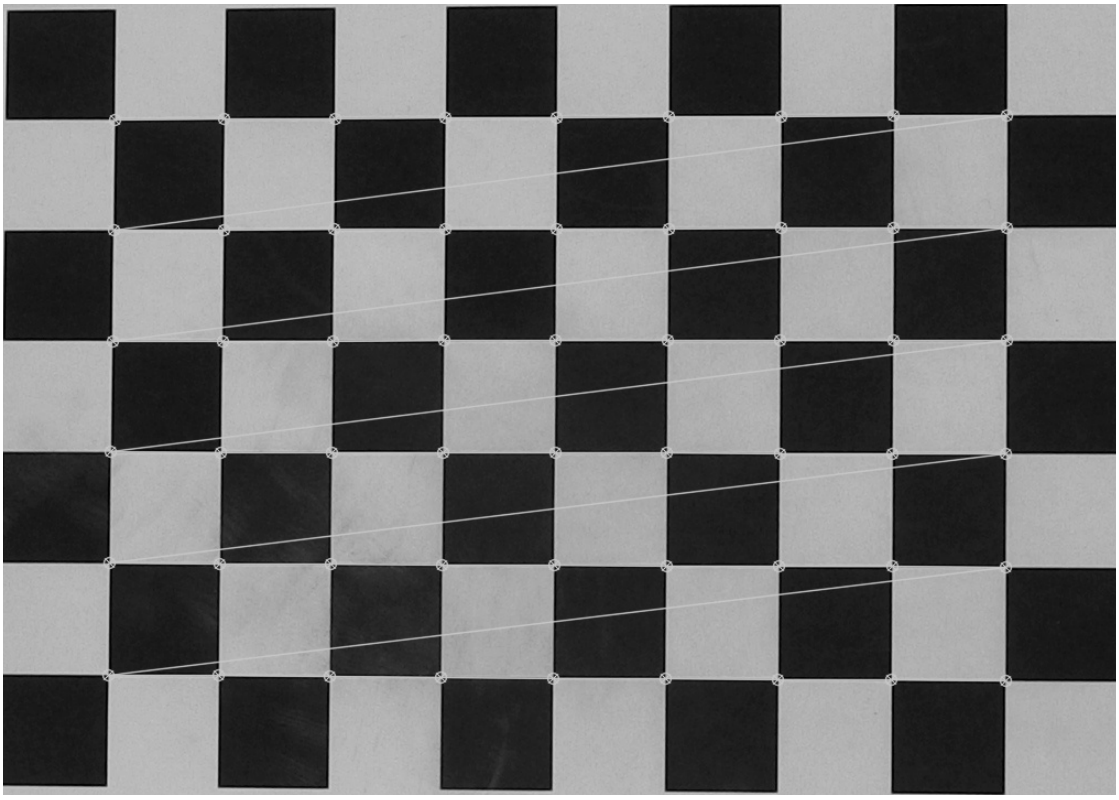


Figure 4.4: Detected corner through the Harris interest point operator and after a refinement by an iterative subpixel localization in the calibration pattern.

For the calibration method OpenCV uses Zhang's method [Zha00] to solve for the focal lengths and offsets. However, the distortion parameters are solved based on Brown's method [Bro71].

4.3 Preprocessing

Undistortion

During the calibration process radial and tangential distortion parameters are computed. There are several other distortions that may occur in the picture taking process, but these are the typical ones. Radial distortion arises as a result of the shape of the lens. Rays farther away from the center of the lens are bent more than those which are closer to the center. This causes so called barrel distortion, which is especially obvious in cheap web cameras. Tangential distortion is caused by manufacturing defects, when the lens is not exactly parallel to the image plane [BKSA].

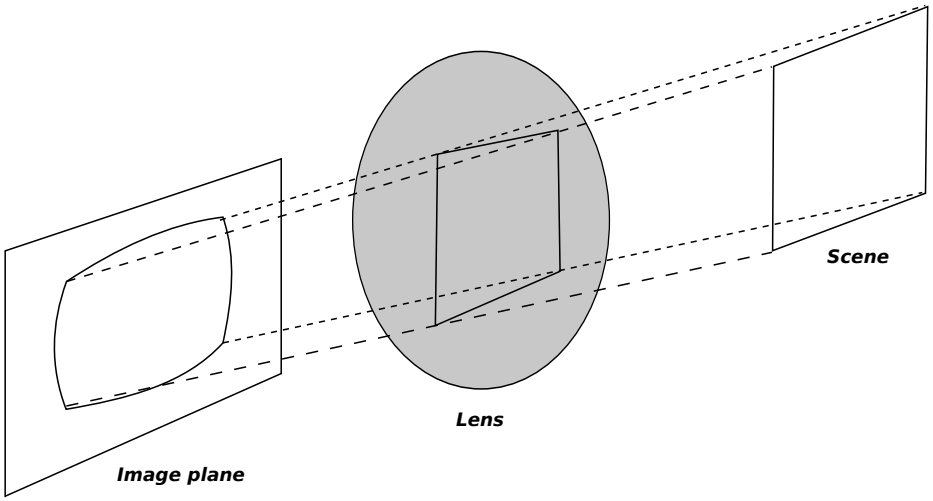


Figure 4.5: Radial distortion, as rays farther from the center of a cheap lens are bent too much [BKSA].

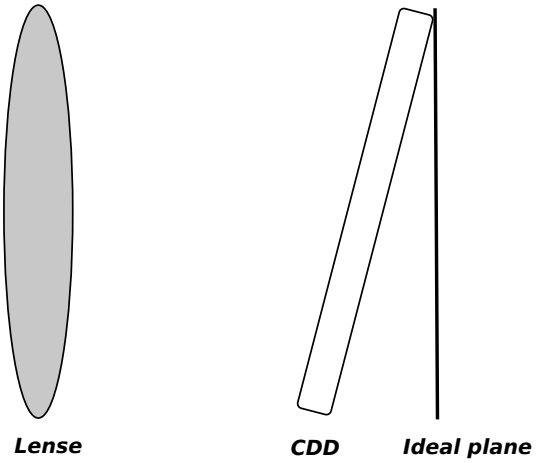


Figure 4.6: Tangential distortion as the camera sensor is not fully parallel to the lens [BKSA].

With the distortion parameters, the images are transformed to compensate for lens distortion with a method from OpenCV. The input camera matrix is changed for the distored image. Bilinear interpolation is applied for filling in the holes.

Rectification

After distortion is compensated, the image needs to be rectified. Again a method of OpenCV is consulted. There are many methods to rectify an image pair. OpenCV implements Bouguet's algorithm for the calibrated case, as it uses the rotation and translation parameters from two calibrated cameras. As this method has never been published, only implemented in the Camera Calibration Toolbox for Matlab [Jea], further details can be taken from [BKSA].

Spatial Restriction for the Reconstruction

To restrict the computational area of the correlation process, we have the option to mark the interesting region by surrounding it with a black frame. The bounds are detected automatically. Setting the frame may improve the computational time.

Figure 4.7: Interesting regions should be surrounded with a black frame.

Further, if the frame is not set, OpenCV's face detector is used. OpenCV implements a face detection method [BKSA] similar to the work of Paul Viola and Michael Jones, also known as the Viola-Jones detector [VJ01]. Later it has been extended by Painer Lienhart and Jochen Maydt [BKSA], also known as the diagonal features [LM02]. In OpenCV this detector is called "Haar classifier" as it uses "Haar-like wavelets that consist of adding and subtracting rectangular image regions before thresholding the result" [BKSA]. During the detection process a rectangular window of specific size is moved over the input image, while for each of these sub-areas the Haar-like features are computed. The result is compared to a learned threshold that classifies the object into a face or non-face. OpenCV offers pretrained object-recognition files for faces, one for a frontal face and one for a profile face. We are using both in our implementation, as the first camera sees the face in frontal position and the second and third cameras obliquely from the side. The detection works sufficiently well on objects that are "consistently textured and mostly rigid" [BKSA].

In case of not finding the bounds, the whole image serves as input for the correlation phase.

Thresholding

Before we consider correlation, we need to compute the saliency of the image pixels for the left image. The saliency may be defined as being inversely proportional to the probability of the occurrence of the image feature [HLS02]. Mismatches during the correlation process, due to noise or lack of texture, will result in uncorrelated noise in the 3D geometry. To reduce these points the saliency is used. In the following description, pixels are processed depending on their contained information. To avoid mismatches due to low textured regions the correlation process will start with pixels, that are highly textured and will end with pixels that are less textured. In specific, we will use the Sobel operator.

The Sobel operator is a linear filter used for edge detection. It computes an approximation of the gradient of the intensity function of the image. A gradient is a vector that represents the edge magnitude with its length and whose direction is orthogonal to the edge direction [Tönny]. Or in other words, the gradient represents the steepness and the direction of the slope.

The Sobel operator is defined in horizontal and vertical direction as follows:

$$G_x = \begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix} \text{ and } G_y = \begin{pmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{pmatrix}.$$

With this small and integer valued 3×3 kernels, that gives the approximation of the derivatives in horizontal and vertical direction, a convolution of the image can be computed. The result gives the information of how abruptly or smoothly the image intensities change at a point. At homogeneous regions the filter will result in zero [Tönny].

The resulting gradient approximations of M_x^{bgr} and M_y^{bgr} are combined to give its edge magnitude as

$$M = \sqrt{(M_x^{bgr})^2 + (M_y^{bgr})^2},$$

whereas M_x^{bgr} and M_y^{bgr} are again combined for every color channel of the color space BGR, that is used by OpenCV, as

$$M_x^{rgb} = \sqrt{(M_x^b)^2 + (M_x^g)^2 + (M_x^r)^2} \text{ and}$$

$$M_y^{rgb} = \sqrt{(M_y^b)^2 + (M_y^g)^2 + (M_y^r)^2}.$$

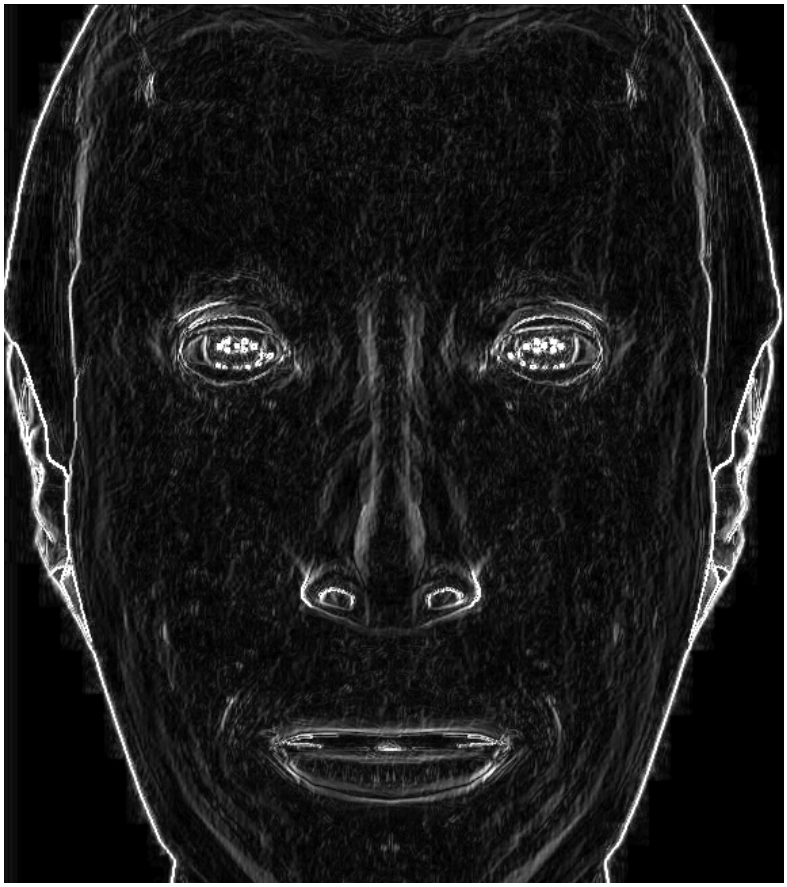


Figure 4.8: Combined gradient image from horizontal and vertical Sobel operator. The value ranges from 0 to 255

We receive a gradient magnitude for every pixel, which expresses the saliency of the image. Further, the maximum is found and the minimum is set to 20. The threshold for every iteration of the computation is calculated as follows

Listing 4.1: Threshold Computation

```
min = 20;
max = getMax(sobelFilterdLeftImage);
step = (max-min)/3;
for(i=1;i<4;i++)
    threshold = min + step(3-i);
    ...
end
```

When performing the thresholding, we receive three different results for the gradient magnitude.

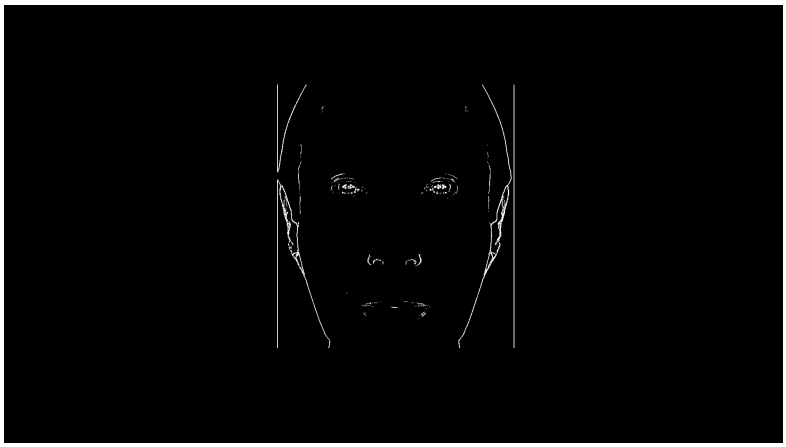


Figure 4.9: The salient pixels in the first iteration of the matching taken from the first image.

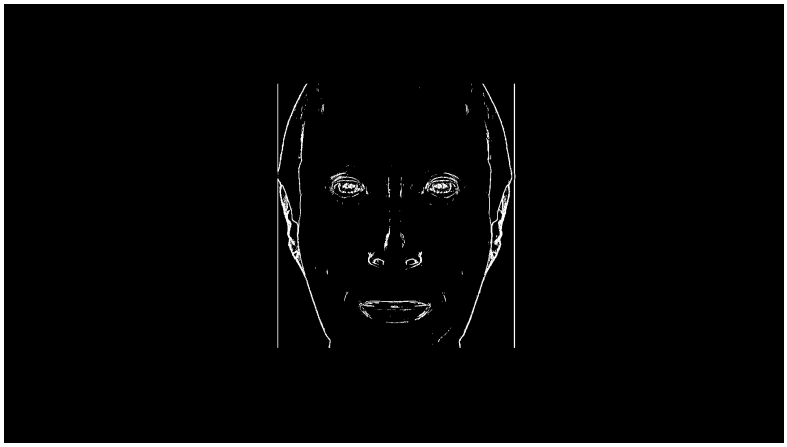


Figure 4.10: The salient pixels in the second iteration of the matching taken from the first image.

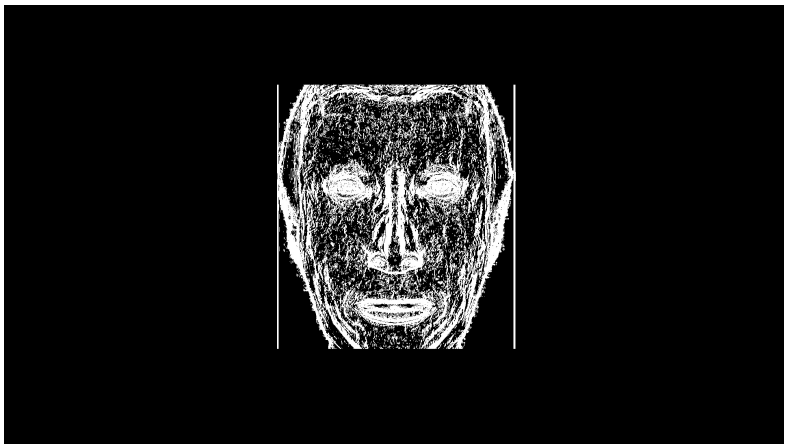


Figure 4.11: The salient pixels in the third iteration of the matching taken from the first image.

4.4 User Input

For the local sparse stereo matching method user input is necessary. The idea is to reduce the search space for the image matching step by defining a bounding area in space for the frontal side of the face. Therefore, we need to tell the application, where the closest point and the farthest point of the face are. With this information we can restrict the search space on a projection ray, that goes through the image point and the optical center.

Figure 4.12: The user input in the first image is marked in this example with red crosses.

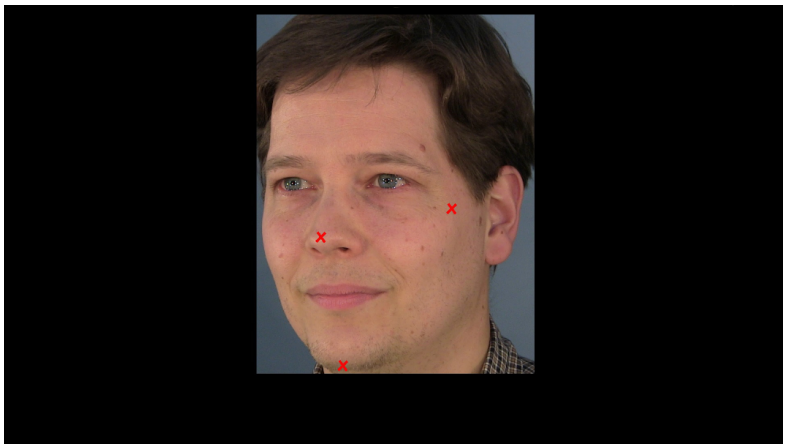


Figure 4.13: The user input in the second image is marked in this example with red crosses.

When seen from the camera, the nose tip is assumed to be the closest point of the face. The farthest point may be defined as a point on the right edge of the face of the input image. For robustness reason the chin point is also required, as the input point serve as initial input for the stereo matching method (see the next section 4.5). The bottom line is an input of six points for each pair of input images. Therefore, we will have three inaccurate correspondences. With these, three scene points, approximately lying on the face in WCS, can be calculated. As a face has bilateral symmetry, its left and right side are symmetrical. Due to this property, three pairs of corresponding points are sufficient. However, it is possible to increase the number of input points.

4.5 A Local Sparse Stereo Matching Method

Iterative process

There are three iterations over the left image to compute the depthmap. Depending on the threshold, explained in section 4.3, an image pixel is considered for the local sparse stereo matching method or is excluded. In the first iteration, the corresponding pixel is computed for very salient pixels. The threshold of the pixel is decreased by every iteration. We have three iterations, as experiments have shown that the algorithm performs well at three iterations.

Structure pixels

In general, the idea is to compute so called “structure pixels“. The initial input are three structure pixels, as we have three pairs of corresponding pixels given by the user’s input. In principle, our algorithm works as follows: for each currently selected pixel a viewing ray is built up with this pixel and the optical center. It is traced back into the scene. As all points on this ray are potential candidates for the projection onto the second image plane, a restriction needs to be formulated. We take the scene point of the nose tip as the closest bound on the ray and the point selected on the edge of the face as the farthest bound. These restrictions are further improved, when the application is progressing, as more structure pixels are computed. The bounds are computed from the closest structure pixel.

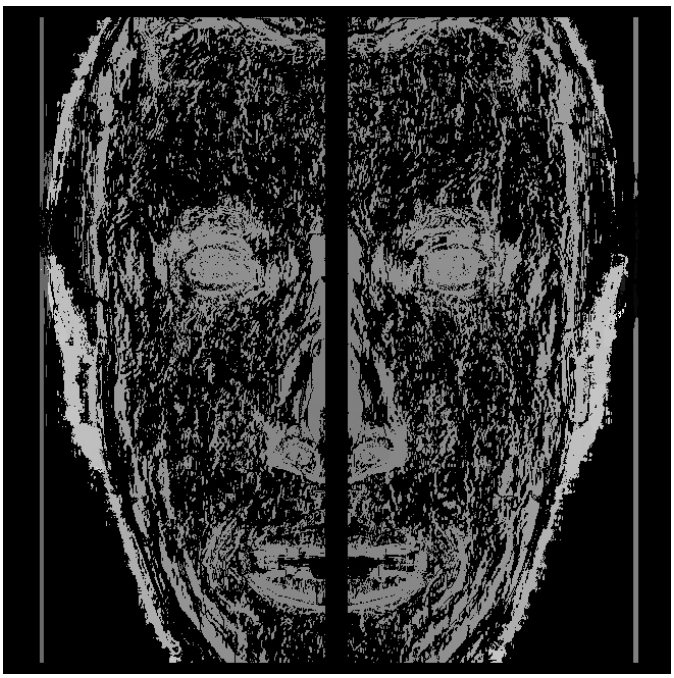


Figure 4.14: Example of structure pixels after the third iteration.

Restriction of Search Space

Subsequently, depending on the depth z of the closest neighbors (our structure pixels) of a currently selected pixel, the search space is restricted by $[(1-\lambda)*z, (1+\lambda)*z]$. The value of λ is crucial for the performance of the matching method. The lower the value, the more restricted is the search space and the more we assume that neighboring pixels have similar depths. This may cause mismatches at depth discontinuities like on the transition between cheek to nose. In contrast to this, if the value is chosen too high, mismatches are likely to appear as it is likely to find a similar pixel in the neighborhood in the second image.

Further, with every iteration, the restriction of the search space is tightening up, as follows.

Listing 4.2: Tightening up of Restriction

```
1              for(i=1;i<4;i++)
2                  ...
3                  interval = λ/i;
4                  minDepth = (1-interval)*z;
5                  maxDepth = (1+interval)*z;
6                  ...
7              end
```

This is necessary, as with every iteration more closer structure pixels are available. Therefore, the accuracy of the neighboring depth is higher, which allows to restrict the search space better.

With this method, the smoothness assumption is met better.

Normalized Mean-Squared Differences

When the bounds are fixed, we rasterize the viewing ray between the maximum depth and the minimum depth, so every hundredth point is projected into the second image plane. Such a projection in the second image plane is called a potential corresponding pixel. To find the best correspondence, a correlation value needs to be computed. For every potential pixel a window is centered around it and it is compared to a fixed template of the pixel in the first image.

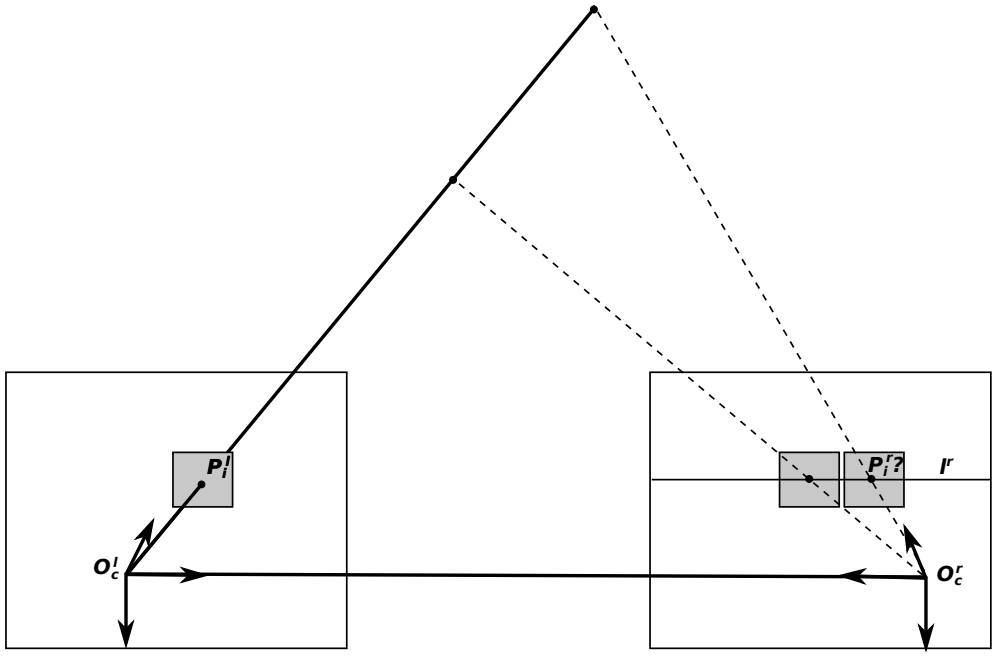


Figure 4.15: In the left image frame a fixed template and in the right image frame a shifting window, depending on the position on the viewing ray, is presented.

The correlation value is computed by the fixed window in the first image and the shifting window in the second one. Every step on the rasterized ray requires the computation of a correlation value. In our approach, we take normalized correlation of intensity values, the so called normalized mean-squared differences [Fua93], and compute the correlation value c by

$$c = \frac{\sum_{i,j}((I_1(y+j,x+i) - \bar{I_1}) - (I_2(y+dy+j,x+dx+i) - \bar{I_2}))^2}{\sqrt{\sum_{i,j}(I_1(y+j,x+i) - \bar{I_1})^2 * \sum_{i,j}(I_2(y+dy+j,x+dx+i) - \bar{I_2})^2}}.$$

I_1 and I_2 are the first and the second image intensities. $\bar{I_1}$ and $\bar{I_2}$ are the average value over the correlation window and dx and dy are the displacements along the epipolar line, which are proportionally to our step size on the rasterized ray. In [Fua93], it is argued that this correlation values are insensitive to linear transformation of the image. This may be caused by slightly different settings of the cameras. Therefore, with this method, the similarity constraint is met better.

Variable Window Size

Further, in [Fua93] it is argued that the probability of mismatches decreases as the size of correlation window increases. Therefore we do not use a fixed window size, but a varying size, that depends on the saliency of the pixel in the first image. A combination that was often used throughout the testing was the following: for the first iteration, the window size was set to 9, for the second iteration a size of 15 was used and for the third 21 was used. For further details see the section in the evaluation chapter 6.3. The size of the window is increasing as the saliency of the pixel is decreasing. However, using large windows is problematical as the fine-detailed geometric information gets lost, whereas small windows are not distinct enough for correlation.

Using this method, we try to avoid partially the foreground fattening problem and to gain some fine-detailed structures in the depth map.

Considering the Neighboring Viewing Rays

An improvement, that we have implemented is not only to consider one pixel in the first image for the stereo matching but to include several pixels from the adjacent area. Therefore, a distance of $+/-1$ for the pixel position was chosen.

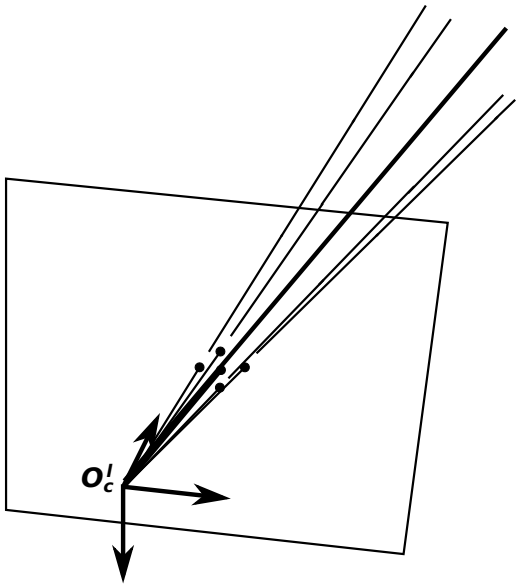


Figure 4.16: The 4 neighbours in the first image that are also considered for stereo matching.

As illustrated in figure 4.16, a bunch of viewing rays is traversed. Further, a common correlation weight for every step on the central rasterized rays is calculated. The common weight is simply calculated by the sum of weights. The correlation values from the neighborhood are weighted by a Gaussian filter. Therefore, correlation values closer to the central pixel have more influence on the common weight than pixels that are farther away. The best match is determined by the minimal common correlation weight.

With this method, we try to implement an approach to partially fulfill the smoothness assumption.

Left/Right Consistency Check

As the restriction of the search space may cause mismatches during the third iteration, due to lack of saliency, we implemented a left/right consistency check, which can be considered as a validity measure. Therefore, the roles of the images are reversed. Once a potential pair of corresponding points is found, the same stereo correspondence process is applied for the found pixel in the second image. For this pixel and its neighborhood, again the best matching pixel is determined by the minimal common correlation value.

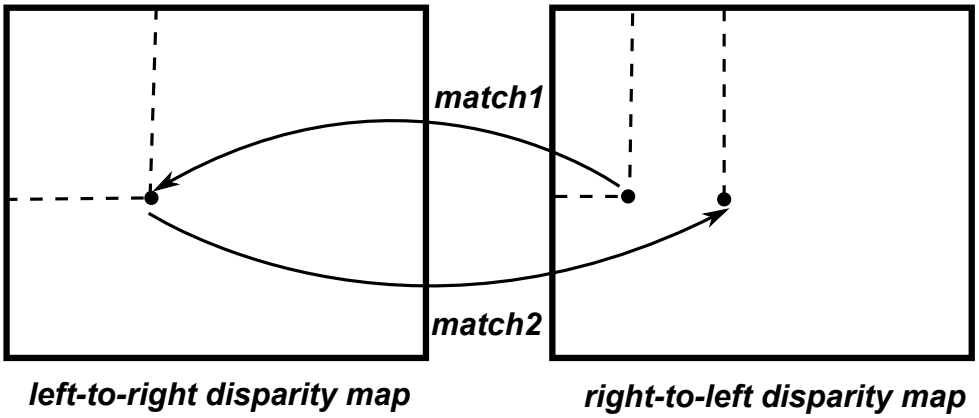


Figure 4.17: The right-to-left disparity map points back to a pixel in the left image. This pixel should point back to the outgoing point in the left-to-right disparity map. As it does not fulfill this condition, the found match is invalid and will be ignored.

The resulting depthmap is sparse, as the same initial pixel, for which the correspondence is searched, has to be found again after the reversion. If it differs in the pixel position the corresponding pair is rejected.

The left/right consistency check can be defined as follows: given a pixel $P_{i,1}$ in the first image I_1 and its neighborhood, after the stereo matching process, let $P_{i,2}$ be the potential corresponding point that was found in the second image I_2. This match is valid, if and only if, the reverse matching process with $P_{i,2}$ from I_2 to I_1 locates again the pixel $P_{i,1}$.

With this method, we try to fulfill the uniqueness constraint.

Calibration Imperfections

As for some calibration parameters, the course of the epipolar line lacks pixel accuracy as it does not go exactly through the corresponding pixels, another improvement was implemented. Instead of matching the template just with the pixel that lies on the epipolar line, we do the same for three pixel above and three pixel below this pixel. This is easy to do, as the images are rectified. Therefore, we reduce again the probability of mismatches.

With this method, we try to cope with the violation of the epipolar constraint.

CHAPTER 5

Setup

5.1 Setup Configuration

Computer Vision Library and IDE

For our application we used OpenCV [The00], an open source computer vision library, that is written in optimized C and C++ [BKSA]. It runs under several plattforms (Linux, Windows and Mac OS X). Therefore, our source code is also written in C++.

In general, OpenCV is a collection of classes that gives support for image processing. It consists of several functions for real-time calculation. OpenCV is structured into five main components, one of it is "cv" for image processing and vision algorithms. Moreover, it can take advantages of multicore processors.

Its alpha release was published in 1999. Since then, OpenCV has been used in many applications and for several research work. The library contains over 500 routines that give solutions for areas including camera calibration and stereo vision [BKSA].

OpenCV was used for this thesis, as it has many routines already implemented and offers a good framework to work with cameras and to perform mathematical operations.

OpenCV was designed to be portable [BKSA] and it was written to compile for instance with MSVC++. As a free version for students was available, we used the commercial product Microsoft Visual Studio 2010 Ultimate as our integrated development environment (IDE). It offers tools for developing and debugging of C++ code.

Optimization

As the execution time of our program was long, we decided to parallelize the core step in the computation. We divided the salient pixel of the left image into several windows, whereas every window was calculated in parallel to the other. Therefore, two arguments have to be given to the program: $d1$ and $d2$. $d1$ divides the image in horizontal direction and $d2$ divides the image in vertical direction, resulting in several windows.

For our evaluation, $d1 = 2$ and $d2 = 1$ was chosen, which results only into two windows. This weak parallelization was necessary, as we had just two cores on the testing machine.

Stereo System

For the stereo system we used three HD camcorder. In specific, we used Canon LEGRIA HF M 46. One of its benefits is the Canon HD CMOS Pro Sensor with a total pixel of 2.37 Megapixel. The sensor uses a Bayer pattern filter. Moreover, it has a high-quality Canon HD Video lens with 10x optical zoom and 40x/200x digital zoom. The focal length is 6.1 - 61.00 mm and the maximum aperture is f1.8 to f3.0.

Camcorder and Lighting Setup

To receive images of faces that are brightly and uniformly illuminated, we built our own lighting setup, shown in image 5.1 and 5.2, that is similar to a light dome.

First of all, a fundamental setup was built out of curved fiber board with the approximate dimensions of 4 x 1 x 1 in meters. To keep its curved shape, we stabilized it with hooks and a steel rope. To mount the three camcorders, we set up three small pedestals and carved three round holes in the board. The camera's field of view wasn't narrowed down by this setup. The baseline between the two camcorders was approximately 30 cm. The distance to the object was about two meters, whereas the zoom was used to approach to the face. Finally, the lighting was realized by twelve swiveling panel mounted light fixtures. For these, again several round holes were carved in the board, whereas the positions were equally distributed. Further, two studio light lamps were set up from above to lighten up the upper area. This setup provides high-intensity light with uniform light distribution, like with a LED panel. Most of the reflections are avoided, in spite of using directed light. Only reflections in the eye were not possible to avoid. Another disadvantage of using regular light fixtures is the radiated heat.

For the camcorder in the center of this constellation, the depthmap is to be calculated, therefore it is equivalent to our first camera. The left and right camcorders are always equivalent to the second camera. For the first stereo pair of images, the left camcorder is the second camera, whereas for the second stereo pair the right camcorder is the second camera.

Figure 5.1: The lighting setup with the cameras from the back.

Figure 5.2: The lighting setup with the cameras from above.

5.2 Taking Images

Our three cameras are directed at the same scene in front of the cameras. Here it was a frontal human face. However, several assumption are made before taking the images.

1. the camcorders should be identical
2. the focal length should be identical
3. the zoom level should be identical
4. the white balance of all camcorders should be set manually
5. the camcorders should be aligned horizontally to each other
6. the baseline should not be too wide, so that the field of view is similar
7. the object in front of the camcorder does no sudden movements, as to avoid motion blurring

The camcorder was set to snapshot mode. Moreover, pictures were recorded in full HD resolution of a capture image size of 1920 x 1080. Therefore, fine details like pores, wrinkles and moles are visible in the images. The pictures were shot simultaneously with a remote controller of the camcorders. The controller worked for all three camcorders at the same time.

Figure 5.3: The lighting setup with a model in front of the camera, while taking pictures.

5.3 Calibration Process

For the calibration process the method by Zhang [Zha00] was used. For more details, see the previous chapter 4. A checkerboard pattern of known dimensions needs to be held in front of the camera and recorded by the camera. It is important that the background of the taken images is uncluttered, otherwise the method will fail. The stereo calibration process provided by OpenCV was used. As several pairs of images are necessary for the calibration process, the left and middle camcorders are calibrated separately as well as the right and middle camcorders.

For this process, two persons are necessary. The first person is handling the checkerboard pattern in front of the camcorders. The second person is handling the remote controller from behind the camcorders. To achieve a satisfying calibration accuracy, about 40 pairs of images from various distances need to be taken. Some examples are shown below.

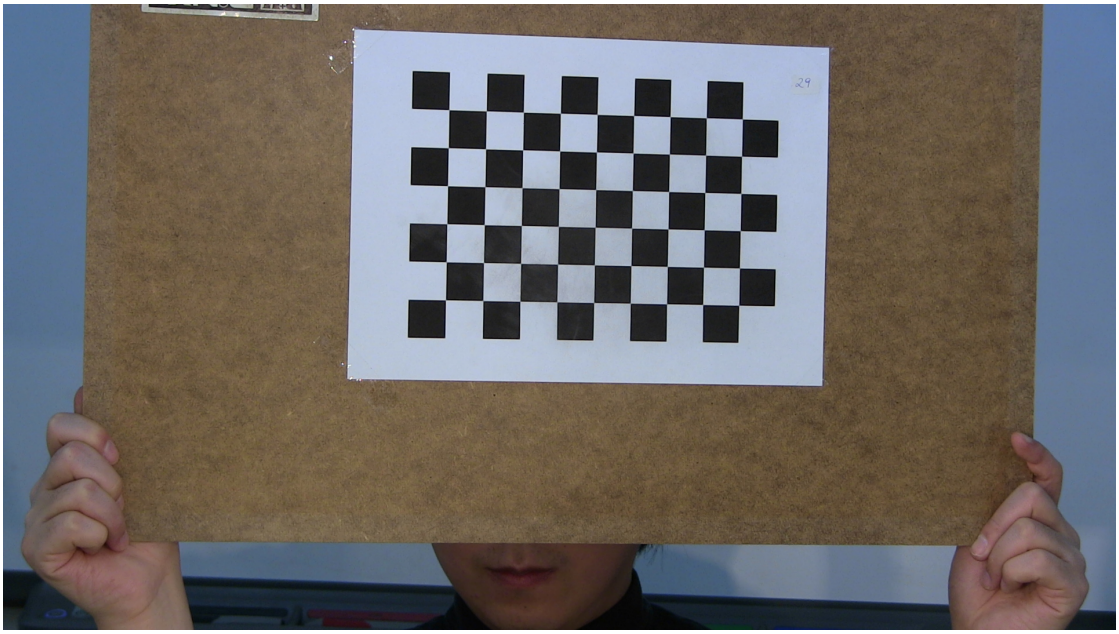


Figure 5.4: The calibration pattern during the calibration process, seen from the first camera.

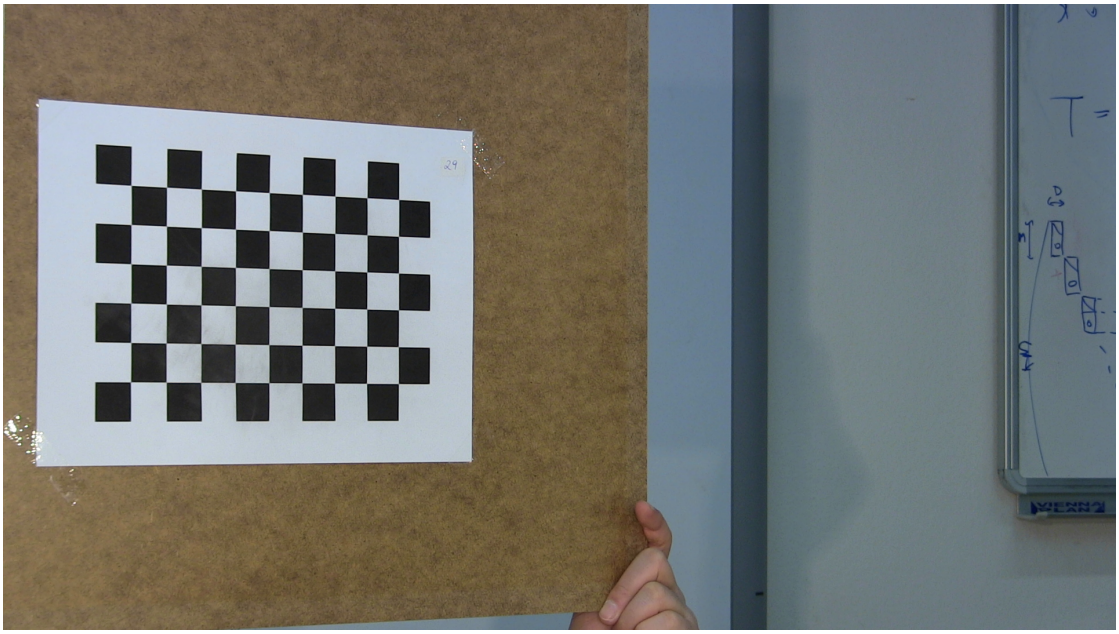


Figure 5.5: The calibration pattern during the calibration process, seen from the second camera.

Calibration Correction

As the calibration method from OpenCV tends to inaccuracies, we implemented a method for the correction of the epipolar line in y-direction, that is described in the following.

We are searching the function that gives us the difference to the correct location of the corresponding pixel in the second image. It may be defined as

$$e(y) = a_0 + a_1 y + a_2 y^2.$$

Therefore, we need to estimate a_0, a_1 and a_2. This can be done by linear regression

$$X\beta = y,$$

whereas

$$X = \begin{pmatrix} 1 & y_1 & y_1^2 \\ 1 & y_2 & y_2^2 \\ 1 & \vdots & \vdots \\ 1 & y_n & y_n^2 \end{pmatrix}, \beta = \begin{pmatrix} a_0 \\ a_1 \\ a_2 \end{pmatrix}, y = \begin{pmatrix} dy_1 \\ dy_2 \\ \vdots \\ dy_n \end{pmatrix}$$

and $n \geq 3$. $(dy_1, dy_2, ..., dy_n)$ are the differences to the correct solution in y-direction. They need to be collected manually. Therefore, the user needs to select a pixel in the first image and receives the epipolar line in the second image. Further, the user can take a closer look at the course of the epipolar line. Through the selection of the correct pixel in the second image

(usually close to the epipoal line), a feedback to the application is given. On that basis, the differences are selected.

If a solution exists, we can solve the regression by Gaussian approximation

$$\beta = (X^T X)^{-1} X^T y,$$

whereas $(X^T X)^{-1} X^T$ is the pseudo inverse of X. Finally, if we search for a corresponding pixel in the second image, we need to add to every potential on the epipolar line the value of $e(y)$. This is defined as

$$y' = y + e(y),$$

whereas y' is our new y-coordinate for the pixel in the second image.

CHAPTER 6

Evaluation

In the following, we will describe the quality metrics that will be used to evaluate the performance of our stereo matching algorithm. Further, we will describe the techniques we used for acquiring the image data set and the ground truth for this evaluation.

6.1 Ground Truth

To evaluate a stereo matching algorithm, a ground truth is necessary, as it yields the correct solution to a given problem, in our case, to a stereo calculation. This appears to be a disparity image of the horizontal displacement between the first and second image.

There are two commonly used ways for generating a ground truth. A real world ground truth can be generated or a synthetic ground truth can be calculated.

Real world ground truth usually comes with some measurement errors. The ground truth for real-world examples is not fully correct. Here, we need to differ between a controlled situation, for instance, with the lighting being controlled in an indoor situation or an uncontrolled situation in an outdoor scene. Depending on the circumstances, errors may appear. In general, it is necessary to apply a measurement technique that is superior to the own stereo matching algorithm.

The typical stereo evaluation benchmark is the Middlebury Stereo Benchmark, that can be found on the following website [Mid]. It provides accurate ground truths, gained from a structured light system. For this approach, different light patterns are projected onto the scene and a high-quality disparity map is gained. In total one can choose from 38 image pairs, that are varying in matching difficulty. Some of them are complex, such that algorithms can be evaluated to their limits. A further example is the use of another active system, e.g. a laser scanner. In [BBB+10a], the ground truth is obtained by scanning a physical mask by a laser scanner. Even though laser scanners achieve high quality results, they also suffer from artifacts and therefore aren't true for hundred-percent. Finally, another technique exists, namely hand-labeled ground truth. An example is the Tsukuba test set of the Middlebury ground truth disparity maps. However, this approach is extremely labor-intensive.

In contrast to these real-world examples, a synthetic ground truth does not always reflect real-world problems. For instance, the input of intrinsic and extrinsic parameters is usually correct for the calculation of the stereo matching algorithm, which isn't the case for real-world calibration data. With two virtual cameras, disparities at occluded areas may also be calculated, which differs from real-world ground truth.

Often the absence of a real-world ground truth to evaluate a stereo matching algorithm causes a major problem in computer vision. For most problems, not a single ground truth solution exists. Therefore, the ranking of stereo matching is not wholly possible and the progress in this field is difficult to trace.

Unfortunately, the Middlebury Stereo Benchmark does not include the evaluation of human faces for stereo matching algorithm. The second problem, that arises with the Middlebury Stereo Benchmark, is the lack of the calibration data, which is necessary for the own approach. In the absence of a real-world ground truth for the constructed lighting system, we decided to generate a synthetic ground truth for our quantitative evaluation process. The advantages are the correctness of the disparity map under artificial conditions. Moreover, we are able to test the algorithm easily under different conditions, for instance different structural conditions.

For the aquisition of the ground truth, we decided to use Blender and a python-script taken from [Ben], which was used in the following paper [BCC05]. Blender is a free and open-source 3D computer graphics software, that allows 3D modeling, UV unwrapping and texturing. The python script is executable in Blender 2.34 under the precondition that the polygon mesh is based on triangles.

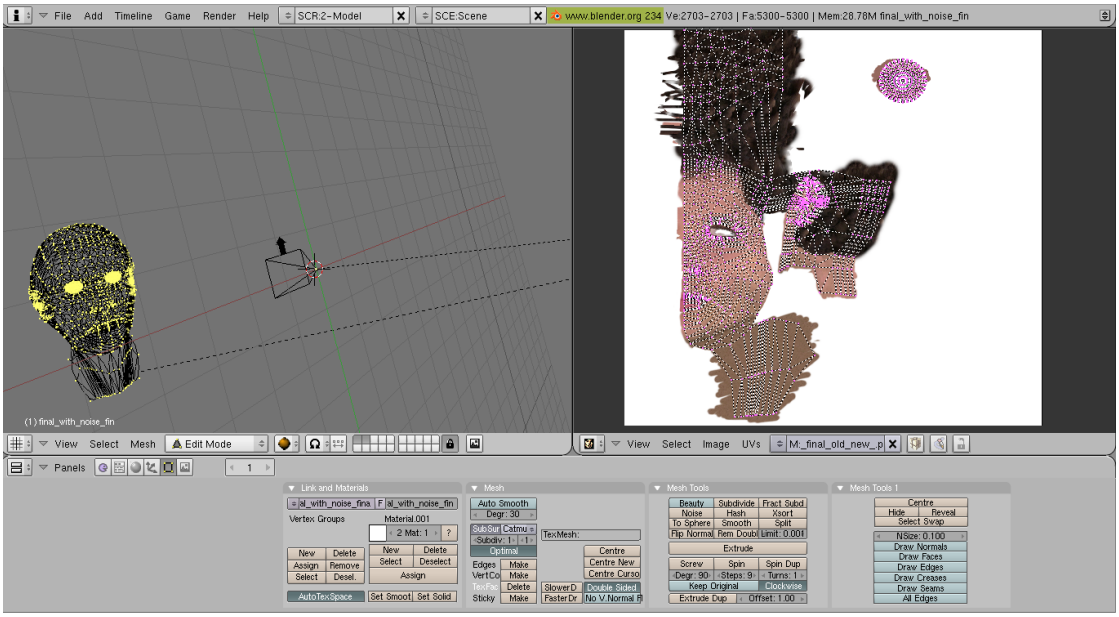


Figure 6.1: The mesh of our head on the left side and the UV image on the right side.

First of all, we built a mesh using Blender 2.63 with a poly-by-poly method out of quads, as it is easier to model the mesh in comparison to older Blender versions. With this method, starting from a quad polygon, more quads are extended from the edges. We used frontal and lateral photographies as references. Afterwards seams were set and UV unwrapping and projection was done. The UV map was textured with a real-world photography from our testing setup to keep the texture more realistic in Adobe Photoshop CS3 Extended. The clone stamp tool and the healing tool were used for the texturing. Finally, the quads were converted into triangles and the mesh was exported in the object format to import it in Blender 2.34. To make some experiments, an additional texture map was overlaid on the face, that should simulate blemishes of the skin. Therefore, the standard random texture *cloud* was chosen. Further, a lighting source was set. Varying conditions were built up with different parameters for the *cloud* texture and the lighting source.

6.2 Quality Metrics

The quality of the computed correspondences needs to be evaluated in a quantitative way, in our case with some quality metrics.

Generally stereo evaluation is done by computing error statistics based on a ground truth for ranking reasons. The Middlebury Stereo Evaluation usually compares different stereo matching algorithms by the following measurements:

- percentage of erroneous pixels in non-occluded regions
- percentage of erroneous pixels in total
- percentage of erroneous pixels near disparity borders

For this thesis, the following two metrics are computed, whereas both quality metrics are computed between the disparity map and the ground truth map.

1. The error measurements, the root mean squared error $RMSE$ [Hal10], is used, similar to the Middlebury stereo website

$$RMSE = (\frac{1}{n} \sum_{(x,y)} |d(y,x) - d_T(y,x)|^2)^{\frac{1}{2}},$$

   whereas n is the total number of pixel that are found for the underlying amount of pixels and d_T represents the ground truth. Every pixel, that fulfills the conditions for the underlying amount of pixels, will contribute to the $RMSE$. A low level of $RMSE$ means that the accuracy of the calculated disparity map is high.

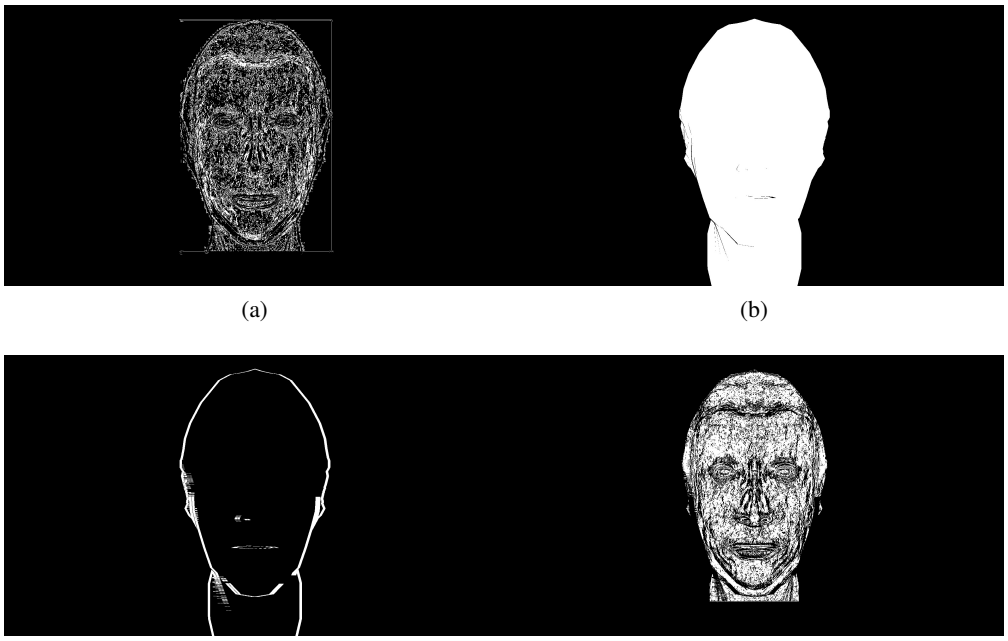
2. The percentage of erroneous pixels PEP is defined as

$$PEP = \frac{1}{n} \sum_{(x,y)} (|d(y,x) - d_T(y,x)| > \sigma_d),$$

whereas σ_d is the threshold of disparity tolerance. Wrong pixels are estimated in a first step by the absolute difference between the own and the ground truth disparity map. If the absolute disparity difference is larger than one pixel, this pixel is counted as an error. Again, only pixels that belong to the underlying amount of pixels will contribute to the PEP.

Similar to [SS02] and the Middlebury Stereo Evaluation [Mid], these metrics are applied to four different kinds of areas in the image. Note that for all of the areas occluded regions are included, as we do not have information about their position. Four segmentations on the basis of the first image and the ground truth need to be done:

- textureless areas: areas where the squared intensity gradient, a combination of horizontal and vertical direction calculated with the Sobel operator, over a square window of a given size is below a given threshold. For the threshold the value $20 - 80$ was chosen, as salient pixels are filtered with a threshold of 20 and the value range is defined as $1 - 255$.
- the whole reference image: only the pixels covered by a ground truth pixel value are counted for the evaluation. Pixels at the borders, also computed by the matching algorithm, are ignored.
- depth discontinuity areas: pixels are considered to be in this area, if they are not farther away than 9 pixels from a disparity jump in the ground truth.
- textured areas not near depth discontinuities: pixels are considered to be in this area, if they are farther away than 9 pixels from a disparity jump in the ground truth and if the Sobel operator gives a value higher than 80.



(a) (b)

(c) (d)

Figure 6.2: (a) textureless area, (b) the whole reference image, (c) depth discontinuity areas and (d) textured areas not near depth discontinuities.

6.3 Experiments with Synthetic Dataset

For the following experiments, the central image pair has the following characteristics throughout most of the experiments:

- a baseline of 0.5
- a noise filter, *cloud*, of noise size 0.096
- a lighting with *sun* of Blender with an energy of 1.0
- a window size of $9, 15, 21$ is used
- a λ value of 0.5 is used

Baseline

To evaluate our stereo matching algorithm we need some datasets, that have a ground truth disparity map. We have collected several datasets by rendering images of our face mesh in Blender within different conditions. First of all, a sequence, consisting of three pairs of images, was rendered with three different baselines: 0.25, 0.5 and 0.75. These baselines are given in

Blender units. With these pairs we want to test the sensitivity of our algorithm to a varying baseline.

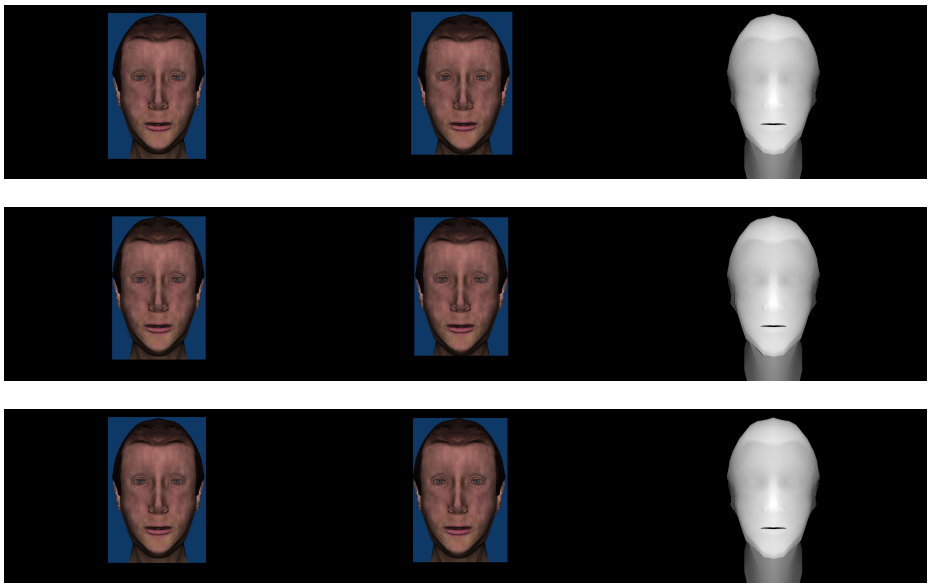


Figure 6.3: Dataset of image pairs with ground truth of 4 different baselines: 0.25 for the first row, 0.5 for the second row and 0.75 for the third row. The first column represents the first image, the second column represents the second image and the third column shows the ground truth.

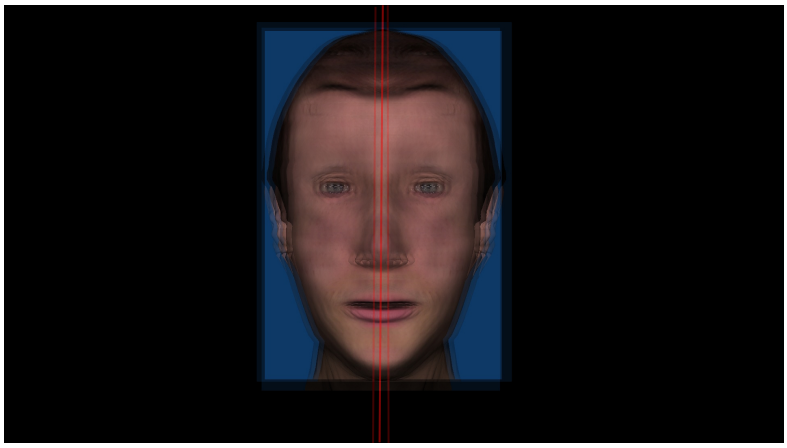


Figure 6.4: Three second images from the dataset *baseline* overlayed in one image, whereas the center of one image is marked by a red line.

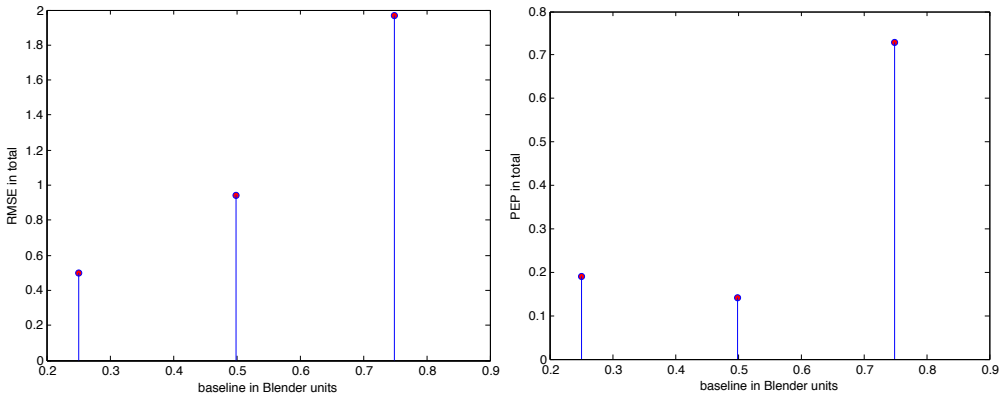
Results and Discussion

In the whole reference image, for a baseline of 0.25, 28% of correspondences from the whole possible amount of relevant pixels were found. For a baseline of 0.5, 29% and, for 0.75, 25% matches were found. So we have computed around 30% of the ground truth.

In the following figures it is obvious, that the baseline at 0.5 gives the best results. In total we reach a percentage of erroneous pixels of 14% for a baseline at 0.5, whereas in area that aren't near depth discontinuities and that are salient we reach 9%.

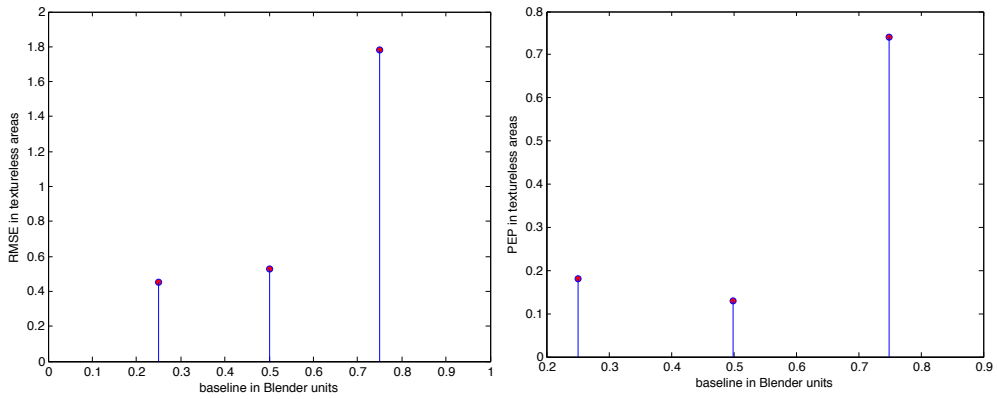
The quality of our algorithm near depth discontinuities is low as we haven't introduced any measurements to cope with these areas. The main reason is also the bad lighting at the border of the face. For a baseline of 0.5 we reach a PEP of 43%, whereas at 0.75, 98% matches are incorrect at depth discontinuities. In textureless areas, a PEP of 13% was reached at a baseline of 0.5, 18% at 0.25 and 74% at 0.75. Again, the results are quite satisfying for a baseline of 0.5. Note that the textureless area is already filtered. Pixels that give very little structural information in their neighborhood are ignored in the matching process.

The $RMSE$ of a baseline at 0.5 in areas that aren't near depth discontinuities and that have edge information is about 0.4 and in total 0.94 is reached. The latter value results from the poor disparity accuracy near depth discontinuities, that has a $RMSE$ of 2.64.



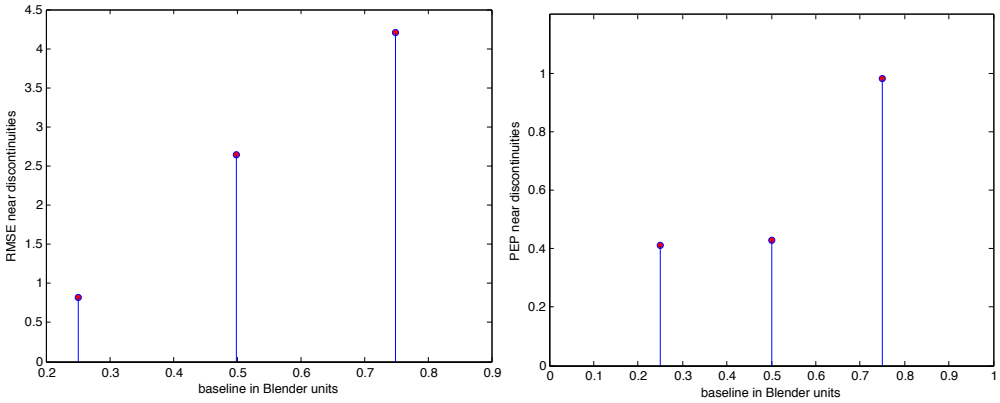
(a) (b)

Figure 6.5: (a) RMSE and (b) PEP in the whole reference image.



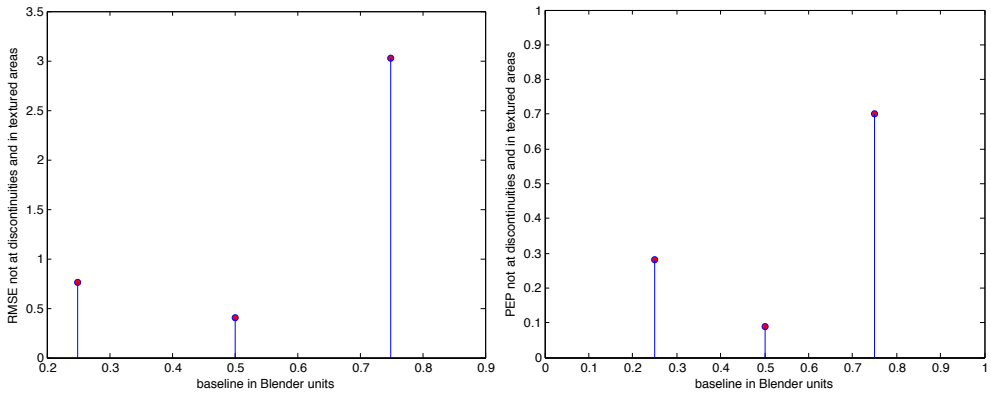
(a) (b)

Figure 6.6: (a) RMSE and (b) PEP in the textureless areas.



(a) (b)

Figure 6.7: (a) RMSE and (b) PEP in regions near discontinuities.



(a) (b)

Figure 6.8: (a) RMSE and (b) PEP in regions not near discontinuities and in textured areas.

Blemishes

Next, a sequence, consisting of three pairs of images, was rendered under three different structural conditions. A standard noise texture *cloud* of Blender was used for that purpose. The first pair has just the natural texture of a skin, without a noise texture. The second pair has a noise of noise size 0.096, which is quite coarse. The third pair has a noise size of 0.012, which is fine. With these pairs we want to test the sensitivity of our algorithm to a varying structure of the face texture, that is equally to blemishes of the skin.

(a) (b) (c)

Figure 6.9: Zooming in of the different skin textures of our dataset. The images illustrate a skin with (a) natural texture, (b) additional coarse noise and (c) additional fine noise.

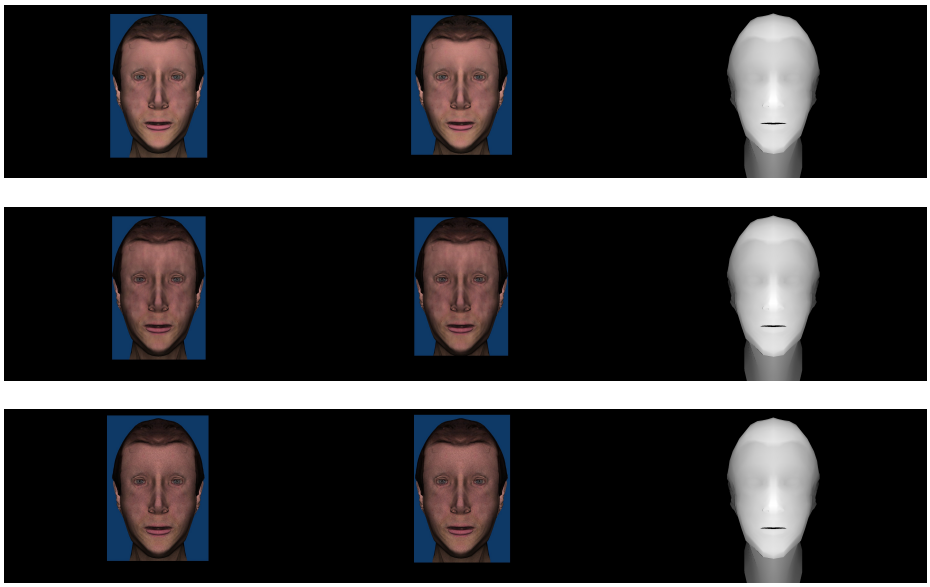


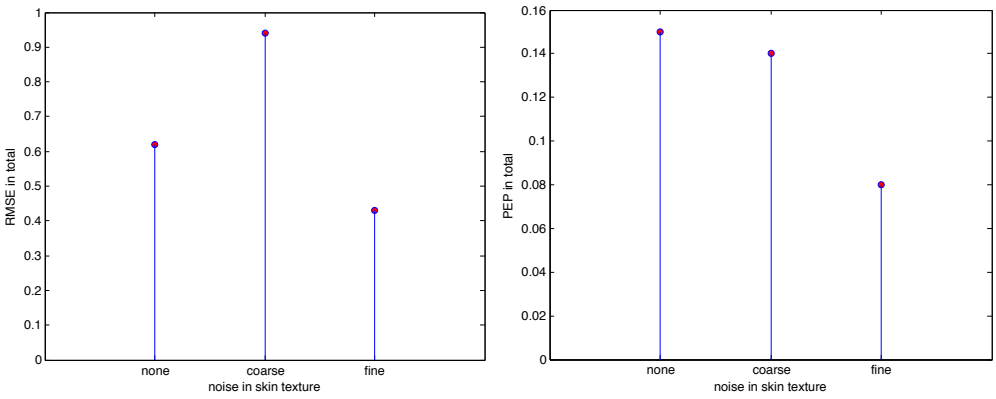
Figure 6.10: Dataset of faces with different texture. The *cloud* noise texture was used as follows: for the first row no noise is visible, for the second row coarse noise is given and for the third row fine noise is used. The first column represents the first image, the second column represents the second image and the third column shows the ground truth.

Results and Discussion

With this dataset $29\% - 44\%$ of correspondences were found. With a very noisy image pair, 44% of the ground truth was found.

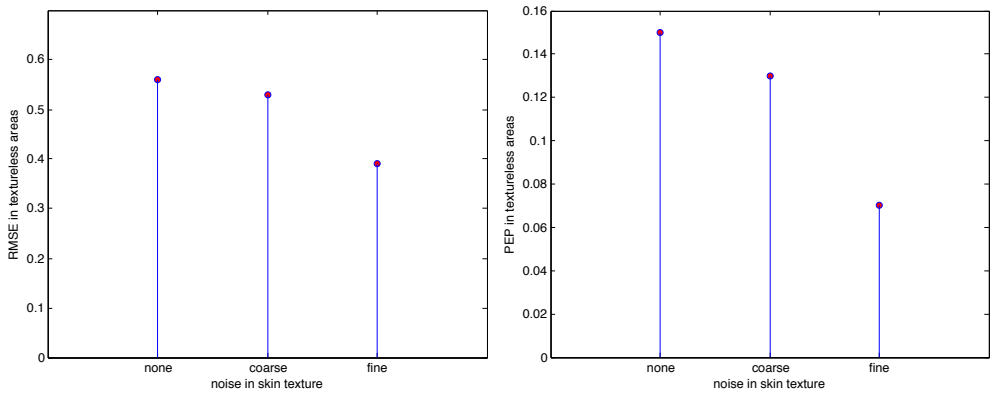
The following figures show that with increasing noise the PEP in total decreases. With no noise the natural skin texture reaches a PEP of 15% and with coarse noise we reach a slightly better result of 14%. With fine noise the percentage of erroneous pixels decreases at 8%. The improvement from none to coarse noise is due to the better results in textureless areas.

The $RMSE$ is quite varying, but shows in areas not near depth discontinuities and in textured areas, that the accuracy is increasing with increasing noise. However in total, the coarser noise reaches the worst result, when considering the $RMSE$. This is due to bad results near depth discontinuities.



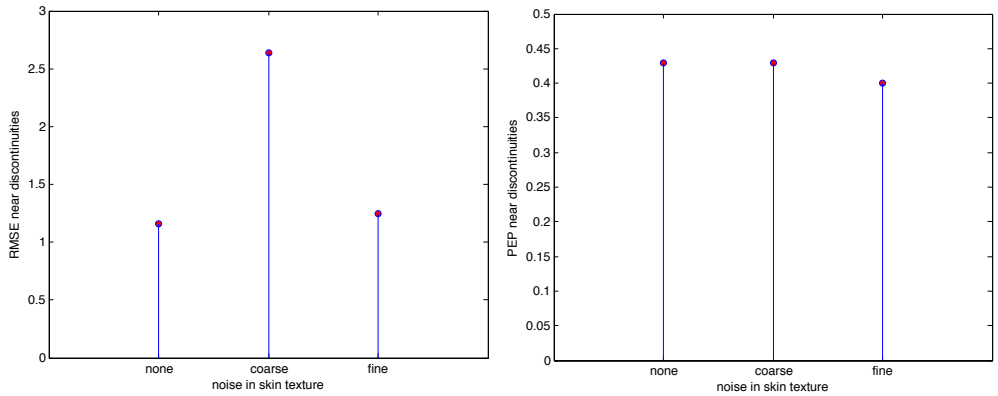
(a) (b)

Figure 6.11: (a) RMSE and (b) PEP in the whole reference image.



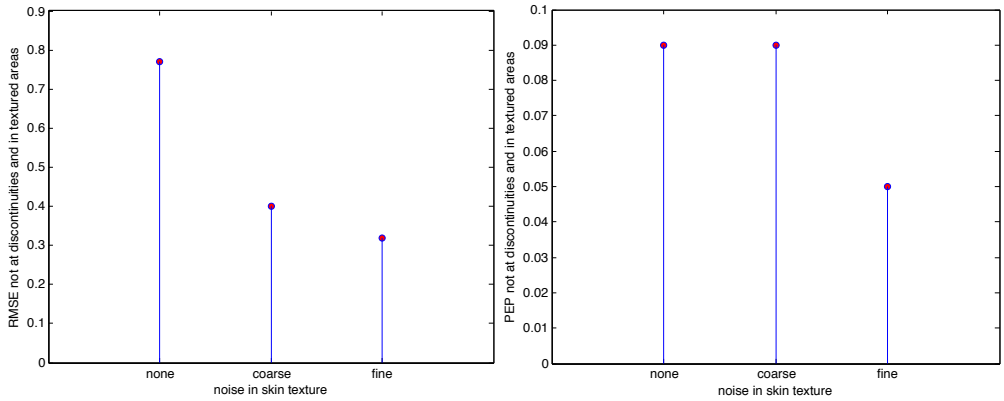
(a) (b)

Figure 6.12: (a) RMSE and (b) PEP in the textureless areas.



(a) (b)

Figure 6.13: (a) RMSE and (b) PEP in regions near discontinuities.



(a) (b)

Figure 6.14: (a) RMSE and (b) PEP in regions not near discontinuities and in textured areas.

Lighting

A sequence, consisting of three pairs of image under different lighting conditions, was collected. The first image reflects the lighting with a natural lighting source, namely the *sun*. The lighting energy was set to 1.0. Under these conditions the skin texture was fully visible, except of the border of the face, which is shadowed. For the second pair, a *lamp* lighting was chosen with an energy of 2.94. Again, the border of the face is shadowed. Finally, a difficult situation was created, namely reflection of the skin, whereas still some skin texture is visible. This was done by six *spot* lamps with an energy of $1.96 - 2.02$. However, in contrast to the other two pairs, the border of the face is lit well.

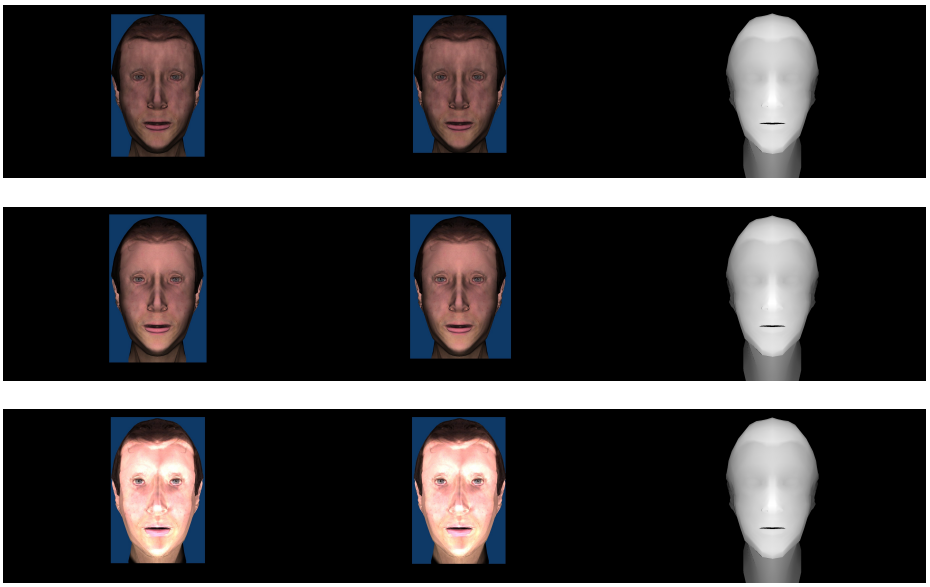
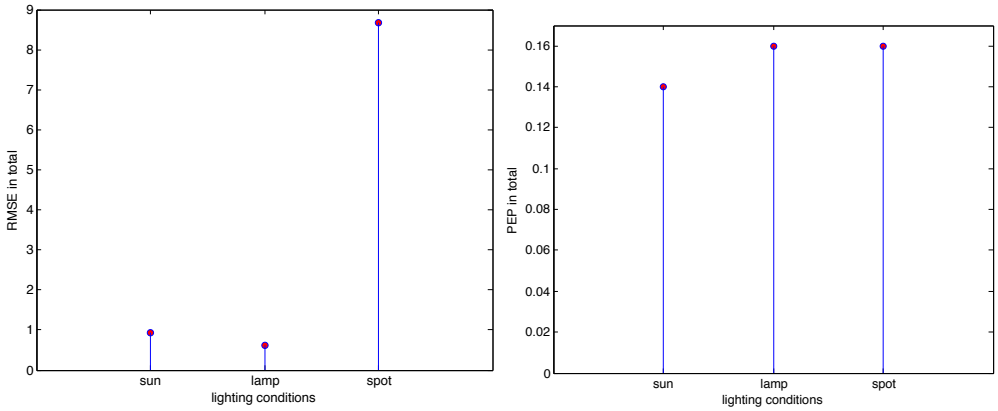


Figure 6.15: Dataset with different lighting conditions: the first row shows the images under a natural lighting, the second row shows the images under a lamp and for the third row images with reflecting skin were created. The first column represents the first image, the second column represents the second image and the third column shows the ground truth.

Results and Discussion

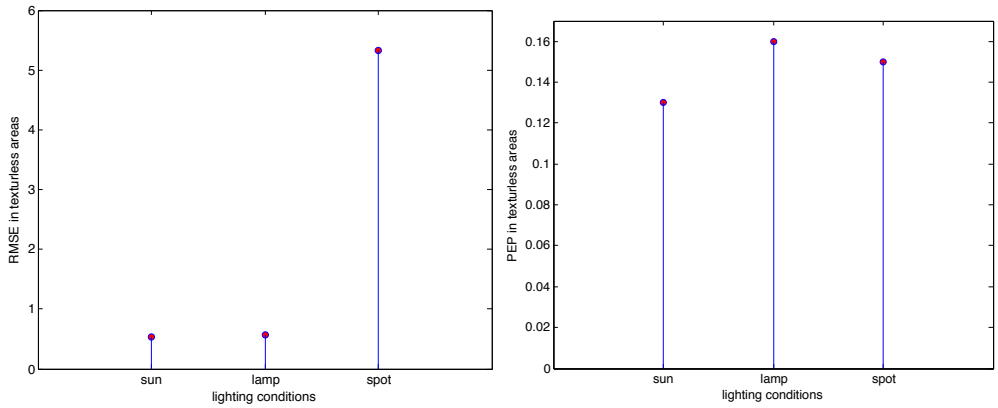
In total $29\% - 49\%$ of correspondences were found. The more energy for the lighting was used, the more matches were found.

The PEP in total reflects that with the *lamp* and *spot* lighting the results are worse, both reach 16%. However, when looking at the $RMSE$, it is obvious, that the lighting using the spots gives the worst results, as it reaches 8.68, whereas for instance the *lamp* lighting reaches 0.61. It is noticeable that the $RMSE$ of the *spot* lighting is very bad in regions with textured areas and not near depth discontinuities. In contrast to that, from near depth discontinuities the results are better than from the other two pairs. This may be caused by the better lighting at the border of the face. This observation is also reflected in the PEP, as *spot* lighting reaches an inaccuracy of 35% at depth discontinuities. *sun* and *lamp* result in a PEP of 43% and 42% near discontinuities. In areas that are textured and not near discontinuities, *sun* has a PEP of 9%, *lamp* has a PEP of 10% and *spot* has a PEP of 16%.



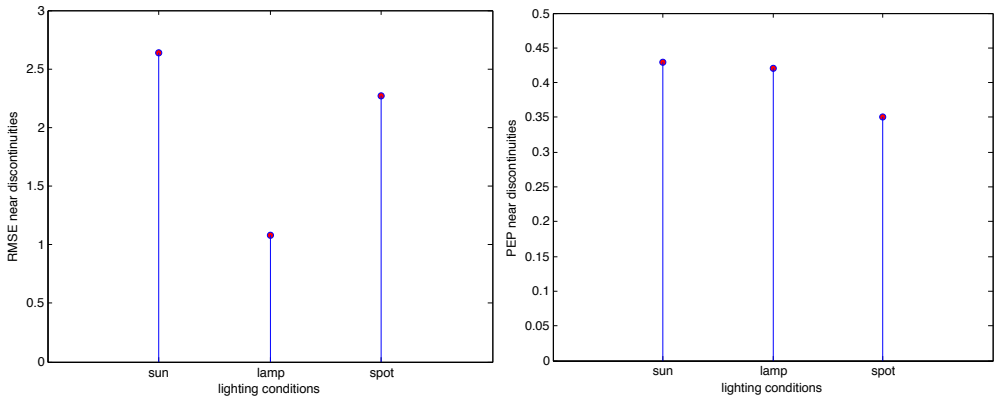
(a) (b)

Figure 6.16: (a) RMSE and (b) PEP in the whole reference image.



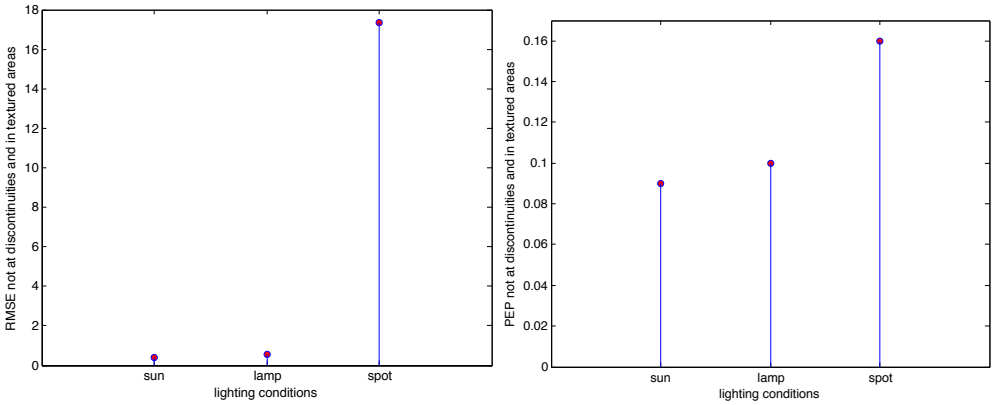
(a) (b)

Figure 6.17: (a) RMSE and (b) PEP in the textureless areas.



(a) (b)

Figure 6.18: (a) RMSE and (b) PEP in regions near discontinuities.



(a) (b)

Figure 6.19: (a) RMSE and (b) PEP in regions not near discontinuities and in textured areas.

Window

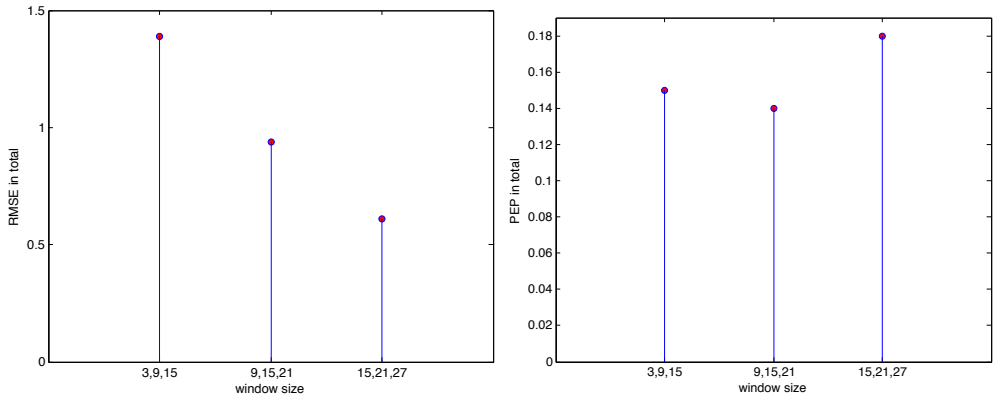
In this subsection, the difference between varying windows for the matching process will be investigated. The first experiment will be done with windows size of 3, 9 and 15. The second experiment will be done with windows size of 9, 15 and 21. The third experiment will be done with windows size of 15, 21 and 27. The following pair and ground truth were used.

Figure 6.20: Dataset for an experiment with different window size for the matching process. The first column represents the first image, the second column represents the second image and the third column shows the ground truth.

Results and Discussion

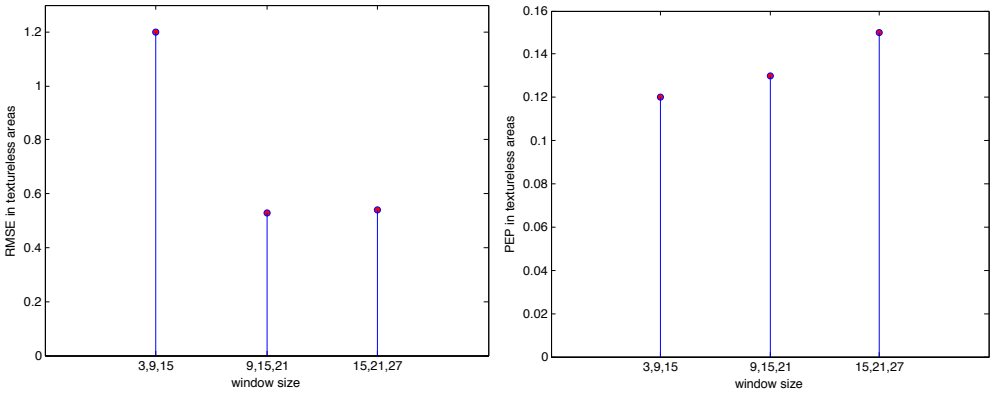
About 29% correspondences were found for the second pair of the dataset. For the first and third pair around 35% matches were found.

The PEP was in total best for the windows $9, 15, 21$, it results in about 14%. For the windows $3, 9, 15$, 15% was reached and for $15, 21, 27$ 18% matches were incorrect. On the one hand, the $RMSE$ reflects, that with increasing window size, the accuracy of the disparity value increases. On the other hand, it is also obvious, when looking at the PEP, that with increasing window size, the errors at depth discontinuities increase. This may be due to the foreground fattening problem. In regions not near depth discontinuities and in textured areas the results of PEP are best for the $9, 15, 21$ combination: 9% for $9, 15, 21$, 14% for $3, 9, 15$ and 18% for $15, 21, 27$. This can be caused by the fact that smaller windows lack distinction for correlation, whereas bigger windows are affected by the foreground fattening problem. Finally, in textureless areas, the $RMSE$ reflects that the inaccuracy of 0.53 is best at $9, 15, 21$. With smaller windows, the accuracy decreases significantly to 1.2, whereas for bigger windows the $RMSE$ is about 0.54, which is just slightly worse than the result with $9, 15, 21$.



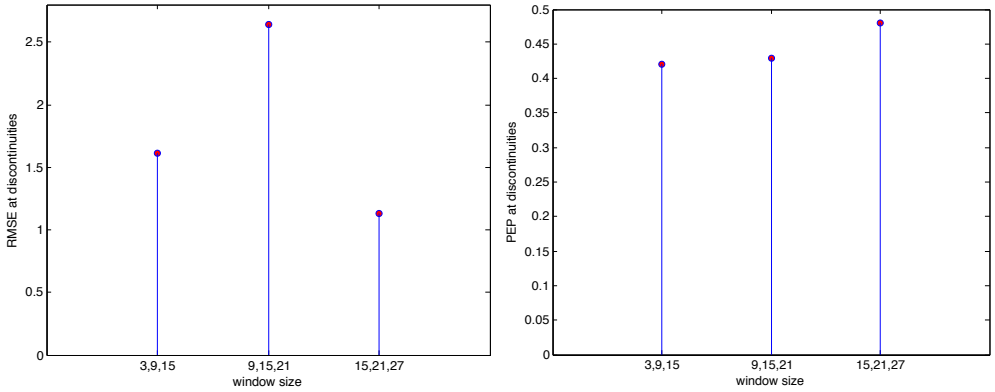
(a) (b)

Figure 6.21: (a) RMSE and (b) PEP in the whole reference image.



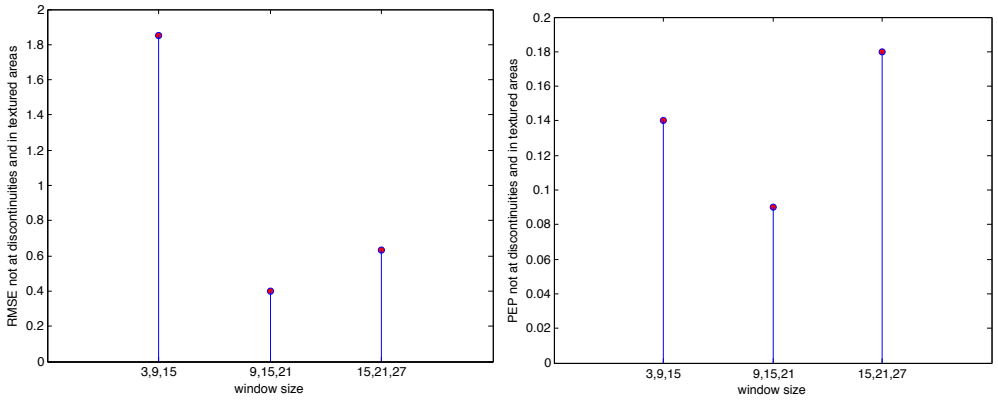
(a) (b)

Figure 6.22: (a) RMSE and (b) PEP in the textureless areas.



(a) (b)

Figure 6.23: (a) RMSE and (b) PEP in regions near discontinuities.



(a) (b)

Figure 6.24: (a) RMSE and (b) PEP in regions not near discontinuities and in textured areas.

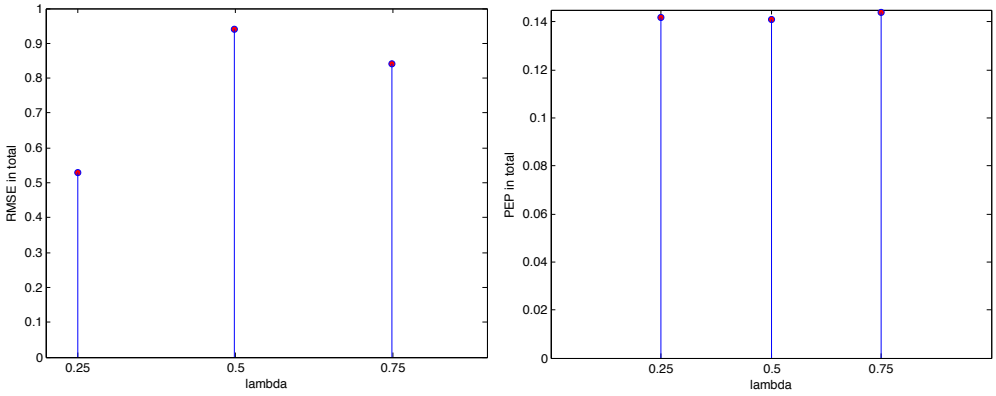
Lambda

For this experiment the same dataset was used as for the subsection 6.3. However, the goal of this experiment is to test the sensitivity of λ, which restricts the viewing ray in space, and therefore restricts the search space on the epipolar line.

Results and Discussion

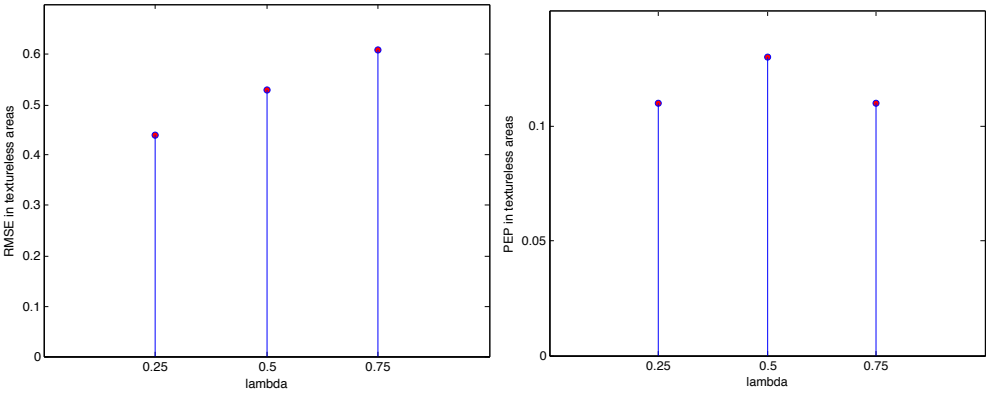
Again, around 30% of correspondences were found in total.

The PEP is in total slightly better for $\lambda = 0.5$. The big difference is obvious in the regions away from discontinuities and in textured areas. Here, we have for $\lambda = 0.5$ a PEP of 9%, for $\lambda = 0.25$ a PEP of 15% and for $\lambda = 0.75$ a PEP of 16%. This is also reflected in the $RMSE$ of these areas. It results from the fact, that a too restricted search space prevents the finding of matches, as the correct match is outside of the search space and a too wide search space increases the possibility for mismatches. However, λ is different from setting to setting. There exists no ideal value for all settings.



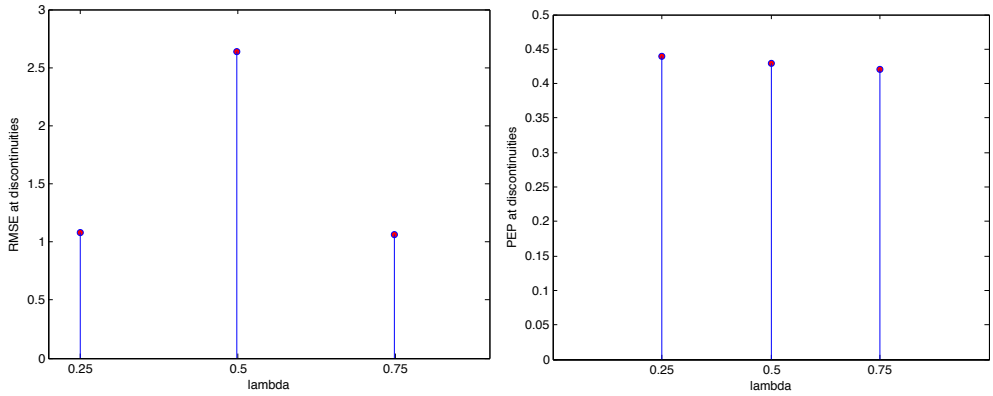
(a) (b)

Figure 6.25: (a) RMSE and (b) PEP in the whole reference image.



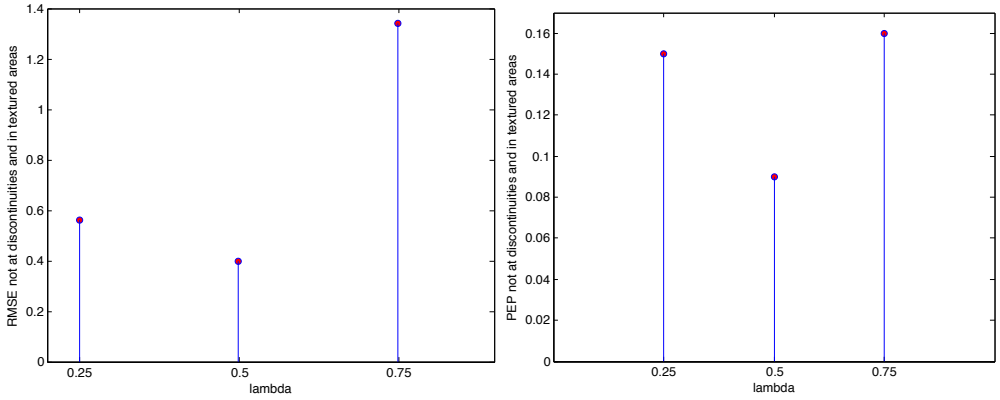
(a) (b)

Figure 6.26: (a) RMSE and (b) PEP in the textureless areas.



(a) (b)

Figure 6.27: (a) RMSE and (b) PEP in regions near discontinuities.



(a) (b)

Figure 6.28: (a) RMSE and (b) PEP in regions not near discontinuities and in textured areas.

6.4 Experiments with Real-World Datasets

In a second phase, we have done experiments with real-world datasets. These images were recorded with the lighting setup, that is described in subsection 5.1, and with three volunteers.

As we do not have any ground truth data, we cannot quantitatively evaluate the resulting disparity maps. The resulting interpolated depthmaps, out of the sparse depthmaps, are shown in the following. Interpolation is done linearly.

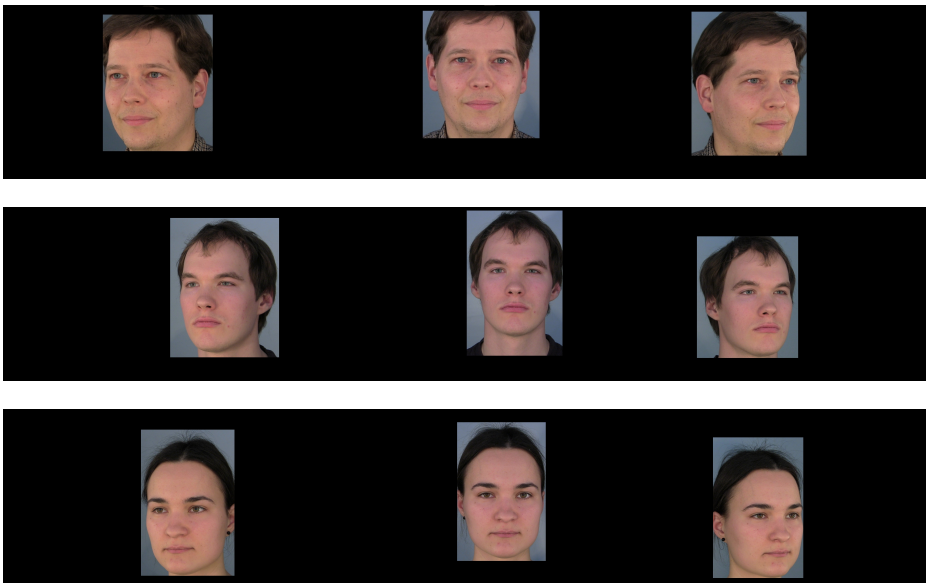


Figure 6.29: The first column shows the right camera images, the second column the middle camera image and the third column the left camera image.

However, we can qualitatively evaluate the results. The results are quite poor in regions near depth discontinuities. In fact, obvious errors appear near the nose areas. Further, due to reflections, errors in depth information appear in the eye regions. The baseline is too big, as areas near one side of the border of the face are not reconstructed correctly. Therefore, the algorithm does not really work in occluded areas, which we already expected, as we haven't implemented any measurements for these areas.

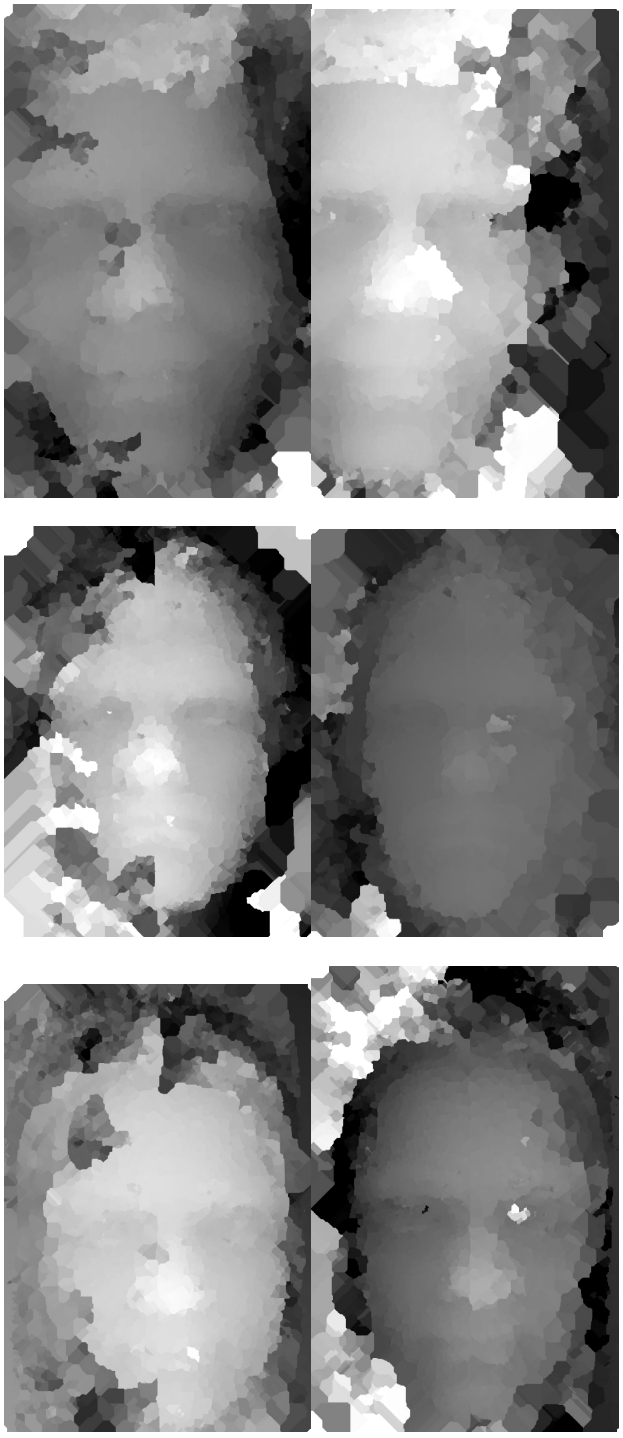


Figure 6.30: The first column shows the interpolated depthmap between middle and right camera and the second column the interpolated depthmap between middle and left camera.

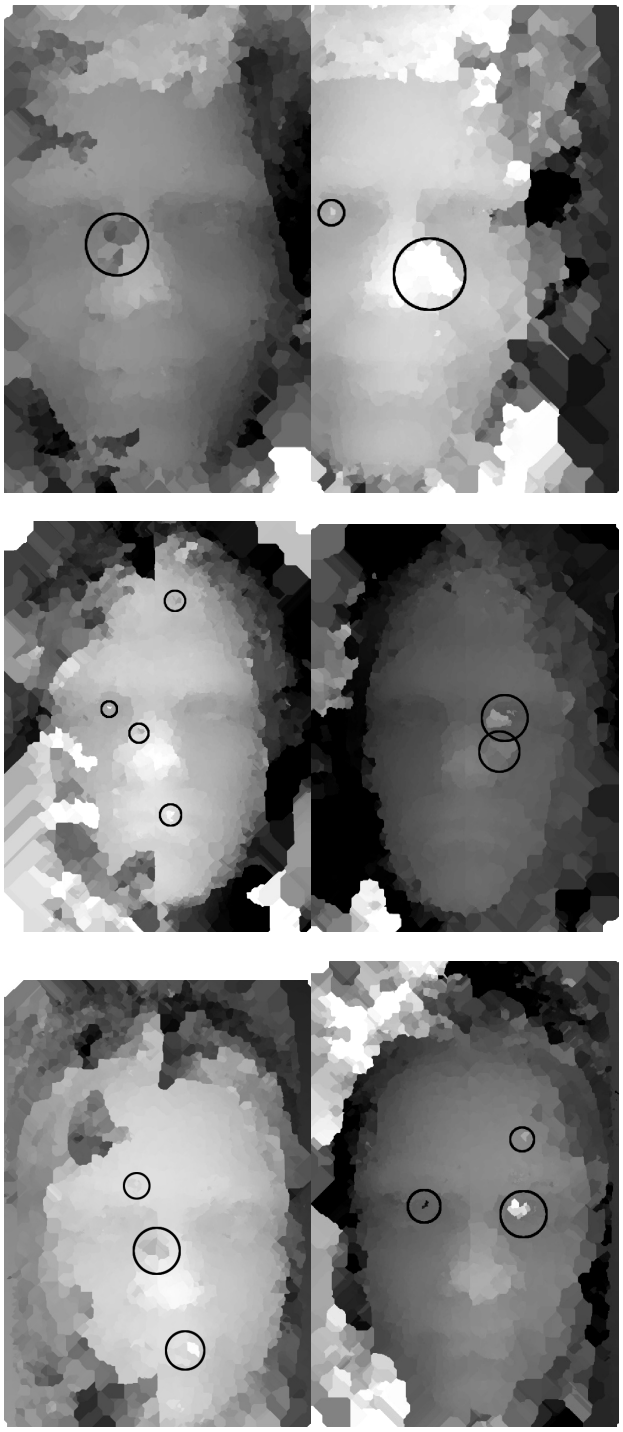


Figure 6.31: The first column shows the error zones of the interpolated depthmap between middle and right camera and the second column the error zones of the interpolated depthmap between middle and left camera indicated with black circles.

CHAPTER 7

Conclusion and Future Work

In this thesis, we have described a method to gain overall satisfying 3D data for the purpose of simpler face models. The main idea of our algorithm is to develop a completely passive reconstruction with passive sensors. Two cameras are aligned horizontally and calibrated carefully. The focus of this thesis lies on the image matching step, solving the correspondence problem. Therefore, the well known epipolar geometry was adapted. The first viewing ray and its neighboring rays, outgoing from a pixel in the first image frame and its four neighboring pixels, are restricted by a minimal and maximal depth. Then every hundredth scene point is projected in the second image frame. A block matching with a winner-takes-all technique for the common weight of the pixels is applied. We used varying windows size for the matching step and an epipolar band, instead of a line, in the second image, to cope with calibration inaccuracies. Further, a left/right consistency check is implemented, to fulfill the uniqueness constraint.

We built a synthetic dataset, as real-world ground truth data were missing, as well as a real-world dataset for testing the own lighting environment, that gives uniform and bright illumination. With this dataset the own algorithm was executed. The evaluation shows that special measurements for depth discontinuities are necessary. For instance, for a baseline of 0.5 in Blender units, a percentage of erroneous pixels of 43% was reached at discontinuities and in textured areas not at discontinuities 9% was measured. In textureless regions 13% was reached. These results are gained on the basis of a sparse disparity map with the synthetic ground truth. Moreover, the evaluation showed, that with higher structure in the face, the accuracy increases, and the brighter the lighting, the more correspondences are found. Shadows, especially at depth discontinuities, are difficult to reconstruct. Reflection in a face leads to loss of information and to inaccuracy in the disparity values. The most difficult areas to reconstruct are the eyes and the nose. The baseline should be chosen closer than what we did in our real-world experiments. Further problems, are low contrast and noise in the real-world image pair, as our algorithm mainly bases on edge detection. For the window size and the λ value, one needs to find a good balance. Fine-detailed matches need to be preserved. The foreground fattening problem needs to be avoided. This is done partially by the left/right consistency check. On the one hand, the

search area should not be too restricted and, on the other hand, too weakly restricted search areas cause also problems.

For future work, we need to increase the accuracy by coping with above mentioned problems. Further, the optimization of the source code is necessary, because our code is mainly unoptimized. Several operations could be easily implemented on the GPU. As a Sobel filter on an area of 3×3 pixels is not very resistant to noise, one can improve the filtering of salient pixels also. The images can be smoothed before submitting them to the Sobel filter, to avoid noise and one can use histogram equalization to stretch the contrast. Finally, the depth information of the two camera pairs can be combined to gain more accurate results.

We have fulfilled all of our goals. An approach for computing 3D depths of a frontal human face was described. We developed a simple method with minimal user input, we evaluated it with different lighting conditions, baselines and structural information of the face and showed, where the main difficult regions of the face are.

Bibliography

[3dM] 3dMD. http://www.3dmd.com. Accessed: 2012-04-05.

[Bar08] Daniel Bardsley. *A Practical Framework for 3D Reconstruction and Its Applications*. Dissertation, University of Nottingham, 2008.

[BBB+10a] Thabo Beeler, Bernd Bickel, Paul Beardsley, Bob Sumner, and Markus Gross. High-Quality Single-Shot Capture of Facial Geometry: Implementation Details. Technical report, Technical Report No. 671, Institute of Visual Computing, ETH Zürich, 2010.

[BBB+10b] Thabo Beeler, Bernd Bickel, Paul A. Beardsley, Bob Sumner, and Markus H. Gross. High-Quality Single-Shot Capture of Facial Geometry. *ACM Transactions on Graphics*, 29(3):40:1–40:9, 2010.

[BBH03] Myron Z. Brown, Darius Burschka, and Gregory D. Hager. Advances in Computational Stereo. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(8):993–1008, 2003.

[BBH08] Derek Bradley, Tamy Boubekeur, and Wolfgang Heidrich. Accurate Multi-View Reconstruction using Robust Binocular Stereo and Surface Meshing. In *CVPR '08 Proceedings of the 26th IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2008.

[BCC05] Benoît Bocquillon, Sylvie Chambon, and Alain Crouzil. Segmentation semi-automatique en plans pour la génération de cartes denses de disparités. In *ORASIS, Fournol*. LASMEA - Université de Clermont-Ferrand, 2005.

[Ben] Benoît Bocquillon. http://www.irit.fr/ alain.crouzil/bocquillon/mycvr/download.php. Accessed: 2012-04-05.

[BH10] Derek Bradley and Wolfgang Heidrich. Binocular Camera Calibration using Rectification Error. In *CRV '10 Proceedings of the 2010 Canadian Conference on Computer and Robot Vision*, pages 183–190. IEEE Computer Society, 2010.

[BHB+11] Thabo Beeler, Fabian Hahn, Derek Bradley, Bernd Bickel, Paul Beardsley, Craig Gotsman, Robert W. Sumner, and Markus Gross. High-Quality Passive Facial Performance Capture using Anchor Frames. *ACM Transactions on Graphics*, 30(4):75:1–75:10, 2011.

[BHPS10] Derek Bradley, Wolfgang Heidrich, Tiberiu Popa, and Alla Sheffer. High Resolution Passive Facial Performance Capture. *ACM Transactions on Graphics*, 29(4):41:1–41:10, 2010.

[BKSA] Gary Bradski and Adrian Kaehler. *Learning OpenCV: Computer Vision with the OpenCV Library*. O'Reilly Media, 1. edition, 2008 (in USA).

[Boo12] Boost.org. Boost C++ libraries. `http://www.boost.org`, 2012. Accessed: 2012-04-05.

[Bro71] Duane Brown. Close-range camera calibration. *Photogrammetric Engineering*, 37(8):855–866, 1971.

[CM01] Qian Chen and Gérard Medioni. Building 3-D Human Face Models from Two Photographs. *Journal of VLSI Signal Processing*, 27(1/2):127–140, 2001.

[Dim] Dimension Imaging 3D. http://www.di3d.com. Accessed: 2012-04-05.

[DS11] Feipeng Da and Yihuan Sui. 3D Reconstruction of Human Face based on an Improved Seeds-Growing Algorithm. *Machine Vision and Applications*, 22(5):879–887, 2011.

[FM98] Pascal Fua and Christian Miccio. From Regular Images to Animated Heads: A Least Squares Approach. In *ECCV '98 Proceedings of the 5th European Conference on Computer Vision*, pages 188–202. Springer, 1998.

[FP10] Yasutaka Furukawa and Jean Ponce. Accurate, Dense, and Robust Multiview Stereopsis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(8):1362–1376, 2010.

[Fua93] Pascal Fua. A Parallel Stereo Algorithm that produces Dense Depth Maps and preserves Image Features. *Machine Vision and Applications*, 6(1):35–49, 1993.

[Gal11] David Gallup. *Efficient 3D Reconstruction of Large-Scale Urban Environments from Street-Level Video*. Dissertation, University of North Carolina, 2011.

[GGW+98] Brian K. Guenter, Cindy Grimm, Daniel Wood, Henrique S. Malvar, and Fredrick H. Pighin. Making Faces. In *SIGGRAPH '98 Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, pages 55–66. ACM, 1998.

[Hal10] Hachem Halawana. *Partial demosaicing of CFA images for stereo matching*. PhD thesis, University Lille 1, 2010.

[Hir01] Heiko Hirschmüller. Improvements in Real-Time Correlation-Based Stereo Vision. In *SMBV '01 Proceedings of the IEEE Workshop on Stereo and Multi-Baseline Vision*, pages 141–148. IEEE Computer Society, 2001.

[HLS02] Daniela Hall, Bastian Leibe, and Bernt Schiele. Saliency of Interest Points under Scale Changes. In *BMVC '02 Proceedings of the British Machine Vision Conference*, pages 646–655. BMVA Press, 2002.

[HS07] Heiko Hirschmüller and Daniel Scharstein. Evaluation of Cost Functions for Stereo Matching. In *CVPR '07 Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2007.

[Huq01] Mohammad Shafikul Huq. *3D Reconstruction and Tracking of Human Faces from a Stereo Image Sequence*. Master thesis, Wright State University, 2001.

[IY96] Horace H. S. Ip and Lijun Yin. Constructing a 3D Individualized Head Model from Two Orthogonal Views. *The Visual Computer*, 12(5):254–266, 1996.

[JDFHSA] Steven Feiner James David Foley, Andries van Dam and John Hughes. *Computer Graphics: Principles and Practice*. Addison-Wesley Professional, 2. edition, 1995 (in USA).

[Jea] Jean-Yves Bouguet. Camera Calibration Toolbox for Matlab. http://www.vision.caltech.edu/bouguetj. Accessed: 2012-04-05.

[KSB11] Ira Kemelmacher-Shlizerman and Ronen Basri. 3D Face Reconstruction from a Single Image using a Single Reference Face Shape. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(2):394–405, 2011.

[LM02] Rainer Lienhart and Jochen Maydt. An Extended Set of Haar-Like Features for Rapid Object Detection. In *ICIP '02 IEEE International Conference on Image Processing*, pages 900–903. IEEE Computer Society, 2002.

[LMPM03] Jinho Lee, Baback Moghaddam, Hanspeter Pfister, and Raghu Machiraju. Silhouette-Based 3D Face Shape Recovery. In *Proceedings of the Graphics Interface*, pages 21–30. A K Peters, 2003.

[LZSA] Zicheng Liu and Zhengyou Zhang. *Face Geometry and Appearance Modeling*. Cambridge University Press, 1. edition, 2011 (in USA).

[LZJC00] Zicheng Liu, Zhengyou Zhang, Chuck Jacobs, and Michael F. Cohen. Rapid Modeling of Animated Faces From Video. Technical report, Microsoft Technical Report MSR-TR- 2000-11, 2000.

[MAP04] Gian Luca Mariottini, Eleonora Alunno, and Domenico Prattichizzo. The Epipolar Geometry Toolbox (EGT) for MATLAB. Technical report, Technical Report 07-21-3-DII University of Siena, Siena, Italy., 2004.

[Mar96] Marc Proesmans and Luc Van Gool and André J. Oosterlinck. Active Acquisition of 3D Shape for Moving Objects. In *ICIP '10 Proceedings of the International Conference on Image Processing*, pages 647–650. IEEE, 1996.

[Men97] Christian Menard. *Robust Stereo and Adaptive Matching in Correlation Scale-Space*. PhD thesis, Vienna Institute of Technology, 1997.

[Mid] Middlebury Stereo Benchmark. http://vision.middlebury.edu/stereo/. Accessed: 2012-04-05.

[Moy00] Tai Jing Moyung. *Incremental 3D Reconstruction Using Stereo Image Sequences*. Master thesis, University of Waterloo, 2000.

[MR10] Yue Ming and Qiuqi Ruan. Face Stereo Matching and Disparity Calculation in Binocular Vision System. In *IIS '10 Proceedings of the 2nd International Conference on Industrial and Information Systems*, pages 281–284. IEEE, 2010.

[OTRT05] Davide Onofrio, Stefano Tubaro, Antonio Rama, and Francesc Tarres. 3D Face Reconstruction with a Four Camera Acquisition System. In *VLBV '05 Proceedings of the International Workshop on Very Low Bitrate Video Coding*, pages 0–0. ?, 2005.

[Schny] Oliver Schreer. *Stereoanalyse und Bildsynthese*. Springer, 1. edition, 2005 (in Germany).

[SS02] Daniel Scharstein and Richard Szeliski. A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms. *International Journal of Computer Vision*, 47(1–3):7–42, 2002.

[SW06] Mikhail Sizintsev and Richard P. Wildes. Coarse-to-Fine Stereo Vision with Accurate 3-D Boundaries. Technical report, Technical Report CS-2006-07, Department of Computer Science, York University, 2006.

[The00] The OpenCV Library. G. bradski. dr. dobb's journal of software tools, 2000.

[Tönny] Klaus Dietz Tönnies. *Grundlagen der Bildverarbeitung*. Pearson Studium, 1. edition, 2005 (in Germany).

[VJ01] Paul Viola and Michael Jones. Rapid Object Detection using a Boosted Cascade of Simple Features. In *CVPR '01 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 511–518. IEEE Computer Society, 2001.

[Wil90] Lance Williams. Performance-Driven Facial Animation. *SIGGRAPH '90 Proceedings of the 17th annual conference on Computer graphics and interactive techniques*, 24(4):235–242, 1990.

[YTCA10] Qingxiong Yang, Kar-Han Tan, Bruce Culbertson, and John Apostolopoulos. Fusion of Active and Passive Sensors for Fast 3D Capture. In *MMSP '10 IEEE International Workshop on Multimedia Signal Processing*, pages 69–74. IEEE, 2010.

[Yue95] Yuencheng Lee and Demetri Terzopoulos and Keith Waters. Realistic Modeling for Facial Animation. In *SIGGRAPH '95 Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, pages 55–62. ACM, 1995.

[Zha00] Zhengyou Zhang. A Flexible New Technique for Camera Calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1330–1334, 2000.