Universität Osnabrück
Institute of Cognitive Science

Bachelor's Thesis

Fusion of 3D Laser Scans and Stereo Images for Disparity Maps of Natural Scenes

Johannes M. Steger
jss@coders.de

March 23, 2006

Supervisors:

Peter König
Neurobiopsychology, Institute of Cognitive Science
University of Osnabrück
Germany

Joachim Hertzberg
Knowledge-Based Systems, Institute of Computer Science
University of Osnabrück
Germany

Abstract

Extracting 3-dimensional structure from multiple views of a scene is a classical problem of machine vision. This task is especially demanding for cluttered natural / outdoor scenes. We directly measured the 3D structure of natural scenes using a laser range scanner. We developed a technique to use these scans to obtain accurate disparity maps for images taken by a stereoscopic camera rig observing the same scene. The disparity maps thus acquired provide the basis for an algorithmically unbiased ground truth for the evaluation of stereo matching algorithms on natural images. Additionally, they are used to synthesize stimuli for psychophysical and electrophysiological experiments on three-dimensional viewing in humans.

Acknowledgements

I would like to thank all those who made this thesis possible.

I am particularly grateful to my supervisors Peter König and Joachim Hertzberg who gave me the opportunity to realize this extraordinary project.

Furthermore, I thank Selim Onat, Daniel Weiller and Saskia Nagel for open ears and valuable advice.

Also, I gratefully acknowledge the support and infinite patience of Andreas Nüchter and Kai Lingemann.

I drew immense motivation from Lina Jansen and Boris Bernhardt who showed me that my work was of instant use. Thank you for making teamwork fun.

I owe a great dept to the people at the Neurobiopsychology Lab who make it a stimulating and fun place to work. Here, special thanks go to Mario Negrello for emotional support and priceless discussions.

Most importanly, I would like to thank my parents whose support made it *all* possible.

Attributions

This thesis would not have been possible without the work of Robert Märtin, who also supplied the contents of sections 2.1.3 and 2.3.
The discussion - part 4 - is a collaborative effort.

Contents

1. Introduction

A classical problem of computer vision is the extraction of three-dimensional structure from multiple views of a scene. To solve this problem, knowledge of view properties and point correspondence between views is needed. Combining these two is sufficient to compute the three-dimensional position of underlying scene points. We can identify the following sub-problems:

1. What is the geometric relationship between different views, i.e., where is camera A with respect to camera B?

2. How can the imaging process, generating a two-dimensional representation of the scene be described by a mathematical model?

3. What is the best way to establish point correspondence between views, i.e., to compute disparity information?

In the field of computer vision, the first two items are known as the extrinsic and intrinsic camera calibration problem, respectively. There is a number of established techniques and tools to solve calibration problems. Establishing point correspondence, to which we shall refer for the case of two views as *stereo matching*, is notoriously hard to do for natural scenes. This is due to the high amount of clutter (e.g. foliage and twigs) causing occlusion and highly view dependent illumination changes (specularities). In addition, we lack an objective measure to assess the performance of stereo matching algorithms when dealing with natural scenes. Usually, these algorithms are evaluated using synthetic stereo views generated from a known three-dimensional structure [Scharstein and Szeliski, 2002]. However, we do not know the true structure of the underlying natural scene. Consequently, we do not possess a *ground truth* disparity map and cannot judge the correctness of computed disparity maps. Of course, one could subjectively evaluate the results and define the "best" algorithm to act as ground truth as long as no (subjectively) better result is available. However, this solution is not appropriate as the ground truth remains biased, both subjectively and algorithmically.

In this study, we developed a means to acquire unbiased high quality disparity maps of natural scene stereo images based on laser range measurements. We present two applications for these disparity maps:

1. They are used as a ground truth for the evaluation of two stereo matching algorithms.

2. They are used to synthesize stimuli for psychophysical and electrophysiological experiments on three-dimensional viewing in humans [Jansen, 2006, Bernhardt, 2006].

2. Methods

2.1. Sensor Fusion

2.1.1. Hardware

The system used for data acquisition consists of three main components: a stereo camera rig for image shooting, a laser range scanner for gaining depth data, and a stand to hold both of them as well as necessary supplies.

2.1.1.1. Stereo Rig

Stereo Images are acquired using a dual camera setup, consisting of two identical cameras mounted in parallel on a solid metal rig as seen in 2.1. We used two Sony [Sony Corp.,

Figure 2.1.: Stereo Rig with two digital cameras mounted. The black cables on the top are connected to the synchronization device.

Tokyo, Japan] Digital Still Cameras (DSC V1). To mimic human inter-eye distance, the baseline was chosen as small as the form factor of the components allowed for, resulting in 7.5cm inter-lens distance.
The cameras offer a variety of configuration options such as focal distance, white balance,

zoom level and image resolution. On the one hand, these allow for very precise optimization of the pictures taken, on the other hand this requires manual synchronization of these options. While this manual operation is feasible for configuration, this is not true for actually shooting a scene. The latency between the two shots (*drift time*) needs to remain below the shutter time of the cameras to avoid mismatches between the resulting images. As the shutter timer is typically around $10ms$, this cannot be achieved by manual triggering. We therefore used a LANC Shepherd by Berezin Stereo Photography Products, Abedul, Mission Viejo, CA, USA. This device connects to both cameras and sends synchronized "shoot" commands upon button press. The resulting drift times are well below 10% of the shutter time.

2.1.1.2. Scanner

The 3D laser scanner consists of two sub-components: a stock 2D laser scanner developed by Sick AG, Waldkirch, Germany and mechanics developed by Fraunhofer Institut für Autonome Intelligente Systeme to rotate that Scanner around it's transverse axis, adding a third degree of freedom.

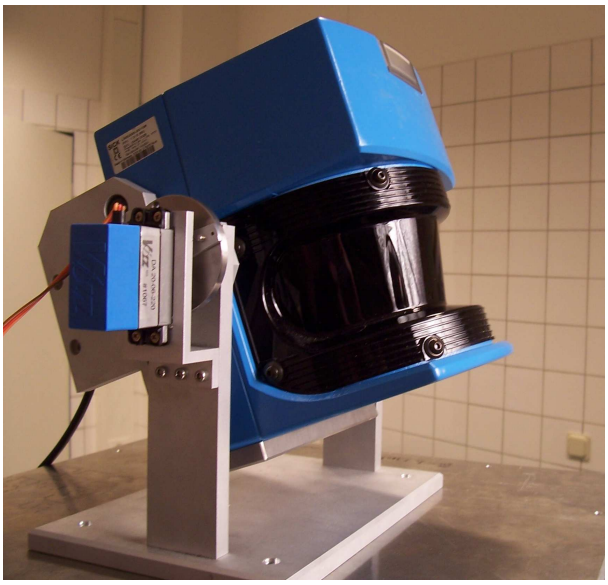


Figure 2.2.: Laser scanner and servo motor

The 2D scanner radially emits laser beams on the horizontal plane, covering 180° with a maximum resolution of 0.25°. The light is monochromatic with a wavelength of $906nm$, which is well out of the range visible to humans. Scan time is dependent on the selected resolution and takes from $13.32ms$ (1.0°) to $53.28ms$ (0.25°). When a laser beam hits an object, it is reflected back to the scanner. Distance is then a function of time passed between emission and receipt. The surface structure is captured by the decline in strength of the beam. This is represented by the *remission* value recorded by the scanner.
The scanner allows for configuration of a minimum reflection value. Data points below

For Device	Manufacturer	Model	Volts	Ampere hours	Nr. needed
Servo	Panasonic	LC-R0612P	7.25	4.8	1
Scanner	Panasonic	LC-R123R4PG	12.0	3.4	2
Scanner	YUASA	NP7-12 (2002)	12.0	7.0	2

Table 2.1.: Battery types used for Scanner and Servo

this value are discarded. This effects a compromise between surface material and depth range: Materials with high reflectance can be scanned of up to $80m$ distance, while extremely low reflectance limits the scan range to $8m$. Although it would be feasible to account for this in every single scene scan, we decided for a constant value, resulting in a maximum range of $32m$.
The laser scanner features a high-speed RS-422 serial interface that is connected to the controlling computer by a Sealevel Systems, Inc., Liberty, SC, USA USB to RS-422 Serial Interface Adapter.

The Servo Motor (VOLZ Servos GmbH & Co.KG, Offenbach am Main, Germany) is directly attached to mechanics that allow for rotation of the Scanner around its transverse axis. The working range of the servo covers $\pm 65°$ with 500 steps, theoretically resulting in about $0.26°$ resolution. The location of the axis is chosen to match the emission point of the laser scanner, so that 2D scans always originate from the same point, simplifying later calculation of 3D points. This setup also makes the elevation angle of the whole system equal to the servo's rotation angle.
However, this is not an axis matching the center of gravity of the scanner, resulting in non-linear rotation dynamics. Therefore the position has to be actively maintained by motor force during operation. A maximum stall torque of $220Ncm$ delivered by the servo accounts for this imbalance.
Fraunhofer Institut für Autonome Intelligente Systeme also supplied an interface that allows for control of the Servo via a standard serial line. In addition, we use a Wiretek, XiaoJieJiao Humen, GuangDong, China RS-232 to USB 1.1 converter cable to connect this interface to the controlling computer.

2.1.1.3. Stand

In order to minimize view differences between camera images and laser scans, a solid platform is needed that allows for sequential placement of both. As the primary application of the data gained will lie in the area of psychophysical experiments, we decided for a stand of about $1.4m$ height (figure 2.3), thereby approximating typical human view height. It is built from solid stainless steel and features four legs adjustable in length to account for ground unevenness. The top platform of the stand exposes two pins to fixate scanner or rig. The controller board for the servo motor is located on the side of the top platform by a hook and lose fastener. A second platform $30cm$ below the top platform carries power supply according to table 2.1, and a notebook used to acquire data from the laser scanner. Cables for power supply and USB interfaces are fixed to the stand to minimize assembly effort between successive scan runs.

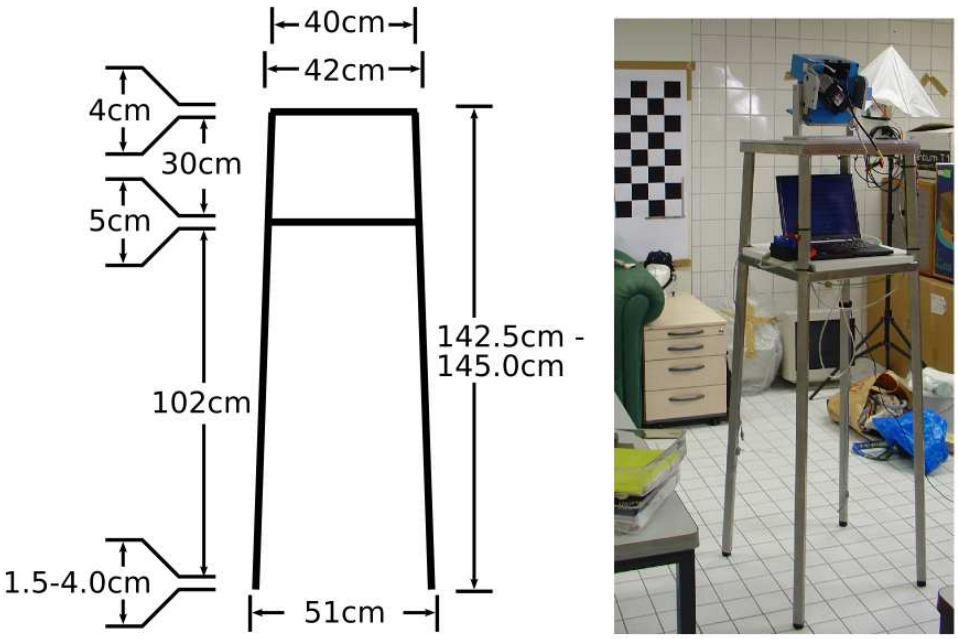


Figure 2.3.: Stand: A stable metal platform where laser scanner, stereo rig, notebook and supplies are mounted on.

2.1.1.4. Interface Software

User's Perspective Images taken by the stereo cameras are downloaded from the devices by standard USB transfer in mass storage mode.
Laser scanning is controlled by a custom software that achieves a maximum of automation:
After initial configuration of the Scanner's resolution and the hardware port it is attached to, the user only needs to increment the software's internal scene counter and hit the *scan* button. Configuration options are saved between runs and typically only change when the hardware setup is modified or data is gained for special purposes. Therefore, a user will usually not be required to change those options. A screen-shot of the scanning window is shown in figure 2.4.

Implementation The scanning application is divided into two parts: a low-level hardware interface that controls the laser scanner and servo motor in real time and a high-level user interface component that offers a convenient way of interacting with the low-level component.

Initial Version In the initial implementation of the scanner software, the hardware interface was mainly based on core components supplied by Fraunhofer Institut für Autonome Intelligente Systeme. They originated from applications in autonomous robotics and are therefore primarily optimized for speed and robustness. According to it's use-cases, it

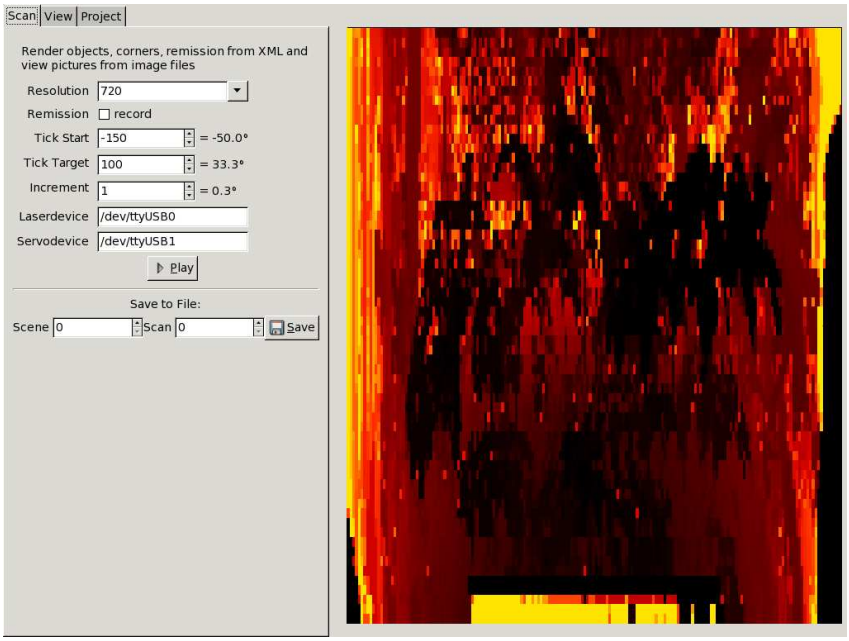


Figure 2.4.: Screen-shot of the Application used to control the laser scanner

allowed for scanning at different resolutions, always using the fastest possible scan mode and covering the whole scan plane of 180°. However, this resulted in some limitations: although the software allows for scanning at the maximum scanner resolution, it does not allow for recording of remission values at the same time. As the latter would double the time required for scanning a single plane, such a mode is not acceptable in the original application. In contrast, our application does not have such real-time requirements, but still has to live with the mentioned constraints.

Free Version In the beginning of 2006, Sick AG, Waldkirch, Germany made the scanner's interface protocols publicly available. We then used the transmission manual [Sick AG, Waldkirch, Germany, 2003] to reimplement the underlying hardware interface, supporting scanning at maximum resolution with simultaneous recording of remission values. This results in a more narrow field of view of only 45° per single scan-run in one scan plane. However, as the opening angle of the cameras used is 45° as well, we do not lose any data needed for our application. Overall scan speed is also entirely unaffected by this change. Furthermore, the new implementation is much more streamlined as it is developed solely for the purposes of this project and thereby also easier to integrate into the application governing the scanning process. We also changed the way we interact with low level kernel access, making the whole program more responsive and less CPU-time consuming at the same time. Additional work was put into developing a free replacement for the driver accessing the servo motor, making integration even easier. In the end, this approach results in an application that is more suited to our needs. Furthermore, it is free software, allowing us to distribute the source code as a whole without leaving out one of the most critical parts.

The resulting new application is shown in 2.5. It features a second graph on the bottom

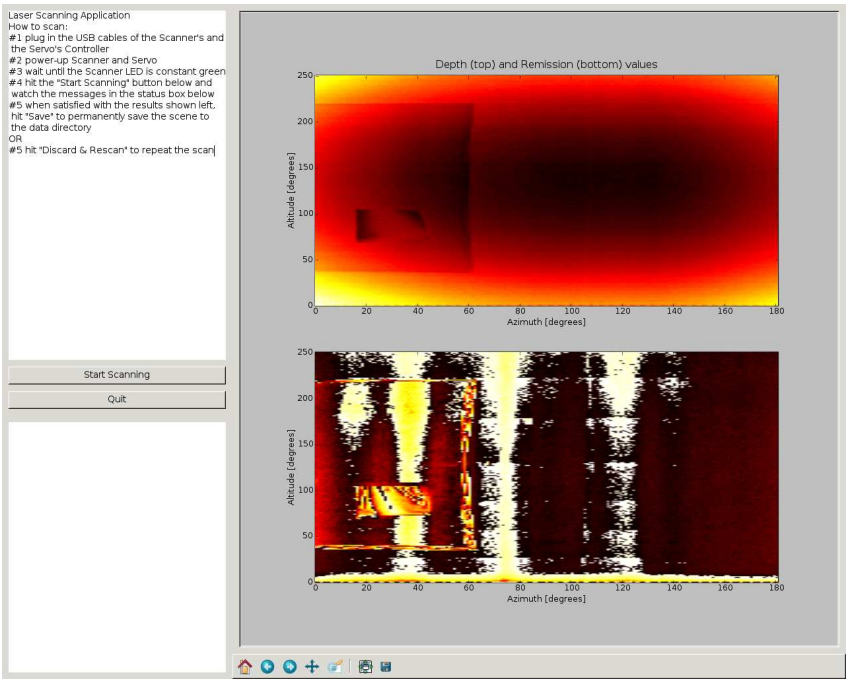


Figure 2.5.: Revised user interface of the scanning application with support for maximum resolution *and* remission values.

for display of remission values.

2.1.2. Scanner Calibration

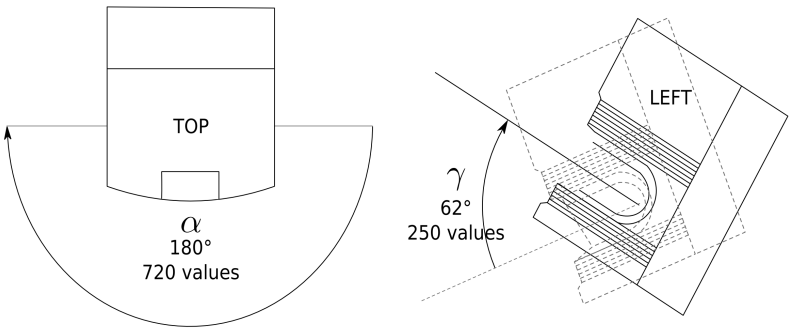


Figure 2.6.: Left: Resolution of the stock 2D laser scanner. Right: Resolution of the servo mechanics.

The 2D laser scanner component of the scanner system is pre-calibrated by it's OEM [Sick AG, Waldkirch, Germany]. For each scan it returns a maximum of 721 distance values d covering 180° with 0.25° resolution from left to right (see figure 2.6). However, no prior knowledge exists with regard to the relation between the ticks of the servo motor

and the elevation angle of the system.
As mentioned above, the rotation axis of the mechanics was chosen to match the emission point of the laser beams in the scanner. Furthermore, the servo motor has a constant step angle (see 2.1.1.2).
Nevertheless, the intrinsic dynamics of the mechanical system do not allow for a direct translation of servo ticks into the elevation angle γ, so we need to empirically find this mapping f:

$$f : T \to \Gamma$$

Before calibration, we can only rely on data acquired by scanning the horizontal plane, where $\gamma = 0°$. The Cartesian Z-coordinates can then be calculated by using only formula 2.1 with $z = z\prime$ and $y = 0$. When scanning a vertical flat object, such as wall, this z-coordinates are constant for a given angle α within the 2D scan plane over any elevation angle γ. Therefore, these values can be used to calculate the angle for any other elevation level and establish a translation from servo ticks:

$$f(t) = \arccos\left(\frac{z}{d}\right)$$

Fortunately, a linear fit suffices to describe the relation between T and Γ. We computed a conversion factor of 0.24° per tick.
Given this mapping f, it is straight forward to translate the input coordinates (α, γ, d) to Cartesian coordinates (x, y, z) in two steps: first, the position in the scan plane is calculated in equation 2.1, second, the elevation angle is taken into account in 2.2.

$$x = d\cos(\alpha) \qquad z\prime = d\sin(\alpha) \tag{2.1}$$

$$y = z\prime\cos(90° - \gamma) \qquad z = z\prime\sin(90° - \gamma) \tag{2.2}$$

In the end, this type of calibration can be said to be fully automatic. Figure 2.7 shows the application developed for this purpose.

2.1.3. Camera Calibration

Now that we are able to measure three-dimensional points, we wish to model our cameras to know how these points are imaged.

As already stated in the Introduction, our goal in camera modeling is to find a suitable mathematical model describing the mapping from a three-dimensional scene onto a two-dimensional image plane which occurs in our cameras. By *suitable*, we mean a model which is both accurate and mathematically tractable, i.e., allows a reliable estimation of its parameters. The pinhole camera model is such a model. Adapting the model parameters to our real cameras is a process known as *camera calibration*. It is an essential part of camera modeling. In the following paragraphs we provide a short treatment of the relevant mathematical concepts followed by an introduction to the pinhole camera model and camera calibration. At the end of this section, we will be able to calibrate a camera given multiple images of a calibration object of known geometry.

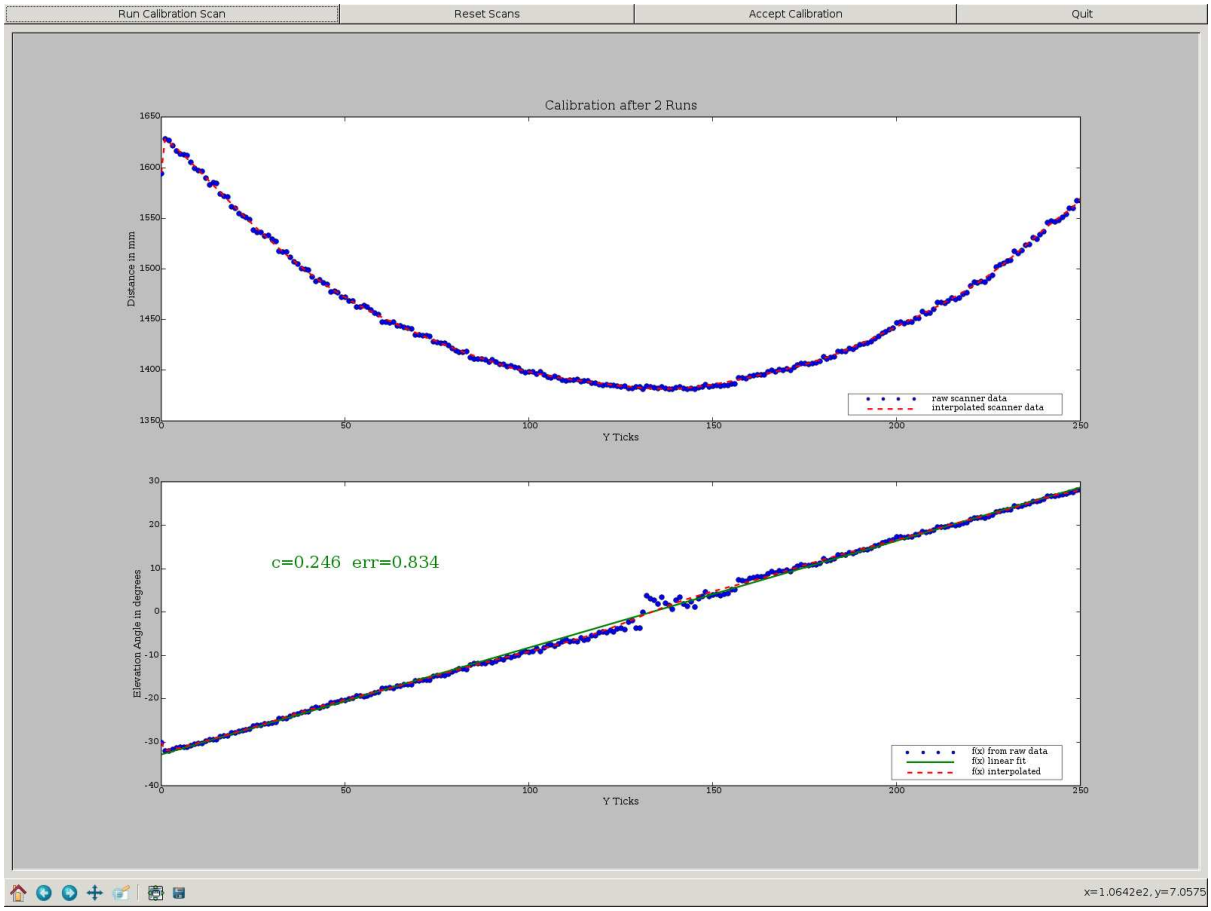


Figure 2.7.: Laser calibration application

2.1.3.1. Mathematical Background

Camera modeling relies heavily on the geometric laws of projective space, which allow a convenient mathematical description of perspective effects. Homogeneous coordinates are an integral part of projective geometry. In this paragraph, we provide a short introduction to projective geometry and homogeneous coordinates. For a more thorough discussion of projective geometry, see Mohr and Triggs [1996]. Hartley and Zisserman [2004] as well as Schreer [2005] deal with virtually all of the issues discussed below including camera modeling and calibration.

Homogeneous Coordinates in Projective Space In projective space, vectors are expressed in homogeneous coordinates. To transform an n-dimensional Euclidean vector to homogeneous coordinates, one simply adds another dimension to that vector, resulting in a vector of length $n+1$. This new element is set to be 1. $[x, y, z]^T$ becomes $[x, y, z, 1]^T$. An equivalence relation holds for all vectors in projective space of the form $[x\alpha, y\alpha, z\alpha, \ldots, \alpha]^T$, where $\alpha \in \Re_{/0}$. By dividing such a vector by α we return to the homogeneous representation introduced above. All elements within an equivalence set possess direction as an invariant property and can be regarded as lying along a *ray* at

position α. To return to an Euclidean vector description it is sufficient to normalize according to α. However, when $\alpha = 0$, we are unable to return to an Euclidean description. Such points are defined to lie in a sub-space at *infinity* and are referred to as *ideal points*. For our purposes, ideal points are used to introduce an abstract entity called the *absolute conic* (see below), which is essential to a popular camera calibration technique [Zhang, 2000] described in subsection 2.1.3.3.

The Plane at Infinity A plane Π can be defined as follows:

$$\forall p \in \Pi : \Pi p = 0; \Pi = \left[n^T, -d\right]^T \tag{2.3}$$

Here, n is the vector normal to the plane and d is a distance to the origin. For points at infinity, whose last component is zero, the plane is defined as

$$\Pi_\infty = [0, \ldots, 0, 1]^T \tag{2.4}$$

Π_∞ is called *the plane at infinity* [Schreer, 2005]. It is useful to us, as the absolute conic is defined to lie on it. The fact that the last components of ideal points is zero causes homographies between Π_∞ and the image plane (see 2.1.3.3) to be invariant of translation.

The Absolute Conic As the concept of the absolute conic is not intuitively accessible, we will just provide the facts needed to understand the arguments used in the calibration technique by Zhang [2000].

- The absolute conic Ω is a circle lying on the plane at infinity (defined in the above paragraph) in three-dimensional projective space.
- $\Omega = I_{3\times3}$. Even though the idea of the absolute conic is very complicated, the fact that it is equivalent to the identity matrix makes easy to use.

We recommend Hartley and Zisserman [2004] for more concise information.

2.1.3.2. The Pinhole Camera Model

The pinhole camera model provides the basis for our camera models (see figure 2.8). Its primary feature is that it performs a perspective transformation according to the laws of central projection. We define a three-dimensional camera coordinate system with its origin at the *focal point*, i.e., the pinhole, of the pinhole camera. The *image/sensor plane* lies parallel to the $X_{cam}Y_{cam}$ - plane (the *focal plane*) of the camera coordinate system at depth $Z_{cam} = f$, where f is called the *focal length*. Points in in the camera coordinate system are projected through the pinhole onto the image plane in a ray-like fashion. The ray that is normal to both image and focal plane and intersects the pinhole is called the optical axis ($[0, 0, Z_{cam}]$). It pierces the image plane at the *principal point*, which serves as an origin for image plane coordinates.

Under central projection, scene depth (value of Z_{cam}) and focal length exert an inversely related effect on the coordinates of projected points. As can be seen in equation 2.5,

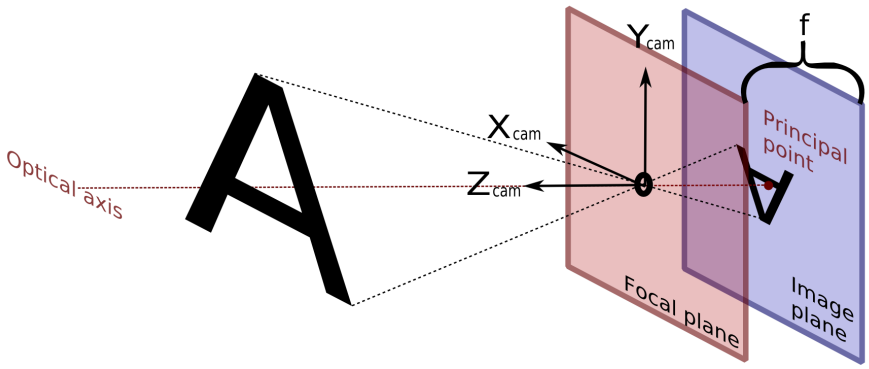


Figure 2.8.: Perspective transformation: A point in 3D space is mapped onto the sensor plane by a simple *pinhole camera* model realizing *central projection*.

points that are farther away from the camera are imaged closer to the principal point ($[0_{img}, 0_{img}]$).

$$u_{img} = \frac{X_{cam} f}{Z_{cam}}, v_{img} = \frac{Y_{cam} f}{Z_{cam}} \tag{2.5}$$

This behavior is implemented using homogeneous coordinates by the *perspective projection matrix* A shown in equation 2.6.

$$\begin{bmatrix} u_{img} \\ v_{img} \\ 1 \end{bmatrix} = \begin{bmatrix} X_{cam} f \\ Y_{cam} f \\ Z_{cam} \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X_{cam} \\ Y_{cam} \\ Z_{cam} \\ 1 \end{bmatrix} \equiv p_{img} = A p_{cam} \tag{2.6}$$

By p_{cam} and p_{img} we denote points described in homogeneous 3D camera coordinates and homogeneous 2D sensor coordinates, respectively. Using homogeneous coordinates, the metaphor of rays passing through a pinhole is elaborated, as all points lying on the ray meeting a certain point on the image plane belong to the same equivalence class, i.e., are the same homogeneous vector.

Intrinsic Camera Properties The simple pinhole model described above is already able to image points abiding the laws of central projection. In this paragraph, we will incorporate certain intrinsic parameters of real cameras into this model. Up to now, imaged points are described in the sensor coordinate system, using the same metrics as the camera coordinate system. However, in a digital camera, incident light is sampled and digitized by a photosensitive *CCD chip* (charge coupled device) positioned in the sensor plane. The conversion to pixel metrics as well as certain geometric properties of the CCD chip are modeled by the *camera matrix* C shown in 2.7.

$$C = \begin{bmatrix} res_x & skew & pp_x \\ 0 & res_y & pp_y \\ 0 & 0 & 1 \end{bmatrix} \tag{2.7}$$

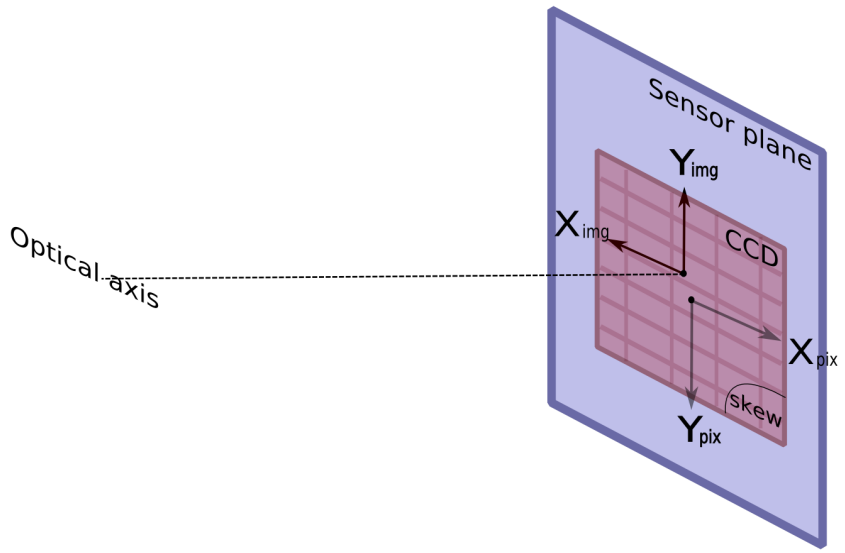


Figure 2.9.: Intrinsic transformation: Points in the sensor plane are converted to the CCD chip's coordinate system and metrics.

The horizontal and vertical sampling resolution (e.g. $\frac{pixels}{cm}$) of the CCD chip are represented by res_x and res_y. These are the factors converting camera coordinate metrics to pixel metrics. The principal point may not be identical to the origin of the CCD chip coordinate system. This may be due to a shift of the chip relative to the principal point or to convention, as most image coordinate systems are defined relative to one of the image corners. The shift between the principal point and the CCD origin is described by $[pp_x, pp_y]$. The metrics of these offset compensation factors is pixels. For the sake of completeness, we also include a *skew* parameter to model non-orthogonal CCD axes configurations. In most cameras, the CCD chip will not be skewed and the corresponding parameter is set to zero. Equation 2.8 converts imaged points from image plane coordinates to pixel coordinates.

$$\begin{bmatrix} u_{pix} \\ v_{pix} \\ 1 \end{bmatrix} = \begin{bmatrix} res_x & skew & pp_x \\ 0 & res_y & pp_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u_{img} \\ v_{img} \\ 1 \end{bmatrix} \tag{2.8}$$

Combining camera matrix and perspective projection matrix yields the enhanced pinhole model: $p_{pix} = CAp_{cam}$.

Extrinsic Camera Properties Usually, the three-dimensional points we want to project onto the image plane are not described in camera coordinates but using an arbitrary *world coordinate system*. Our world coordinate system is defined with respect to the laser scanner, measuring points in its own frame of reference. When the relative translation and orientation of the two coordinate systems is known, we can transform points from one frame of reference to the other by a rigid (Euclidean) transformation. This transformation

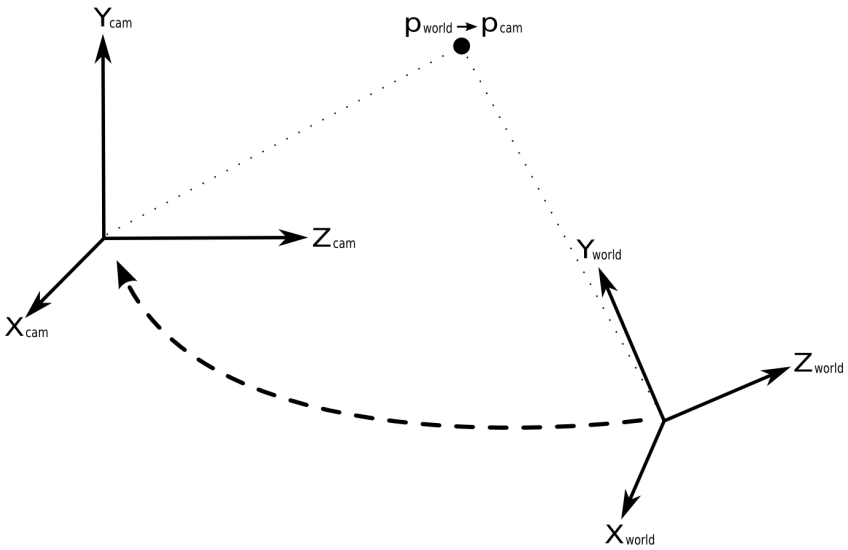


Figure 2.10.: Extrinsic transformation: We transform the 3D-Data from an arbitrary world coordinate system to a camera-centric coordinate system. This is a purely Euclidean transformation.

is known as *extrinsic transformation* (2.9) which is depicted in figure 2.10.

$$E = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.9}$$

Rotation is performed by R, the upper 3×3 sub-matrix of E, while translation along each axis is coded in t. As E is a rigid transformation, R is orthonormal. This is an important constraint used in optimization of the extrinsic parameters obtained during camera calibration. In equation 2.10, the complete pinhole camera model including intrinsic and extrinsic parameters is shown.

$$p_{pix} = \begin{bmatrix} res_x & skew & pp_x \\ 0 & res_y & pp_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} p_{world} \tag{2.10}$$

Lens Distortion Real lenses cause nonlinear distortions of the projected image. These can be divided into radial and tangential distortions. Knowledge of lens distortion parameters enables us to either mimic or correct for these effects. An important application for this knowledge is undistortion of images prior to stereo matching. Even though the distortion is dominated by the first component of radial distortion [Schreer, 2005, Zhang, 2000], the OpenCV functions we used return the first two radial and tangential components. Consequently, we used these parameters as well. For a more detailed account of

lens distortion, we refer to Brown [1971], Fryer and Brown [1986] and Forsyth and Ponce [2002].

2.1.3.3. Intrinsic and Extrinsic Calibration

The camera model introduced above contains eleven unknown parameters. There are five intrinsic unknowns and six extrinsic unknowns. Focal length need not be known, as it is only a scaling factor that can be incorporated into the metrics conversion factors res_x and res_y. Consequently, we assume that $f = 1$. Lens distortion parameters are approximated after solutions for the other parameters have been found.

In order to determine the values of these parameters, we used functions provided by the OpenCV library. Its calibration functions (*CalibrateCamera2* and *FindExtrinsicCameraParams2*, see B.2.3) implement a method developed by Zhang [2000]. The procedure can be organized as follows:

1. Computing homographies ($N \geq 3$) between a calibration plane and its images.
2. Extracting intrinsic camera parameters from these homographies.
3. Computing the extrinsic parameters for each homography.
4. Approximate distortion parameters.
5. Optimize solution by minimizing projection error.

In the following paragraphs, we will elaborate on each of these points.

Homography computation We acquire at least three images of a planar calibration body at various orientations in space (see figure 2.11).

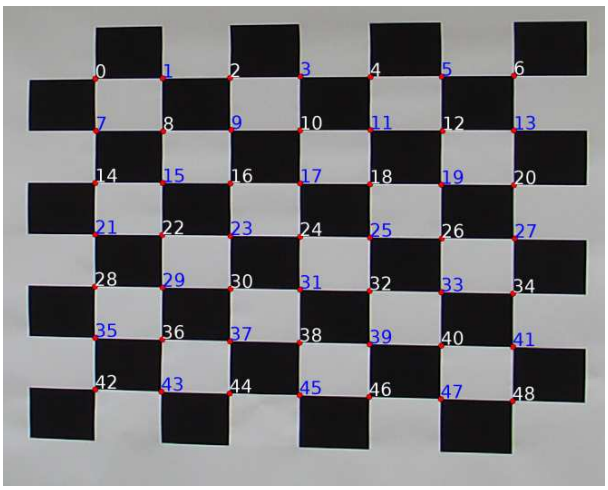


Figure 2.11.: Our calibration plane with checkerboard pattern attached. Automatically detected inner corners are shown.

We assume the calibration plane to lie in the X_{world}, Y_{world} plane, i.e., that $Z_{world} = 0$ for all points in this plane. This reduces equation 2.11 to equation 2.12.

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} res_x & skew_x & pp_x \\ skew_y & res_y & pp_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ 0 \\ 1 \end{bmatrix} \tag{2.11}$$

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} res_x & skew_x & pp_x \\ skew_y & res_y & pp_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & t_1 \\ r_{21} & r_{22} & t_2 \\ r_{31} & r_{32} & t_3 \end{bmatrix} \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} \equiv p_{pix} = A\,[r_{i1}, r_{i2}, t]\,p_{world} \tag{2.12}$$

$H = A\,[r_{i1}, r_{i2}, t]$ is a linear, nonsingular 3×3 matrix constituting a *homography* between calibration plane and image plane. As the homography is defined in projective space, it is defined up to scale. Consequently, we may arbitrarily choose one element of H, leaving us with eight unknown parameters.

In order to impose constraints on these unknowns, we have to establish point correspondence between points on the calibration plane and their image coordinates. Each point correspondence provides us with two constraints, so we require at least four pairs of points. Using the direct linear transform (DLT) algorithm all of the unknowns in H can be determined [Shapiro, 1978].

Point correspondence is established by finding salient points in images of a planar calibration body. In our case, these salient points are the inner corners of a black and white checkerboard pattern attached to the calibration plane (see figure 2.11). For Zhang's method, we do not need to know the world position of the calibration body, as he assumes a local world coordinate system attached to the calibration plane.

As we wish to know the extrinsic parameters for transforming laser scanner coordinates to camera coordinates, we will use Zhang's method to compute the intrinsic parameters of the camera and calculate the extrinsic coordinates separately (see 2.1.3.3). In practice, this means that we will use *optimal* coordinates. As mentioned above, the calibration plane is defined to lie at $Z_{world} = 0$. Furthermore, we define the world coordinates to be described in laser scanner metrics (cm) and the 7×7 inner corners to be equally spaced on a grid of $57.75cm \times 57.75cm$. This means that even though we use fictional world coordinates, the metrics conversion parameters will be computed as if laser scanner metrics were used.

In order to find the image points corresponding to our three-dimensional checkerboard corner coordinates, one has to detect the intersections of high contrast gradients in the images. These are the most likely candidates for inner corners of a checkerboard pattern. In order to detect these salient points, one usually uses adaptive thresholding to convert the image to a binary representation. This is followed by optional image erosion [Haralick et al., 1987], edge detection [Canny, 1986], optional dilation [Haralick et al., 1987], line detection [Duda and Hart, 1972] and computation of line intersections. We simply used the OpenCV (see B.2.3) function *FindCornerSubPix* providing us with inner corner coordinates at sub-pixel accuracy.

Extracting Intrinsic Parameters This paragraph uses the concept of ideal points and the absolute conic. We introduce these terms in 2.1.3.1. Every conic in projective space can be projected onto another conic by projective transformation H(2.13).

$$C^B = H^{-T} C^A H^{-1} \tag{2.13}$$

This is also true for the absolute conic Ω. When we define H to be our pinhole camera model CE, we have the *image of the absolute conic* (IAC, ω) projected onto our image plane. As is true for all ideal points, points on the absolute conic are of the form $[\ldots, 0]$. In this manner, CE is reduced to the homography CR; $R = [r_{i1}, r_{i2}, r_{i3}]$, as the translation column of E is multiplied by zero. As $\Omega = I_{3\times3}$, 2.14 can be reduced to 2.15.

$$\omega = (CR)^{-T} \Omega (CR)^{-1} \tag{2.14}$$

$$\omega = (CR)^{-T} I_{3\times3} (CR)^{-1} = C^{-T} R R^{-1} C^{-1} = C^{-T} C^{-1} \tag{2.15}$$

This means that ω depends solely on the intrinsic properties of the camera. Inverting ω yields $\omega^{-1} = CC^T$. If ω can be obtained, C can be extracted using Cholesky decomposition [Press et al., 1992] because C is of upper triangular form. $C^{-T}C^{-1}$ is symmetric and determined by six parameters. To determine ω, we need to compute at least three homographies as shown in 2.1.3.3. Each homography imposes the following two constrains on ω [Zhang, 2000]:

$$h_1^T \omega h_2 = 0 \tag{2.16}$$

$$h_1^T \omega h_1 - h_2^T \omega h_2 = 0 \tag{2.17}$$

Here, h_i is the ith column of the homography shown in equation 2.12. Given N homographies, we can create a $2N \times 6$ matrix M composed of the stacked linear equations 2.16 and 2.17 and a vector b containing the unknowns of ω such that $Mb = 0$. Solving for b yields ω from which C is extracted.

Calculating Extrinsic Parameters Once the internal camera parameters are determined, it is very simple to extract and refine the extrinsic parameters from each homography [Zhang, 2000]. To obtain the extrinsic parameters for the rigid transformation from laser coordinates to camera coordinates, we used the OpenCV (see B.2.3) function *FindExtrinsicCameraParams2* which minimizes the projection error using known intrinsic parameters and distortion coefficients (see equation 2.18). To find the checkerboard in a laser scan, we developed a tool that allows the user to manually select the board in a laser scan. However, we experienced considerable instabilities when calculating extrinsic coordinates. In the end we had to manually measure the extrinsic parameters. We address this problem and potential solutions to it in section 4.2.

Optimizing the results To refine the results and to estimate lens distortion coefficients d the sum of squared projection errors over all inner corner points is to be minimized. The error function is shown below, where d are distortion coefficients while C and E represent the camera and extrinsic matrix, respectively (as used in e.g. 2.10).

$$\sum_{corners} \|p_{pix} - \text{project}(C, E, d, p_{world})\|^2 \tag{2.18}$$

The optimization is initialized using C and E as obtained above and $d = 0$.

2.1.3.4. Summary

In this section, we introduced the popular pinhole camera model and the meaning of its intrinsic, extrinsic and lens distortion parameters. We now are able to use images of a calibration body of known geometry to impose constraints on these free parameters. This is done in the following manner:

- We approximate the homography containing both intrinsic and extrinsic parameters. To do this, we require knowledge of 3D→2D point correspondence for views of a calibration body. These point correspondences constrain the free parameters of the homography. Given enough constraints, the homography can be computed. For each view, we can find one homography.
- Intrinsic and extrinsic parameters are contained in the homography computed above. Given at least three of these homographies, the intrinsic parameters can be extracted using the image of the absolute conic (IAC) which is independent of the extrinsic parameters.
- The intrinsic parameters are assumed to be constant in each view and in the homography associated with it. As the intrinsic parameters are known now, we can easily extract the extrinsic parameters from each homography.
- Intrinsic and extrinsic parameters are refined in an optimization procedure minimizing projection error. Lens distortion parameters are also approximated in this step.
- To obtain extrinsic parameters for an arbitrary world coordinate system (e.g. the laser scanner's frame of reference), one fixes intrinsic and distortion parameters and minimizes the projection error as a function of the extrinsic parameters.

2.2. Image Rectification

Rectification is the transformation of two images of a stereo pair such that pairs of conjugate epipolar lines become collinear and parallel to the horizontal image axes.
This is achieved by defining two new virtual cameras by rotating the actual ones around their optical centers until their focal planes becomes coplanar. Thereby, we ensure that the epipoles are located at infinity, and therefore all epipolar lines are parallel. This

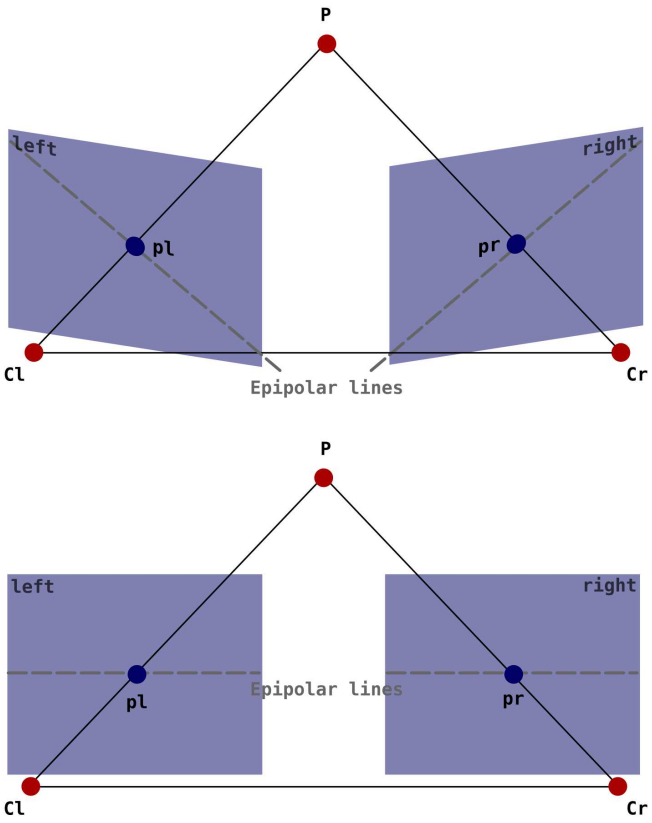


Figure 2.12.: Epipolar lines before (top) and after (bottom) rectification

rotation is depicted in figure 2.12. This is an important feature greatly simplifying later processing in stereo matching (see 2.3), as corresponding pixels in the two images are located along these horizontal epipolar lines.
To actually carry out the required rotation, knowledge of the actual camera matrices is necessary. The previous section 2.1.3 described camera calibration under a set of preconditions: stereo image pairs showing a calibration pattern and corresponding world coordinates are available.
In the course of this project we also dealt with uncalibrated stereo images acquired earlier. Even though we lack intrinsic and extrinsic parameters for these images, the epipolar geometry can still be described in terms of a *fundamental* matrix $\mathbb{F}$.

$$p_{right}^T \mathbb{F} p_{left} = 0 \tag{2.19}$$

$$\text{with } p = (u, v, 1)^T \tag{2.20}$$

$\mathbb{F}$ maps points of one image to their corresponding points in the other image.
Based on this, we implemented a rectification technique that does not depend on any calibration matrices acquired earlier and works solely on stereo image pairs. In the following, we will outline the main steps of the algorithm:

Find Features in one image that are very likely to be unique. We use a standard method of computer vision: The Harris corner detector [Harris and Stephens, 1988]. We calculate

the local covariance matrix of gradients M (2.21) and find it's minimum Eigenvalue.

$$M = \begin{bmatrix} \sum_{p \in P} \left(\frac{\Delta Img}{\Delta x} \right)^2 & \sum_{p \in P} \left(\frac{\Delta Img}{\Delta x} \cdot \frac{\Delta Img}{\Delta y} \right) \\ \sum_{p \in P} \left(\frac{\Delta Img}{\Delta x} \cdot \frac{\Delta Img}{\Delta y} \right) & \sum_{p \in P} \left(\frac{\Delta Img}{\Delta y} \right)^2 \end{bmatrix} \tag{2.21}$$

with a neighborhood of P pixels and image luminance values Img.
This yields a quantitative description of "how unique" a corner is with respect to it's neighbors. Applying non-maxima suppression and cutting off Eigenvalues below a given threshold then delivers the strongest corners. As a last step, a minimum distance between these corners is ensured to ease matching against the other image of the pair and make mismatches highly unlikely.
This functionality is implemented in `cvGoodFeaturesToTrack` (c.f. B.2.3).

Find Match For every point found in the previous step, we extract a 11×11 patch around it from the first image and cross-correlate it with a 200×200 patch around the point in the second image. If the operation yields a maximum correlation value greater than 0.95, we accept this point pair as valid and use it for the next step. Figure 2.13 shows a sample image pair with points found and the vectors between corresponding points.

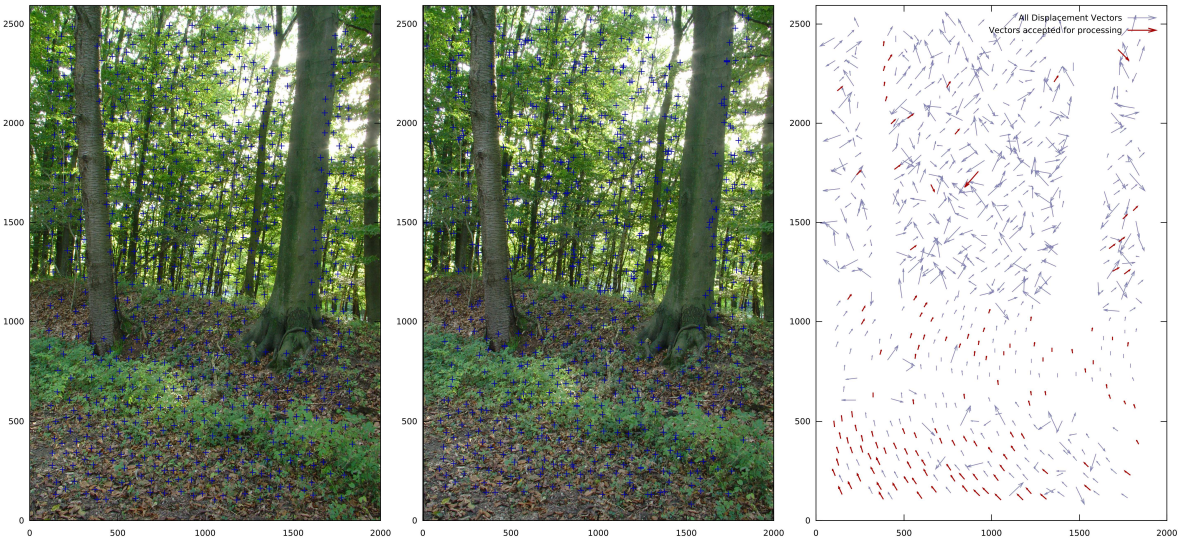


Figure 2.13.: Point correspondence between two images of a stereo pair, used for calculating the fundamental matrix.

Calculate Fundamental Matrix The point pairs identified are then used in the *RANSAC* algorithm. In a nutshell, it repeatedly builds a fundamental matrix based on a sub-sample of all points and calculates the resulting error for the whole set. It terminates when the

error has reached a given threshold. A detailed account on this technique is given in Hartley and Zisserman [2004]. For our implementation, we use `cvFindFundamentalMat` (c.f. B.2.3).

Create Scan-lines The resulting fundamental matrix allows for calculation of coordinates of dense epipolar lines in both images (`cvMakeScanlines` (c.f. B.2.3)). These one pixel wide lines are cut out of the original image and reassembled in a new image, where they are stacked horizontally. The assembled image is then the rectified version of the input image.

2.3. Stereo Matching

The goal of stereo matching is to find corresponding points in two views of a scene. Correspondence means that both image points are projections of the same point in three-dimensional space. One assumes that the two images of this scene point are exceptionally similar. Consequently, the stereo matching problem can be formulated as a search for similar features in a pair of images. The disparity value for a certain pixel is obtained by calculating the difference vector between this pixel's image coordinates and the coordinates of its corresponding point. As we only deal with the horizontal component of this vector, we use *disparity* and *horizontal disparity* synonymously. In this section, we will introduce two stereo matching algorithms, namely normalized cross-correlation (2.3.1) and belief propagation (2.3.2). We tested the performance of these algorithms on the stereo images of natural scenes. Results, including a comparison to scanned ground truth, can be found in section 4.1.

On the epipolar constraint According to the *epipolar constraint*, matching pixels in the other image are to be searched along the *epipolar lines*. This simplifies stereo matching to one-dimensional search. The stereo matching algorithms used in this section use horizontal scan lines , i.e., search only along rows. As a consequence, they require a reasonably parallel stereo setup with horizontal epipolar lines to fulfill the epipolar constraint. It is possible to *rectify* a stereo image pair to align the epipolar lines with the horizontal axis. We refer to Loop and Zhang [1999], Schreer [2005] and section 2.2 for further information. Before stereo matching, we undistorted the images using the distortion coefficients obtained during camera calibration. We used the OpenCV function `cvUndistort2` (c.f. B.2.3) for this purpose. Furthermore, the global disparity was minimized by shifting the images with respect to one another according to the maximum response of a normalized cross-correlation similarity measure. This makes it more comfortable to view the stereo images on a 3D monitor. Another effect of this shift is that disparity may now also be negative.

2.3.1. Normalized Cross-Correlation

The first stereo matching algorithm we used to calculate a dense disparity map was a simple *block matching* algorithm using normalized cross-correlation (NCC) as a similarity

measure [Onat et al.]. For every pixel in image l, a rectangular region around this pixel is compared to equally sized blocks in image r whose center is shifted along the scan line. We denote the similarity value at shift Δ as $s(\Delta)$. Finding $\Delta_{max} = max(s(\Delta))$ yields the most probable disparity value. The block matching method is illustrated in figure 2.14. In the following subsection, similarity computation using NCC is described in more detail.

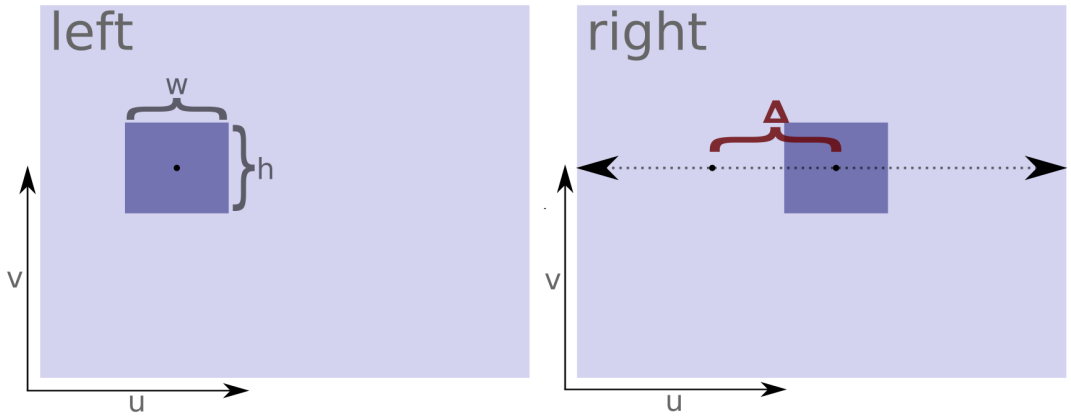


Figure 2.14.: In a *block matching* approach such as *NCC*, the similarity s of neighborhoods at disparity Δ is computed. Maximizing $s(\Delta)$ yields the most probable disparity.

2.3.1.1. NCC as a similarity measure

Plain cross-correlation can be regarded as a sliding template matching algorithm. In signal processing, it is used to find a known (template) signal in a second signal. The template signal vector is compared to a portion of the second signal vector using the dot product operation. This is equivalent to a convolution of the two signals, where the template becomes the convolution kernel. In our case, the signal vectors are taken from two-dimensional image regions. The template l is a correlation window (taken from the left image) which is compared to an equally sized region r on the scan line in the right image. This is shown in equation 2.22.

$$\mathrm{CC}(u,v) = \sum_{y=0}^{h-1} \sum_{x=0}^{w-1} l(x,y) \times r(u+x, v+y) \tag{2.22}$$

This is equivalent to a dot product which can be formulated as shown below (2.23).

$$\langle l, r \rangle = \cos(\angle(l,r)) \times ||l|| \times ||r|| \tag{2.23}$$

Here $||r||$ denotes the vector norm (length) of r. While cross-correlation can be called a *sliding dot product*, NCC is a *sliding correlation coefficient* (see equation 2.24). NCC

compares two data vectors, i.e. image patches, using the cosine of the angle between them, which is equivalent to their correlation coefficient. The cosine is extracted by dividing equation 2.23 by $||l||$ and $||r||$.

$$\mathrm{NCC}(u,v) = \frac{\mathrm{CC}(u,v)}{\sqrt{\sum_{y=0}^{h-1}\sum_{x=0}^{w-1} l(x,y)^2 \times \sum_{y=0}^{h-1}\sum_{x=0}^{w-1} r(u+x,v+y)^2}} \tag{2.24}$$

As pure cross-correlation contains both cosine and length components, it is sensitive to global illumination differences corresponding to the length of the data vectors. This is not desired, as the important measure is the relative difference between vectors corresponding to the angle between them. Thus, NCC is the preferred method for stereo matching.

After a short period of experimentation we chose a correlation window size of 15×15 pixels which we used with the NCC implementation in `cvMatchTemplate` (c.f. B.2.3). This function is extremely optimized, yet does not use color information. The only way to compute NCC any faster is to exploit the Fourier representation of convolution which is widely used for CC [Bracewell, 1986] and has been extended to NCC by Lewis.

2.3.2. Belief Propagation

In this section, we will give a brief overview of the belief propagation algorithm and how we used it for stereo matching. For a thorough introduction to belief propagation we recommend Kschischang et al. [2001] and Yedidia et al. [2003]. For the representation of images as MRFs and inference thereupon, we refer to Geman and Geman [1988]. The technique we employed for stereo matching is taken from Felzenszwalb and Huttenlocher [2004]. Consequently we have adopted a nomenclature similar to that of Felzenszwalb and Huttenlocher [2004].

Belief Propagation is a fast algorithm for inferencing on graphical models, such as Markov random fields (MRF). It accurately estimates the marginal probability of a certain graph state by passing messages between neighboring nodes.

2.3.2.1. Markov Random Fields

A Markov random field is an undirected graphical model. Its vertices represent random variables while its edges represent relations or interaction between these variables.

Estimating the most probable scene underlying one or more images can be formulated as an inference problem on a pairwise MRF [Geman and Geman, 1988]. In this pairwise MRF, the scene is represented by a two-dimensional lattice x of *hidden* nodes connected to a second, equally sized lattice of *observable* nodes y, which may be the image of x. As can be seen in figure 2.15, there is an observable value y_i (e.g. a pixel value) for every scene node x_i. Furthermore, hidden nodes (scene nodes) interact with one another within the neighborhood defined by the Markov property of the MRF.

In our case, the scene is represented by a disparity map. Thus, x is a MRF with a node for each pixel and the label f of a node is the disparity at this scene point. We want to find the most probable disparity labels f for x given the evidence y. The evidence is drawn from a stereoscopic photograph of the scene. At scene point x_i, the evidence y_i

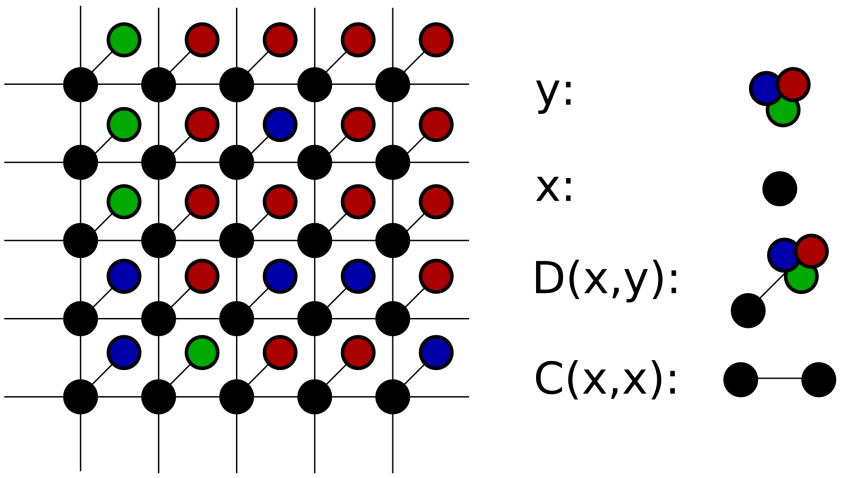


Figure 2.15.: An image and the underlying scene can be represented as a rectangular pairwise MRF. Here, x are the hidden nodes representing scene properties while y is an observed image property (e.g. pixel color value) at each node of x. This *evidence* influences the scene nodes via the *data costs* $D(x, y)$. Between neighboring scene nodes there exists an *compatibility* relation (e.g. smoothness). It is denoted by $C(x, x)$.

is a vector whose dimensionality corresponds to the number of possible x-labels, i.e., the disparity range. The entries of y_i are the costs of assigning the corresponding disparity value f_i to x_i. As this is based on the observed stereo images, it is also referred to as *data cost*. According to Felzenszwalb and Huttenlocher [2004], the data cost for label f_q is $D(f_q) = \lambda \min(|Img_{left}(x, y) - Img_{right}(x - f_q, y)|, \tau)$, where τ is a truncation value and λ is a scalar multiplier.

2.3.2.2. Belief Propagation on MRFs

Belief propagation allows us to compute marginal probabilities of MRF states by passing messages between neighboring nodes. Each node receives messages from its neighbors concerning their beliefs about their states. These messages, along with local data costs and compatibility relations constitute this nodes' own belief (see figure 2.16).

There are two versions of belief propagation, *sum-product* and *max-product*. They use slightly different message update rules and compute different posterior probabilities [Cowans, 2004]. Sum-product computes the posterior probability of each label, while max-product computes the joint posterior probability of all labels. Choosing the most probable state of each individual node does not automatically result in globally maximal probability. We wish to achieve the latter. As done in Felzenszwalb and Huttenlocher [2004], we use the max-product version reformulated as a min-sum using negative log probabilities. The message update rule is shown in equation 2.25. It defines the message node p sends to node q at iteration t. Here, $\mathcal{N}(p)$ denotes the neighborhood of node p, while $\mathcal{N}(p) \backslash q$ refers to all neighboring nodes of p except q.

$$m^t_{p\to q}(f_q) = \min_{f_p}(C(f_p, f_q) + D(f_p) + \sum_{s\in\mathcal{N}(p)\setminus q} m^{t-1}_{s\to p}(f_p)) \tag{2.25}$$

$$b_q(f_q) = D(f_q) + \sum_{p\in\mathcal{N}(q)} m^t_{p\to q}(f_p) \tag{2.26}$$

After a sufficient number of iterations, a final belief vector is computed for each node. Then, the label f_q minimizing $b_q(f_q)$ is chosen to be the final label at each node. The composition of a belief vector is shown in equation 2.26 and figure 2.16.

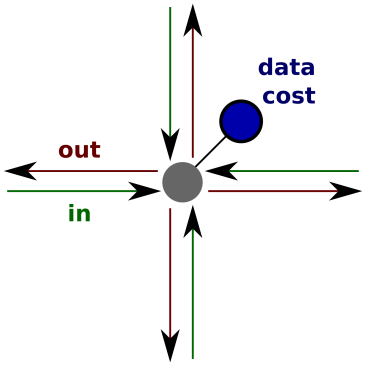


Figure 2.16.: The belief a hidden node has about the probability of its labels is composed of local data cost and messages from its neighbors.

2.3.2.3. Implementation

Felzenszwalb and Huttenlocher [2004] developed three techniques to speed up belief propagation for inferencing on MRFs that represent images.

1. The compatibility function $C(x_i, x_j)$, which is used to enforce local smoothness is reduced to a difference measure between the labels f_i and f_j. $C(x_i, x_j) = V(f_i - f_j)$, where V is a linear or quadratic cost function of the label difference. Using a difference measure instead of explicit comparison of neighboring labels reduces complexity significantly.

2. As neighborhoods simply consist of the four nearest neighbors on a 2D grid, messages can be updated according to a *checkerboard update scheme*, reducing memory requirements for storage of messages by 50%.

3. It usually takes a large number of iterations to account for long-range interactions. This number is reduced by employing a coarse-to-fine approach.

Source code implementing these techniques is provided by Felzenszwalb and Huttenlocher [2004]. We reimplemented the algorithm to match our requirements. The main changes include:

- *Input Data* can be any file format supported by OpenCV (png, tif, jpg), allowing for direct application without previous conversion.
- *Disparity Range* is a run-time rather than a compile-time parameter, significantly reducing adaptation efforts per image.
- *Discontinuity Costs* can be fed from an additional data source such as laser scanning results.
- *Fast Standard Routines* from the OpenCV libraries replace self-written, unoptimized routines.
- *Pixel dissimilarity*, which is the basis for data cost computation, is measured in the color domain as described below.

A new dissimilarity measure for data costs As the pixel dissimilarity measure used by Felzenszwalb and Huttenlocher [2004] is only defined for gray-scale images, we extended it to include color values. The new difference measure was inspired by Grest et al. [2003] and is based on the HSL representation of color space. The two components of the measure can be seen in equation 2.27 and 2.28, where the variable names are the same as in figure 2.17.

$$A = \min(|H_l - H_r|, |H_r - H_l|, 64) \cdot \frac{2 \cdot S_l \cdot S_r}{255^2} \tag{2.27}$$

$$B = |L_l - L_r| \cdot \left(1 - \frac{S_l \cdot S_r}{255^2}\right) \cdot 0.5 + |L_l - L_r| \cdot 0.5 \tag{2.28}$$

Two colors are compared using a weighted difference of their hue angles A and their luminance values, B. The full dissimilarity measure is defined by $A+B$ and is represented by a value between 0 and 255. The maximum angular difference between two hues is set to 64. As H, S and L are stored using 8 bit integer values, this corresponds to a truncation of the angular difference at $\frac{\pi}{2}$. The maximum contribution of hue (A) to the dissimilarity measure is 128. This number is weighted by the saturation. If the saturation of both colors is high, the luminance difference B and the hue difference A both have the same influence on the dissimilarity measure. As soon as one color is of weak saturation, the effect of B is strengthened at the expense of A.

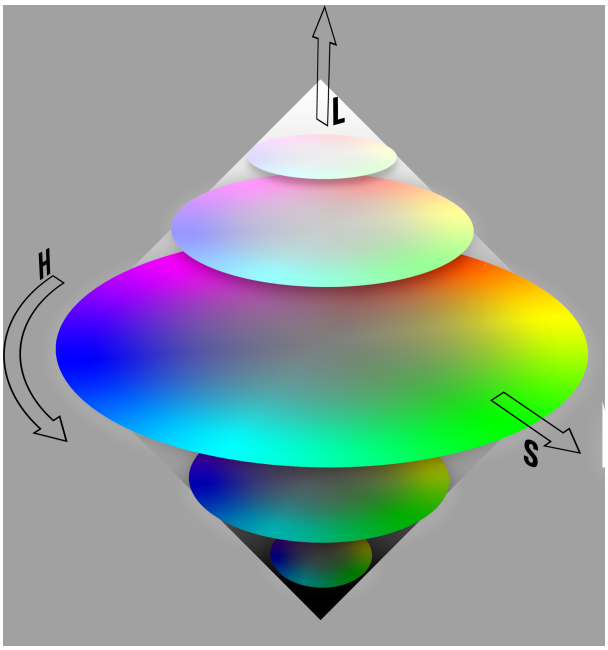


Figure 2.17.: The HSL representation of color space. For our difference measure, H, S and L are represented by values from 0 to 255. (Figure Copyright ©by Alexandre Van de Sande, 2005.)

3. Application

The target application of the generated data primarily lies in the field of psychophysical and electrophysiological experiments at the Neurobiopsychology Lab, Univ. Osnabrück. The technology employed during further use limits the resolution of images to a maximum of 1280×1024. We therefore use selected regions of the full size images acquired as input to the following processing. Nevertheless, all methods are suitable for processing of other input sizes and have been successfully tested with full size images.

3.1. Data Acquisition

3.1.1. How to scan

In the following we sketch a typical scan session. The steps from *Placement* on are repeated for every scan run.

Preparation at the Lab includes charging of the batteries (typically 12hrs each) and carrying out a test-run to validate the systems operational readiness. Components not fixed to the stand, such as stereo rig, laser scanner, batteries and notebook are securely transported in a stable padded box. Participating personnel is instructed about security measures regarding operation of a laser scanner according to Deutsches Institut für Normung [2001].

Preparation on Site consists of connecting power and USB cables, and then activating the power switches. All cables are long enough to allow for simultaneous operation of stereo rig and laser scanning, removing the need for powering up and down when switching between acquisition modes. As the necessary USB devices are auto-detected and the scanning application is auto-started, no more manual interaction is necessary.

Placement of the stand requires either solid ground, with the adjustable legs accounting for ground unevenness, or forest soil that allows for driving the legs in firmly. The field of view can then be checked by placing the stereo rig on top of the stand and examining the LCD displays of the cameras.

Laser Scanning is started by placing the Scanner on the two metal pins on top of the platform and hitting the "Start Scanning" button in the application running on the notebook. After about 30 to 60 seconds, depending on horizontal and vertical resolution, the application signals completion of the scanning process and allows the user to visually inspect and optionally save or discard the acquired scan.

Image Acquisition requires replacing the scanner with the rig and manual synchronization of camera parameters. When the LCD display of the cameras shows a satisfactory image, photos are taken by pressing the "shoot" button on the external synchronization device. The cameras take about 20 seconds to store the data on their internal memory sticks, which leaves enough time for replacing the rig by the laser scanner to prepare for the next run.

Data Storage needs to link laser data and the corresponding image pairs. For this purpose, a hot-plug event daemon on the notebook watches for plug-in of the stereo cameras. On connection, it identifies the location of the camera (left/right) by its device id, downloads all available images and assigns them to stored laser scans.

3.1.2. Locations

We conducted six scan sessions at six different sites in mixed forests around Osnabrück, each of them yielding ten to twenty different scenes, with a total of 80 scenes. All scan sessions took place during early October 2005, so that all images show vegetation typical for the summer season. An overview of sites is shown in figure 3.1.

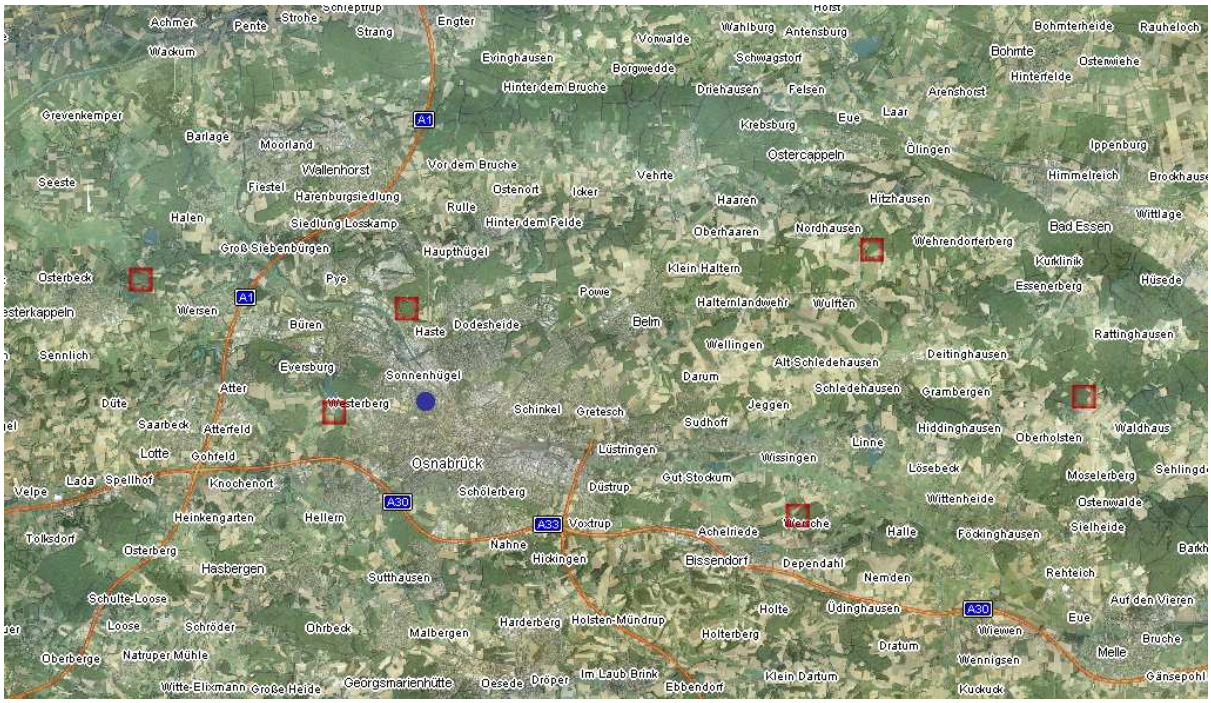


Figure 3.1.: Scanning sites for natural scenes in October 2005

3.2. Disparity Map Generation

With the camera model from section 2.1.3, any 3D point captured by the Laser Scanner can be projected onto the two planes of the acquired images. This is done be the func-

tion `cvProjectPoints2` (c.f. B.2.3). The horizontal component of the difference vector between two corresponding points is then the disparity. The number of projected points depends on the location of the projection pane with respect to the laser scanner's field of view and typically lies between 9000 and 12000 per camera image of size 1280×1024. Figure 3.2 clearly shows the sparse distribution of those projected points.
For creation of dense disparity maps we used a B-Spline interpolation algorithm `MpShepard2d`

Figure 3.2.: Laser Scanner data projected onto one image pane

(c.f. B.2.3) developed by Gammel [1991-2003]. The parameters were fixed to $nq = 33$, $nw = 40$ and $nr = \sqrt{n}$, with n denoting the number of projected points.
A comparison of the disparity maps generated with this method against stereo-matching algorithms described earlier can be found in Appendix A.

3.3. Stimulus Synthesis

Disparity maps generated in 3.2 do not only allow for *description* of view differences between two given images, but also allow for *creation* of artificial image pairs:
As one value in the disparity map describes the displacement of one pixel in an image with respect to the its corresponding point in the second image, we can create e.g. a right image by shifting the pixels of the left image according to the disparity map. However, this does not guarantee a correct depth order of the elements in the resulting image. To achieve this, we apply the classic Painter's algorithm [Foley et al., 1996]: elements are drawn according to their Z-coordinate from back to front, so that near ones *paint over* far ones. This algorithm is a simpler version of the Z-Buffer algorithm, which can also

deal with non-discrete data like polygons with corners of different Z-values. Figure 3.3 shows how a gradient graphic, acting as left image, is transformed to the corresponding right image using a disparity map.

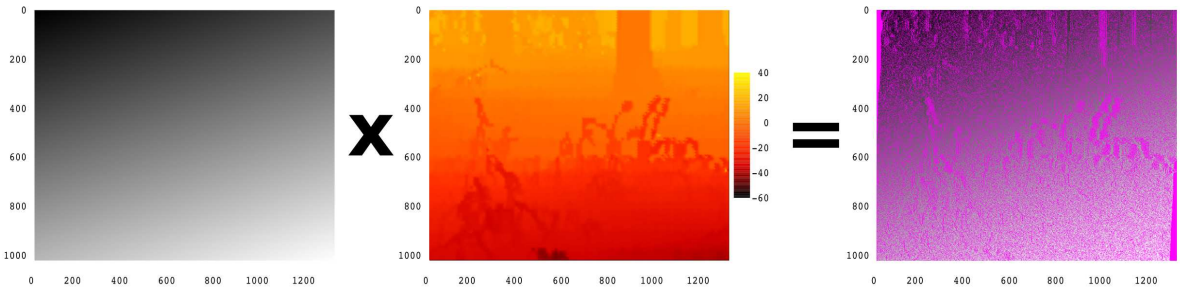


Figure 3.3.: Stereo pair synthesis: Source (left) image and disparity map produce the destination (right) image with Painter's Algorithm. Pink areas indicate that no information is available for that area due to occlusion.

3.4. Visualization

Visual inspection of the results gained in 3.2 is an important step in assessing their quality. However, just superimposing a disparity map on one image or viewing them side-by-side yields little to no insight. We therefore created a viewer application that maps photos onto the 3-dimensional structure of a scene. It allows for real-time interactive manipulation of the users viewpoint. The application was built employing the Visualization Toolkit [Martin et al., 1993-2006], which uses state of the art rendering techniques and an OpenGL output interface. In addition, it features an anaglyph mode, where screen output is rendered in a way suitable for viewing with cyan/magenta glasses, yielding a "real" 3D effect without further hardware requirements.

Figure 3.4.: 3D Visualization of a forest scene

4. Discussion

4.1. Summary and Results

We successfully created a means to measure the three-dimensional structure of natural scenes using a laser range scanner. From these measurements we created dense, high-resolution disparity maps for stereoscopic photographs of these scenes. We claim that these disparity maps are algorithmically unbiased and thus can be used to assess the performance of stereo matching algorithms on natural images. Consequently, we evaluated the performance of belief propagation and block-matching using normalized cross-correlation. The results of this comparison are shown in appendix A. Furthermore, we developed techniques to synthesize stereograms for psychophysical and electrophysiological experiments on three-dimensional viewing in humans.
We collected 80 pairs of scans and images from six sites in the forests surrounding Osnabrück. Up to now about half of these sets were used for data analysis and stimulus synthesis.
As is the case with many practical projects, progress is the result of continuous problem solving. By adapting hardware and software to serve the needs of the users on-site and to improve measurement quality, we reached a high degree of usability and reliability. However, as the scope of this thesis is finite, some problems still are to be solved. In section 4.2, we will provide an overview of these remaining problems and potential solutions to them.

4.2. Problems

4.2.1. Unnecessarily low Signal to Noise Ratio

We operated the cameras at their maximum resolution of 2592×1944 pixels. During the course of this project, we noticed that this was a bad decision. Due to this high resolution we are more susceptible to image noise and violation of the epipolar constraint. The stimulus presentation equipment only handles resolutions up to 1024×1280 pixels we worked with sub-regions of our original images. This artificially decreases the size of the view frustum and the portion of the scene observed by the camera. Consequently, we lose scanner resolution as well, as a smaller part of the scene is sampled less frequently (see figure 3.2).

In the future, we will not crop the images to fit the stimulus presentation requirements but downsample them or use a lower recording resolution. Thus we increase the signal to noise ratio and the number of projected points per image.

4.2.2. Stability of Stereo Rig Configuration

We assume our stereo rig to be reasonably parallel and to maintain its physical configuration across trials. However, there are image pairs where the orientation of the two cameras appears to be massively altered. As we did not notice this during recording, this calls for an improvement of the stereo rigs physical stability.

Image Rectification We rectified a number of image pairs, but were not satisfied with all of the results, as we experienced significant distortion and resampling effects. In a number and of images we were neither able to obtain a reasonable *fundamental matrix* from point correspondence (possibly due to the low SNR) nor were we able to guarantee the correctness of the extrinsic parameters (see 4.2.3 below). Thus, the epipolar geometry remains unknown for these views, rendering them less usefull to stereo matching.
We have to physically enforce the parallel orientation of our cameras and find a stable image rectification algorithm which minimizes image over- or under-sampling and distortion. Such an algorithm could benefit from incorporating known parameters, i.e. intrinsic matrix and lens distortion coefficients.

4.2.3. Quality of Extrinsic Parameters

We were unable to obtain stable and accurate extrinsic parameters using the OpenCV implementation of Zhang's calibration algorithm [Zhang, 2000]. As a solution, we manually measured these parameters. Subjective comparison of projected disparity maps and underlying images tells us that our extrinsic parameters are sound. However, we will have to find a technique to automatically obtain stable extrinsic parameters. Preferably, such a technique will be combined with the fundamental matrix, computed for rectification, to cross-validate extrinsic parameters.

Keep world coordinate noise out of the optimization During extrinsic calibration, we experienced surprisingly unstable results when using inner corner world coordinates measured by the laser scanner. When using optimal (i.e. not measured) world coordinates, results improved a lot. In his calibration algorithm, Zhang [2000] assumes a world coordinate system which is attached to the calibration plane. This is necessary to compute the homographies between calibration plane and image plane that are used to constrain the intrinsic coordinate parameters. In this algorithm, as implemented in the Open Computer Vision library, it is possible to use another kind of world coordinate system, as long as an *initial guess* of the intrinsic parameters is supplied to start the optimization process. However, to get the extrinsic parameters for a "laser-centric" world coordinate system, we propose to not include inner corner coordinates acquired by the laser-scanner into the calibration procedure, as they will introduce additional measurement noise to the optimization. Instead, we should use optimal inner corner world coordinates. Once the extrinsic transformation matrix (E_1) is obtained, we can detect the checkerboard plane in our laser scan and compute the rigid transformation (E_2) necessary to move it to lie as in the optimal coordinate system used for calibration. Then we concatenate the two

transformations : $E_{new} = E_1 E_2$. E_{new} now describes the rigid transformation from laser coordinates system to camera coordinate system which is our new extrinsic matrix.

Improve Scan Quality Some problems with the extrinsic calibration may originate from the way the checkerboard coordinates are found in a laser scan. Up to now, one has to manually select three of the four outer corners of the calibration plane in a program showing the raw depth map of a scan. A more precise selection was not possible as the inner corners of the chessboard where not detectable.
With the new scanning application we also gain remission values, allowing for automatic detection of those corners with sub-pixel accuracy. Mapping these findings from the remission layer to the depth layer then allows for fitting a model of the calibration body to the data. Its refined 3D coordinates will be much more accurate and thereby enhance the performance of the extrinsic calibration.

4.2.4. Irregularities in Servo Movement

During our recording sessions, the scanner's rotation around its horizontal axis was choppy and irregular at times. This lead to decreased vertical resolution and and erroneous measurements and is one of the reasons why we had to discard about half of the data sets.

However, we have pinpointed the source of this problem to lie in the strength of the electrical current our batteries were able to supply. When using stronger batteries, movements become more regular. Future recordings will benefit from this fact, as less data will have to be discarded.

4.3. Outlook

Naturally, future work will focus on the realization of the improvements proposed above. In addition, we wish to achieve the following:

- Public accessibility of data sets. Stereo images, raw scan data and disparity maps will be made available to the research community.
- Improve documentation of the interaction between different parts of our software package.
- Combine laser scans and stereo matching algorithms. We will assess in how far depth or disparity measurements may be used to provide initial guesses or constraints to stereo matching algorithms.
- Knowledge transfer. We will continue to actively support further psychophysical and electrophysiological experiments using three-dimensional stimuli. Furthermore, we will contribute to computer vision applications in autonomous robots and on a binocular robot head. The latter will be done as a part of the European FP6-IST project *Perception On Purpose*.

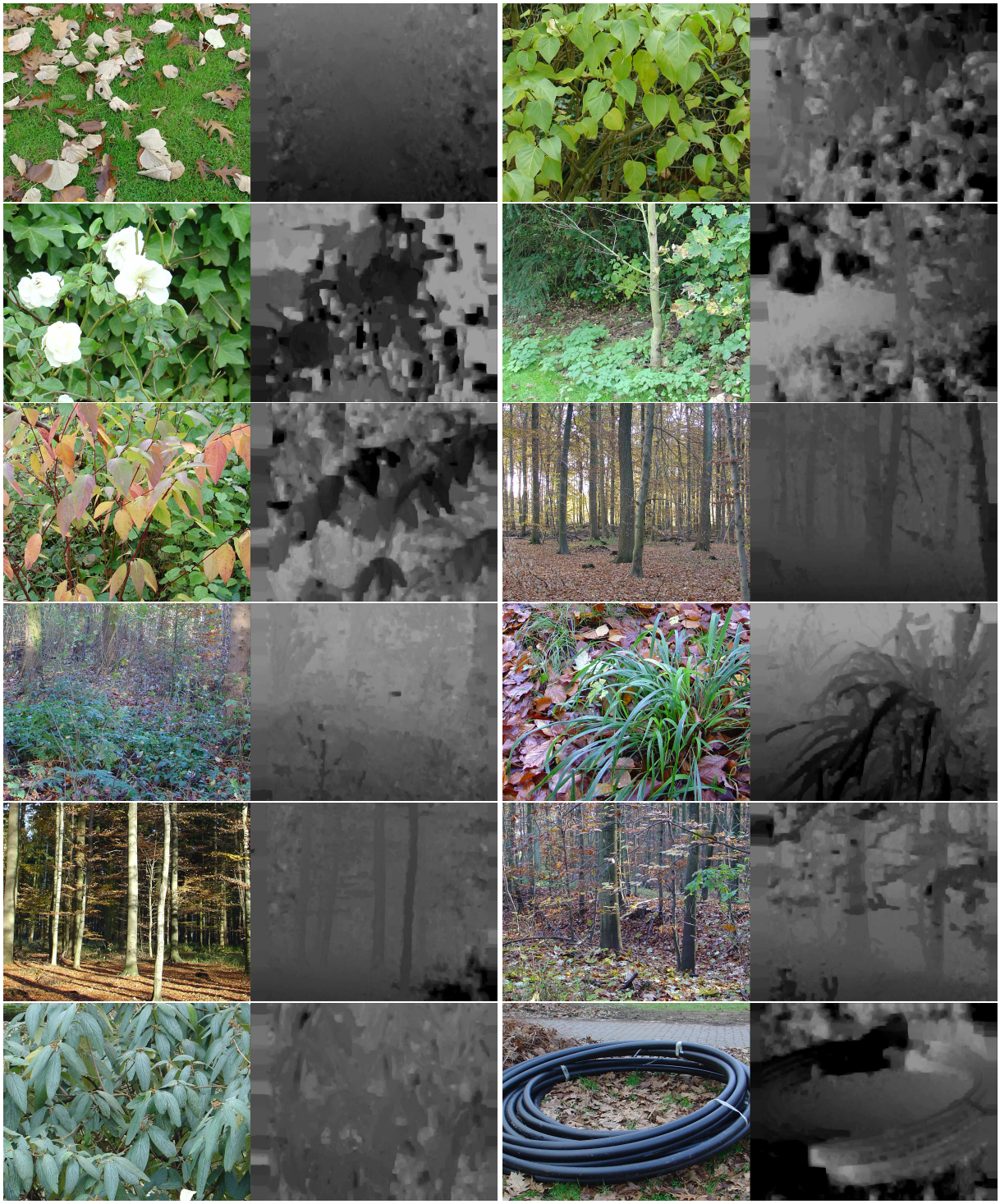
A. Descriptive Scene Statistics

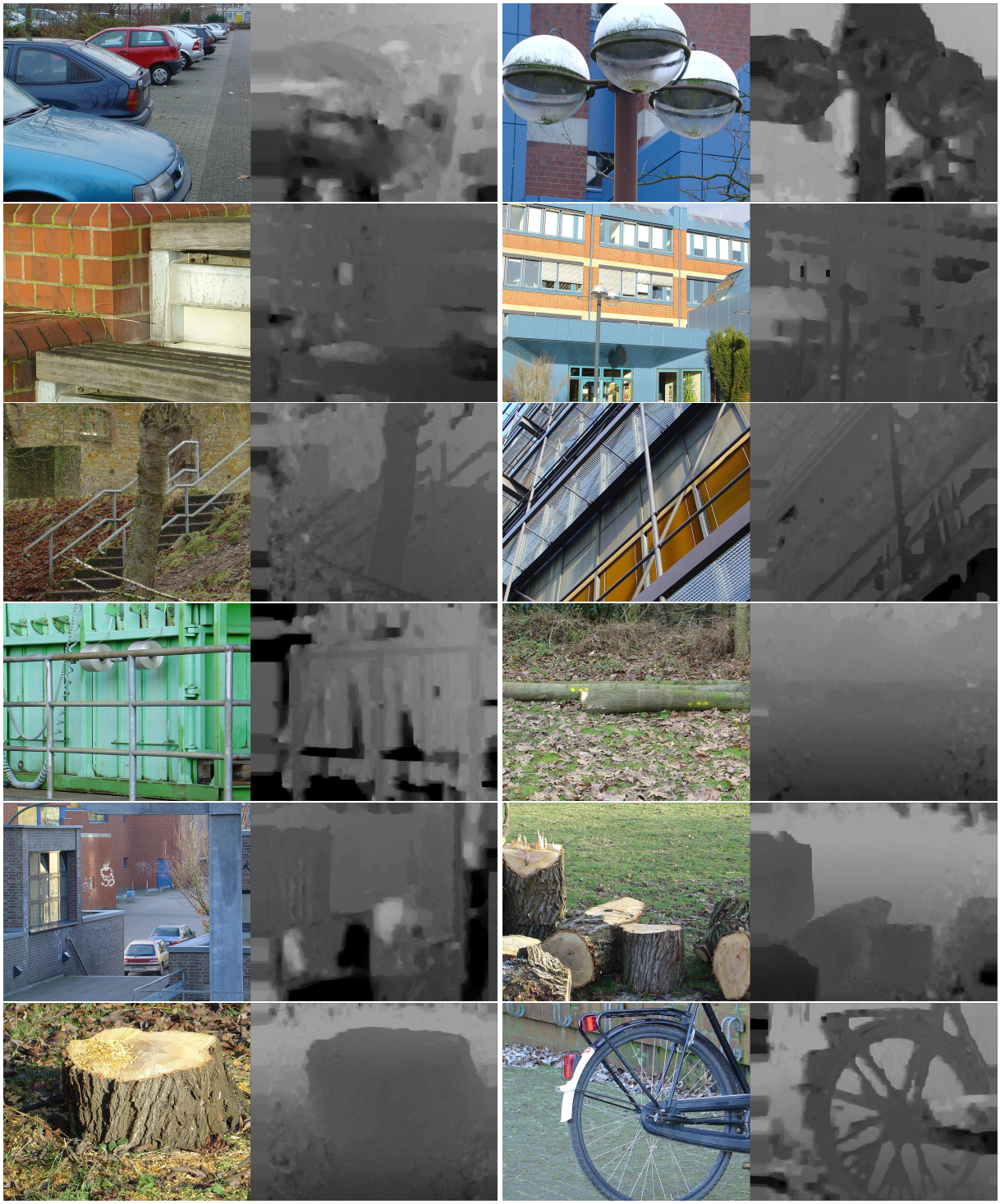
A.1. Uncalibrated data

Beginning on page 43, we present disparity maps of stereo images that were created using data from a previous experiment (see Jansen [2006] for details). As no calibration data was available, we used the rectification technique described in section 2.2 and the belief propagation algorithm described in section 2.3.2. The results were significantly better than those obtained by cross-correlation techniques used before.

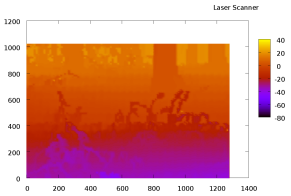
A.2. Calibrated data

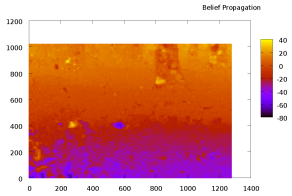
About half of the pairs of scans and images are shown from page 45 on. We also present disparity maps computed by belief propagation and normalized cross-correlation. The disparity maps generated from laser range scans carry the closest resemblance to the actual photographs. Their resolution and coherence is superior to both stereo matching algorithms. Belief propagation seems to come close to the quality of laser scans, although its results are more noisy and patchy. Normalized cross-correlation performs worse in almost all cases, yielding even noisier data. The correlation coefficients on page 49 clearly supprts this impression.
However, the laser scans also exhibit some typical errors: there are comb shaped edges diagonal to disparity gradients as well as bigger blob-shaped singularities that are due to single erroneously measured points with big depth difference to their neighbors.

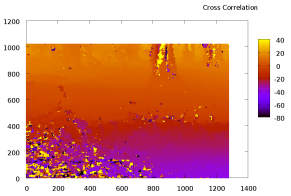




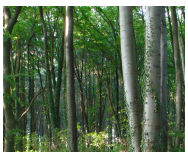
01

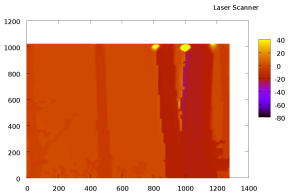


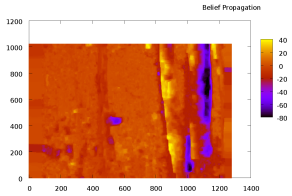


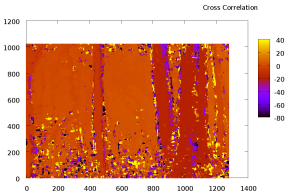


02

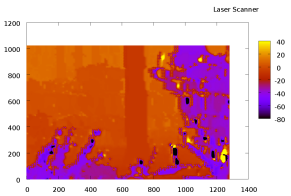


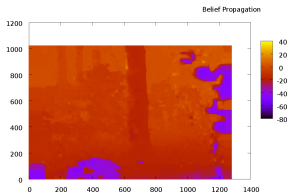


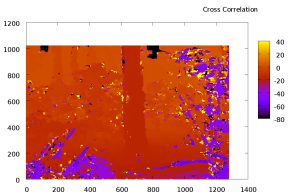




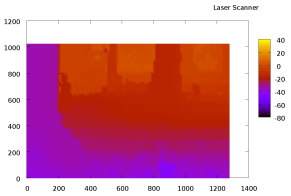
03

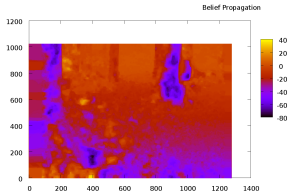


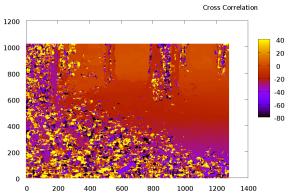




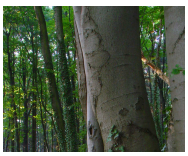
04

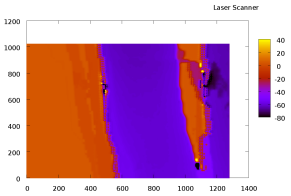


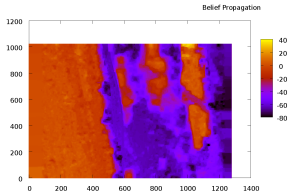


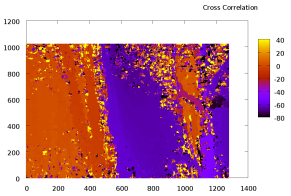


05

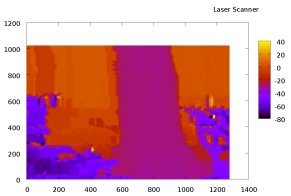


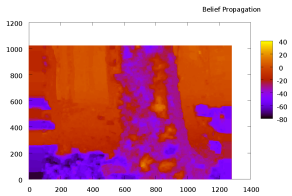


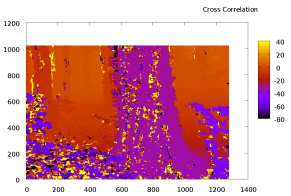




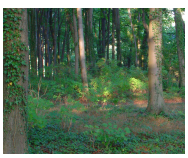
06

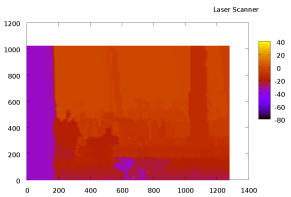


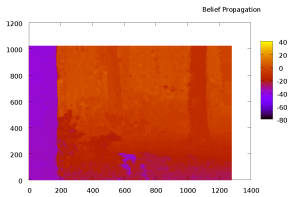


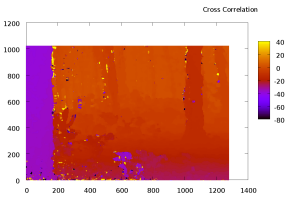


07

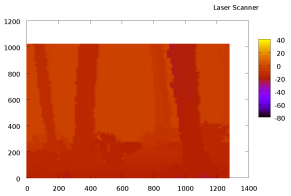


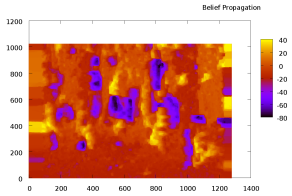


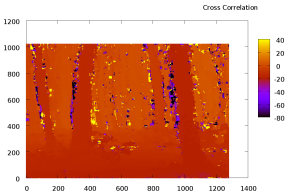




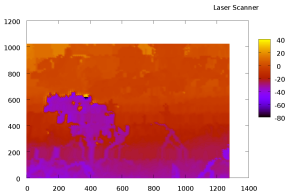
08

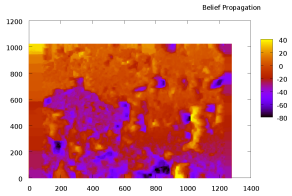


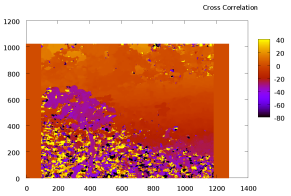




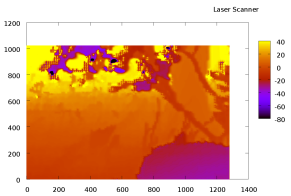
09

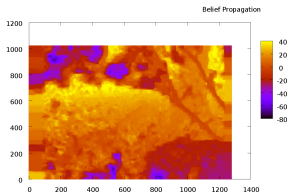


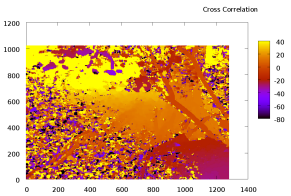




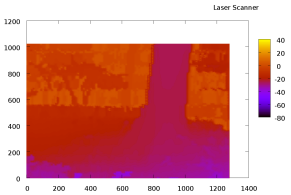
10

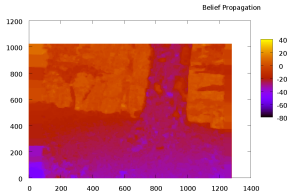


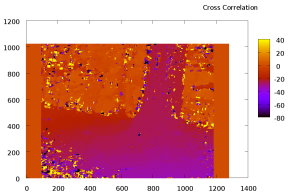




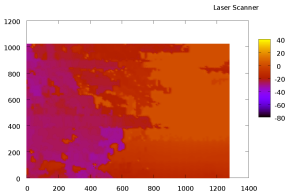
11

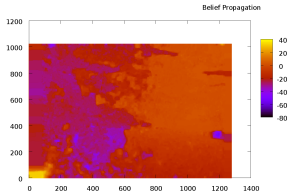


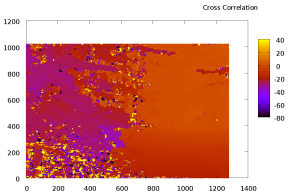




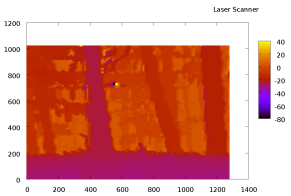
12

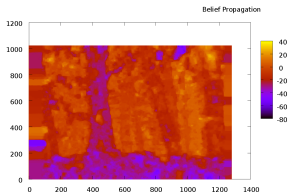


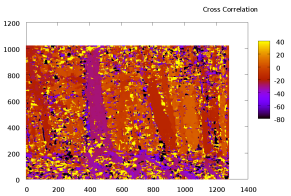




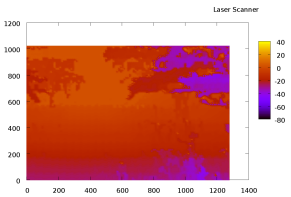
13

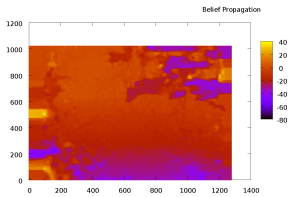


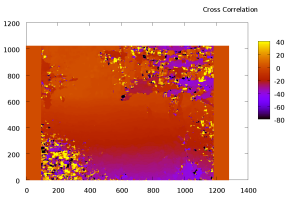




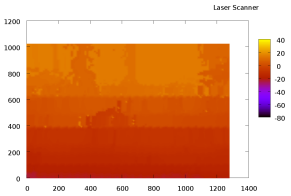
14

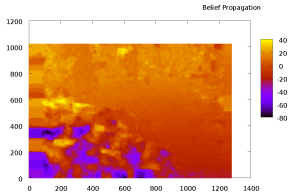


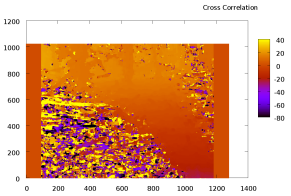




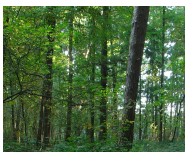
15

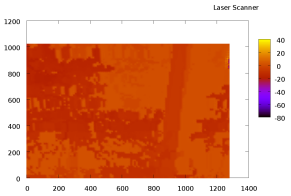


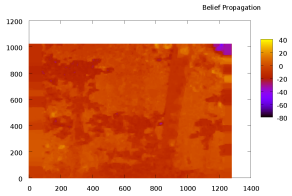


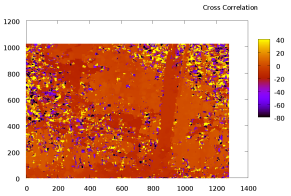


16

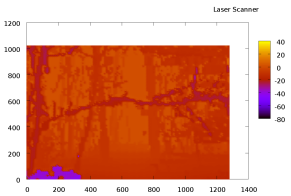


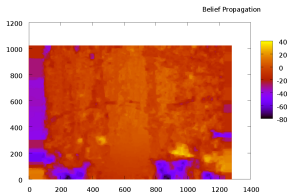


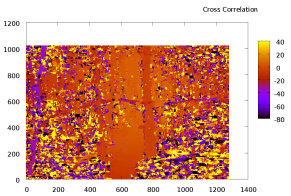




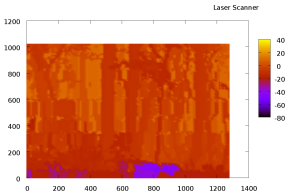
17

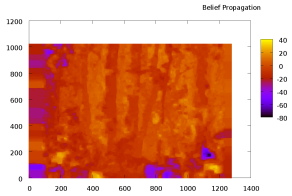


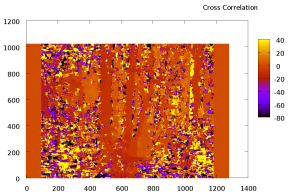




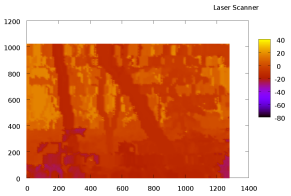
18

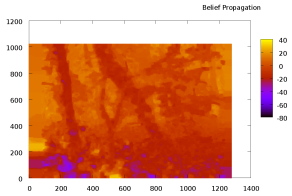


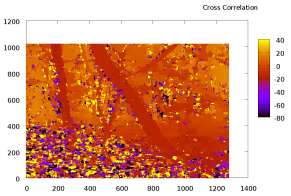




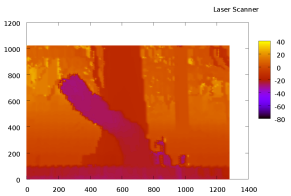
19

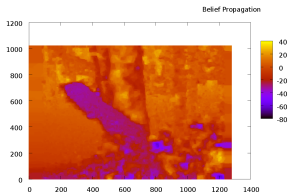


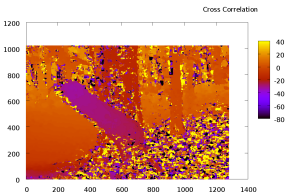




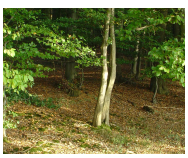
20

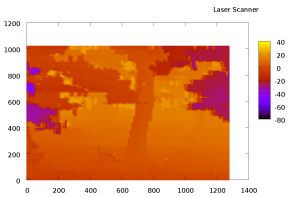


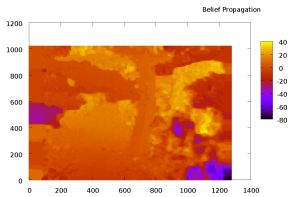


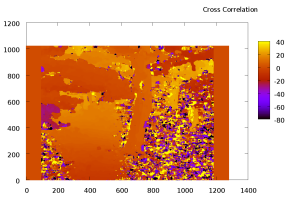


21

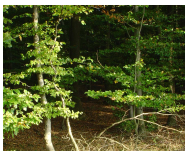


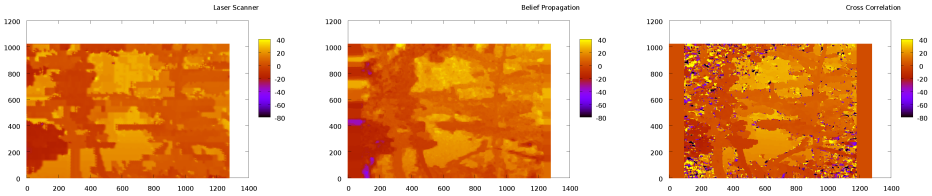




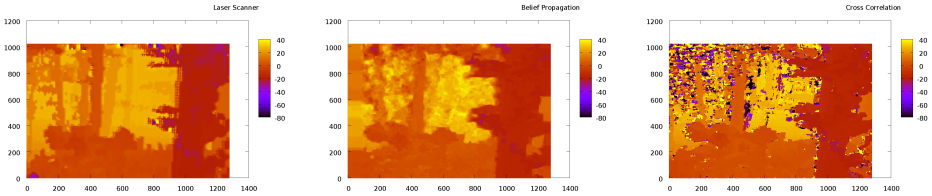


22

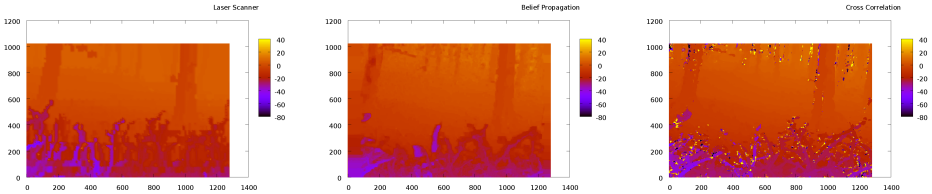




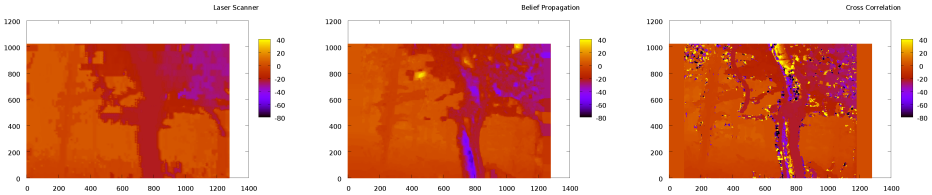
23



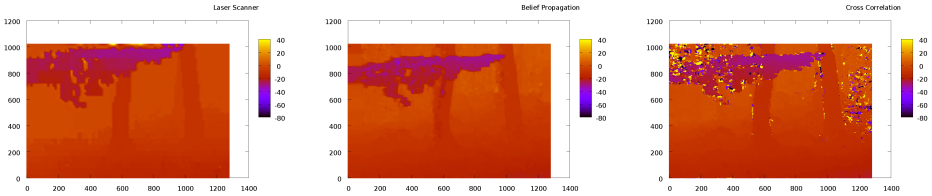
24



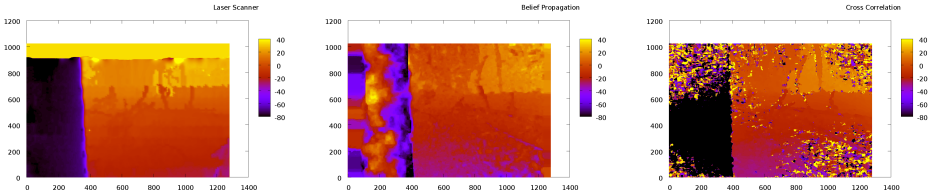
25



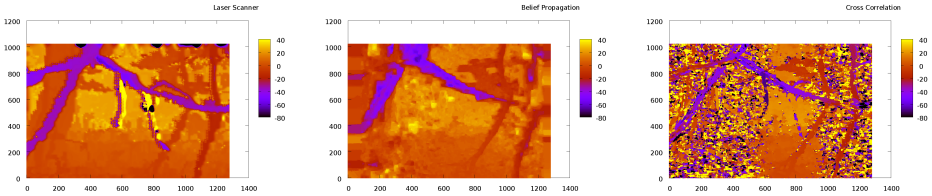
26



27



28



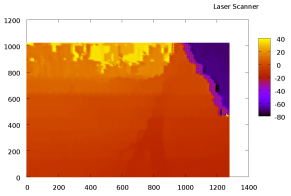
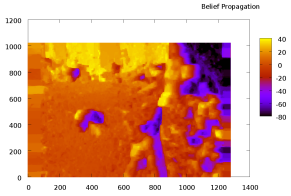
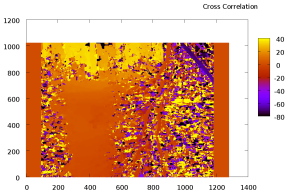
29				

Table A.1.: Images and Disparity maps created by laser scanning, belief propagation and cross-correlation. We show a representative subset of 29 images of 80 images total.

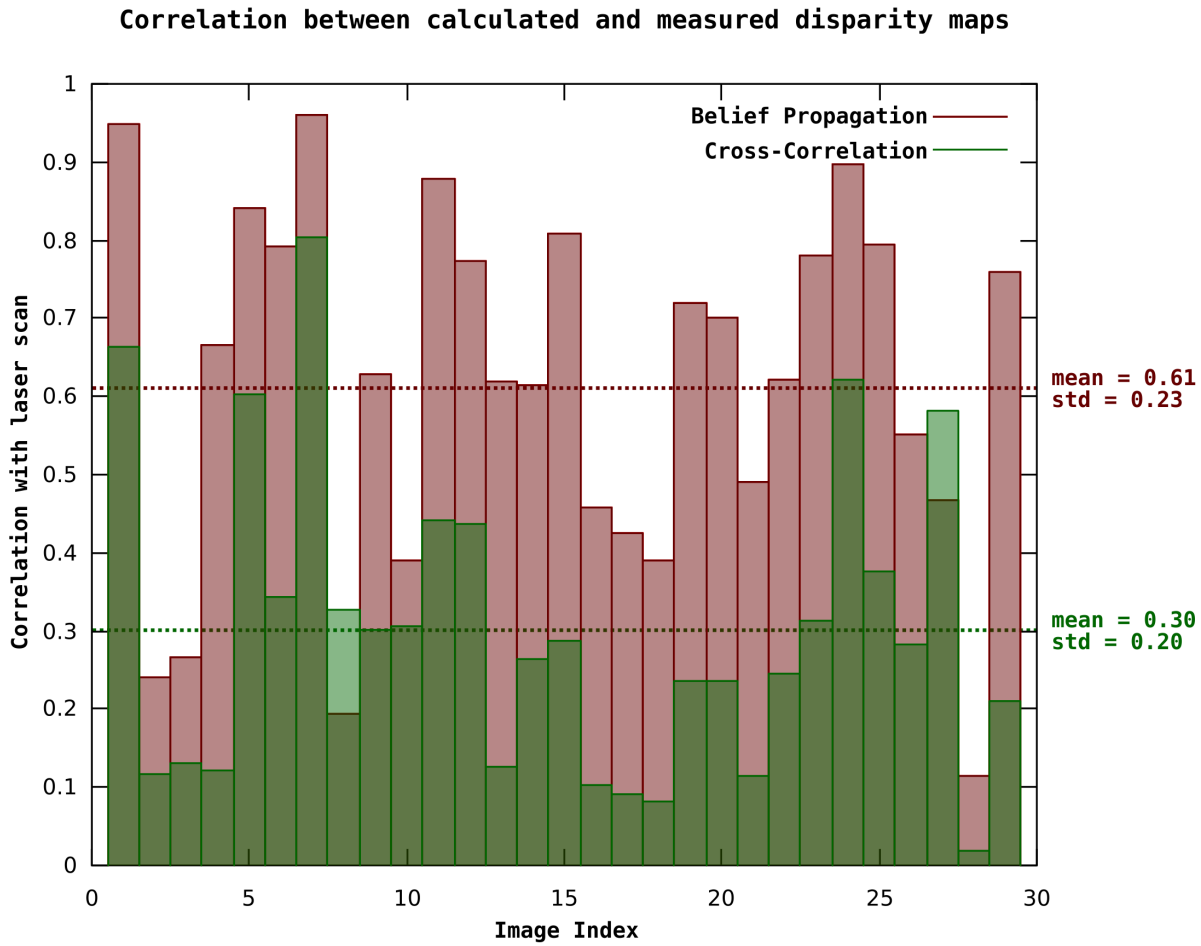


Figure A.1.: Correlation between disparity maps measured by laser scans and disparity maps generated by belief propagation and cross-correlation algorithms

B. Source Code

B.1. How to obtain

The development of the accompanying software was a collaborative effort, with writing, testing and application distributed among several systems as well as developers. In order to keep track of changes and to synchronize local versions, we made extensive use of a Version Control System. Specifically, we used Subversion [CollabNet] which is a modern, feature complete system which is easy to use. A short introduction for project members was created and is available at `http://coders.de/uos/bsc/doc/internal/subversion.txt?view=markup`. We refer to Collins-Sussman et al. [2006] for a more concise description of the system's capabilities and usage.
Besides the native clients offered by the Subversion Project itself, all features can also be accessed via a custom web interface located at `http://coders.de/uos/bsc/`. One may either browse the directory structure and inspect individual files' properties and contents by clicking on them, or download the whole structure as a standard tar archive by following the corresponding link.
Please note that the `src/scan3d` subdirectory is not publicly available. This is due to license constraints induced by using software copyrighted by Fraunhofer Institut für Autonome Intelligente Systeme. However, the free (GPL) reimplementation (c.f. 2.1.1.4) of the software is available in the `src/freescan` directory.

B.2. Structure

We will first outline the broader concepts employed during development in B.2.1. The following section B.2.2 will pick up some of the central software modules and finally we will list important functionality imported from external libraries.

B.2.1. General Organization

We try to make the software as platform independent as possible. A key to that lies in obeying the *B*ase Standard set by The Open Group [December 2001], which is a superset of the POSIX specifications. We thereby reach source-level compatibility with all current *NIX systems, including but not limited to Linux®, OpenBSD and Mac OS X®. The design follows well established Unix[1] principles by creating small yet generic programs with defined capabilities. Each of those programs can be used on it's own for a specific

[1] *Unix* here denotes the system in general and no particular vendor's implementation.

purpose. Alternatively, standard methods in shell scripting can be employed to plug those programs together, allowing for a huge variety of possible interactions among them.

B.2.2. Important Programs

Besides the short description here, every program features a rudimentary explanation of usage in its source code files. More complex programs like `disparity` or `stereo` come with full command line help, available by calling the program with the `--help` parameter.

B.2.2.1. scan3d.py

runs the graphical user interface of the scanning application. It is usually called by the script `detect.sh`, which detects the USB interfaces the hardware components are attached to.

B.2.2.2. disparity

is the core program of stimulus synthesis (c.f. 3): it projects raw laser data, interpolates disparity maps and creates artificial 3D images.

B.2.2.3. calibrate

processes point correspondences found in the stereo images by `corners` and gained with the laser scanner by `scan3d.py`. Depending on its input, it either calculates initial intrinsic matrices and lens distortion or refines them and additionally creates extrinsic matrices.

B.2.2.4. undistort

corrects the image by applying lens distortion coefficients.

B.2.2.5. crop

takes an image pair and pane coordinates of the left image to find the corresponding pane in the right image by maximizing correlation between both. This is a crucial preparation for later usage in psychophysical experiments [Jansen, 2006].

B.2.2.6. corr

implements different stereo matching algorithms: normalized cross-correlation from 2.3.1, fast cross-correlation and optical flow from the OpenCV library.

B.2.2.7. stereo

is the implementation of the belief propagation disparity algorithm from 2.3.2.

B.2.2.8. mkstimuli.sh

is the core script for batch-processing of acquired data. When executed with the `all` parameter, it sequentially executes the following operations on all input scenes:

crop as described in B.2.2.5

disparity as described in B.2.2.2

mean calculates the mean luminance and standard deviation over all images

normalize applies these values to all images

stripe generates images suitable for display on a 3D screen

xml generates machine-readable information about the images

graphics creates informational images such as histograms and disparity maps superimposed on camera images

B.2.3. Imported Library Functions

In the following, we list key functions imported from third party libraries. The description is purely semantic and does not include detailed calling conventions. However, we include a direct link to the source code repository for each function, which allows for investigation of the context it is used in.

FindCornerSubPix

Source Library	OpenCV
Input Arguments	Image with checkerboard, initial corners, search window size
Output Arguments	refined corners
Used in	corners.cpp

Refines the input corners on sub-pixel level by following the white/black gradient in an area of the given window size.

FindExtrinsicCameraParams2

Source Library	OpenCV
Input Arguments	Calibration points on the image plane and in world coordinates, intrinsic calibra
Output Arguments	extrinsic calibration matrix
Used in	calibrate.cpp

Iteratively calculates extrinsic camera parameters, minimizing projection-error.

cvMatchTemplate

Source Library	OpenCV
Input Arguments	Image 1, smaller Image 2
Output Arguments	Normalized correlation
Used in	crop.cpp

Computes the normalized cross-correlation on the luminance values of the supplied images.

cvUndistort2

Source Library	OpenCV
Input Arguments	Image, distortion coefficients
Output Arguments	Undistorted Image
Used in	undistort.cpp

Corrects a given image for lens distortion.

MpShepard2d

Source Library	Matpack
Input Arguments	Sparse 2D grid
Output Arguments	dense 2D grid
Used in	disparity.cpp

Interpolates a sparse 2D data set using a B-spline algorithm.

cvProjectPoints2

Source Library	OpenCV
Input Arguments	3D points, projection matrix
Output Arguments	2D projection of points
Used in	disparity.cpp

Projects given 3D points to the 2D image plane according to the supplied camera projection matrix.

cvGoodFeaturesToTrack

Source Library	OpenCV
Input Arguments	Image
Output Arguments	2D coordinates
Used in	epiLines.cpp

Finds corners of high discriminability.

cvFindFundamentalMat

Source Library	OpenCV
Input Arguments	2×2D point correspondences
Output Arguments	Fundamental Matrix
Used in	epiLines.cpp

Finds the Fundamental Matrix by sampling the point correspondences using RANSAC.

cvMakeScanlines

Source Library	OpenCV
Input Arguments	Image size, Fundamental Matrix
Output Arguments	Scan-runs
Used in	epiLines.cpp

Calculates scan-runs, i.e. coordinates of epilines that entirely cover the plane of an image of the given size.

cvPreWarpImage

Source Library	OpenCV
Input Arguments	Image, Scan-runs
Output Arguments	Image data
Used in	epiLines.cpp

Copies data from the source image along lines of one pixel height along the given scan-runs.

B.3. Usage

B.3.1. Dependencies

Dependencies on external software packages are listed in the *REQUIRES* file. We do not provide a generic installation routine for them, as their installation is typically dependent on the local distribution and platform in use. However, we list the corresponding Debian™package names, which also match packages in a broad range of other distributions.

B.3.2. Configuration

On systems with custom *include* directories, it might be necessary to change the corresponding paths for inclusion. This is easily done by editing the `PATH` variables in the various *Makefiles*.
Additionally, the global search path for *libraries* might need to be changed according to the installation location of third party libraries. Depending on the rights of the user, this might be achieved by:

- appending the corresponding paths to `/etc/ld.so.conf` and running `ldconfig`,

when root privileges are available

- exporting them to `LD_LIBRARY_PATH`, for ordinary users

B.3.3. Compiling

We provide a top-level *Makefile* that automatically builds all programs and libraries, and installs them into the `bin` directory. It is called by executing `make install` in the `src` subdirectory. It is also possible to selectively compile certain parts of the distribution, either by executing `make` in a subdirectory of the `src` branch, or by choosing a different *target*, i.e. `make disparity` for only compiling the disparity program.

C. GNU Free Documentation License

Preamble

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or non-commercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The **"Document"**, below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as **"you"**. You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A **"Modified Version"** of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A **"Secondary Section"** is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The **"Invariant Sections"** are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The **"Cover Texts"** are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A **"Transparent"** copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called **"Opaque"**.

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The **"Title Page"** means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

A section **"Entitled XYZ"** means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as **"Acknowledgements"**, **"Dedications"**, **"Endorsements"**, or **"History"**.) To **"Preserve the Title"** of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.

C. State on the Title page the name of the publisher of the Modified Version, as the publisher.

D. Preserve all the copyright notices of the Document.

E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.

H. Include an unaltered copy of this License.

I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.

N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.

O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties–for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

D. Affirmation

Hereby I confirm that I wrote this thesis independently and that I have not made use of any other resources or means than those indicated.

Hiermit bestätige ich, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.

Osnabrück, March 23, 2006

_____

Johannes M. Steger

Bibliography

The websites referred to below where last accessed on March 23, 2006. In case of unavailability at a later time, we recommend visiting the Internet Archive.

Berezin Stereo Photography Products, Abedul, Mission Viejo, CA, USA. Official website. http://www.berezin.com/.

Boris C. Bernhardt. How disparity fits into the statistics and the neural processing of natural scenes. Bachelor's thesis, Institute of Cognitive Science, University of Osnabrueck, 2006.

R. N. Bracewell. *The Fourier Transform and its Applications*. McGraw-Hill Series In Electrical Engineering. Circuits And Systems. McGraw Hill, New York, second edition, 1986.

D.C. Brown. Close-range camera calibration. *Photogrammetric Engineering*, 37:855–866, 1971.

J. Canny. A computational approach to edge detection. *IEEE T. Pattern. Anal.*, 8: 679–714, 1986.

CollabNet. Official subversion homepage. http://subversion.tigris.org/.

Ben Collins-Sussman, Brian W. Fitzpatrick, and C. Michael Pilato. *Version Control with Subversion*, 2006. http://svnbook.red-bean.com/.

P. J. Cowans. The sum-product and max-product algorithms for graph partitions. Draft technical note, Microsoft Research, 2004.

Deutsches Institut für Normung. *EN 60825-1 : 2001-11*, 2001.

R. O. Duda and E. H. Hart. Use of the hough transformation to detect lines and curves in pictures. *Commun. ACM*, 15(1):11–15, 1972.

Pedro F. Felzenszwalb and Daniel P. Huttenlocher. Efficient belief propagation for early vision. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 261–268, 2004.

James Foley, Andries van Dam, Steven K. Feiner, and John F. Hughes. *Computer Graphics. Principles and Practice. 2nd Edition in C.* Addison-Wesley, 1996. ISBN 0-201-84840-6.

David A. Forsyth and Jean Ponce. *Computer Vision: A Modern Approach*. Prentice Hall Professional Technical Reference, 2002.

Fraunhofer Institut für Autonome Intelligente Systeme. Official website. http://www.ais.fraunhofer.de.

J.G. Fryer and D.C. Brown. Lens distortion for close-range photogrammetry. *Photogrammetric Engineering and Remote Sensing*, 52:51–58, 1986.

Berndt M. Gammel. Matpack c++ numerics and graphics library release 1.8.1, 1991-2003. http://matpack.de.

Stuart Geman and Donald Geman. *Neurocomputing: foundations of research*, chapter Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images, pages 611–634. MIT Press, Cambridge, MA, USA, 1988.

D. Grest, J-M. Frahm, and R. Koch. A color similarity measure for robust shadow removal in real time. VMV 2003, Nov. 19 - 21, Munich, Germany, 2003.

R. M. Haralick, Sternberg S. R., and Zhuang X. Image analysis using mathematical morphology. *IEEE T. Pattern. Anal.*, 9(4):532–550, 1987.

C. J. Harris and M. Stephens. A combined corner and edge detector. In *Proceedings of the 4th Alvey Vision Conference, Manchester*, 1988.

R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, second edition, 2004.

Lina M. Jansen. Human eye movements in 3-dimensional natural scenes. Bachelor's thesis, Institute of Cognitive Science, University of Osnabrueck, 2006.

F. R. Kschischang, B. J. Frey, and H-A. Loeliger. Factor graphs and the sum-product algorithm. *IEEE T. Inform. Theory*, 47(2):498–519, 2001.

LANC Shepherd. Lanc shepherd product description. http://www.berezin.com/3d/Lanc/index.html.

J. P. Lewis. Fast normalized cross-correlation. Technical report, Industrial Light and Magic. http://www.idiom.com/~zilla/Work/nvisionInterface/nip.html.

C. Loop and Z. Zhang. Computing rectifying homographies for stereo vision. Technical Report MSR-TR-99-21, Microsoft Research, 1999.

Ken Martin, Will Schroeder, and Bill Lorensen. The visualization toolkit, 1993-2006. http://vtk.org.

R. Mohr and B. Triggs. Projective geometry for image analysis. Tutorial given at ISPRS, Vienna, July 1996.

Neurobiopsychology Lab, Univ. Osnabrück. Official website. http://www.cogsci.uos.de/~NBP.

S. Onat, C. Kayser, and P. König. On the time course of disparity in natural visual stimuli. To be published.

W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical recipes in C: the art of scientific computing*. Cambridge University Press, second edition, 1992.

D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *Int. J. Comput. Vision*, 47(1/2/3):7–42, April-June 2002. http://www.middlebury.edu/stereo.

O. Schreer. *Stereoanalyse und Bildsynthese*. Springer, 2005.

Sealevel Systems, Inc., Liberty, SC, USA. Official website. http://www.sealevel.com/.

R. Shapiro. Direct linear transformation method for three-dimensional cinematography. *Res. Quart.*, 49:197–205, 1978.

Sick AG, Waldkirch, Germany. *Telegramm Listing - Telegramme zur Bedienung / Konfiguration der Lasermesssysteme 2XX*. Sick AG, Waldkirch, Germany, 2003. http://www.sick.com/home/factory/downloads/downloads_auto_ident/de.downloadnewpar.0012.filePdf.tmp/TLLMS2xxD_8007953_04042003.pdf.

Sick AG, Waldkirch, Germany. Official website. http://www.sick.com.

Sony Corp., Tokyo, Japan. Official website. http://www.sony.net.

The Open Group. *IEEE Std. 1003.1-2001: Standard for Information Technology – Portable Operating System Interface (POSIX). Open Group Technical Standard: Base Specifications*, December 2001. http://www.opengroup.org/austin/.

VOLZ Servos GmbH & Co.KG, Offenbach am Main, Germany. Da 20-06-220. http://www.volz-servos.com/file/Industrial-pages0405_Ncm.pdf.

Wiretek, XiaoJieJiao Humen, GuangDong, China. Official website. http://www.wiretek.com.cn/.

Jonathan S. Yedidia, William T. Freeman, and Yair Weiss. *Exploring artificial intelligence in the new millennium*, chapter Understanding belief propagation and its generalizations, pages 239–269. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2003.

Z. Zhang. A flexible new technique for camera calibration. *IEEE T. Pattern. Anal.*, 22 (11):1330–1334, 2000.

List of Tables

List of Figures