Design and Test of Algorithms for the Evaluation of Modern Sensors in Close-Range Photogrammetry

A dissertation

submitted in partial fulfillment of the requirements for the degree of
Doctor of Philosophy in Computer Science

to the Faculty of Mathematics

of the Georg-August-University at Göttingen

by

Karsten Scheibe

born in Lauchhammer

Göttingen 2006

D7
referee: Prof. Dr. Robert Schaback
coreferee: Prof. Dr. Reinhard Klette
coreferee: Prof. Dr. Herbert Jahn
date of disputation:

Summary

The thesis describes a general approach for scanning and visualizing panoramic (360°) indoor scenes. It combines range data acquired by a laser rangefinder with color pictures captured by a rotating CCD line camera. The thesis defines coordinate systems of both sensors, specifies the fusion of range and color data acquired by both sensors, and reports about different alternatives for visualizing the generated three-dimensional data sets. Compared to earlier publications, this approach also utilizes an improved method for calculating the spatial (geometric) correspondence between laser diode of the laser rangefinder and the principal point of the rotating CCD line camera. Calibration is also a subject in this thesis. A least-square minimization-based approach is proposed for the rotating CCD line camera. The thesis also deals with particular problems occurring in the process of merging range data and color texture, such as achieving straight edges or planarity in the scene where supported by range or color data. Finally the thesis verifies the possibility to combine terrestrial panoramic images with airborne data. Altogether, the thesis verifies that the described multi sensor approach allows a digitization of indoor or outdoor scenes, illustrates important progress, but also indicates areas of future research.

Contents

List of Figures

List of Tables

1 Introduction

Photography contributed to entertainment from its beginning when the optic camera was invented in the early 19th century, see [Ger82]. However, engineers of many disciplines were also quick to realize the potential of this technology as a tool for both civilian and military applications. This was true for areas such as architecture, and more significantly in cartography. In particular, some pioneering photographers attempted to apply their art to the science of measuring physical spaces at an accuracy as supported by their latest tools. Of course, photography is in general more concerned with creating images that meet aesthetic and commercial demands than with capturing accurate images as required for scientific measurements.

1.1 Basic Terms

The *position* (of a camera, a Laser Rangefinder, and so forth) is defined by coordinates in a given coordinate system. Whereby position means the *principle point* of a device, and is given, for example, by the *projection center* (the main point of a lens in a single thin lens system) for a camera, or the main point of the laser diode for a Laser Rangefinder.

The *principle axis* of a lens is an imaginary line which path through the principle point and is perpendicular to the curved surface of the lens.

The *direction* is defined by angles and normally characterized by a vector (where we do not require in this thesis that a direction is a unit vector).

The *attitude* is defined by position and direction.[1] Note that “attitude” is already defined this way for airplanes or in airborne photogrammetry, and we simply extend this meaning also to sensors in general.

A *3D point* is a location in the Euclidean space $\mathbb{R}^3$. A *cloud of 3D points* is a finite set of such locations.

2.5D is the discrete representation of a partial 3D environment in a two dimensional grid. Each grid point has one value which is the “z-value” (a orthogonal distance of a 3D point to this grid).

1.2 Photogrammetry

Close-Range photogrammetry, as a method for recording and monitoring architecture, had its origins in the work of Albrecht Meydenbauer (as reported by [Sch71]), a German architect, who used the *graphic intersection method* as early as in 1867. A true pioneer in the use of photogrammetry, he is credited for the first photogrammetric recording of Islamic architecture in the Middle East in the 1870s.

Photogrammetry became the engineering field concerned with this task, and its results improved with developing technologies. *Photogrammetry* is a measurement technology in which the three-dimensional (3D) coordinates of points of an object are determined by measurements made in two or more photographic images taken from different positions or with different viewing directions. Standard references about photogrammetry are, for example, [AK89, Luh03, Kra97, GH01] and [LRKI06] as a recent book focused on close-range photogrammetry and its techniques and applications.

The past development of photogrammetry can be subdivided into four periods (see Figure 1.1). Each period is characterized by technologic or methodological innovations which made photogrammetry incrementally more flexible and more effective, see [AW96].

[1] Attitude is called “Orientierung” in German photogrammetry. Following mathematics, *orientation* specifies an angular order (e.g., clockwise, or to the left) in this thesis.

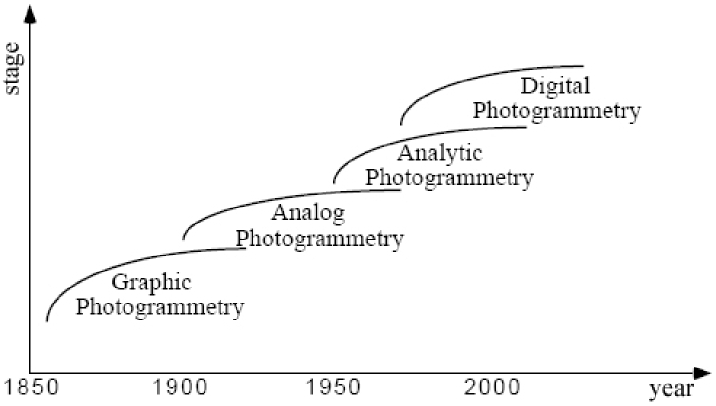


Figure 1.1: A rough sketch of periods in the development of photogrammetry.

1.3 Panoramic Photography

Based on calibration, modern digital cameras can be integrated with or into other technologies, such as further sensors, or helicopters, airplanes or satellites. Digital cameras already replaced analog cameras in astronomy or remote sensing (e.g., in space research), and more recently also (mostly) in aerial photography. The resulting availability of high-quality pictures has created a consumer expectation that maps should not only be detailed and accurate but also always up to date. In particular, pictures or maps should have a very high resolution.[2]

Increasingly more applications (e.g., travel industry, online maps, navigation systems, real estate, gaming industry) are taking advantage of these camera technologies.

Panoramic photography is one of the possible options for creating 3D (or pseudo 3D) visualizations. Panoramic images (*panoramas* for short) often provide very adequate representations for one attitude. This can be enhanced by allowing stereo-viewing (e.g., using anaglyphs), independent (possibly interactive) animations of 3D objects within a panoramic scene (e.g., just waving of leaves, or random drives of cars through a scene), or special effects such as illumination changes in a panoramic scene (e.g., a sunset). Re-

[2] The book [KR04] discusses in Chapter 1 different meanings of "resolution", and "high resolution" is here understood with respect to *geometric resolution* (i.e., the metric resolution, defined in physical scales by the implemented geometric projection). Of course, high geometric resolution typically also means a high screen or picture resolution (i.e., more pixels per inch on a screen, or the total size of a picture measured in rows and columns).

port [WHK98b] introduced a new panorama visualization approach, based on a general optimization of color filters for full-color anaglyph stereoscopic viewing. Panoramic and stereo image acquisition setups are incompatible intrinsically; the cited report proposed an error measure for anaglyphic panoramic images and provided a formula for controlling this error, allowing to adjust the camera positioning at the time of photo shooting.

Panoramic photography[3] is a style of photography that aims to create images with wide fields of view, but is sometimes also referring to photographs that are cropped to a relatively wide aspect ratio (such as 1:a, with $a \geq 2$). There is no generally accepted definition for the angle at which "wide-angle" or "panoramic" starts. As a rule of thumb, let us specify that "truly" panoramic images capture a field of view comparable to, or greater than, that of the human eye (i.e., more than $160°$ by $75°$), and should do so while maintaining to capture details across the entire picture.

A panoramic image is ... defined by whether the image gives the viewer the appearance of a "panorama", regardless of any arbitrary technical definition (see [WIK]).

Panoramic imaging is one of the standard options for visualizing 3D objects or spaces, with a dominating preference for cylindric panoramas. Many applications benefit from using such panoramas; see, for example areas such as stereoscopic visualization [HH98, PBE99, WHK98a, WHK99], stereo reconstruction [IYT92, Mur95, KS97, HWK99, SS99, HWK01, Hir05], walk-through or virtual reality [Che95, MB95, KD97, SS97, RB98, SH99], multimedia and teleconferencing [RB98, NMO97], localization, route planning or obstacle detection in robot-navigation or mobile vehicle contexts [YK90, Hon91, IYT92, ZT92, OHS99, Yag99], tracking and surveillance in 3D space [ISI97], and (certainly) many others.

Panoramic cameras[4] have been produced for at least 160 years (as of 2005), in various types and sizes. Most of the cameras produce an image of at least $110°$ wide, and many take a picture of $360°$. [McB] lists panoramic cameras, build since 1843.

Analog panoramic cameras are often based on the *spinning lens* idea, patented by Joseph Puchberger of Retz, Austria, in 1843. His camera had an 8-inch focal length lens and covered a 150 degree arc. Friedrich von Martens, a German living in Paris, made

[3] As defined on Wikipedia in March 2006.

[4] From [McB].

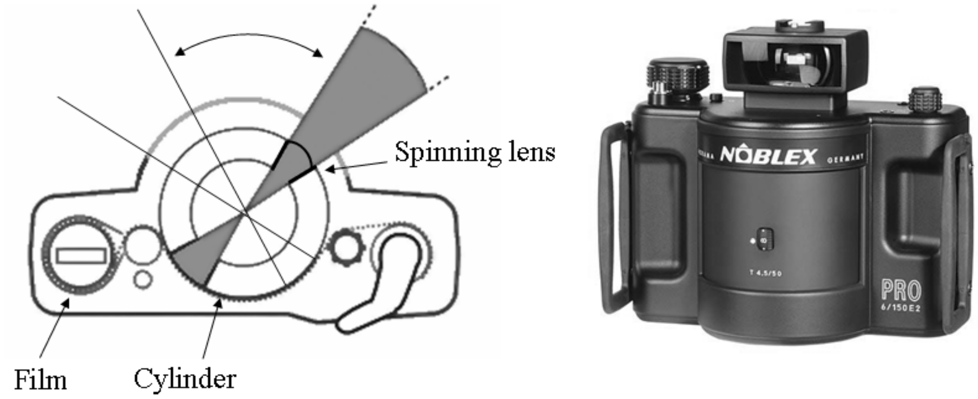


Figure 1.2: Left: Exposing a film, mounted on a cylinder, by a spinning lens (rotating slit). Right: Noblex camera, an example for this approach.

in 1844 the Megaskop camera, which featured a swing lens, operated by a handle and gears. Films of more recent spinning lens cameras are moved into a cylinder and exposed by a rotating lens (see Figure 1.2). In the field of panoramic analog photography, there are further ways for creating panoramas, for example with short focal lengths or fisheye lenses. The disadvantage of using such lenses are heavy image distortions. To avoid such distortions, special mechanical setups had been invented.

Since about 1994, the development of panoramic cameras started to utilize digital technology. A popular way is that a digital matrix camera (i.e., a *frame camera*) is placed on a tripod to take a few pictures while the camera is rotating, and then the pictures are “stitched” together by applying related software. Panoramic digital images became widely popular starting with [Che95] or [SK95], and stitching-based panorama generation is today a common technique for home, commercial, or professional photography.

A recent trend is also to offer 3D animated videos or interactive 3D models for 3D scene visualization. Often these are very simplified models, built by programs in computer aided design (CAD). For creating accurate models (e.g., as required by architects), either close-range photogrammetry or 3D object reconstruction (in computer vision, see [RK98]) are applied. The acquisition of 3D range data is crucial for creating 3D models, and different alternatives are provided in close-range photogrammetry or computer vision.

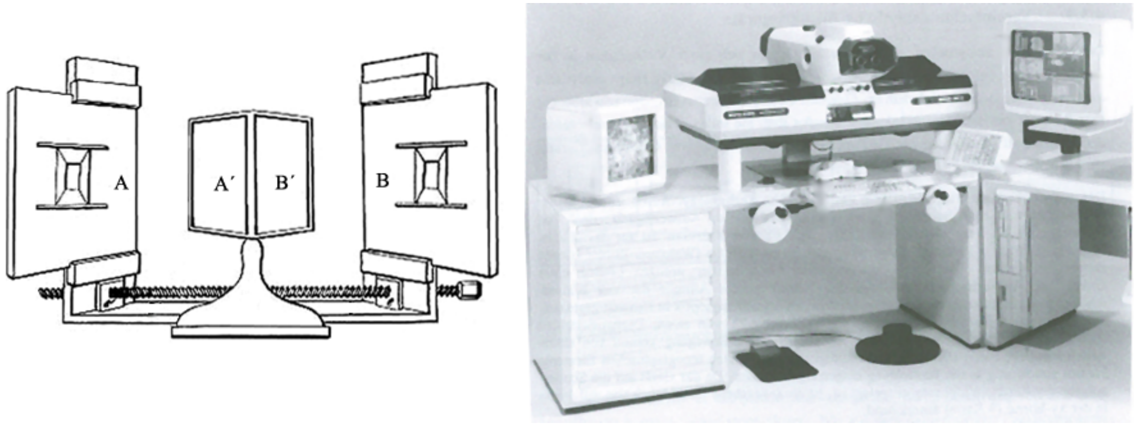


Figure 1.3: Left: Basic design of a stereoscope which uses angled mirrors (A′ and B′) to merge stereoscopic drawings (A and B) into a single stereo view. Right: a modern stereo comparator (Aviolyt BC3).

1.4 3D Scene Capture in Computer Vision

Modeling a 3D scene based on captured images, possibly including further sensors or interaction with the scene, can be achieved in different ways. The classical way in photogrammetry for obtaining 3D information is to evaluate binocular images by a human operator. The basic challenge of this method is to identify corresponding features in stereo pairs of images.

For many years, human operators used *stereoscopes* to identify all the corresponding points manually. A stereoscope is a measurement system which merges an image pair into a 3D view. A virtual marker is blended into this view. Images can be translated via wheels on the instrument, and when marker and object point coincide then corresponding image coordinates in both images of the stereo pair are given. The first powerful stereo comparator was invented by Carl Pulfrich in 1901 [Kra97]. A modern analytical evaluation instrument[5] is shown in Figure 1.3 (on the right); it is based on the stereoscope technology invented by Charles Wheatstone in 1833 (shown on the left in the figure).

Modern computer vision technology (i.e., automated stereo matching) allows partial automation; human interaction is still required to ensure desired 3D accuracy, but required time has been drastically reduced with digital technologies. *Stereo imaging* is one of the

[5] For an overview about today’s instruments, see [Pet90].

main subjects in computer vision [RK98, KR04], also considering cases without any a-priori knowledge (e.g., a model or descriptions of objects in a scene) or additional 3D data (e.g., from a laser rangefinder). Over the past decade, there has been a rapid development in the understanding and modeling of the geometry of multiple views in computer vision; see, for example, [Har92, Fau93, Kan93, LV94, SW95, SA96, AS97, Hey97, HZ00, FL01, DP04, DK06]. Most of this work was carried out based on projective geometry and the *pinhole camera model* (i.e., ideal central projection onto a rectangular image in the focal plane). Geometric studies of multiple views can be categorized according to the number of views: binocular as in [Har92, Fau93], trinocular as in [SW95, AS97], or polynocular (i.e., more than three views) as in [LV94, SA96].

A further, also widely studied approach for 3D object scene modeling, is *structured lighting*. The point-correspondence problem of stereo imaging is solved by using projected light patterns. Important variants of structured lighting are time-multiplexing (see [PA82] and [HK99]), spatial codification (see [ZCS02, G0̈1]), and direct codification (see [CH85]). The time-multiplexing technique has the highest resolution, allows for good accuracy, and it is easy to implement. The time-multiplexing technique is not able to measure dynamic (i.e., moving) objects. Spatial codification is able to measure dynamic objects, but has lower resolution and has also difficulties to deal with occlusion. Direct encoding is very sensitive to image noise, is not able to measure dynamic objects, but has also high resolution and allows for good accuracy.

Structure from motion (SfM) is one of the more recent approaches within computer vision, which aims on estimating 3D structure from 2D image sequences. 3D shape recovery by SfM (based on correspondences) follows the classical stereo approach, but is generalized to the use of uncalibrated cameras [Hua94]. Difficulties are not only finding corresponding points or features (e.g., corners or "landmarks" in a video sequence), but also robust estimations of fundamental or essential matrices. SfM can be stabilized by using a-priori knowledge about the scene; see [LKR05]. For overviews on related work, see [Koc94, Koc99].

A relatively new technique is structure from motion without correspondences, see [DSTT00]. This method recovers 3D scene structure and camera motion from multiple images without using correspondence information. The problem is formulated as finding the maximum likelihood 3D structure and motion, given only a few 2D measurements, by integrating over all possible assignments (i.e., of 3D features to available 2D measurements).

1.5 Laser Rangefinder Applications

Todays solutions towards 3D scene recovery often apply multiple tools, do not restrict itself to the use of cameras as the only sensor option, and merge theoretical approaches as well; see [KR05]. This thesis discusses 3D scene modeling based on using a laser rangefinder (LRF) together with digital panoramic cameras. LRFs have been used for close-range photogrammetry (e.g., acquisition of building geometries) for several years, see [Nie95, Wie01]. Today, LRFs are provided by several suppliers (e.g., Mensi, Cyra, Riegl or Leica), which offer systems with different parameters. Basically, there are two different kinds of scanning techniques, the frequency-to-distance converter technique (called *phase shift*), and the pulsed converter technique (called *time-of-flight*). A phase-shift LRF has sub-millimeter accuracies for sensor-to-surface distances in a range of less than one meter and up to 15 meters, and accuracies of 3 to 10 mm for distances of less than 50 meters.

It becomes more and more common to use panoramic LRFs. The typical *field of view* (FOV) of such a LRF is $360°$ horizontally and $180°$ vertically. The vertical FOV may be restricted because the physical appearance of the device itself limits the FOV. A panoramic LRF allows that a scene is 2.5D documented by a single scan (i.e., of course, only for all surfaces which are visible from the LRF's attitude). The rays of a LRF scan, hits surface points which are documented by 3D coordinates, that means such a scan creates a cloud of 3D points on visible surfaces. Such a cloud of 3D points is a discrete representation of 2.5D surfaces, being part of a 3D scene.

A LRF also captures intensity values which can be shown as gray-level images. However, applications related to 3D documentations of buildings, art, statues, and so forth require true-color surface textures. Some LRF systems also have the option of capturing color 3D data by mounting a camera near the LRF laser diode, or by complex mechanical mirror systems. In such cases it is common to use a "normal" CCD frame camera (i.e., of standard resolution).

1.6 Recent Camera Technologies

The development of digital cameras is very fast, and illustrated in Figure 1.4. The resolution of a single picture, captured by such camera, increases recently every month. Low-cost cameras already have a resolution of 4 to 6 Megapixel, and today's professional cameras have up to 20 Megapixel. The FOV of such cameras depends on the used lens (i.e., their focal length) and sensor size. Typically we have a FOV of between $15°$-$50°$ vertically, and $20°$-$65°$ horizontally.

Omnidirectional cameras (e.g., *catadioptrical* panoramic systems) are often used for capturing $360°$ panoramas of dynamic scenes. Catadioptric sensors are imaging sensors built by combining hyperbolic or parabolic mirrors (i.e., catadioptrics) and lenses (i.e., dioptrics). The family of catadioptric panoramas [ZT92, GNT98, BN99, Svo99] provides real-time and highly portable imaging capabilities at affordable cost. Applications include robot navigation, tele-operation, and 3D scene reconstructions [DK06]. [YNY95] used a conic-shaped mirror for mobile vehicle navigation. [Tho03] proposed a panoramic camera with a spherical mirror for navigating a mobile vehicle. [YYY95] detected obstacles using a panoramic sensor with a hyperbolic mirror. [ZT92] analyzed features of panoramic images and proposed applications for mobile robot navigation.

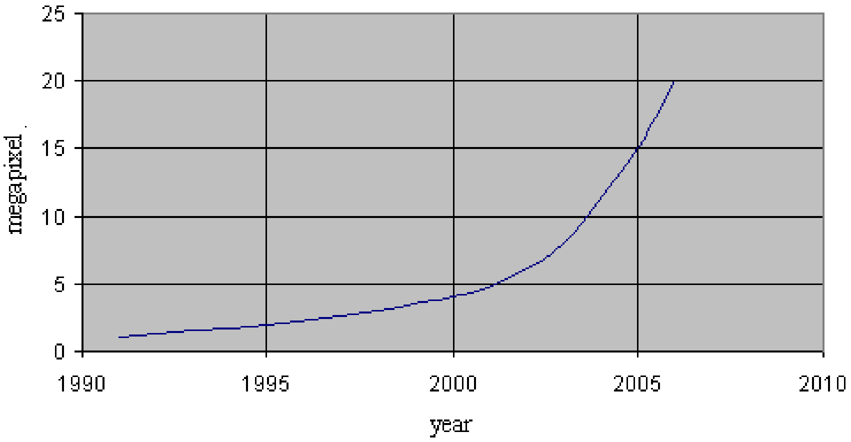


Figure 1.4: Development of "normal" CCD frame cameras dates back to 1991: The first digital camera (DSC-100) was introduced by Fujifilm.

Generally, a mirror is mapping a $360°$ view into a CCD frame, causing varying geometric resolution at different pixel positions. Geometrical dependencies are shown in a recent publication in [GD01]). The first catadioptrical system was invented by A. S. Wolcott in 1840, see [Hic]. Major drawbacks of these approaches include low resolution near the image's center, non-uniform spatial sampling, inefficient usage of images (i.e., there is a self-occluded or mirror-occluded area in each captured image), and severe distortions and image blurring due to aberrations caused by coma, astigmatism, field curvature and chromatic aberration. These drawbacks suggest that the catadioptric panoramas are not suitable for applications of recognition or inspection type (but very useful for robot navigation).

Rotating line cameras are another option for capturing panoramas; they originated from line cameras designed for aerial photography based on the *push-broom principle*[6] (i.e., one or several CCD lines translate or "fly" across a scene) were developed to acquire high resolution aerial images. Since 1995, line cameras are also used for capturing $360°$ panoramas; now instead of translating such a camera is mounted on a tripod and rotates (what was basically "kind of an unexpected use" of this technology).

The use of CCD line sensors was also introduced in 1982 as "dynamic photogrammetry" in remote sensing [HNE82]. Today's high-end parameters of CCD-line based airborne systems are defined by the *High Resolution Stereo Camera* (HRSC-AX),[7] see [WSG00, SGW02, SG04], and Leica's *Airborne Digital Scanner* (ADS40), see [BKM$^+$00, Eck02, ER01].

Arising problems of line cameras are related to the photogrammetric interpretation of captured aerial or panoramic images, namely geometric rectification (alignment of independently captured image lines) and motion blur or other problems related to dynamic scenes.

[6] The push-broom principle was invented when designing the first optomechanical scanners, see [B8̈5]. Here, a mirror rotates from left to right (”sweeping like a broom”) to acquire an image. Later on, this principe was applied to larger CCD lines in airborne sensors, and also called push-broom.

[7] AX is the acronym for the airborne version of the original HRSC, which was developed for the Mars mission in 1996.

1.7 Combining LRF Data and Panoramic Images

This thesis introduces an approach for data fusion of two independent sensor systems, namely an LRF and a panoramic camera; the latter one is based on the push-broom principle. The thesis also informs about contributions to the developed camera. Experimental results are regarded in the context of a *Neuschwanstein project* (by the company Illustrated Architecture), which is under way since 2002, and which aims at a complete 3D photogrammetric documentation of this Bavarian castle. Main intentions are the archiving of data for cases of restoration or redevelopment. The project is supervised by the Bavarian Construction Department, by order of the Free State of Bavaria, and the project volume is in the order of about 500,000 Euro.

As an alternative, the thesis also discusses the fusion of panoramic images with 3D data of the HRSC-AX. Here, parts of an airborne HRSC-AX scan are rendered by terrestrial panoramic images to improve the texture's resolution.

The fusion of range data and pictures, see [RSS01, SPS02, KS05], is a relatively new approach for 3D scene rendering. [Ker01] discusses the combination of range data with images acquired by a video camera. Combinations of panoramic images and LRF data (see [BK01]) provide a new technology for high-resolution 3D documentation and visualization. Applying two different *sensors* (i.e., LRF or camera, both characterized by very high resolution and accuracy) has drawbacks as well as advantages. Drawbacks are obvious (the need of unifying data captured at different positions and different sensors). One advantage is that both sensors can be dealt with separately, allowing to adjust parameters according to one goal only; either to collect range data or to collect texture data. The independence of the camera has also the advantage to use 3D data from alternative sensors (e.g., by using structured light, or one of the techniques briefly reviewed in Section 1.4). The collection of 3D data with alternative methods is necessary if an application of the LRF is not satisfactorily (e.g., specular surface materials). We illustrate such surface problems later by examples obtained within the Neuschwanstein project (e.g., stairs made out of marble). The following approaches briefly introduce common procedures to combine data captured by an LRF and a camera.

1.7.1 Ideal Correspondence of Sensor Attitudes

When using a panoramic camera and a LRF, then an ideal situation would be that both *principal points* (i.e., the origins of measurement rays) of the LRF and the camera are at the same position, and that both rotation axes and their directions coincide (i.e., both sensors have identical attitude). Under such ideal conditions, processing of the data would be straightforward, and we could apply rendering algorithms that work in (or nearly in) real time. Intensity data of the LRF can simply be replaced by color information of the camera. The result is a *colored cloud of 3D points* in world coordinates, where each 3D point in this set is labeled by a color (i.e., RGB) value.

Experiments have shown that it is possible in principle to implement this approach with minor errors. Both devices were located, one after the other, on the same tripod. To assure that both devices have the same attitude, we measured the principal points of both devices with a telescope. This approach works well, and a colored cloud of 3D points can be calculated. However, the approach is time-consuming and requires an ideal coordination of laser and panoramic scans which is in general not feasible in practice. More important, this approach is suitable for single-point of view situations, but would only have minor advantages in general multi-point of view (i.e., multiple scan) scenarios. The merging problem of multiple scans remains.

1.7.2 Different Attitudes of Capturing Devices

In this thesis, the panoramic camera is considered to be an independent device. The main argument is that this allows to use different types of 3D data (e.g., also HRSC-AX data). Secondly, our experience shows that scenes require multiple scans, and it is not always possible to scan a scene with the panoramic camera from the same position as the LRF. The *depth of field* is also not always compatible (for an LRF it might be limited, for example, to a distance between 1m and 25m).

Orthophotos

Orthophotos are pictorial representations of orthogonal projections of textured surfaces into *orthoplanes* (i.e., selected planes in $\mathbb{R}^3$ which are "behind" the object of interest, allowing orthogonal projections of all those surface points by different scans into such a plane which are visible from the plane along those orthogonal projection rays). High-accuracy orthophotos are a common way of documenting existing architecture. In architecture or CAD it is common that results are mapped into several orthoplanes, which can then be transformed into a unified (simple) 3D model; see Figure 1.5. Note that such a simplified 3D model (e.g., obtained by 3D surface reconstruction techniques in computer vision, or by combining a few LRF scans) cannot represent a "complex" 3D scene properly, and required reconstruction accuracy depends on numbers of laser scans, panoramas, and number and attitude of the selected orthoplanes.

The fusion of range data and panoramic images acquired by a rotating CCD line camera (which we sometimes call *camera* for short in this thesis) has been discussed in [HWK$^+$02b, KGW$^+$03]. The calibration of range sensors [HWK$^+$02b] and of rotating CCD line cameras [HWK02a] provides necessary parameters for data fusion. We demonstrated in [RSS01] how to fuse LRF data with pictures, which produce a *colored surface texture*. This approach was based on two single scans (i.e., one camera position and one LRF position) and a simplified camera model.

Note that range data, acquired at one LRF position, provide 2.5D surface data only, and a full 3D surface acquisition (if possible at all) can only be obtained by merging data acquired at several LRF positions. More than one single camera position is required as well. Therefore we use a cloud of points in 3D space (as given by several LRF scans) and also a set of panoramas (i.e., color panoramas, captured by the camera during 360° scans). This approach, addressed for the first time within this PhD project, integrates multi-position data acquisitions using laser range scanners and rotating line cameras. This also includes a special intention of mapping LRF and image data into specified orthoplanes.

For both simplified approaches (ideal correspondence, or a set of orthophotos), no complicated *ray tracing*[8] or volume rendering procedures are needed (i.e., an absence

[8] In general ray tracing follows the path of a light ray and estimates on each surface contact the absorption, reflection and refraction, follows the new arisen ray until the eyepoint is reached. In 3D computer

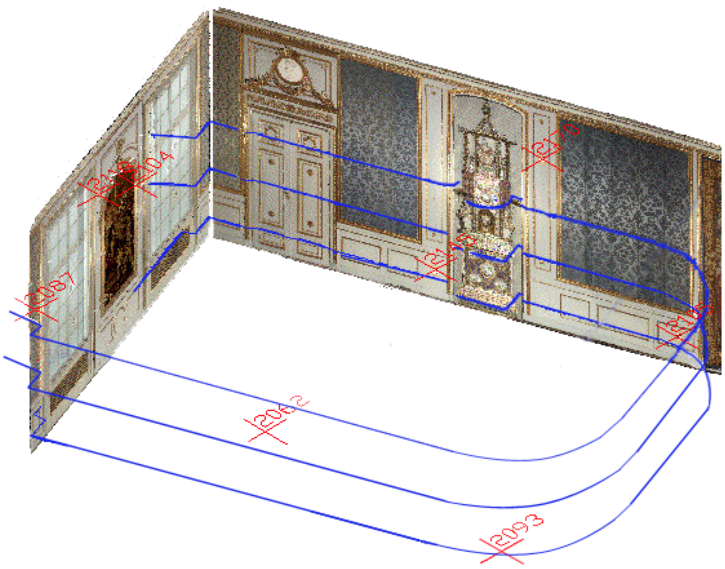


Figure 1.5: A (simple) 3D CAD model consisting of two orthoplanes.

of occluding objects between LRF or camera and the required orthoplane is assumed). For the second approach, ray tracing (i.e., from camera into the scene) cannot assume ideal correspondences between 3D points, defined by LRF data, and captured surface texture, but the search space is limited by the minimum and maximum depth value of the 2.5D map, as long as there is no obstacle between orthoplane and camera. Optimizations techniques (e.g., early ray termination, empty space skipping, depth buffer overwriting) accelerate this rendering process.

Volume Rendering

This thesis is about the use of multiple scans for 3D modeling of “complex” 3D scenes. Necessary volume rendering is here based on ray tracing in geometrically complex 3D scenes. Figure 1.6 illustrates the visibility problem in a simplified scheme. Object 1 caused a shadow. Therefore, Object 2 is not visible from the LRF position. A mapping routine possibly maps a wrong texture (i.e., pixel value) onto Object 3 (e.g., if the shown *camera ray* is used for rendering). Such situations already occur if LRF and camera are just in “slightly different” positions.

graphic it is often used as a technique which follows rays from the eyepoint outward, rather than originating at the light sources. One goal is the calculation of visible surfaces.

Our initial experiments for colorizing “real” 3D clouds of points were designed along the following steps:

(i) We calibrate attitudes of the LRF and the camera within one world coordinate system for the assumed space $\mathbb{R}^3$. Requirements for calibration accuracy are defined by the desired geometric resolution in the 3D scene space. Then all LRF scans are merged into a regular volumetric grid. The grid size is also defined by the desired geometric resolution.

(ii) A *viewing direction* is a ray emerging from a principle point to infinity. There are viewing directions for LRF and camera. For the LRF, each viewing direction and the measured distance determine one *3D sample* (or *3D point*). For the camera, each viewing direction identifies one color value at one pixel of the panoramic image (i.e., the ray, described by camera position and the pixel, “hits” a surface which has this color value at that 3D point). Due to the non-linearity of the sensors, the calculation of viewing directions requires a combination of affine and non-linear transforms. Their specification is supported by the use of *control points* (or *calibration marks*) and standard photogrammetry software (e.g., Neptan, Pictran, Australis), and also uses the calibration results of Step (i).

(iii) For any viewing direction of the camera, we select a point in the given grid of 3D points which is closest to this ray (defined by one pixel of the camera), and assign the color value of this pixel to this 3D point. This finally leads to a (partially) *colored cloud of 3D points*.

We implemented the selection of closest 3D points using *backward ray tracing*. This means that at first, the viewing direction (which is indexed by image coordinates i and j of the chosen pixel) is calculated, and then this camera ray is checked for occluding objects. *Forward ray tracing* tries to hit an object for each given camera ray, and this would be an alternative variant. With respect to our data, backward ray tracing was faster in experiments because of the given quantitative relation between 3D samples and numbers of pixels in a panorama (the rotating line camera allows to generate much more pixels compared to 3D samples of the LRF).

An *octree* data structure can be used for an efficient implementation of this ray tracing approach. An octree data structure splits a cuboidal subset of the 3D world in differ-

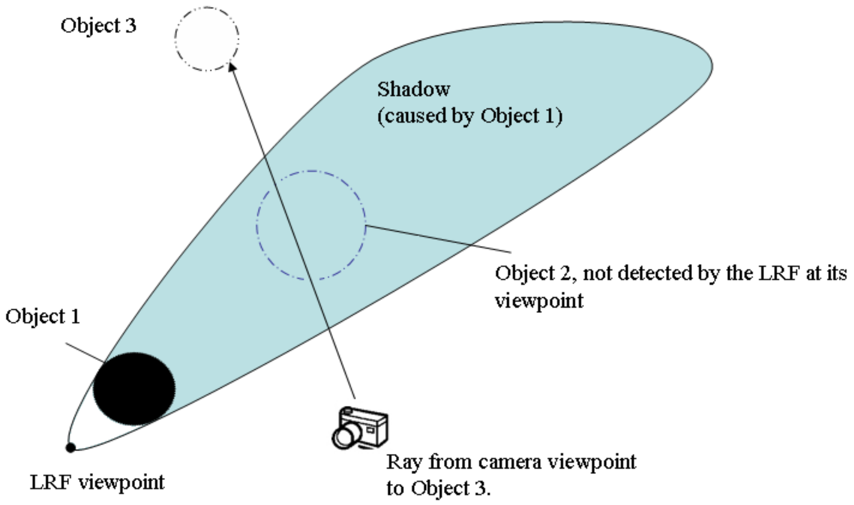

Object 3
Shadow
(caused by Object 1)
Object 2, not detected by the LRF at its viewpoint
Object 1
LRF viewpoint
Ray from camera viewpoint
to Object 3.

Figure 1.6: Ray tracing problem when combining one LRF scan with data from one camera position.

ent cubes, and addresses these cubes in form of a tree data structure. Each node of the tree represents a cuboidal volume. Each node has eight children. All eight child nodes partition the same volume of space as represented by the parent node. This data structure allows to minimize necessary ray tracing checks with respect to available 3D points calculated based on all LRF attitudes.

However, this ray tracing approach often colorizes available 3D points incorrectly, because it is common that we have less LRF 3D points for these ray tracing checks than camera data. Due to sparse LRF points we may miss a solid surface by tracing "in-between" of those points. The density of 3D points depends on the given resolution of the LRF, scanning distance, and on effects of filter processes. If we choose a larger search radius surrounding a voxel in the Euclidian space (i.e., choosing a bigger volumetric grid size) then we create problems at edges in the LRF data, or for fine detailed structures of objects in 3D scene. In fact, the calculated ray from the camera principle point to an object may just miss a closer object (which actually contributed its color value), because of "object density" or thickness of the closer object. Figure 1.6 illustrates this problem. A camera ray travels through the 3D points captured for this surface of Object 2 even if this is visible. No 3D point of Object 2 is on the shown camera ray.

As long as we do not have *triangulated 3D points* it is difficult to do ray tracing properly. Triangulation is the common way of connecting 3D points into a polygonal *3D* *surface mesh* (also called *wireframe*). This finally defines the approach which will be detailed in this thesis. Instead of fusing a single LRF scan with color information, followed by merging several colored LRF scans into a single 3D model, we prefer that all LRF scans are merged first into one unified depth representation of the 3D scene (again a cloud of 3D points, followed by triangulation, simplification of triangulation, etc.), and then all camera data are used for coloration of this unified depth representation by using a ray tracing routine. The unified textured 3D representation (in form of an binary file or in a standard format *Virtual Reality Modeling Language*) can then be followed by 3D visualization, animations, or orthophoto generations. All these data are of extremely large size due to given sensor resolutions. Several standard algorithms had to be redesigned with regard to the data volume. As an example, a single panorama is about 1,200 Megapixel, and a computer screen can only show a very small “keyhole” section at a time at full resolution. A special viewer was developed within this PhD project to load and analyze LRF or camera data, which can be more than 10 Gigabyte for a single scan.

1.8 Organization of this Thesis

Chapter 2 is defining some assumptions, and introduce the used notations in this thesis.

Chapter 3 deals with the developed camera for archiving high resolution texture data. The author’s part (within this PhD project) was the development of the software for the electronic components, driver development for the digital communication interface, grabbing of the camera data and its visualization. Only subject relevant work as the photogrammetric interpretation of these data are specified. The data preprocessing, basic transformation, the camera coordinate system and the formalized sensor model are explained in this chapter. A least-square minimization approach will be presented, which is a new method for the calibration of a rotating CCD line camera; it also allows to estimate the parameters of exterior and interior attitude of the camera. Calibration results allow then a specification of coordinate transformations for data fusion.

Chapter 4 briefly reviews advantages and drawbacks of available LRFs. LRFs are not main topic of this thesis, but the calibration process, error analysis and the coordinate system of the used LRF are explained more detailed in this chapter.

Chapter 5 starts with a short introduction in 3D visualization, continues with a discussion of possible ways of visualizations (i.e., data projections). Possible applications are the generation of orthophotos, interactive 3D animations (e.g., for virtual tours), and so forth. The chapter explains the 3D data pre-processing concerning our data and deals with the subject meshing, the basis for 3D visualization.

Chapter 6 shows the optimizing of LRF data and its resulting mesh. Of course, here many literature about this subject is available, but the developed filters are focused on appearing effects in 3D point clouds specially caused by data capturing with LRFs. Advanced approaches for general point clouds and its problematic nature (e.g, holes and connectivity problems) are shown in this chapter. We also deal with particular problems occurring in the process of merging range data and color texture, such as achieving straight edges or planarity in the scene where supported by range and color data. We leave surface patches of superlinear shape complexity (e.g., starting with a cylinder modeling a column) to future work.

Chapter 7 formalizes the relation between object coordinates and image coordinates based on the sensor model. The calculation of the requested image coordinates by given object coordinates is the basis for each data fusion. This chapter embraced the data fusion of our panoramic data with the meshed 3D model. In this texture mapping process a ray tracing routine is explained. Secondly, as an example of data fusion, the processing of high resolution orthophotos is shown. Here each pixel is mapped separately to a given surface model. As an alternative the fusion of our panoramic images with achieved 3D data of stereo processed airborne data will discussed at the end of this chapter.

Chapter 8 concludes this thesis, summarizes and defines future work.

2 Definitions and Assumptions

Definition of Image Coordinates

This thesis discusses imaging systems. Image coordinates address pixels in an image positioned on a two-dimensional (2D) manifold, where (in general) i is the *column*, and j the *row* in the image coordinate system. The image size $W \times H$ is given by the physical count of pixels on the $W \times H$ CCD matrix sensor, or on H-elements of a CCD line sensor, where W is given by the number of captured lines. Depending on manifold geometry, coordinates can be Cartesian, cylindric, spheric, and so forth.

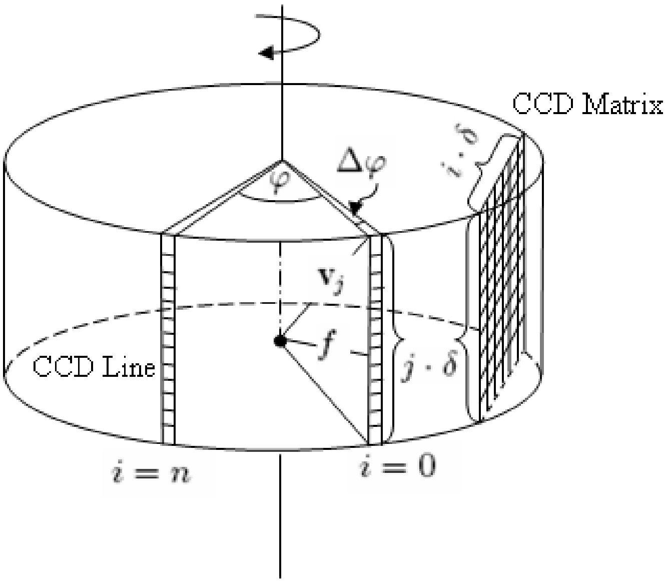


Figure 2.1: Image coordinates with respect to CCD line sensor or CCD matrix sensor.

In physical terms, the coordinates represent metric values, such as $j \cdot \delta$, where δ is a particular metric unit (e.g., $\delta = 0.007mm$, the pixel size for each element of the used Kodak CCD line sensor), or an absolute angle $\varphi = i \cdot \Delta\varphi$. An angular increment $\Delta\varphi$ is given by the resolution of the measuring system, or by the sampled image.

We store panoramic images in a format such that the coordinate origin is at the upper left corner, and such that all coordinates i and j are always positive. Assuming an ideal case of having a camera with camera coordinate origin at the center of the line sensor (i.e., j runs from $+\frac{H}{2}$ to $-\frac{H}{2}$), the assumed storage format means that j is replaced by $\frac{H}{2} - j$. In case that i represents an absolute angle, we perform quadrant checks where needed (e.g., for calculating trigonometric functions). Furthermore, we consider φ modulo 2π which ensures that $\varphi \in [0, 2\pi)$.

Notation of Vectors and Matrices

Matrices are given in bold capitals, and vectors in (small) bold letters. For the length of vector **p** we write $|$**p**$|$. The unit vector of **p** is **p**$^\circ$. Unit vectors along the *x*-, *y*-, or *z*-axis are defined as $\mathbf{e}_x = (1, 0, 0)^T$, $\mathbf{e}_y = (0, 1, 0)^T$, or $\mathbf{e}_z = (0, 0, 1)^T$, respectively.

Notation for 3D Points and Base Planes

A 3D point **p** is (typically) given in a 3D Cartesian coordinate system (e.g., the world coordinates). For 3D point coordinates we use XYZ for world coordinates or xyz for local coordinates. In the 3D world coordinate system we consider the *XY-plane* to be the *base plane* (i.e., the plane where the objects are standing on). The *XZ-plane* is sometimes also called the *front plane*. Often we consider a 3D point also as a vector, writing, for example, $\mathbf{p} = (p_x, p_y, p_z)^T$, where superscript *T* denotes the transpose.

Note that in computer graphics the use of homogeneous coordinates is common practice to accelerate further calculations. Homogeneous coordinates, introduced by August Ferdinand Möbius, allows that an affine transformation is represented by one matrix. The vector of a 3D point has to be extend by one dimension $\mathbf{p} = (p_x, p_y, p_z, p_w)^T$ with $w = 1$. The inverse transformation is given as follows:

$$(p_x^{'}, p_y^{'}, p_z^{'}, p_w^{'})^T \rightarrow \left(\frac{p_x^{'}}{p_w^{'}}, \frac{p_y^{'}}{p_w^{'}}, \frac{p_z^{'}}{p_w^{'}} \right) = (p_x^{''}, p_y^{''}, p_z^{''}) \tag{2.1}$$

This has the advantage that also perspective transformations are describable within one matrix. Matrices has to extend by one dimension as well, such they given as a 4×4 matrix.

Definition of Attitudes

The *interior attitude* is the position and direction of the sensor in the local coordinate system, and defined by six degrees of freedom.

The *exterior attitude* is defined by a transformation from the local coordinate system into the world or reference coordinate system. The transformation is given by a translation $\mathbf{r}_0$ followed by a rotation in form of a matrix $\mathbf{R}$; in photogrammetry rotation angles are denoted by ω, ϕ, and κ. Due to a notational conflict with camera parameter ω, we use the symbol ψ instead. The rotation about the *X*-axis with angle ψ is defined by the rotation matrix

$$\mathbf{R}_\psi = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(\psi) & -\sin(\psi) \\ 0 & \sin(\psi) & \cos(\psi) \end{pmatrix} \tag{2.2}$$

The rotation about the *Y*-axis with angle ϕ is defined by the rotation matrix

$$\mathbf{R}_\phi = \begin{pmatrix} \cos(\phi) & 0 & \sin(\phi) \\ 0 & 1 & 0 \\ -\sin(\phi) & 0 & \cos(\phi) \end{pmatrix} \tag{2.3}$$

and the rotation about the *Z*-axis with angle κ by the rotation matrix

$$\mathbf{R}_\kappa = \begin{pmatrix} \cos(\kappa) & -\sin(\kappa) & 0 \\ \sin(\kappa) & \cos(\kappa) & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{2.4}$$

Sometimes we use different notations for specifying a rotation (e.g., $\mathbf{R}_\varphi$, a deliberately rotation of the sensor about the *z*-axis to create the panoramic image), and also to identify particular rotations such as the interior or exterior rotations. In any case, all rotation matrices are notated by **R**. If no index is given, a rotation about all axes $\mathbf{R} = \mathbf{R}_\psi \cdot \mathbf{R}_\phi \cdot \mathbf{R}_\kappa$ is assumed. The resulting matrix **R** is then given as

$$\begin{pmatrix} C\varphi \cdot C\kappa & S\varphi \cdot S\kappa & S\varphi \\ C\psi \cdot S\kappa + S\psi \cdot S\varphi \cdot C\kappa & C\psi \cdot C\kappa - S\psi \cdot S\varphi \cdot S\kappa & -S\psi \cdot S\varphi \\ S\psi \cdot S\kappa - C\psi \cdot S\varphi \cdot C\kappa & S\psi \cdot C\kappa + C\psi \cdot S\varphi \cdot S\kappa & C\psi \cdot C\varphi \end{pmatrix} \tag{2.5}$$

where κ, ϕ, ψ are the rotation angles about the Z-, Y-, and X-axis, respectively. Symbols C and S stand short for the cosine and sine function, respectively.

Viewing Direction

The *viewing direction* $\mathbf{v}_{ij}^\circ$ or $\mathbf{R}_\varphi(i) \cdot \mathbf{v}_j^\circ$ is given by image coordinates i and j as an *image vector* for a matrix camera, and in combination with a rotation for the line camera . The image vector is given as

$$\mathbf{v}_{ij} = \begin{pmatrix} i \cdot \delta \\ f \\ j \cdot \delta \end{pmatrix} \tag{2.6}$$

and applying our notation for unit vectors

$$\mathbf{v}_{ij}^\circ = \frac{\mathbf{v}_{ij}}{|\mathbf{v}_{ij}|} \tag{2.7}$$

for the matrix camera; for a rotating CCD line senor we have $\mathbf{R}_\varphi(i) \cdot \mathbf{v}_j$ with

$$\mathbf{v}_j = \begin{pmatrix} 0 \\ f \\ j \cdot \delta \end{pmatrix} \quad \text{and} \quad \varphi = i \cdot \Delta\varphi \tag{2.8}$$

and $\mathbf{v}_j^\circ$ for the unit vector. The rotation matrix $\mathbf{R}_\varphi(i)$ is also abbreviated as $\mathbf{R}_\varphi$ in this thesis, if the value of i is obvious.

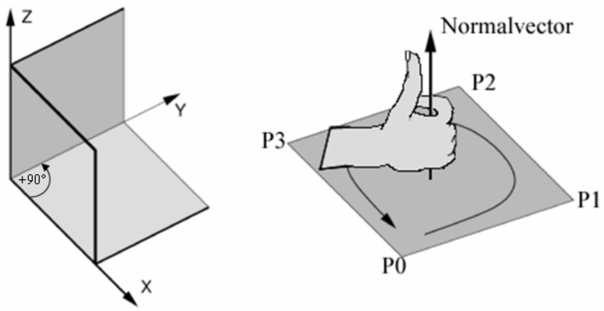


Figure 2.2: Right-hand Cartesian coordinate system and defined planes.

Orientations

We always assume a *right-hand Cartesian coordinate system* (i.e., if thumb, index finger and middle finger of the right hand are held so that they form three right angles, then the thumb indicates the *x*-axis, the index finger the *y*-axis and the middle finger the *z*-axis.)

A common method to determine the normal vector of polygons in computer graphics applications follows this right-hand rule: When an observer looks at the polygon from one side, and the points of the polygon are defined in counter-clockwise orientation, then the normal vector points towards the observer. This can be symbolized by a "thumbs up" closed fist of the right hand. The four remaining fingers point in counter-clockwise orientation (the sequence of the tips), while the thumb shows the direction of the normal vector, and this side is the *front side* of a polygon.

Quaternion

Rotations about a specified axis in 3D space are often necessary. Those rotations are the combination of two rotations about the spatial axes at least. The use of quaternions is of benefit for an efficient implementation. Quaternions are defined as follows:

$$\bar{q} = \left(q_r, \mathbf{q}_v^T\right)^T \qquad \text{with } q_r \in \mathbb{R}, \mathbf{q}_v \in \mathbb{R}^3 \tag{2.9}$$

Whereby q_r is a scalar value and represent a rotation about an arbitrary axis. The rotation about an arbitrary axis **x**, with $|\mathbf{x}| = 1$ by the angle ϕ is represented by the quaternion $\bar{q}$ as follows:

$$\bar{q} = \left(\cos(\tfrac{\phi}{2}), \mathbf{x}^T \sin(\tfrac{\phi}{2})\right)^T \qquad (2.10)$$

with $q_r = \cos(\frac{\phi}{2})$, and $\mathbf{q}_v^T = \mathbf{x}^T \sin(\frac{\phi}{2})$.

We have $|\bar{q}| = 1$ in this case. The rotation of a vector **p**, expressed as a quaternion $\bar{p} = (0, \mathbf{p})^T$ is represented as the following quaternion product:

$$\bar{p}\,' = \bar{q} \circ \bar{p} \circ \bar{q}\,^* \qquad (2.11)$$

with $\bar{q}\,^*$ as the conjugated quaternion of $\bar{q}$, defined as follows:

$$\bar{q}\,^* = \left(q_r, -\mathbf{q}_v^T\right)^T \qquad (2.12)$$

and their product (non-commutative) as follows:

$$\bar{q} \circ \bar{p} = \begin{pmatrix} q_r \cdot p_r - \mathbf{q} \cdot \mathbf{p} \\ q_r \cdot \mathbf{p} + p_r \cdot \mathbf{q} + \mathbf{q} \times \mathbf{p} \end{pmatrix} \qquad (2.13)$$

In matrix form (with its corresponding submatrix to **R**) we have:

$$\begin{aligned} \bar{p}\,' &= \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 - 2(q_2^2 + q_3^2) & 2(q_1 q_2 - q_0 q_3) & 2(q_1 q_3 + q_0 q_2) \\ 0 & 2(q_1 q_2 + q_0 q_3) & 1 - 2(q_1^2 + q_3^2) & 2(q_2 q_3 - q_0 q_1) \\ 0 & 2(q_1 q_3 - q_0 q_2) & 2(q_2 q_3 + q_0 q_1) & 1 - 2(q_1^2 + q_2^2) \end{pmatrix} p \qquad (2.14) \\ &= \begin{pmatrix} 1 & \\ & \mathbf{R} \end{pmatrix} \begin{pmatrix} 0 \\ \mathbf{v} \end{pmatrix} \end{aligned}$$

Geometric Dependencies

Geometric dependencies of optic systems with a thin single lens are shown in Figure 2.3. The *focal length* is shown as f (a distance to the focal plane where all rays from infinity

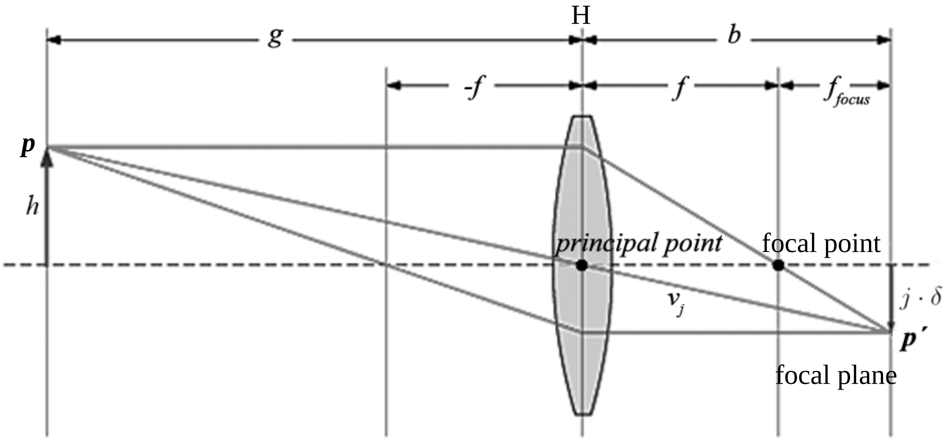


Figure 2.3: Geometric dependencies at a thin lens.

are bundled to one point). Rays coming from infinity and being parallel to the principle axis are bundled in the focal point. Figure 2.3 depicts that the focal plane has to be shifted by f_{focus} for a sharp image mapping (for $g \neq \infty$).

However, the essential image vector $\mathbf{v}_j = (0, f, j \cdot \delta)^T$ of each pixel j, as shown in Figure 2.3, is given by the pre-calibration of the sensor and its fictive focal length, which is used by resampling the uncalibrated image. In this calibration process also a tilted focal plane is considered. The calibrated image fulfils the geometric dependencies as shown in Figure 2.3. The relation between the object distance (g) and the image distance (b) is given as follows:

$$\frac{1}{g} = \frac{1}{f} + \frac{1}{b} \tag{2.15}$$

and

$$\frac{h}{g} = \frac{j \cdot \delta}{b} \tag{2.16}$$

Multiple lens systems are geometrically reducible to a single, thin lens system, whereby the main plane H (in the center) of a single thin lens is replaced by H_1 (front main plane) and H_2 (back main plane) in an optical system. The optical path between these planes is specified by the manufacturer of the optical system. For our geometric calculations the front main plane and its corresponding effective focal length are decisive.

3 Camera

A panoramic camera (see left of Figure 3.1) was developed at DLR Berlin between 1999 and 2001, which allows the acquisition of high-resolution texture maps (a single image is several 100*s* Megapixel, up to multiples of Gigapixel). The camera is basically a rotating CCD line sensor. Three CCD lines (i.e., for the Red, Green or Blue channel) form a linear CCD array, which is mounted vertically on a focal plane and rotates clockwise, describing a cylindric surface during a full 360° rotation. Scanned data are stored in cylindric coordinates, line by line, and according to the sensor geometry.

3.1 Major Components

The camera consists of three major components: a camera head (I), an optic bench (II), and a drive (III).

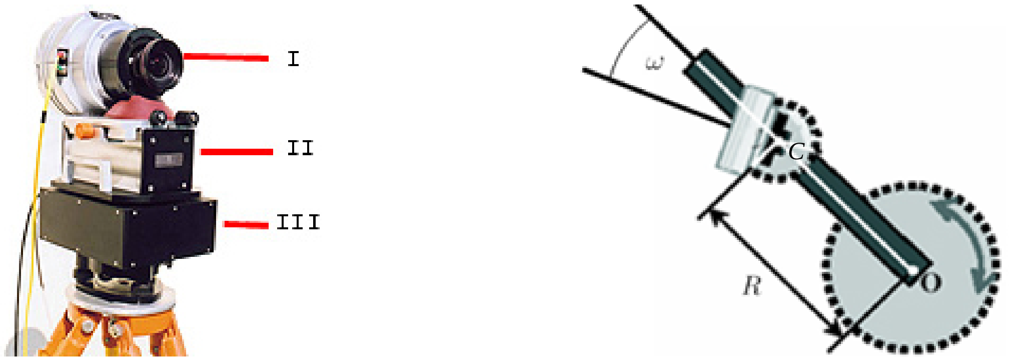


Figure 3.1: Left: EyeScan M3metric, developed at DLR, Berlin (in cooperation with KST GmbH, Dresden). Right: Basic stereo set-up.

(I) The camera head consists of optic and electronic components, fibre-optic (elements for bidirectional data transfer), and the CCD line sensor.

(II) The optic bench supports the camera head and allows to adjust the camera head parallel to the *base plane* (as defined in [HWK$^+$02b]). Point O in Figure 3.1 is the intersection point of the rotation axis with the base plane. Point C is the (ideal) *projection center* (also called *principal point*) of the camera head. The camera head can be positioned, away from the rotation axis, at an *off-axis distance* R on the optic bench, tilted by a *principal angle* ω (to be defined in the next section).

Summarized, the optic bench can be used for a physical movement of the camera head. This allows, for example, to ensure that $R = 0$ (i.e., optic center is exactly at the rotation axis; ω is meaningless in this case). In this situation we capture a *single-projection-center* panoramic image, also *normal panorama* for short. Marks on the optic bench support the acquisitions of normal panoramas; they specify various points which allow that the principal point of a chosen lens coincides with point O.

(III) The drive rotates the optic bench together with the camera head (for a continuous CCD line rotation). The drive contains an angle-increment measuring system which allows to specify the number of lines to be captured during one $360°$ rotation. It also supports camera calibration.

3.2 Off-Axis Rotation and Principle Angle

If R is set (on the optic bench) to a non-zero value, then the camera is at *off-axis position*, which is one possibility to acquire stereo images (using different values of ω, such as symmetric pairs ω and $-\omega$). In this situation we capture a *multi-projection-center* panoramic image. The projection center is moving on a circle with radius R. As defined in [HWK$^+$02b], this circle specifies the *base cylinder*.

The radius of the *image cylinder* is the sum of off-axis distance R and effective focal length. A principal angle $\omega = 0°$ defines a situation where the viewing direction of the linear CCD array coincides always with the normal of the cylinder. As another example,

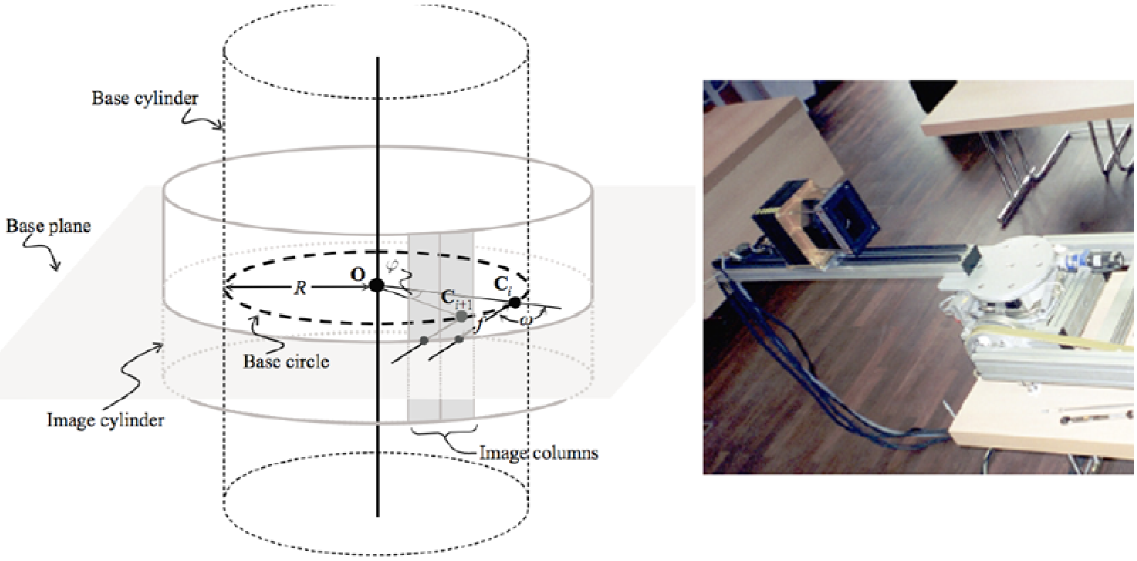


Figure 3.2: Left: the stereoscopic panoramic imaging model. Right: stereoscopic panorama camera at the German Aerospace Center.

$\omega = 90°$ or $\omega = 270°$ define viewing directions which are tangential to the cylinder.

Parameters R and ω are characterized in [HWK$^+$02b], in particular how to optimize these for stereo viewing of a scene characterized by closest and furthest distance between objects of interest and the camera.

R and ω are two important parameters of this camera, and their parameter intervals are crucial for specifying the accuracy or flexibility of the camera. For example, the aim might be to have $R = 0$, but it is important to calibrate the actual deviation from this ideal case. Figure 3.1 illustrates (on the right) the parameters R and ω in a simplified mechanic scheme, and Figure 3.2 illustrates all geometric components.

3.3 Field of View and Focus

The basic advantage of using panoramic cameras is, of course, the extended field of view. Depending on the used lens a whole sphere could be documented during a single scan (using a fisheye lens for a full sphere). Obviously, occluded objects require scans from different positions.

Furthermore, with such a panoramic camera it is possible to have very high resolution data compared to single frame cameras. The camera, typically used in our experiments, possesses a 70 mm Kodak CCD line sensor. Each image has a vertical resolution of 10,200 pixels (for each of the three color channels), and during one $360°$ rotation we have up to 160,000 image columns depending on the used lens.

Note that the number W of columns is different for each lens, due to the aim to ensure square pixels. Table 3.1 shows dependencies between image resolution and used lens. The image size in this table is based on having 48 bit (i.e., 3×16 for RGB images) for each pixel, and a full $360°$ scan. Actually, the electronic unit has a 14 bit analog-digital converter, but data are stored in the common 16 bit format.

The horizontal field of view (H_{FOV}) is the number W of columns multiplied with the *instantaneous field of view*, the angular increment $\Delta\varphi$ of a pixel or one column (in radian), and is calculated for pixel size δ and focal length f as follows:

$$\begin{aligned}\Delta\varphi &= 2 \cdot \arctan\left(\frac{\frac{1}{2}\delta}{f}\right) \approx \frac{\delta}{f}\,, \qquad f \gg \delta && (3.1)\\ H_{FOV} &= W \cdot \Delta\varphi && (3.2)\end{aligned}$$

For example, the number W of columns for a full $360°$ scan is calculated as follows:

$$W = \frac{2\pi}{\Delta\varphi} \tag{3.3}$$

The vertical field of view V_{FOV} is defined by the focal length and the physical length of the CCD (i.e., the number H of pixels with pixel size δ). We have

$$V_{FOV} = 2 \cdot \arctan\left(\frac{H \cdot \delta}{2 \cdot f}\right) \tag{3.4}$$

The focal plane contains a focussed projection $\mathbf{p}'$ of a 3D point $\mathbf{p}$; it is indicated by image distance b in Figure 2.3. The used camera allows to shift focal plane by the distance f_{focus} (in five *focus positions*, named 0,1...,4) within an accuracy of some micro meters. After changing the focus (i.e., f_{focus}), an reproducible accuracy of one pixel is given (as verified in laboratory experiments). Zoom lenses are not able to fulfill photogrammetric

Focal length in mm	Image size in GByte	Resolution in degree	Resolution in mm @10m
25	1.28	0.0160	2.80
35	1.79	0.0115	2.00
45	2.30	0.0089	1.66
60	3.07	0.0067	1.16
100	5.12	0.0040	0.70
180	9.21	0.0022	0.39

Table 3.1: Image resolution in dependency of used lenses, for pixel size ($\delta = 7\ \mu$m).

focal length in mm \ focus position	0	1	2	3	4
35	1.0	1.5	2.2	3.6	9.9
45	1.9	3.1	4.6	8.5	70
60	3.3	4.9	8.0	14.7	122.0
100	9.2	14.6	22.0	40.0	336.0
180	30.0	45.0	70.0	130.0	1080.0

Table 3.2: Ideal object distances of the panoramic camera.

requirements because of hysteresis effects. Table 3.2 shows ideal object distances of the panoramic camera; the shown values in this table are the accurately calculated object distances g. Each optic system has a depth of field which depends on the aperture. As a rule of thumb, the depth field has a range of $[\frac{g}{3}, \frac{5g}{3}]$. As an example, for $g = 8$ m we have [5.3 m, 13.3 m].

3.4 Examples and Drawbacks

Figure 3.3 shows an example of a panoramic image. The image was taken near the Harbour Bridge in Auckland; the distance to downtown is approximately 2 km. A 60 mm lens was used for this image. Figure 3.4 shows the original resolution on the right; it is possible to see the advertisement in the city. The image size is 3 GByte.

A disadvantage of this camera is the difficulty to achieve correct geometric modeling of captured texture data. Basically, a cylindric model would be sufficient for an ideal camera of the described architecture. But in reality, several mechanic errors (e.g., non-uniform rotations, tumbling of axis, eccentricities) can occur; see also related work on

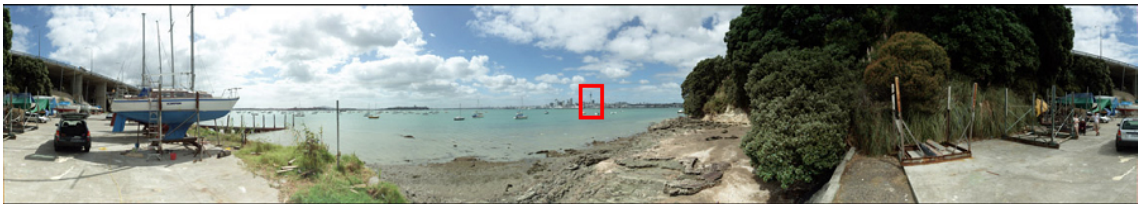


Figure 3.3: This panoramic image (380°) was taken at Auckland's Northcote point, and shows a view near the harbour bridge towards downtown Auckland. The region marked with a rectangle is enlarged in Figure 3.4).

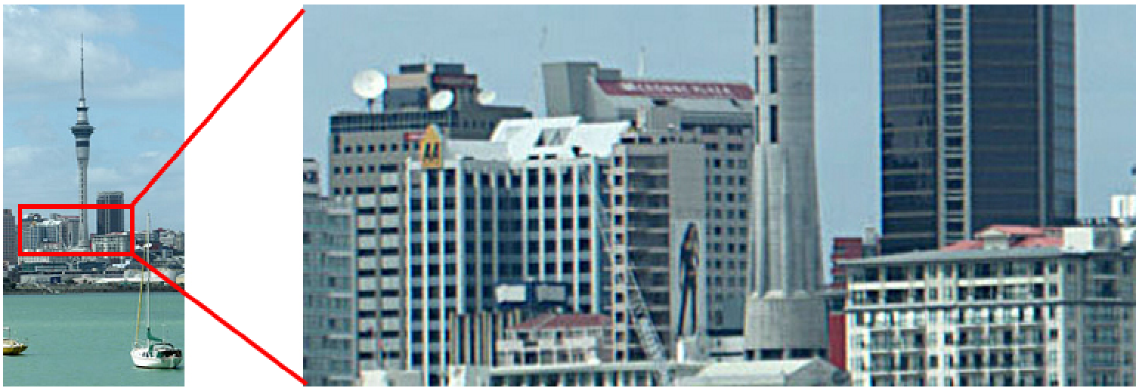


Figure 3.4: Left: a zoom into the city. Right: the original resolution (i.e., a detail of 500×200 pixel of the image shown in Figure 3.3, which has originally 56,580×10,200 pixel @15cm×2.7cm $\approx$ 10,000 dpi).

this topic in [SM04]. Furthermore, the camera does not take images in a single shot. The image is "composed" line by line, and the acquisition of 160,000 lines requires at least 10 minutes in bright daylight. During this time, objects in the scene should not move. The camera is basically designed for archiving static objects.

Also, the camera should not move during image acquisitions (e.g., due to vibration on a bridge). Otherwise an *inertial measuring unit* (IMU) is necessary to measure the movement of the camera. An IMU with sufficient accuracy is expensive. However, a CCD line scanner moveable freely in 3D space can be of interest, and it is discussed in [GBS+04] and [RWK+03]. Figure 3.5 illustrates such distortions. An image based algorithm to correct such distortions is described in Section 3.6.5.

For further technical details of this panoramic camera, see [SKR+01].

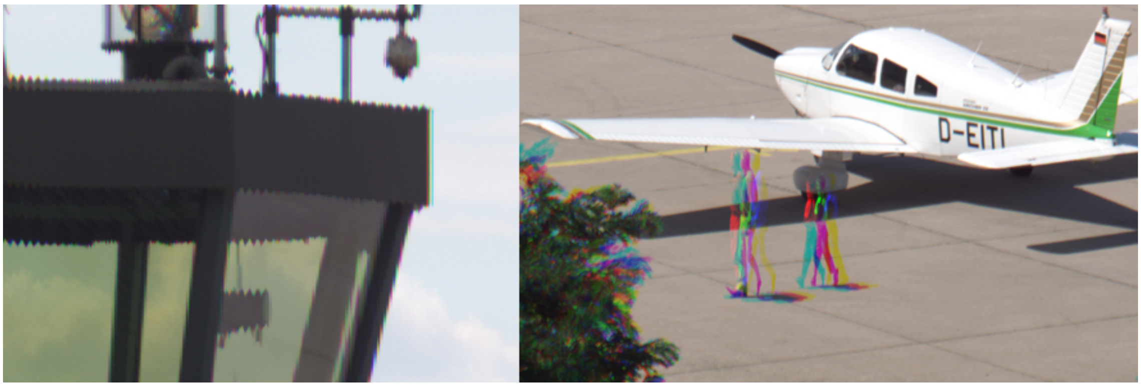


Figure 3.5: Left: Image of an airport tower with distortions caused by vibrations, and people (”ghosts”) move when the rotating line sensor was scanning in their direction. Moving leaves or water waves are examples of critical factors in landscape acquisitions.

3.5 Coordinate Systems

In an ideal focal plane each viewing direction of a combined pixel triple (i.e., all three color channels) of a CCD line is defined by a vector $\mathbf{v}_j$. The rotation axis of the camera is incident with the principal point of the lens. The focal plane is located at effective focal length $f = f + f_{focus}$ (measured along the positive *y*-axis), without any offset $\mathbf{\Delta}$ (i.e., the intersection of the principle axis with the focal plane, also called main point displacement). Scans begin at a horizontal angle of 0 degree. These deviations are illustrated in Figure 3.6. We have the following:

$$\mathbf{v}_j = (v_x, v_y, v_z)^T = (0, f, j \cdot \delta)^T \tag{3.5}$$

The used CCD line has a length of approximately 70 mm or 10,200 pixels, with a pixel size of $\delta = 7$ μm, and pixels are indexed by j. Each scanned surface point is identified by the image coordinates i, and j. The camera rotation is defined as follows:

$$\varphi = \Delta\varphi \cdot i \tag{3.6}$$

The transformation to a reference coordinate system is described by a reference vector (e.g., in world coordinates) for the camera coordinate system and a rotation matrix $\mathbf{R}$ as follows:

$$\mathbf{r} = \mathbf{r}_0 + \mathbf{R} \cdot \mathbf{R}_\varphi \cdot \lambda \cdot \mathbf{v}_j \tag{3.7}$$

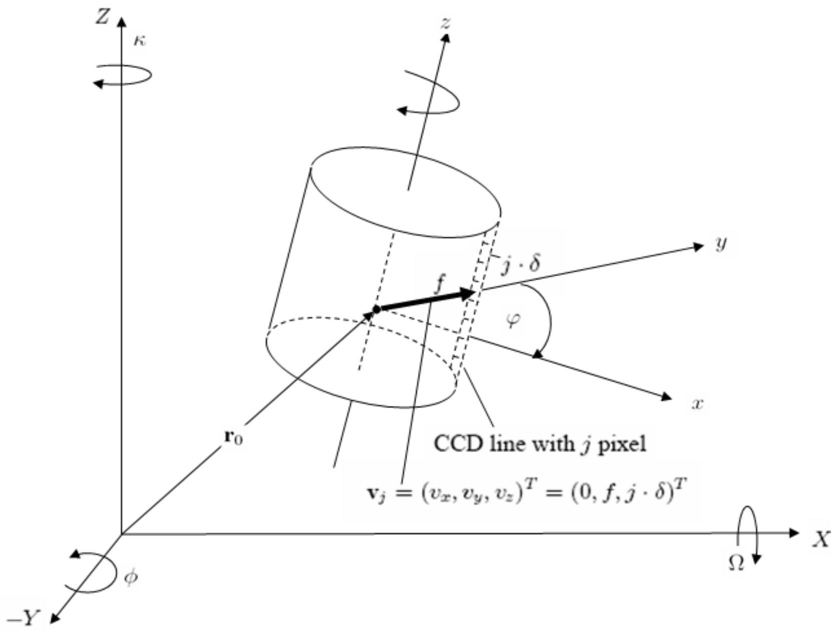


Figure 3.6: The xyz-coordinate system of the rotating line camera: the effective focal length f defines the position of an image column (i.e., the position of the CCD line at this moment) parallel to the z-axis.

or for the off-axis case as follows:

$$\mathbf{r} = \mathbf{r}_0 + \mathbf{R} \cdot \mathbf{R}_\varphi \left(\lambda \cdot \mathbf{v}_j + R \cdot \mathbf{e}_y \right) \tag{3.8}$$

λ is an unknown scale factor of the camera coordinate system (for the 3D scene). If LRF and camera coordinate systems have the same origin, then λ corresponds to the distance measured by the laser scanner. We also model the following deviations from an ideal case (as shown in Figure 3.7):

- The CCD line is tilted by three angles $\mathbf{R}_i(\alpha, \beta, \gamma)$ (interior attitude about all three axes) regarding the principal point.
- The CCD line has an offset vector $\boldsymbol{\Delta}$ regarding the principal point.
- The principle axis is rotated by $\mathbf{R}_x(\xi)$ (rotation about the *x*-axis)
- The CCD is rotating with an eccentricity or a desired off-axis $R > 0$.
- The viewing direction is changed by a fix angle ω, illustrated in Figure 3.2.

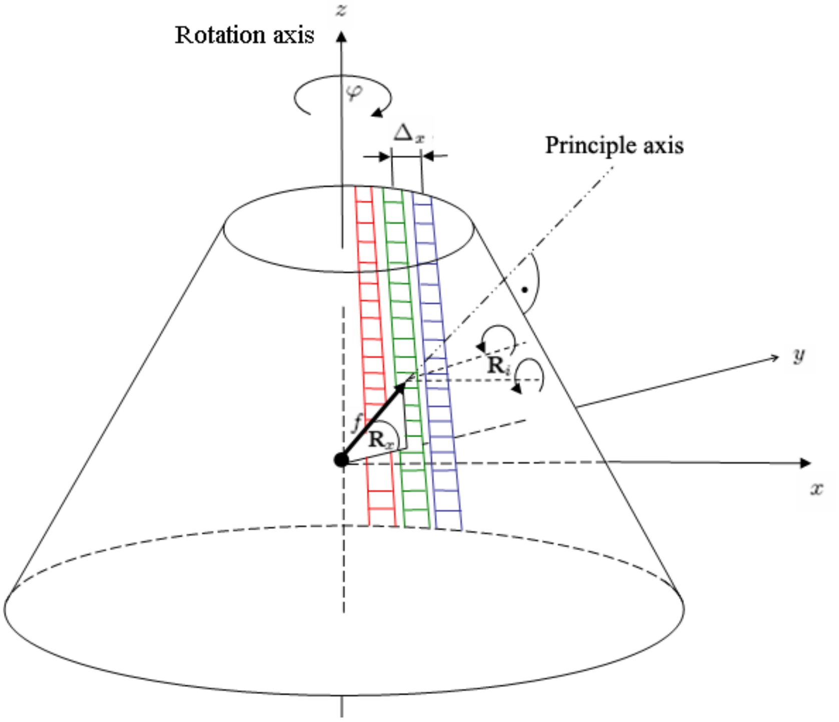


Figure 3.7: The xyz-coordinate system of the rotating line camera: the effective focal length f defines the position of an image column, which is tilted by $\mathbf{R}_x(\xi)$ (principle axis regarding the rotation axis z), with an assumed offset Δ for the center of this image column (i.e., the intersection of the principle axis with the focal plane). Tilting of the CCD line to the principle axis is specified by ($\mathbf{R}_i(\alpha, \beta, \gamma)$).

The difference between the parameters $\mathbf{R}_x$ and $\mathbf{R}_i$ are the description of a rotation of the whole camera head (principle axis) which is similar to a rotation of the CCD line around the focal point without moving the lens, or a rotation $\mathbf{R}_i$ which is about the CCD line itself. Differences between both rotations are illustrated in Figure 3.7 and Figure 3.8. Therefore the image vector $\mathbf{v}_j$ has to be split in two terms to:

$$\mathbf{v}_j = \mathbf{v}_{j0} + \mathbf{v}_{f,\Delta} = (0, 0, j \cdot \delta)^T + (\Delta_x, \Delta_y + f, \Delta_z)^T \tag{3.9}$$

The following equation describes these deviations:

$$\mathbf{r} = \mathbf{r}_0 + \lambda \cdot \mathbf{R}\mathbf{R}_\varphi \left(\mathbf{R}_x \left[\mathbf{R}_i \begin{pmatrix} 0 \\ 0 \\ j \cdot \delta \end{pmatrix} + \begin{pmatrix} \Delta_x \\ \Delta_y + f \\ \Delta_z \end{pmatrix} \right] \right) \tag{3.10}$$

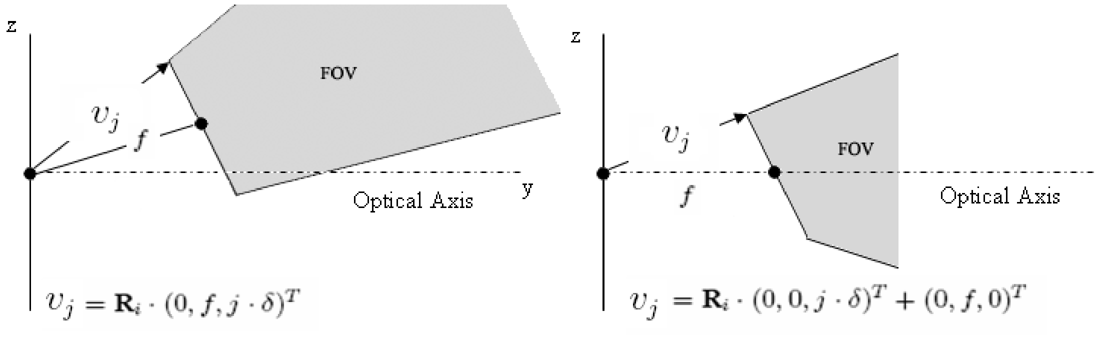

z
FOV
v_j
f
Optical Axis
y
$v_j = \mathbf{R}_i \cdot (0, f, j \cdot \delta)^T$
z
v_j
FOV
f
Optical Axis
y
$v_j = \mathbf{R}_i \cdot (0, 0, j \cdot \delta)^T + (0, f, 0)^T$

Figure 3.8: Difference between a rotation of the CCD line around the focal point (left), or about the CCD line itself (right). For the first case, it does not matter whether the CCD line within the camera head or the camera head as whole is rotated.

In case of an off-axis rotation with $R > 0$ and $\omega = 0$, the typical case of an eccentricity (i.e., the camera is moving on a circular way around the rotation axis) the Equation (3.10) has to be expanded by R as follows:

$$\mathbf{r} = \mathbf{r}_0 + \mathbf{R}\mathbf{R}_\varphi \left(\lambda \cdot \mathbf{R}_x \left[\mathbf{R}_i \begin{pmatrix} 0 \\ 0 \\ j \cdot \delta \end{pmatrix} + \begin{pmatrix} \Delta_x \\ \Delta_y + f \\ \Delta_z \end{pmatrix} \right] + R \cdot \mathbf{e}_y \right) \tag{3.11}$$

Using this off-axis approach to acquire stereo data, the viewing direction has to be changed by ω and it is shown in a general equation.

$$\mathbf{r} = \mathbf{r}_0 + \mathbf{R}\mathbf{R}_\varphi \left(\lambda \cdot \mathbf{R}_x \cdot \mathbf{R}_\omega \left[\mathbf{R}_i \begin{pmatrix} 0 \\ 0 \\ j \cdot \delta \end{pmatrix} + \begin{pmatrix} \Delta_x \\ \Delta_y + f \\ \Delta_z \end{pmatrix} \right] + R \cdot \mathbf{e}_y \right) \tag{3.12}$$

The geometric dependencies of the parameters $\mathbf{R}_\omega$ and R are illustrated in Figure 3.2. For calculation of these parameters and their optimizations regarding a best stereo view on a screen, see [HWK$^+$02b, HWK01]. For calibration of these parameters see related work in [HWK02a] as well. An adjustment calculation for all parameters of a rotating CCD line camera is introduced in Section 3.8.

3.6 Data Preprocessing

For further usage of the camera data it is necessary to preprocess the data and to convert them into a common image format. Therefore a number of corrections concerning the raw data need to be executed. The electronic offset of each channel, which reads out the CCD line (i.e., RGB data and the odd/even detector elements[1]) need to be measured. Other basic operations are the correction of the color shift between channels Red, Green and Blue, caused by the spatial alignment of the three RGB lines, compensation of shading caused by optical components, and an optional data reduction from 16 to 8 bit with suitable scaling functions preserving contrast and information content of the image.

3.6.1 Offset, PRNU and DSNU Correction

The offset for an electronic channel is notated as $O(j)$, where the index j is the physical pixel number of the detector, and the used channel is a function of j. The signal of a detector element is notated as $S_j(t)$. Value t addresses the element of the sensor at time t.

When processing images, the *dark signal non-uniformity* (DSNU) and the *photo response non-uniformity* (PRNU) need to be eliminated. These effects are caused by changes in temperature or integration time, and by different sensitivities of the detector elements with regards to at-sensor radiances, respectively. For measuring these effects, the camera is radiometrically and geometrically gauged in a laboratory at DLR.

The DSNU is measured for total absence of light and a long integration interval. Note that each detector element has its own dark signal noise depending on the integration time. Actually, each pixel should have a digital value of zero (i.e., black). Because of the dark signal noise, the pixels have a digital value > 0. Therefore, a normalized detector factor D_i is estimated at first during calibration, and then saved as a profile. During image acquisition, the level of DSNU is determined by observing the first detectors of the CCD line (i.e., pixel 10 to 30, notated as $\overline{D}(t)$): these elements are covered such that they are not illuminated, and then the DSNU level is multiplied with each normalized detector

[1] A detector element is the physical pixel on the CCD line. Note that a usually CCD line has more than one electronic channel to read out the data, and these are different by offset, noise, and so forth.

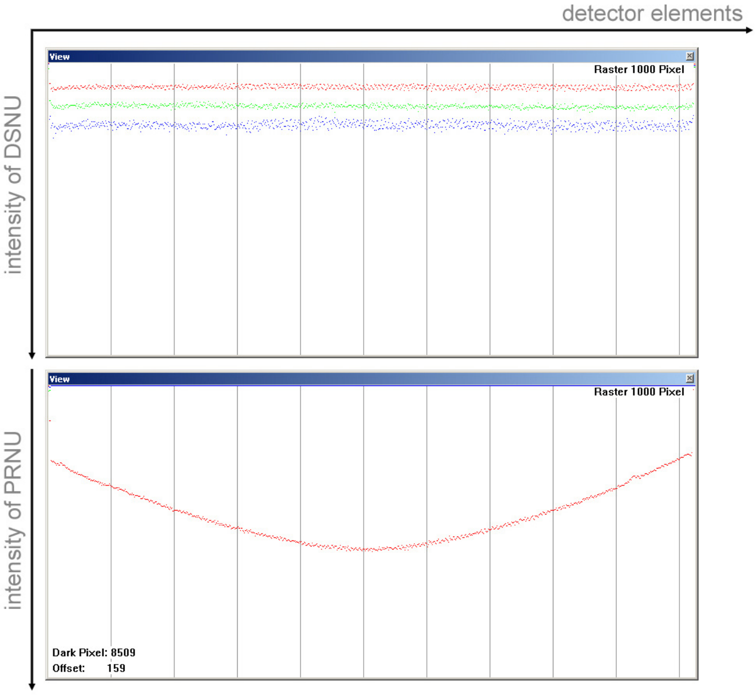


Figure 3.9: DSNU and PRNU, the latter one showing an optical gradient of the CCD line sensor.

factor and subtracted from the current signal (i.e., each illuminated pixel). The signal regarding the DSNU is calculated as follows:

$$S_j^{'}(t) = S_j(t) - \overline{D}(t) \cdot D_j \tag{3.13}$$

with the estimated calibration profile

$$D_j = \frac{1}{n} \cdot \sum_{i=0}^{n-1} \frac{S_j(t_i)}{\overline{D}(t_i)} \tag{3.14}$$

in case of n iterations.

For the PRNU correction, the CCD line sensor is illuminated with homogeneous light (e.g., Ulbricht-Sphere). For each detector element, a factor P_j is determined to normalize each element to the same sensitivity:

$$P_j = \frac{1}{n} \cdot \sum_{i=0}^{n-1} \frac{\max(\{S_j(t_i)\})}{S_j(t_i)} \tag{3.15}$$

Finally, the actual signal is reduced by noise and shifted to the origin after the three basic corrections as follows:

$$S_j^{'}(t) = (S_j(t) - O(j) - \overline{D}(t) \cdot D_j) \cdot P_j \tag{3.16}$$

3.6.2 Correction of the Color Shift

The color shift depends on spatial alignment (i.e., the distance Δ_x between the sensor lines for Red, Green and Blue), the pixel size δ, and the acquisition geometry (i.e., the off-axis parameter R, and the attitude of the principle axis to the rotation axis, defined by the matrix $\mathbf{R}_x(\xi)$.

Single-Projection-Center

In case of a rotation around principle point of the camera (i.e., $R = 0$), also assuming that the principle axis is perpendicular to the rotation axis (i.e., $\xi = 0$), the horizontal color shift is the relative angle $i \cdot \Delta\varphi$, which the camera has to be rotated to see the same object at distance $\mathbf{h} = (0, h, 0)^T$. Therefore, an object is seen by a color line assuming relative angle $\varphi = 0$:

$$\begin{pmatrix} 0 \\ h \\ z \end{pmatrix} = \lambda \begin{pmatrix} 0 \\ f \\ j \cdot \delta \end{pmatrix} \tag{3.17}$$

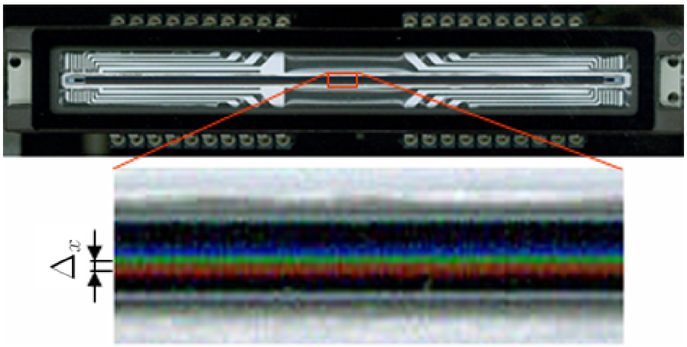


Figure 3.10: Kodak CCD line with illustrated spacing between the sensor lines Green and Red.

The same object is seen by a shifted color line when the camera is rotated by $\mathbf{R}_\varphi(i \cdot \Delta\varphi)$; we have

$$\begin{pmatrix} 0 \\ h \\ z \end{pmatrix} = \lambda \cdot \mathbf{R}_\varphi \begin{pmatrix} \Delta_x \\ f \\ j \cdot \delta \end{pmatrix} \tag{3.18}$$

By using the first component of the equation, the horizontal color shift i (in pixel) can be calculated as follows:

$$\begin{aligned} 0 &= \Delta_x \cdot \cos(i \cdot \Delta\varphi) - f \cdot \sin(i \cdot \Delta\varphi) && (3.19) \\ i &= \arctan\left(\frac{\Delta_x}{f}\right) \cdot \frac{1}{\Delta\varphi} && (3.20) \end{aligned}$$

Consequently, for RGB lines with spacing Δ_x (the camera is rotating around the principle point), and because of Equation (3.1) and Equation (3.20), the horizontal color shift is given as:

$$i = \frac{\Delta_x}{\delta} \qquad \text{with } f \gg \delta \tag{3.21}$$

It can be seen that the horizontal color shift is independent of the object distance h, and is constant for the whole panoramic image. A stereo approach is not applicable, when the camera is rotating around its principle point.

For the determination of the vertical color shift, the Equation (3.18) is solved for λ. The addition of the squares of the first both components eliminates $\mathbf{R}_\varphi$, and λ is determined as follows:

$$\lambda^2 = \frac{h^2}{\Delta_x^2 + f^2} \tag{3.22}$$

Now, substitute λ in the third component of Equation (3.18) and solve its for j gives the following:

$$j = \frac{z \cdot f}{\delta \cdot h}\sqrt{1 + \frac{\Delta_x^2}{f^2}} \tag{3.23}$$

or with assuming $f \gg \Delta_x$, j can estimated as follows:

$$j = \frac{z \cdot f}{\delta \cdot h}\left(1 + \frac{1}{2}\frac{\Delta_x^2}{f^2}\right) \tag{3.24}$$

Because of the determination of j_1 for a nadir line (i.e., $\Delta_x = 0$) with

$$j_1 = \frac{z \cdot f}{\delta \cdot h} \tag{3.25}$$

the vertical color shift (in pixel) is given as:

$$|j_1 - j| = \frac{1}{2}\,\frac{z \cdot \Delta_x^2}{\delta \cdot f \cdot h} \tag{3.26}$$

It can be seen that the vertical color shift is depending on the ratio z/h, and is less than one pixel if following condition is fulfilled.

$$\frac{z \cdot \Delta_x^2}{h \cdot \delta \cdot f} < 2 \tag{3.27}$$

That means a vertical color shift can be excluded because of using RGB lines (i.e., $f \gg \Delta_x$). An unfavorable ratio z/h is implausible. However, the Chapter Data Fusion considered also these cases in a general case. Here each line (i.e., RGB) is mapped separately. The correction of the color shift in a pre-processing step generates distortion free panoramic images.

Multi-Projection-Center

If the camera is rotating eccentrical (e.g., $R \neq 0$), the color shift (defining stereo lines with a small distance in-between) depends on the relation between the object distance h and the rotation radius R, and is shown in Equation (3.35). Following equation describes the off-axis case:

$$\begin{pmatrix} 0 \\ h \\ 0 \end{pmatrix} = \mathbf{R}_\varphi \left[\lambda \begin{pmatrix} \Delta_x \\ f \\ j \cdot \delta \end{pmatrix} + \begin{pmatrix} 0 \\ R \\ 0 \end{pmatrix} \right] \tag{3.28}$$

Multiplying this equation by $\mathbf{R}_\varphi^{-1}$ we have these three components:

$$-\sin(i \cdot \Delta\varphi) \cdot h = \lambda \cdot \Delta_x \tag{3.29}$$

$$\cos(i \cdot \Delta\varphi) \cdot h = \lambda \cdot f + R \tag{3.30}$$

$$0 = \lambda \cdot j \cdot \delta \tag{3.31}$$

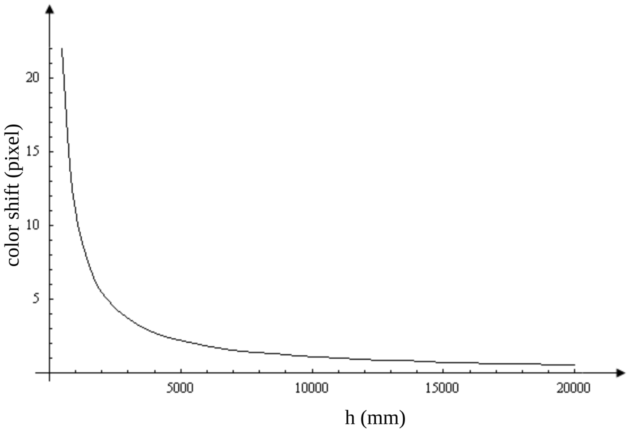


Figure 3.11: Color shift of the RGB lines in off-axis mode for objects at distance h with a typical setup (see text).

λ can be determined by addition of the squares of Equation (3.29) and Equation (3.30) to:

$$h^2 = \lambda^2 \cdot \Delta_x^2 + \lambda^2 \cdot f^2 + 2\lambda \cdot f \cdot R + R^2 \tag{3.32}$$

and its quadratical normal form:

$$0 = \lambda^2 + 2\lambda \frac{f \cdot R}{\Delta_x^2 + f^2} + \frac{R^2 - h^2}{\Delta_x^2 + f^2} \tag{3.33}$$

$$\lambda_{1,2} = \frac{1}{\Delta_x^2 + f^2} \left(-f \cdot R \pm \sqrt{h^2(\Delta_x^2 + f^2) - R^2 \cdot f^2} \right) \tag{3.34}$$

We can assume that λ is positive, because of the forward looking viewing direction of the camera, thus we have only one valid solution λ_1. The color shift is determined by dividing Equation (3.29) and Equation (3.30), now by known λ-value, as follows:

$$i = \arctan\left(\frac{\lambda \cdot \Delta_x}{\lambda \cdot f + R} \right) \frac{1}{\Delta\varphi} \tag{3.35}$$

Figure 3.11 shows the color shift for objects in close range when the camera rotates off-axis. The figure depicts the color shift for a typical setup (i.e., $\Delta_x = 154$ μm, $R = 500$ mm, and $\delta = 7$ μm) based on Equation (3.35). It can be seen that the color shift is less than one pixel, for objects at distance $h > 11$ m with $R = 0.5$ m. Figure 3.12 shows the relation between R and h. If the relation $\frac{R}{h} > \frac{1}{25}$ is valid then no relevant color shift is given.

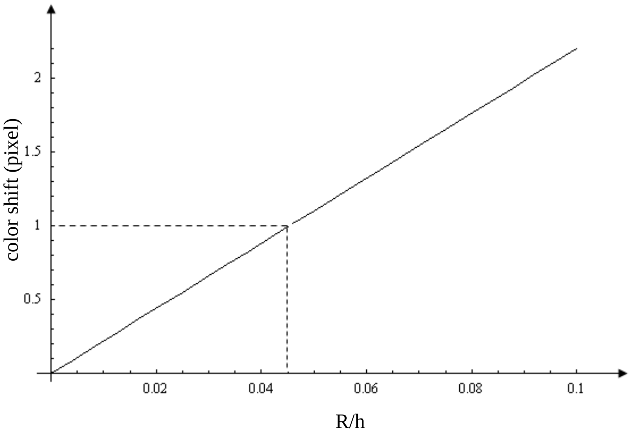


Figure 3.12: Color shift of the RGB lines in off-axis mode for objects in relation to ratio R/h

Tilted principle axis ($\xi \neq 0$)

Figure 3.7 illustrates the "conical acquisition mode" (i.e., $\xi \neq 0$). With an tilted principle axis, the color shift depends on the pixel position j of the CCD line and the angle ξ. Therefore, the Equation (3.19) is expanded by this rotation to the following:

$$0 = \Delta_x \cdot \cos(i \cdot \Delta\varphi) - \sin(i \cdot \Delta\varphi)[\cos(\xi) \cdot f - \sin(\xi) \cdot j \cdot \delta] \tag{3.36}$$

$$i = \arctan\left(\frac{\Delta_x}{\cos(\xi) \cdot f - \sin(\xi) \cdot j \cdot \delta}\right)\frac{1}{\Delta\varphi} \tag{3.37}$$

It can be seen that the color shift is different for each pixel position j on the CCD line.

3.6.3 Radiometric Corrections

Radiometric adjustments of different images are implemented as alignments of their cumulative histograms C_h. The cumulative histogram is defined by partial sums of the

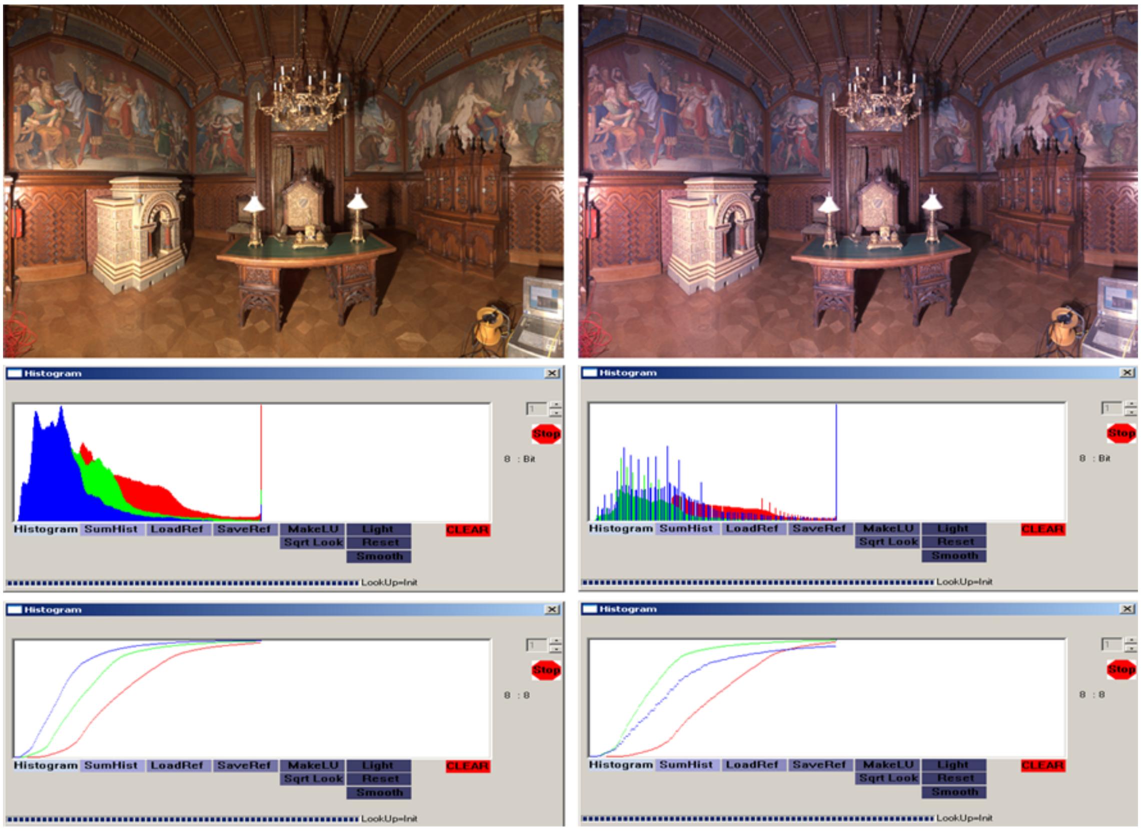


Figure 3.13: Reference image and its histogram and cumulative histogram (left), and an image (right) with its histograms, where this image is taken under light conditions different to the situation on the left.

histogram h, and given as follows:

$$C_h(c, j) = \sum_{i=0}^{j} \frac{h(c, i)}{W \cdot H} \qquad \text{with } 0 \leq j < 2^n \tag{3.38}$$

n is the radiometric resolution (in bit), W and H are the image dimensions (in pixel). The index c addresses the channel (e.g., Red, Green or Blue). After determining a reference cumulative histogram from the reference image, each image can be transformed with respect to the reference image by using the defined transfer function (i.e., given by C_h). This function is implemented in form of a *look-up table* (LUT).

Figure 3.13 shows a reference image with its histogram, cumulative histogram, and an image (for different light conditions) and its histograms. The transformed image, its histogram, and the LUT is shown in Figure 3.14.

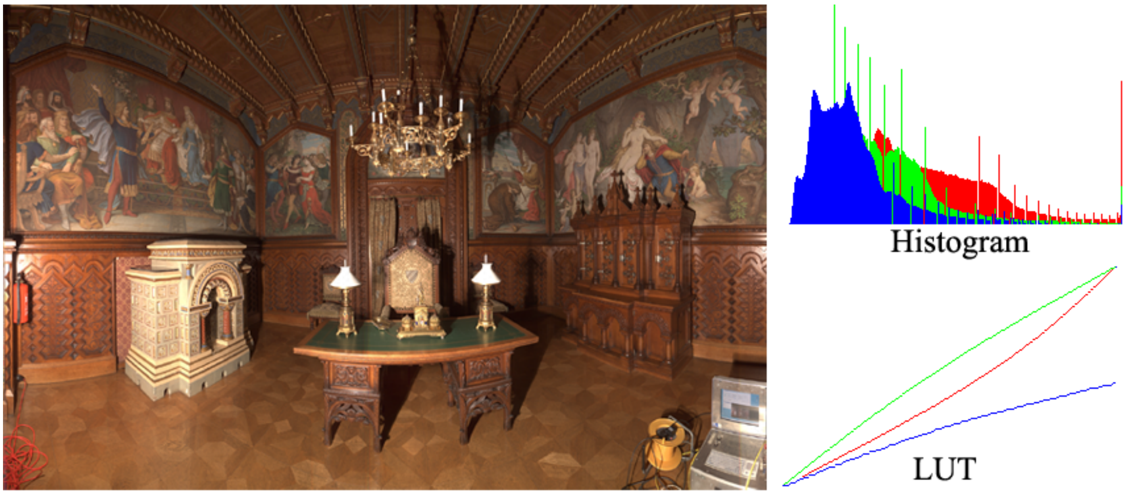


Figure 3.14: Transformed image with its histogram and used LUT.

3.6.4 Geometric Corrections

Geometric errors caused by lens distortions or any other filters (e.g., Ultraviolet- or Infrared-filter), or a tilt of the CCD line, have to be corrected and, therefore, need to be measured. For this purpose, the camera is mounted on a *manipulator*, which is basically a high-precision turntable, which can be rotated with an accuracy of one thousandth of a degree. A sufficient number of detector elements of the CCD line is then illuminated by a *collimator ray*. A collimator is a device that renders divergent or convergent light rays such that they are nearly parallel and illuminate an object from infinity.

Figure 3.15 depicts (in a simplified scheme) the setup. After measuring each detector element about two angles α and β, which are the horizontal and vertical axis of the manipulator, the spatial attitude of the CCD line is mapped into a fictive ideal focal plane, where

$$x^{'} = f \cdot \frac{\tan(\beta)}{\cos(\alpha)} \qquad (3.39)$$

$$y^{'} = f \cdot \tan(\alpha) \qquad (3.40)$$

define the position of each detector element $(x^{'}, y^{'})$ in this ideal focal plane; f is the fictive focal length which should be used for further calculations.

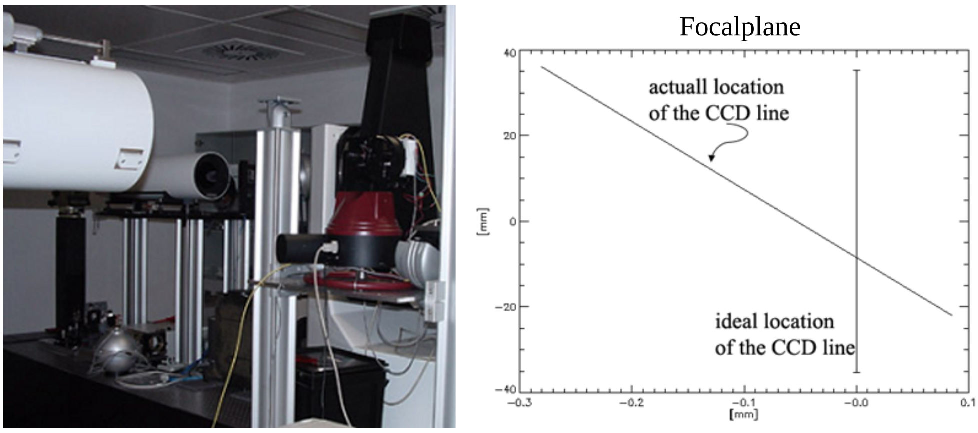


Figure 3.15: Left: Camera mounted on a manipulator to measure geometric properties. The white tube is the collimator. Right: Calculated fictive position of each detector element in an ideal focal plane.

3.6.5 Correction of Mechanical Vibrations

A very simple image- based approach to correct distorted images, caused by vibrations during the image acquisition, is applying a *Normalized Cross-Correlation* (NCC) algorithm. Here, the best correlation between two lines, or a region of these lines, at times t and $t+1$ will be found by shifting both lines with τ pixels to each other. The correlation coefficient is given as:

$$\mathrm{NCC}(\tau) = \frac{\sum\limits_j [S_j(t) - \overline{S}_j(t)] \cdot [S_{j+\tau}(t+1) - \overline{S}_j(t+1)]}{\sqrt{\sum\limits_j [S_j(t) - \overline{S}_j(t)]^2 \cdot \sum\limits_j [S_{j+\tau}(t+1) - \overline{S}_j(t+1)]^2}} \tag{3.41}$$

The best correlation is found at the maximum NCC, with $-1 < \mathrm{NCC} < 1$. The lines are identical if the NCC $= 1$, and inverse if NCC $= -1$. Using the NCC algorithm is just a simple approach and only works when comparing two complete lines or larger regions of lines, because otherwise curved edges will be corrected in straight ones. Using regions on straight edges allows to correct image vibrations reasonably, but at first these edges have to be estimated if they exist. In short, such a simple approach is insufficient, works only partially, and needs to be improved. Figure 3.16 shows the corrected airport tower from Figure 3.5 by using the NCC algorithm.

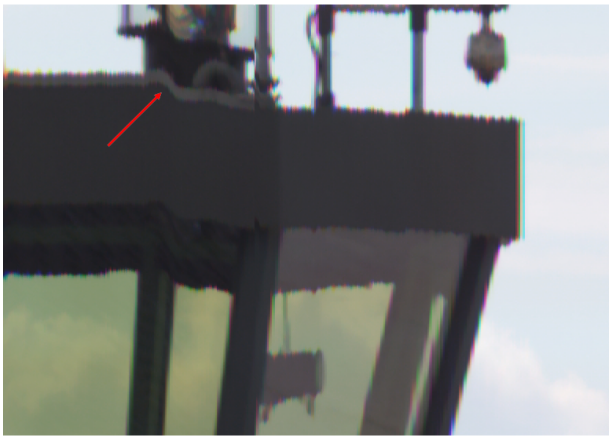


Figure 3.16: Corrected airport tower by using the NCC algorithm. The arrow points at an image part where the restriction on using a small region (see text) caused a false correlation.

3.7 Basic Transformations

To visualize panoramic images in common viewers (on computer screens), and internet applications it is necessary to compute cylindric or spheric transformations. A captured image is basically given in cylindric coordinates because of the rotation of the camera system. Nevertheless, we always compute rectified coordinates for all further calculations. In this step, the estimated calibration parameters (i.e., $\mathbf{R}_i$, $\mathbf{R}_x$, $\mathbf{R}_\omega$ and offset parameters, see Sections 3.5) are applied, to ensure correct image coordinates. Altogether, all parameters after the actual camera rotation $\mathbf{R}_\varphi$ (for $R = 0$) can be computed; they specify the image transformations. Equation (3.10) is simplified to Equation (3.7), and is valid for all further calculations. All other parameters are applied by calculating the object coordinates, see Chapter 7. Our images are then in ideal cylindric or spheric coordinates.

For calculating ideal spheric or cylindric image coordinates, the image vector $\mathbf{v}_j$, see Equation (3.9), has to be transformed by the tilt parameters $\mathbf{R}_x(\xi)$ and $\mathbf{R}_\omega$ for the principle axis, the interior sensor attitude $\mathbf{R}_i(\alpha, \beta, \gamma)$ and the offset parameter $\mathbf{\Delta}$, and the rotation

$\mathbf{R}_\varphi$ to $\mathbf{v}_j^{'} = \mathbf{v}_j(\xi, \omega, \alpha, \beta, \gamma, \mathbf{\Delta}, \varphi)$:

$$\begin{aligned} \mathbf{R}^{-1} \cdot (\mathbf{r} - \mathbf{r}_0) &= \mathbf{R}_\varphi(\lambda \cdot \mathbf{R}_x \cdot \mathbf{R}_\omega[\mathbf{R}_i \cdot \mathbf{v}_{j0} + \mathbf{v}_{f,\Delta}] + R) && (3.42) \\ &= \lambda \cdot \mathbf{v}_j^{'} + \mathbf{R}_\varphi \cdot R \cdot \mathbf{e}_y && (3.43) \\ \mathbf{R}_{\varphi'} \left[\lambda^{'} \begin{pmatrix} 0 \\ f^{'} \\ j^{'} \cdot \delta^{'} \end{pmatrix} + \begin{pmatrix} 0 \\ R \\ 0 \end{pmatrix} \right] &= \lambda \cdot \mathbf{v}_j^{'} + \mathbf{R}_\varphi \cdot R \cdot \mathbf{e}_y && (3.44) \end{aligned}$$

The left side of Equation (3.44) corresponds to an ideal hollow cylinder, whereby $\lambda^{'}$ is a scaling factor of the object coordinates. The parameter $f^{'}$ is the new fictive focal length or radius of this cylinder, and it is freely selectable. The parameter $\delta^{'}$ is the selectable pixel size, and $\Delta\varphi^{'}$ the new angular resolution of this cylinder. To retain the image ratio, the pixel size is given as follows:

$$\delta^{'} = \frac{\delta \cdot f^{'}}{f} \cdot \frac{\Delta\varphi^{'}}{\Delta\varphi} \qquad (3.45)$$

In case of $R = 0$, the spheric image coordinates are as follows:

$$\begin{aligned} i^{'} &= \arctan\left(\frac{v_{xj}^{'}}{v_{yj}^{'}}\right) \cdot \frac{1}{\Delta\varphi^{'}} && (3.46) \\ j^{'} &= \arccos\left(\frac{v_{zj}^{'}}{\sqrt{v_{xj}^{'2} + v_{yj}^{'2} + v_{zj}^{'2}}}\right) \cdot \frac{1}{\Delta\theta^{'}} && (3.47) \end{aligned}$$

with $i^{'}$ and $j^{'}$ being the rectified image column and row, which present the absolute angles $\varphi = i^{'} \cdot \Delta\varphi^{'}$ and $\theta = j^{'} \cdot \Delta\theta^{'}$. The cylindric image coordinates are as follows:

$$\begin{aligned} i^{'} &= \arctan\left(\frac{v_{xj}^{'}}{v_{yj}^{'}}\right) \cdot \frac{1}{\Delta\varphi^{'}} && (3.48) \\ j^{'} &= \frac{f \cdot v_{zj}^{'}}{\sqrt{v_{xj}^{'2} + v_{yj}^{'2}}} \cdot \frac{1}{\delta^{'}} && (3.49) \end{aligned}$$

with $i^{'}$ being the same index like in spheric coordinates, and $j^{'}$ is the image row, which presents the vertical position of each pixel onto the rectified cylinder.

Unfortunately, such a data preprocessing is only possible for single-projection-center

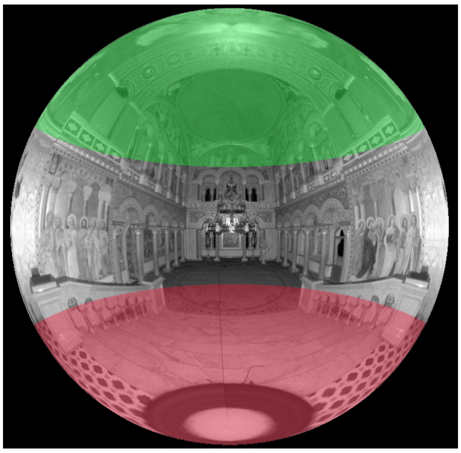


Figure 3.17: Illustration of the projected sphere to a tangential plane. Note that Equation (3.46) describes a flipped open sphere similar to Figure 4.6.

panoramic images, and simplifies the equations for data fusion. In case of multi-projection-center panoramic images (i.e., $R > 0$) these projections into rectified cylinders are possible but change the viewing directions relatively to the center of these cylinders. By using a hollow cylinder, shown in Equation (3.44), it is seen that the equation has four unknown parameters $(\lambda', \varphi', j', \lambda)$. A correct projection would be possible by known object coordinates. Single-projection-center panoramic images can be rectified without knowing of object coordinates because of the possibility of a substitution of λ by λ/λ'.

Figure 3.17 depicts an example of projecting image coordinates into a sphere. Because of the small vertical field of view of the used camera it was necessary to combine three scans. The upper green and the lower red part are acquired with a tilted principle axis $\mathbf{R}_x$. The angle of the principle axis can be changed from $-30°$ to $+30°$. In case of a tilted principle axis the unrectified image coordinates are basically in *conic coordinates*.[2]

The preprocessed images are now in rectified cylindric image coordinates. Straight horizontal lines in object space seem to be curved because the images are isogonal. For a better visualization it is necessary to transform these coordinates into central- perspective Cartesian coordinates. Geometrically, this is the projection of each cylindric pixel onto a tangential plane. Figure 3.18 depicts the Gendarmenmarkt in cylindric coordinates (top), and at the bottom the Gendarmenmarkt is shown as central perspective projection into a tangential plane in Cartesian coordinates. In Figure 3.19, its geometric dependencies are

[2] The coordinates are given as an unrolled cone deformed to a rectangle.

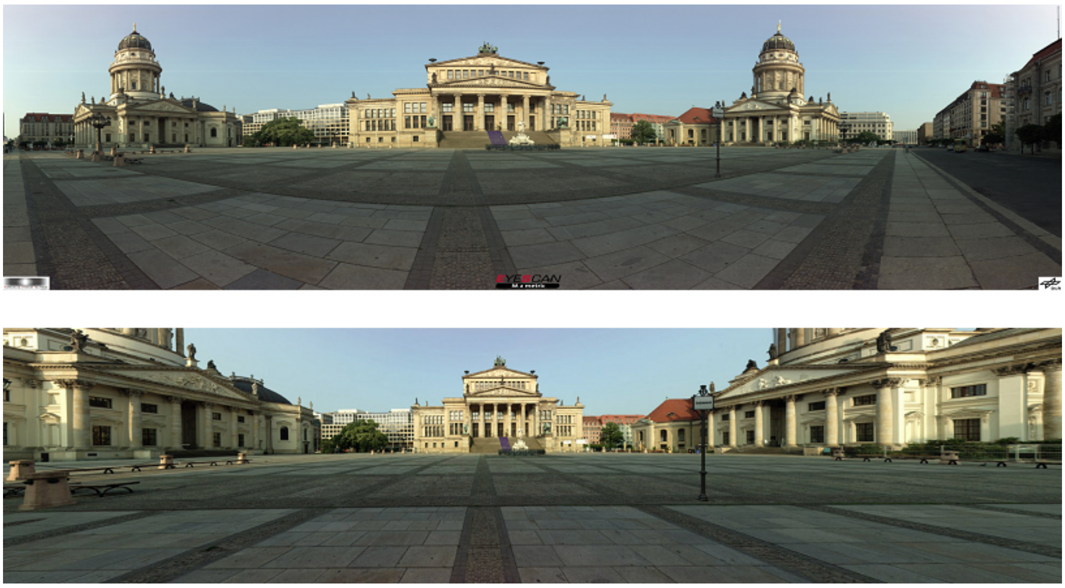


Figure 3.18: Top: Gendarmenmarkt in Berlin in cylindric coordinates. Bottom: Gendarmenmarkt in a rectified, central perspective view.

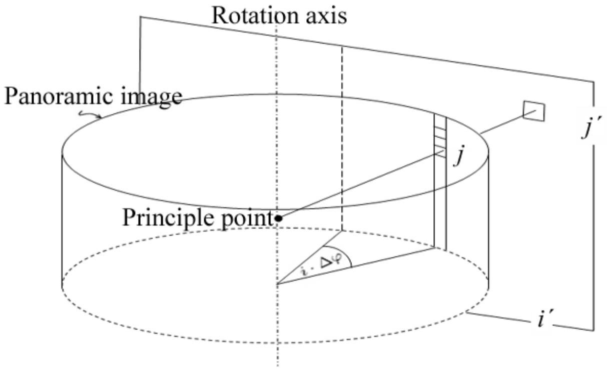


Figure 3.19: Geometric illustration of a projection into a tangential plane.

explained. The projection is only possible for angles less than $180°$, otherwise the image projection is getting divergent to the border. The projection is defined as follows:

$$i' = f \cdot \tan(i \cdot \Delta\varphi) \cdot \frac{1}{\delta} \tag{3.50}$$

$$j' = \frac{j}{\cos(i \cdot \Delta\varphi)} \tag{3.51}$$

3.8 Calibration

In Section 3.6 we described how to precalibrate rotating line cameras at a specially designed calibration site. The disadvantage of this approach is to measure only systems focused to infinity. In close range photogrammetry it is also important to use near-range focusing systems. For example, in the project *bb.Fokal* [SJS04] an approach was examined which uses holographic optic elements to calibrate cameras.

However, this section describes a standard least-square approach, but adapted to rotating CCD lines. This approach determines the m unknowns of the exterior attitude of the camera [i.e., $\mathbf{R}(\Omega, \phi, \kappa), \mathbf{r}_0, R, \mathbf{R}_\omega, \mathbf{R}_x(\xi)$], and the modeled physical terms (i.e., not just correction terms) of the interior attitudes [i.e., $\mathbf{R}_i(\alpha, \beta, \gamma), \Delta, f]$. Given are n observations, with $n \geq m$. Known are n object coordinates XYZ, all as vector $\mathbf{r}$ and its corresponding image coordinates i and j.

For the rotation angle φ the internal measuring system of the turntable is used. This system determines the angle for each line with an accuracy of 1/1000 degree.

Based on general Equation (3.12), and substituting $\mathbf{r}-\mathbf{r}_0 = \tilde{\mathbf{r}}$, $\mathbf{A} = \mathbf{R}\cdot\mathbf{R}_\varphi$, $\mathbf{B} = \mathbf{R}_x \cdot \mathbf{R}_\omega$ and $\mathbf{C} = \mathbf{B}\cdot\mathbf{R}_i$ (with the matrix elements $a_{11}, ..., a_{33}$, $b_{11}, ..., b_{33}$ and so forth) the equation is given as follows:

$$\begin{aligned} \mathbf{r} &= \mathbf{r}_0 + \mathbf{A}(\lambda \cdot \mathbf{B}(\mathbf{R}_i \cdot \mathbf{v}_{j0} + \mathbf{v}_{f,\Delta}) + R \cdot \mathbf{e}_y) & (3.52) \\ \mathbf{A}^{-1} \cdot \tilde{\mathbf{r}} - R \cdot \mathbf{e}_y &= \lambda \cdot \mathbf{C} \cdot \mathbf{v}_{j0} + \mathbf{B} \cdot \mathbf{v}_{j,\Delta} & (3.53) \end{aligned}$$

and the following three components:

$$\begin{aligned} & & (3.54) \\ a_{11} \cdot \tilde{\mathbf{r}}_x + a_{21} \cdot \tilde{\mathbf{r}}_y + a_{31} \cdot \tilde{\mathbf{r}}_z &= \lambda(c_{13} \cdot j \cdot \delta + b_{11} \cdot \Delta_x + b_{12}(\Delta_y + f) + b_{13} \cdot \Delta_z) \\ a_{12} \cdot \tilde{\mathbf{r}}_x + a_{22} \cdot \tilde{\mathbf{r}}_y + a_{32} \cdot \tilde{\mathbf{r}}_z - R &= \lambda(c_{23} \cdot j \cdot \delta + b_{21} \cdot \Delta_x + b_{22}(\Delta_y + f) + b_{23} \cdot \Delta_z) \\ a_{13} \cdot \tilde{\mathbf{r}}_x + a_{23} \cdot \tilde{\mathbf{r}}_y + a_{33} \cdot \tilde{\mathbf{r}}_z &= \lambda(c_{33} \cdot j \cdot \delta + b_{31} \cdot \Delta_x + b_{32}(\Delta_y + f) + b_{33} \cdot \Delta_z) \end{aligned}$$

The coefficients $a_{11}, ..., a_{33}$ are finally transposed because of the inversion of matrix **A**. (For a rotation matrix we have $\mathbf{E} = \mathbf{R} \cdot \mathbf{R}^T$, and consequently $\mathbf{R}^{-1} = \mathbf{R}^T$.)

A special case would be if the interior attitude of the CCD line $\mathbf{R}_i$ is applied around the focal point, illustrated in Figure 3.8. The advantage is the simplification of the equations. All rotation matrices can be combined into one, and its linear expansion is easily derived. For the important parameter of the interior attitude, a tilt of the CCD about the *y*-axis [$\mathbf{R}_i(\beta)$] has no consequences. Furthermore, it is valid for all rotations $\mathbf{R}_i$ in combination with a translation in Δ. The three components would be simplified as follows:

$$\begin{aligned} a_{11} \cdot \tilde{\mathbf{r}}_x + a_{21} \cdot \tilde{\mathbf{r}}_y + a_{31} \cdot \tilde{\mathbf{r}}_z &= \lambda(\Delta_x) \\ a_{12} \cdot \tilde{\mathbf{r}}_x + a_{22} \cdot \tilde{\mathbf{r}}_y + a_{32} \cdot \tilde{\mathbf{r}}_z - R &= \lambda(f + \Delta_y) \\ a_{13} \cdot \tilde{\mathbf{r}}_x + a_{23} \cdot \tilde{\mathbf{r}}_y + a_{33} \cdot \tilde{\mathbf{r}}_z &= \lambda(j \cdot \delta + \Delta_z) \end{aligned}$$

Therefore, the co-linearity equations for this special case (general see below) are defined as follows:

$$\frac{\Delta_x}{f + \Delta_y} = \frac{a_{11}\,(r_x - r_{x0}) + a_{21}\,(r_y - r_{y0}) + a_{31}\,(r_z - r_{z0})}{a_{12}\,(r_x - r_{x0}) + a_{22}\,(r_y - r_{y0}) + a_{32}\,(r_z - r_{z0}) - R} \tag{3.55}$$

and

$$\frac{j \cdot \delta + \Delta_z}{f + \Delta_y} = \frac{a_{13}\,(r_x - r_{x0}) + a_{23}\,(r_y - r_{y0}) + a_{33}\,(r_z - r_{z0})}{a_{12}\,(r_x - r_{x0}) + a_{22}\,(r_y - r_{y0}) + a_{32}\,(r_z - r_{z0}) - R} \tag{3.56}$$

The unknown parameters are functions (F_x, F_z) of these collinearity equations [right side of Equations (3.55) and (3.56)], the focal length f, and the unknown parameter $\mathbf{\Delta}$, which is determined iteratively. The known object coordinates and the image coordinate i are parts of F_x and F_z, respectively. Thus, we have

$$\frac{\Delta_x}{f + \Delta_y} = F_x \tag{3.57}$$

$$\frac{j \cdot \delta + \Delta_z}{f + \Delta_y} = F_z \tag{3.58}$$

The general co-linearity is given by F_x and F_z defined in Equation (3.55) and Equation (3.56) and the separated interior attitude $\mathbf{R}_i$ [see Equation (3.54)] as follows:

$$\begin{aligned} G_x := \frac{c_{13} \cdot j \cdot \delta + b_{11} \cdot \Delta_x + b_{12}(\Delta_y + f) + b_{13} \cdot \Delta_z}{c_{23} \cdot j \cdot \delta + b_{21} \cdot \Delta_x + b_{22}(\Delta_y + f) + b_{23} \cdot \Delta_z} &= F_x \\ G_z := \frac{c_{33} \cdot j \cdot \delta + b_{31} \cdot \Delta_x + b_{32}(\Delta_y + f) + b_{33} \cdot \Delta_z}{c_{23} \cdot j \cdot \delta + b_{21} \cdot \Delta_x + b_{22}(\Delta_y + f) + b_{23} \cdot \Delta_z} &= F_z \end{aligned} \tag{3.59}$$

By linearization of these equations it is possible to estimate iteratively the unknown parameters **u** $= (r_{x0}, r_{y0}, r_{z0}, \Omega, \phi, \kappa, \xi, \alpha, \beta, \gamma, R, \omega)$. The upper index k is the number of the iteration step. For the special case we have

$$\nabla F_{x,z} = \left(\frac{\partial F_{x,z}}{\partial u_1}, \frac{\partial F_{x,z}}{\partial u_2}, ..., \frac{\partial F_{x,z}}{\partial u_m} \right) \tag{3.60}$$

$$\frac{\Delta_x}{f + \Delta_y} - F_x^k = \nabla F_x^k \cdot \Delta \mathbf{u} \tag{3.61}$$

and

$$\frac{j \cdot \delta + \Delta_z}{f + \Delta_y} - F_z^k = \nabla F_z^k \cdot \Delta \mathbf{u} \tag{3.62}$$

For the general case, the linearization is given as follows:

$$\nabla(F_{x,z} - G_{x,z}) = \left(\frac{\partial F_{x,z}}{\partial u_1} - \frac{\partial G_{x,z}}{\partial u_1}, \frac{\partial F_{x,z}}{\partial u_2} - \frac{\partial G_{x,z}}{\partial u_2}, ..., \frac{\partial F_{x,z}}{\partial u_m} - \frac{\partial G_{x,z}}{\partial u_m} \right) \tag{3.63}$$

$$\begin{aligned} F_{x,z}^k + G_{x,z}^k &= \nabla(F_{x,z} - G_{x,z})^k \cdot \Delta \mathbf{u} && (3.64) \\ l &= \mathbf{M} \cdot \Delta \mathbf{u} && (3.65) \end{aligned}$$

For $n = m$, the solution is $\Delta \mathbf{u} = \mathbf{M}^{-1} \cdot l$. For $n > m$ observations, the following equation has to be solved: $v = \mathbf{M} \cdot \Delta \hat{\mathbf{u}} - l$. By applying the method of least-square error minimization, the minimum error is defined as follows:

$$min = v^T v = (\mathbf{M} \cdot \Delta \hat{\mathbf{u}} - l)^T (\mathbf{M} \cdot \Delta \hat{\mathbf{u}} - l) = \Delta \hat{\mathbf{u}}^T \mathbf{M}^T \mathbf{M} \cdot \Delta \hat{\mathbf{u}} - 2 l^T \mathbf{M} \cdot \Delta \hat{\mathbf{u}} + l^T l$$

We obtain

$$\frac{\partial \left(v^T v \right)}{\partial \Delta \hat{\mathbf{u}}} = 2 \Delta \hat{\mathbf{u}}^T \mathbf{M}^T \mathbf{M} - 2 l^T \mathbf{M} = 0$$

which leads to the following solution:

$$\Delta \hat{\mathbf{u}} = \left(\mathbf{M}^T \mathbf{M} \right)^{-1} \mathbf{M}^T l \tag{3.66}$$

A minimum is found if the unknowns do not change significantly anymore (e.g., $\sum_{i=0}^{m} |\Delta \mathbf{u}_i| < \epsilon$, with $\epsilon = 10^{-9}$).

For an approximation of the start values $\Delta \mathbf{r}_0^0$ (approximated camera location) and the

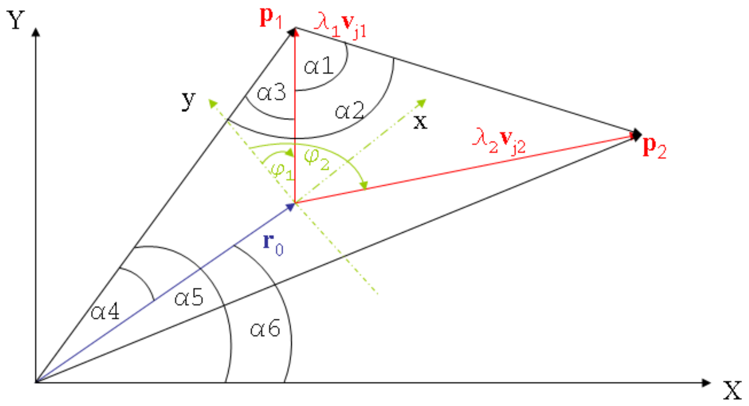


Figure 3.20: Geometric dependencies to approximate the start values for the least square approach.

elements of the rotation matrices $\mathbf{R}$, two control points are sufficient, if the camera is aligned nearly horizontally. In practice this is the normal case. The elements of $\mathbf{R}$, excepted the start value $\Delta\phi^0$ for the rotation about the z-axis, are small and can be set to zero. Figure 3.20 depicts the geometric dependencies of the projected control points into the xy-plane. The control points $\mathbf{p}_1$ and $\mathbf{p}_2$ are known, therefore the vectors $\mathbf{r}_1$ and $\mathbf{r}_2$ are given in the reference system. The column indices i_1 and i_2 of the panoramic image are given in the local camera system, where $\varphi_2 - \varphi_1$ is the angle between $\mathbf{p}_1$ and $\mathbf{p}_2$. The vector lengths $|\lambda_1\mathbf{v}_{j1}|$ and $|\lambda_2\mathbf{v}_{j2}|$ are given as well. Because of the law of sines, the following condition is given:

$$\frac{|\mathbf{r}_1 - \mathbf{r}_2|}{\sin(\varphi_2 - \varphi_1)} = \frac{|\lambda_2\mathbf{v}_{j2}|}{\sin\alpha_1} \tag{3.67}$$

Therefore,

$$\alpha_1 = \arcsin\left(\frac{|\lambda_2\mathbf{v}_{j2}| \cdot \sin(\varphi_2 - \varphi_1)}{|\mathbf{r}_1 - \mathbf{r}_2|}\right) \tag{3.68}$$

and $\alpha_3 = \alpha_2 - \alpha_1$, with the scalar product of the vectors:

$$\mathbf{r}_1 \cdot (\mathbf{r}_2 - \mathbf{r}_1) = |\mathbf{r}_1| \cdot |\mathbf{r}_2 - \mathbf{r}_1| \cdot \cos(\alpha_2) \tag{3.69}$$

$$\alpha_2 = \arccos\left(\frac{\mathbf{r}_1 \cdot (\mathbf{r}_2 - \mathbf{r}_1)}{|\mathbf{r}_1| \cdot |\mathbf{r}_2 - \mathbf{r}_1|}\right) \tag{3.70}$$

Because of the law of cosines, $|\mathbf{r}_0|$ can be estimated as follows:

$$|\mathbf{r}_0| = \sqrt{|\mathbf{r}_1|^2 + |\lambda_1 \mathbf{v}_{j1}|^2 - 2|\mathbf{r}_1| \cdot |\lambda_1 \mathbf{v}_{j1}| \cdot \cos \alpha_3} \tag{3.71}$$

and α_4 with the law of sines:

$$\alpha_4 = \arcsin \left(\frac{|\lambda_1 \mathbf{v}_{j1}| \cdot \sin \alpha_3}{|\mathbf{r}_0|} \right) \tag{3.72}$$

$$\alpha_5 = \arctan \left(\frac{\mathbf{r}_{y1}}{\mathbf{r}_{x1}} \right) \tag{3.73}$$

Therefore, the requested angle α_6 is given and $\mathbf{r}_0$ can estimated as follows:

$$\mathbf{r}_{x0} = \cos \alpha_6 \cdot |\mathbf{r}_0| \tag{3.74}$$

and

$$\mathbf{r}_{y0} = \sin \alpha_6 \cdot |\mathbf{r}_0| \tag{3.75}$$

For testing the algorithm, we used the calibration courtyard at the Institute for Photogrammetry of the University of Applied Sciences Berlin (TFH). The control points along the building are measured with a theodolite (see Figure 4.5), and are photogrammetrically balanced. The deviation of our calculated points to the given control points is shown in Table 3.8. Further experiments confirmed that the estimation of the parameters of the interior attitude is stable and does not change significantly by changing the parameter of the exterior attitude, and therefore these parameters are properly separated from each other.

To ensure this separation, a *bundle adjustment* balanced calculation is applied in photogrammetry. The idea is using a set (called "bundle") of images, taken from different positions. The exterior parameters change but not the interior parameters, thus the dependency between these parameters is minimized. The introduced calibration approach actually does include such bundle adjustment because of capturing lines at different positions ($R > 0$). Furthermore 360° panoramas stabilize the calibration.

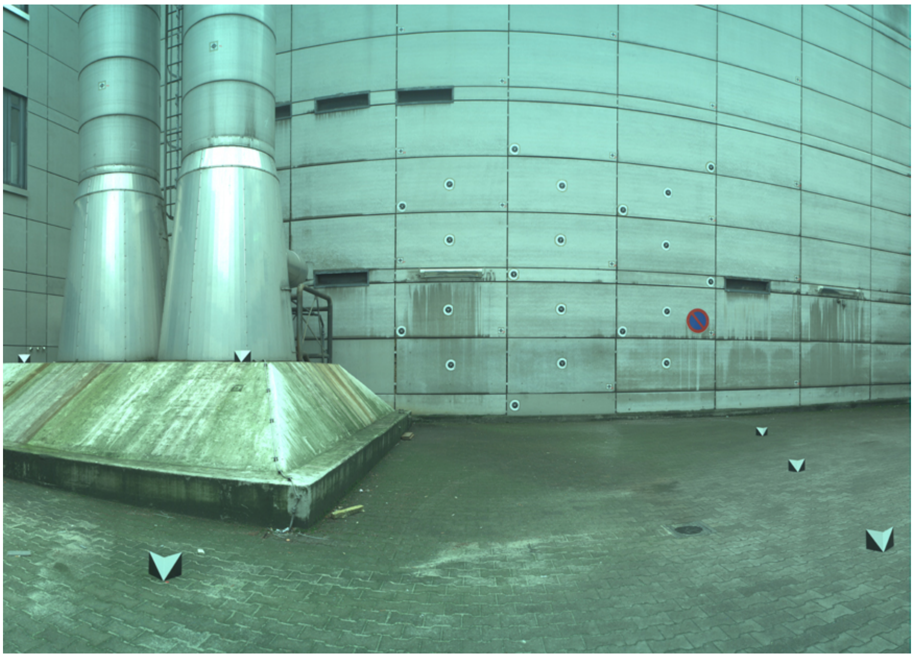


Figure 3.21: Calibration courtyard at the Institute for Photogrammetry at TFH Berlin.

Control Point	Error in Pixel (i)	Error in Pixel (j)
7	0.38	0.03
9	0.09	-0.44
10	0.09	0.08
11	-0.23	0.54
12	-0.17	0.28
14	-0.16	0.05
15	-0.14	-0.37
16	-0.27	-0.35
17	0.26	-0.01
19	0.22	0.30
50	0.01	-0.12
51	-0.09	0.11

Table 3.3: Result of panoramic adjustment. The table shows the deviation of calculated image coordinates and its reference coordinates.

4 Laser Rangefinder

A *laser rangefinder* (LRF) or *laser scanner* determines distances to opaque objects; it is also known as LIDAR (Laser Imaging Detection and Ranging). Such a device determines the distance to an object or surface using laser pulses (similar to radar technology, which uses radio waves instead of light). Applications of laser scanning are, for example, in the building industry, geology, seismology, remote sensing, atmospheric physics and cultural heritage. This thesis uses an LRF for a dense scan of surfaces rather than for measuring isolated distances.

4.1 General Principles

For laser scanners three measured principles are prevailed. A laser scanner is based on the *time of flight* principle: 3D coordinates of a surface point are derived by measuring the time delay between a transmission of a pulse and the detection of the reflected signal, and the direction of the transmitted pulse. Such time-of-flight systems allows unambiguous measurements of distances up to several hundreds of meters. Dense surface scans (to measure distances to thousands or million of surface points) can take up to several hours.

Besides the time-of-flight principle, the *phase measurement principle* is also applied for medium ranges. The phase shift of a modulated wave is measured when transmitting and receiving this wave. The phase shift depends on the distance. In combination with the time-of-flight technique, uniqueness (note that phase repeats at 2π) is ensured. High acquisition rates and high densities of measured 3D surface points are supported by phase shift systems.

Measuring System	Range [m]	Accuracy [mm]	Scan Rate [point/sec]
Time of flight	<1500	<20	up to 12,000
Phase shift	<100	<10	up to 625,000
Triangulation	some meters	<0.1	up to 10,000

Table 4.1: List of survey examples, classified according to their measurement characteristics (for data in time-of-flight and phase-shift, see [Sgr05]).

A third laser- based approach is similar to the technique of structured light. The *triangulation laser scanner* technique uses a point laser to scan the 3D scene. This technique samples a surface point by observing the reflected laser dot with a camera and determines its position by triangulation (see [RK98]). The surface point distance is determined by projection angle between surface point and camera, and the known distance and angle between camera and laser transmitter. Instead of a single laser dot, a laser stripe is used to accelerate the acquisition process. Very high accuracy (i.e., of some micro meters), but a limited scan range of some meters are the main characteristics of a triangulation laser scanner. Table 4.1 briefly summaries the differences between time-of-flight, phase-shift, and triangulation- based systems, related to accuracy, range, and scan rate.

Table 4.2 lists some LRFs and their advantages and disadvantages. Figure 4.1 illustrates conventional laser rangefinders with respect to their deviations in range measurements. In this simple test, a planar surface was scanned at varying distances to the LRF. The resulting deviation of measured range data are a reliable source of information for

System	Advantages	Disadvantages
Callidus	Very large FOV.	Very coarse vertical resolution (0.25).
Cyrax2500	Good accuracy.	Small scanning window (40 x 40).
S25	Very high accuracy for short ranges	Does not work in sunlight. Not suited for long ranges.
GS100	Large FOV.	Large noise.
Riegl Z210	High ranges possible. Large FOV.	Low accuracy.
Riegl Z420i	Very high ranges possible. Large FOV.	Large noise
Z+F	Very high scanning speed. Large FOV.	Low edge quality. Limited angular resolution (0.018).

Table 4.2: List of some LRFs with their advantages and disadvantages (based on [BVM03]). FOV stands short for “field of view”.

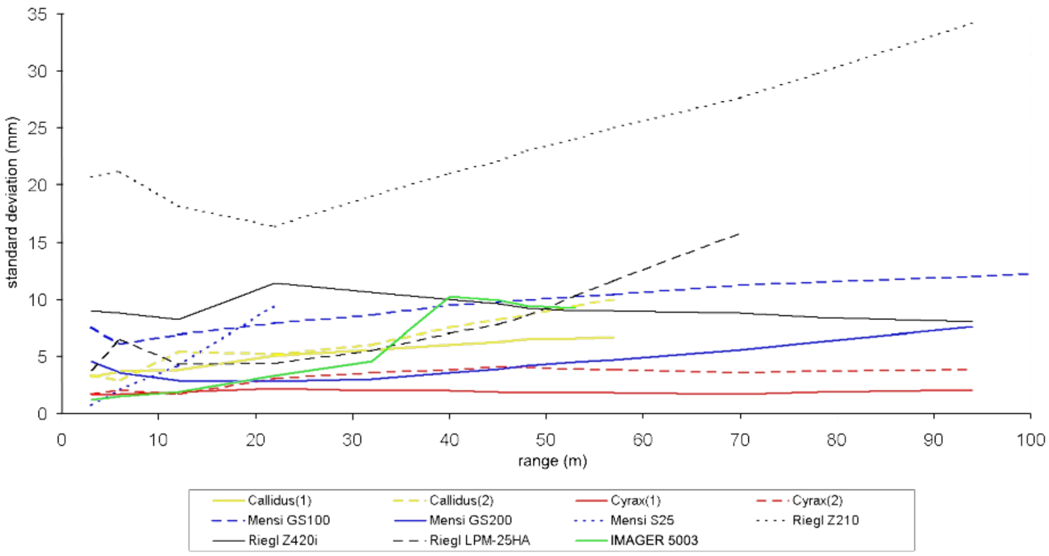


Figure 4.1: Standard deviations of range data at distances between 2 and 100 meters (from [BVM03]).

the relative accuracy of range measurements. These measurements were repeated and averaged for three different surface paints of the used planar surface with intensity of about 80%, 50% or 8% (note that 100% is "totally white" and 0% is "totally black").

4.2 Laser Scanner IMAGER 5003

For our project we used the laser scanner IMAGER 5003 (see Figure 4.2). This system is based on the phase shift measurement principle [FHF01]. The LRF scans, point by point, are uniformly defined in two dimensions, vertically by a rotating deflecting mirror, and horizontally by rotating the whole measuring system. The vertical scan range of the IMAGER 5003 is 310° (which leaves 50° uncovered), and the horizontal scan range is 360°. Actually a horizontal scan range of 180° is sufficient to measure each visible 3D point once, because the LRF scans overhead. However, we always scan the full 360° horizontally, therefore we have each visible 3D point twice. The redundancy can be used for calibration. Figure 4.3 depicts a raw data set without redundancy (i.e., 310° times 180°), captured by the LRF shown in Figure 4.2.

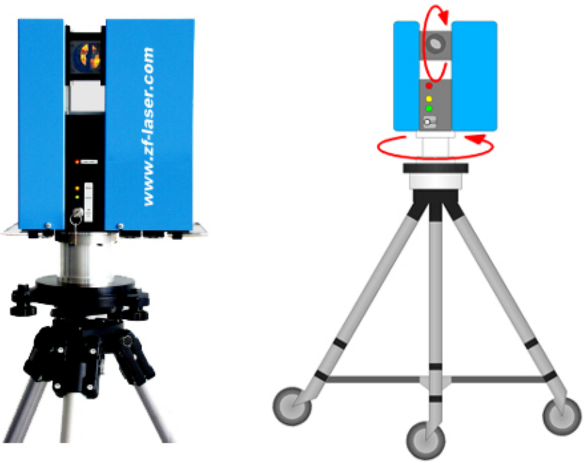


Figure 4.2: Left: IMAGER 5003 Zoller and Fröhlich with (from left to right) electronic module, optics (basically a golden mirror), and mechanic module (for motion control and transmission). Right: visualization of both scan direction (vertical and horizontal).

Note that 3D points, calculated at a single attitude of an LRF, are inaccurate, and we assume an error $\varepsilon > 0$ describing a sphere $U_\varepsilon(\mathbf{p})$ around a measured 3D point $\mathbf{p}$. The correct 3D point is in this sphere. Practically, we can assume that ε is specified by several components such as:

- a distance error (caused by the LRF measurement unit),
- the eccentricity of the scan center,
- the incident angle (angle between the laser ray and the surface),
- different surface material properties, such as of wood or metal,
- a collimation axis error (error of the principle axis),
- a vertical or horizontal axis error,
- the trunnion axis error (i.e., the oscillation around axes),
- and a scale factor.

We measured errors of distances on a dark gray, planar surface (with 20% of maximum intensity), where scans hit this surface about in normal direction (i.e., perpendicular to the surface's tangential plane). In related work, [SI04] provided a detailed analysis of

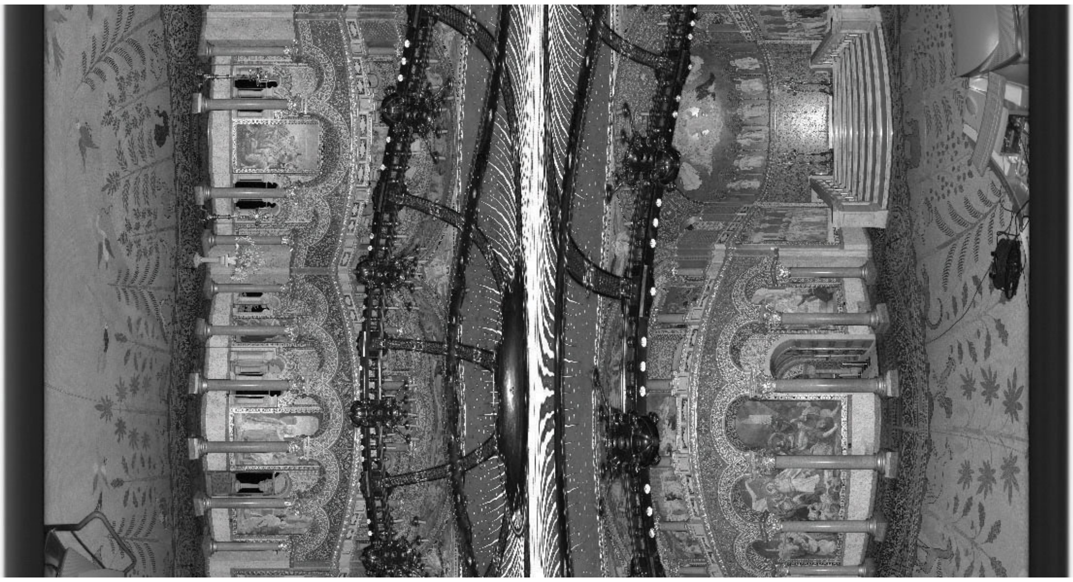


Figure 4.3: Raw data (without redundancy) of an uncalibrated LRF scan. The uncalibrated image geometry is caused by motions of the mirror or the LRF (one row corresponds to one mirror rotation).

accuracy of LRF distances by using an interferometer.[1] This allows measurements within an accuracy of a thousandth of a millimeter (i.e., of one micron). These distances are a thousand times more accurate than the measured distances of the laser scanner. Thus, they can be treated as *nominal distances*. LRF distance errors are illustrated in Figure 4.4. The standard deviation and the differences between nominal distances and measured distances (mean distance of 1,000 measurements) are shown in this figure; it can be seen that absolute errors are about five millimeters or less, increase towards a range about 20 meters and then decrease again slightly. Some systematic corrections (e.g., addition of a constant of approximately five millimeters) or a negative scaling factor (for distances $>$ 20 meters), can be used to be specified by calibration. The distance accuracy of the laser scanner only influences the longitudinal direction.

Photogrammetry specifies ways for calibrating rotational measuring devices (e.g., theodolite systems), how to measure errors along rotation axes. These errors are classified into *vertical, horizontal*, or *principle collimation errors*, and *scale factors*. Figure 4.5 illustrates the principle Z-axis as an axis orthogonal both to the corresponding vertical rotation

[1] This is an instrument that uses the principle of interference of electromagnetic waves for purposes of measurement. Note that an interferometer may be used to measure a variety of physical variables, such as distance, temperature, pressure, and strain, but cannot replace an LRF because it calculates distance to surface patches rather than to surface points.

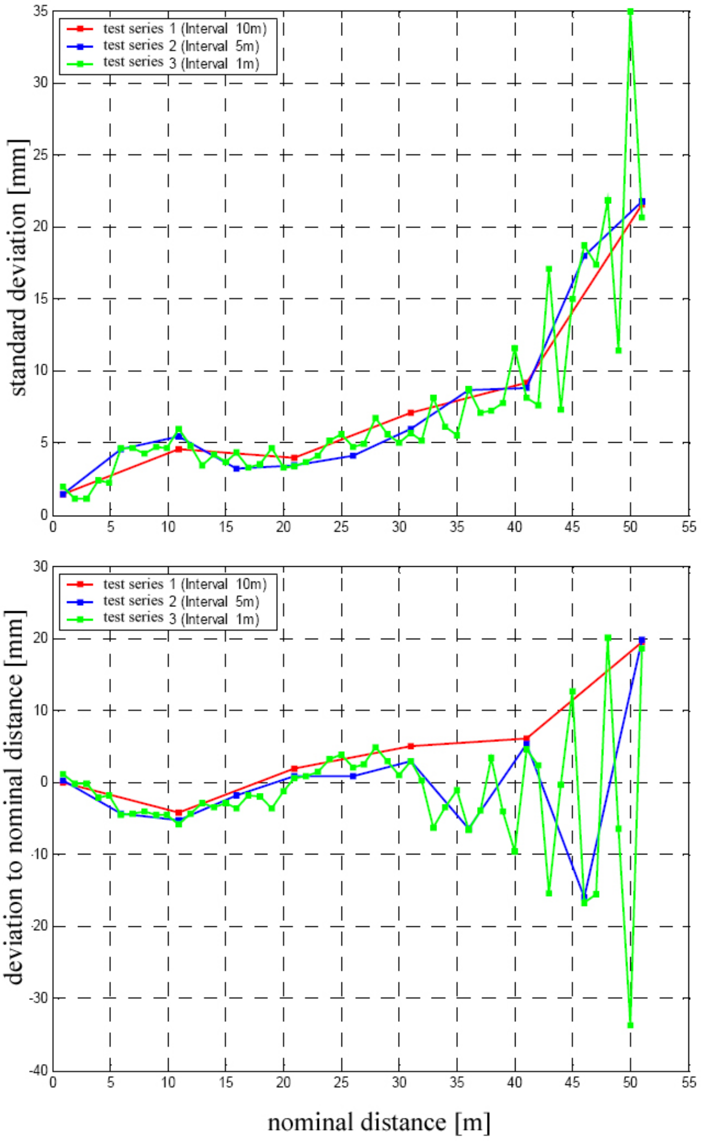


Figure 4.4: Top: Standard deviation, Bottom: Deviations between minimum and maximum measured distances (mean values of thousand measurements) and nominal distances. [SI04]

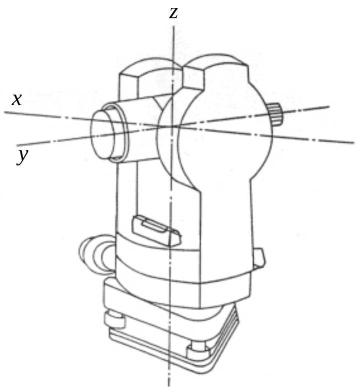


Figure 4.5: Theodolite with three axes: the (German) terms 'Zielachse', 'Kippachse' and 'Stehachse' specify in photogrammetry the principle y-axis and an orthogonal x-axis and z-axis. A rangefinder measures along a variable y-axis, which may be effected by horizontal (i.e., along the x-axis) or vertical (i.e., along the z-axis) errors.

axis V and to horizontal rotation axis (the *tilt-axis*) K.

Note that we have to consider the elimination order of the errors because of interdependencies between errors. To measure a 3D point twice at one LRF position means typically that a 3D point on a surface is first measured, then both rotation axes are turned by about $180°$, and the same surface point is measured again [DS02]. We use this redundancy (each visible point is scanned twice) for calibration.

To determine the *Zielachsfehler* (i.e., errors caused by the principle axis), 3D points are measured twice near the *equator* (i.e., the "horizon" of the LRF data set). The Zielachsfehler describes the deviation of the principle axis with respect to an ideal line, orthogonally to the horizontal axis.

For the determination of the *Kippachsfehler* (i.e., errors caused by the horizontal axis) the Zielachsfehler has to be known first. The Kippachsfehler describes an error between the vertical axis and the horizontal axis. Ideally, both axes have to be orthogonal. To determine the Kippachsfehler, a point near the *pole* has to be measured twice. The pole column indicates the "north pole" of a theodolite system. If the LRF is aligned horizontally then the pole column indicates the correct *zenit*, which is the point on the scanned surface exactly "on top" of the LRF (i.e., $\vartheta = 0$), and this point will be scanned repeatedly for any rotation of the LRF at this position. As an important test we have to confirm that the zenit is actually uniquely defined in 3D space for the whole combined scan of $360°$.

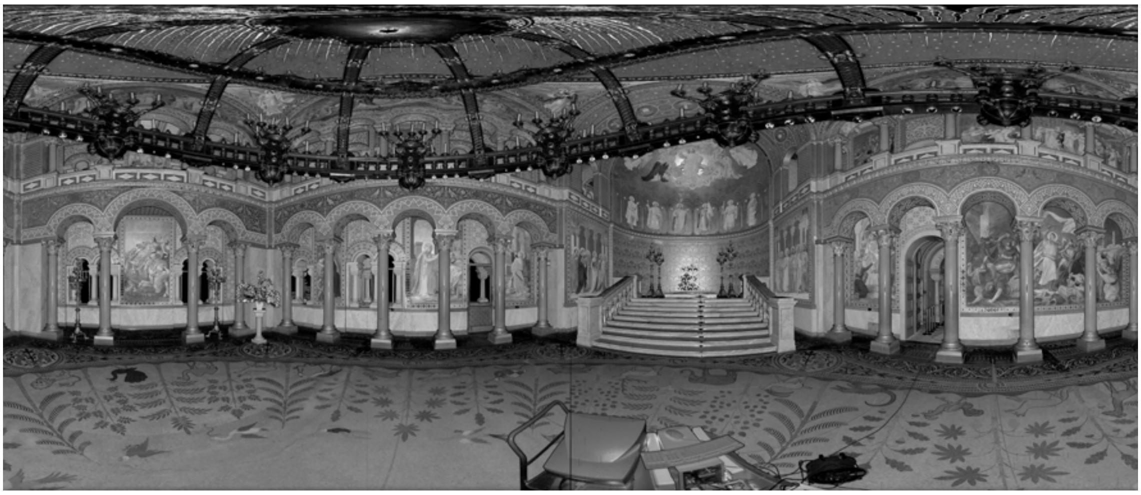


Figure 4.6: Data of a calibrated LRF image in spherical coordinates.

The scale factor needs to be determined for each scan. The scale factor provides the largest or most critical error for the used LRF (which is about 1 mm per 1 m distance). The scale factor depends on temperature changes and is determined via control points or corresponding points in different scans.

The collimation axis error is about 3 milli-degree whereas the horizontal axis error is about 30 milli-degree. Reasons for the larger value of the horizontal axis error lie in the obtainable accuracy and limits of leveling the laser scanner (precise levels and compensators are necessary and recommended for a precise leveling of the laser scanner). Figure 4.6 depicts a calibrated LRF scan (i.e., after removing errors as discussed). The described inaccuracies of the LRF can be due to errors of the distance measurement unit, or to mechanical errors. LRF measurements can be basically either reflector-less, or reflector-based (e.g., using a prism). Measuring with reflectors is much more accurate because of a unique returning ray. In case of measurements on natural or synthetic surfaces without reflectors or prisms, the intensity E of the returning ray is decisive for the achievable accuracy of distance measuring. Note that the intensity E is inversely proportional to the square of the distance s, formally expressed by $E \sim \frac{1}{s^2}$. The intensity is influenced by different parameters (e.g., the angle between laser ray and surface tangential; we call this the *incident angle*). Figure 4.7 illustrates dependencies between the incident angle and the accuracy. An optimal incident angle is orthogonal to the tangential plane at the surface point. Furthermore the intensity is influenced by properties of surface materials. Resulting errors are illustrated in Figure 4.8. In case of styrofoam materials, the laser ray

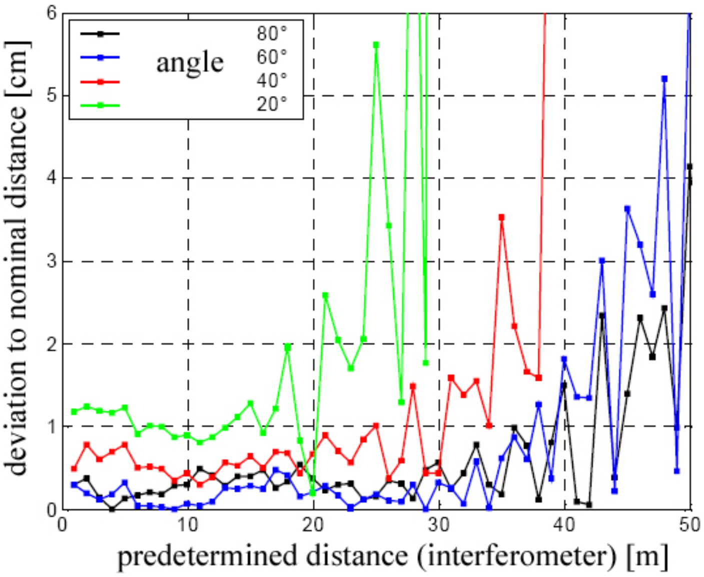


Figure 4.7: Dependencies between incident angle of the laser ray and the surface [SI04].

“sinks” into the surface. The returned intensity is to large for metallic surfaces due to specularities.

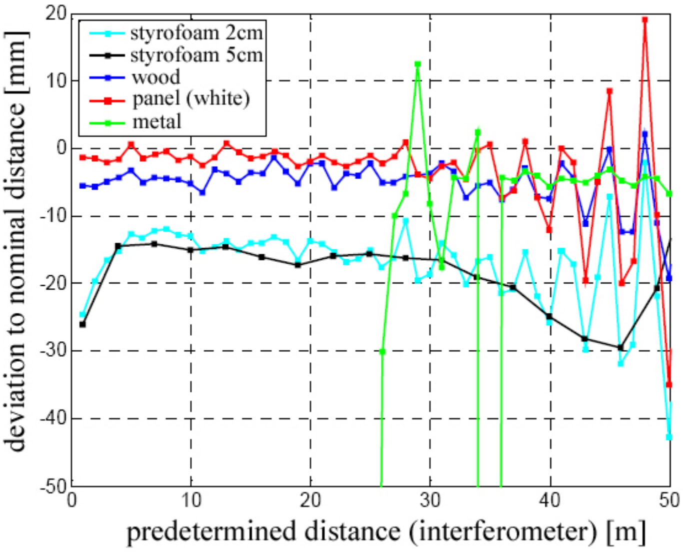


Figure 4.8: Analysis with respect to different surface materials [SI04].

4.3 Coordinate System of the LRF

We transform all LRF data at a given attitude into a polar coordinate system with an horizontal range of $360°$ and a vertical range of $180°$. At this step, all LRF calibration data are available and required. Each 3D point obtained with the LRF is described either in polar coordinates by the triple $(\lambda, \vartheta, \varphi)$, or in Cartesian coordinates as a vector **p**, which are related to one-another as follows:

$$\begin{aligned} p_x &= \lambda \cdot \sin\vartheta \cdot \cos\varphi \\ p_y &= \lambda \cdot \sin\vartheta \cdot \sin\varphi \\ p_z &= \lambda \cdot \cos\vartheta \end{aligned}$$

The laser scanner rotates clockwise. The first scan line starts at the positive y-axis in the LRF coordinate system at the horizontal angle of 0 degree. Figure 4.9 depicts the coordinate system of the LRF. The attitude of the LRF with respect to a reference vector **r** in the world coordinate system is defined by one rotation matrix $\mathbf{R} = \mathbf{R}_\Omega \cdot \mathbf{R}_\phi \cdot \mathbf{R}_\kappa$ and a translation vector $\mathbf{r}_0$:

$$\mathbf{r} = \mathbf{r}_0 + \mathbf{R} \cdot \mathbf{p} \tag{4.1}$$

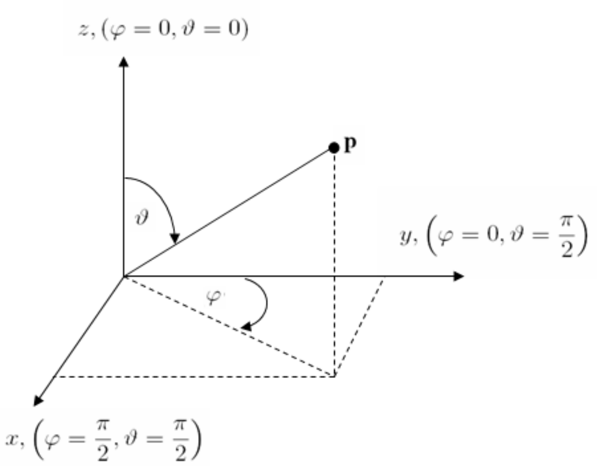

$z, (\varphi = 0, \vartheta = 0)$
p
ϑ
$y, \left(\varphi = 0, \vartheta = \frac{\pi}{2}\right)$
φ
$x, \left(\varphi = \frac{\pi}{2}, \vartheta = \frac{\pi}{2}\right)$

Figure 4.9: Rangefinder coordinate systems: the z-axis of Figure 4.5 points towards **p**, and is defined by slant ϑ and tilt φ.

5 3D Modeling and Visualization

So far we discussed data acquisition and the photogrammetric interpretation of captured data. This chapter deals with fundamentals in 3D modeling and its visualization as used in this PhD-Project. 3D computer graphics is more diverse as briefly reported in this chapter. This thesis is not about 3D computer graphics, and this chapter is not to be understood as an introduction into this topic. However, this chapter lists (with short comments) those techniques which are used for our data. A rather comprehensive treatment of 3D computer graphics can be found, for example, in [Wat02].

To visualize 3D objects (by identifying object coordinates) it is necessary to project each 3D point (i.e., in general a *voxel*) from the $\mathbb{R}^3$ space into a pixel on a 2D surface, that means into a point in $\mathbb{R}^2$. This is done by projections. Projections can be comfortably implemented with OpenGL (Open Graphic Library) [Gra06, RWL05]. Generally, OpenGL is an *Application Programming Interface* which allows the user to *render* simple 3D primitives (e.g., points, lines, polygons) into a selected *buffer*. Rendering in this context means the projection of each primitive under given geometric and radiometric conditions. The target is not necessarily the final “screen”; a buffer such as front-, back-, stencil- or z-buffer can also be selected. This allows the user the freedom of using such a buffer for creating visual effects (e.g., shadow calculations, blending, and so forth).

OpenGL is designed as a state machine. All primitives of a given list go through a graphic pipeline where their states are initialized by the user. In this way the user has the possibility to render “thousands” of primitives using the same conditions, then we can change the states of the machine or the states of modules within this pipeline, and call the next list of primitives. As an example, a state can be the color or material property with which the primitives should be rendered. The properties of the *virtual camera* (the camera through which we see the rendered scene) are defined by matrices, and are another example for a state of the pipeline.

An important module of this pipeline is lighting. This module can be manipulated by various states for reaching the wanted effect. The result depends on light properties (e.g., the ambient or diffuse part of light), the location of the light source, the material properties, the attitude of a primitive (i.e., position and the normal vector), the attitude of the virtual camera, and so forth. This part of the rendering process is called *shading*, and includes the calculation of an intensity or color which is returned by an illuminated surface, based on a *shading model* (e.g., Flat, Gouraud[1], or Phong[2]). These shading models are based on Lambert's cosine (emission) law and are extended for concerning also specularities. We implemented the Gouraud ("smooth") shading model because it is fast and has adequate visual results, but it does not provide varying lighting effects within a single polygon.

Note that shading procedures do not include complex volume- based algorithms (e.g., modeling reflections or shadows). For such "effects" the scene has to be raytraced, which has two meanings in this context. First, energy tracing from a light source through the scene, which actually also includes the second meaning, namely tracing for occluded objects from a ray's origin. The reason is the simplification of both approaches for realtime applications. Note that movie productions often use more realistic physics- based models (with necessary extensions), but these rendering engines do not (in general) support realtime visualizations.

We use, for example, Crow's approach for shadow calculations. This approach (to explain it in short) renders all object silhouettes in such a way into a stencil buffer such that it is masking the visible scene. This mask is then used to cast (blend) shadows into the scene. Further details can be found in [Cro77]. Figure 5.1 shows a rendered image using the Gouraud shading model and Crow's shadow algorithm.

Important information can be saved in arrays (e.g., vertex coordinates, vertex index lists defining how vertices are connected, normal vectors, or texture coordinates). Another possibility is using pre-compiled display lists. In such a list, all 3D objects are defined (or build by using OpenGL base functions such as glRotate, and so forth). Such a list is

[1] Invented by H. Gouraud in 1971 [Gou71]

[2] Invented by B. Phong in 1975 [Pho75]

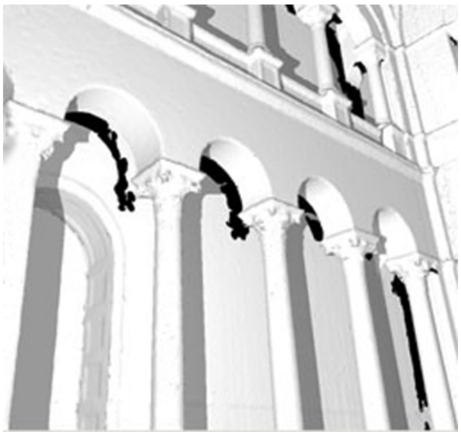


Figure 5.1: Rendered LRF data with casted shadows behind the pillars.

then compiled at once by starting the program, and this pre-compiled objects will only be called by changing a state of the pipeline (e.g., changing the camera view). We use vertex arrays because these allow the possibility of (geometry- based) dynamic scenes (e.g., opening or closing a window or door).

Transformations and projection parameters are stored in different types of matrices. The rendering pipeline multiplies all matrices into a transformation matrix, then it transforms each object coordinate by multiplying the current transformation matrix with the vector of the object coordinates. Different kinds of matrices can be stored in stacks to manipulate different objects by different matrices. The main transformation matrix $\mathbf{M}_T$ is given as follows:

$$\mathbf{M}_T = \mathbf{M}_V \cdot \mathbf{M}_N \cdot \mathbf{M}_P \cdot \mathbf{M}_M \tag{5.1}$$

$\mathbf{M}_V$ is the view port matrix, which is the transformation to the final window coordinates (window scaling and proportions), $\mathbf{M}_N$ is the normalization matrix (perspective division) of the device coordinates; $\mathbf{M}_P$ is the projection matrix, and $\mathbf{M}_M$ is the model matrix to transform the coordinates (e.g., a rotation, scaling, or translation).

We use the OpenGL low-level design[3], which requires a good knowledge about the graphic pipeline but gives also the possibility to render our scenes (characterized by a huge data size, both of LRF and texture data) in a reasonably time, and also provides the freedom to implement novel algorithms and data management routines.

[3] That means close to the kernel of this library.

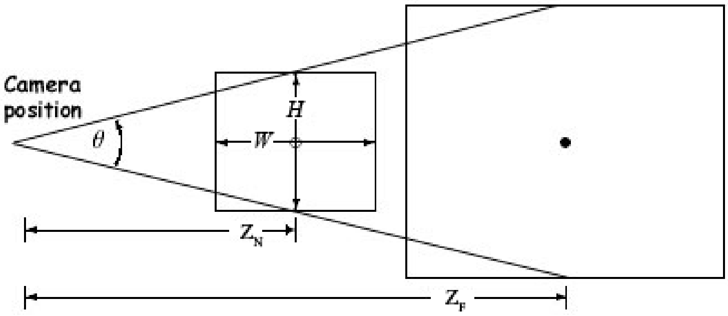


Figure 5.2: Central projection of objects in the range interval $[Z_N, Z_F]$ into a screen (or window) of size $W \times H$.

The following sections describe possible projections as applied to our data, and characterize or illustrate achieved results.

5.1 Central Projection

A central perspective projection considers the projection of an object or scene on a perspective plane. All rays, emerging at visible object surface points, go through the same point (i.e., the principal point of the virtual camera system). The actual homogeneous matrix for this projection is given in as follows:

$$\mathbf{M}_p = \begin{pmatrix} \cot\frac{\theta}{2}\cdot\frac{h}{w} & 0 & 0 & 0 \\ 0 & \cot\frac{\theta}{2} & 0 & 0 \\ 0 & 0 & \frac{Z_F+Z_N}{Z_F-Z_N} & -\frac{2\cdot Z_F\cdot Z_N}{Z_F-Z_N} \\ 0 & 0 & -1 & 0 \end{pmatrix} \tag{5.2}$$

Such a projection is illustrated in Figure 5.2 and stated in Equation (5.2). In Figure 5.2, the clipping planes are drawn as *Far* (we use the symbol Z_F) and *Near* (we use the symbol Z_N). The clipping planes can be seen as a bounding box which specifies the depth of the scene. Figure 5.3 depicts a 3D model rendered by central projection based on image data as shown in Figure 4.6. The figure shows a measured cloud of 3D points with gray levels (as measured by an LRF at a single attitude).

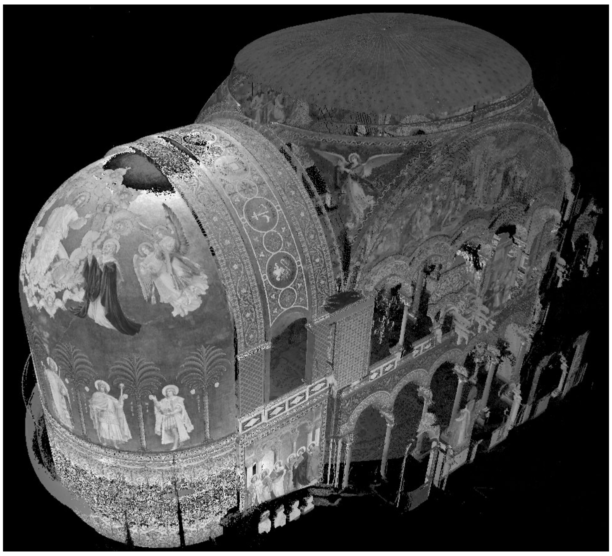


Figure 5.3: Central projection as measured by an LRF of the hall as shown in Figure 4.6.

5.2 Orthogonal Projection

An orthogonal projection considers the projection of each 3D point orthogonally to a specified plane. For each point $\mathbf{p} = (p_x, p_y, p_z)^T$, the transformed point would be:

$$\mathbf{p}^{'} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} p_x \\ p_y \\ p_z \end{pmatrix} = \begin{pmatrix} p_x \\ p_y \\ 0 \end{pmatrix} \tag{5.3}$$

In computer graphics it is common practice that matrices, used for orthographic projection, are defined by a 6-tuple (left, right, bottom, top, near, far), which defines the clipping planes. These planes form a cube with one corner at (left, bottom, near) and another corner at (right, top, far). The cube is translated so that its center is at the origin, then it is scaled to the unit cube which is defined by having one corner at (-1,-1,-1) and another corner at (1,1,1). The orthographic transformation can then be given by the following matrix:

$$\mathbf{M}_P = \begin{pmatrix} \frac{2}{R-L} & 0 & 0 & \frac{R+L}{R-L} \\ 0 & \frac{2}{T-B} & 0 & \frac{T+B}{T-B} \\ 0 & 0 & \frac{2}{F-N} & \frac{F+N}{F-N} \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{5.4}$$

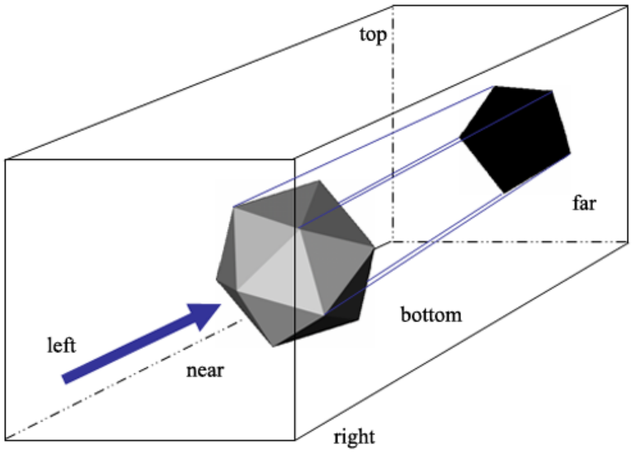


Figure 5.4: Orthogonal parallel projection: the screen (window) can be assumed at any intersection coplanar to the front (or back) side of the visualized cuboidal scene.

which is defined by the chosen values for F (far) and N (near), L (left) and R (right), and T (top) and B (bottom). Figure 5.4 and Equation (5.4) illustrate and represent the dependencies.

5.3 Stereo Projection

Model viewing can be modified by changing the matrix $\mathbf{M}_V$; this way the 3D object can rotate or translate in any direction. The virtual camera attitude can also be modified. It is possible to fly into the 3D scene, and to look around from any attitude within the scene.

Furthermore, it is possible to render more than one image from a virtual camera position in the same rendering context, and to create (e.g., anaglyphic) stereo pairs this way. There are different methods for setting up a virtual camera, and for rendering stereo pairs. Actually, many methods are basically incorrect since they introduces an "artificial" vertical parallax. As an example, we cite the *toe-in method*, see Figure 5.5 (left). Despite being

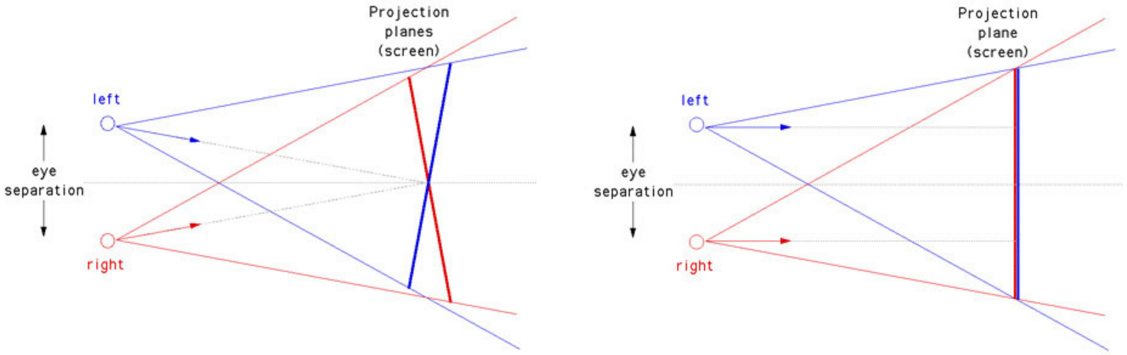


Figure 5.5: (Incorrect) toe-in [Bou] stereo projection (left), and correct stereo projection based on asymmetric camera frustums (right).

incorrect it is still often in use because a correct asymmetric frustum method requires features not always supported by rendering packages [Bou]. In the toe-in projection, both cameras have a fixed and symmetric aperture, and both cameras are directed into a single point. Images created using the toe-in method still allow a stereoscopic effect, but it introduces a vertical parallax which causes visual discomfort. The introduced vertical parallax increases with the distance to the center of the projection plane, and becomes more disturbing when the camera FOV increases.

The correct way to create stereo pairs is the *asymmetric frustum* method. It introduces no vertical parallax. It requires an asymmetric camera frustum, and this is supported by some rendering packages, in particular by OpenGL.

5.4 Modeling

In Section 5.1 we processed voxel data. After the projection of each voxel into the perspective plane, including all interpolation routines, the final result is produced. In computer graphics it is called *rasterization* because the projected voxels are sorted in a non-floating point raster (the final image). Figure 5.3 shows the measured 3D points with (LRF) gray levels. High point density makes the point cloud look like a surface, but single 3D points become visible when zooming in. This is the common way to visualize the captured data in medical applications. Special hardware is used to manage a large number of voxel.

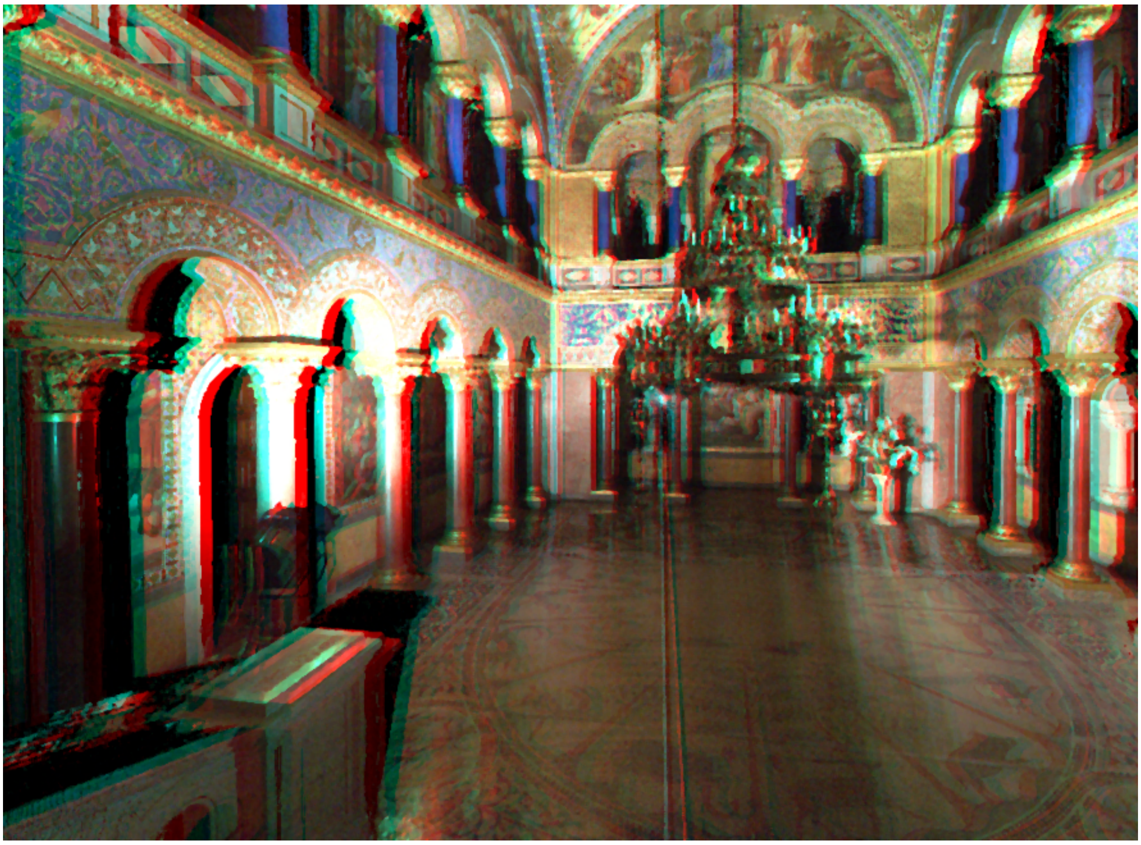


Figure 5.6: Correct stereo projection of the same hall as shown in Figure 7.1; the anaglyph uses red for the left eye.

Standard graphic adapters (with built-in 3D acceleration) support fast rendering of triangles. This includes the rasterization (as a part of the discussed graphic pipeline) of each triangle, the rendering of the triangulated surface (interpolation) and of triangle edges (anti-aliasing), depending on triangle properties (e.g., normal vectors, color, or texture) of each *vertex*[4] (i.e., a vertex of a triangle). Therefore, each given 3D point has to be triangulated into the triangle mesh.

Of course, it is possible to use other primitives as, for example, polygons. But, because of the need of unified primitives for hardware accelerated rendering, the first step of the pipeline would be the *tessellation* of these polygons into triangles. The fastest and data size saving way is rendering a scene by using “triangle strips”. They are defined by one triangle (the first), and each next 3D point defines a new triangle with the last two vertices of the triangle before.

[4] In computer graphics, a vertex is a corner point of a primitive.

The main challenge within 3D model generation is data reduction, namely to triangulate a given cloud of points by a reduced number of triangles or triangle stripes without loosing geometric resolution. This process is called *meshing*, and is separately discussed in the next chapter.

Creating a Digital Surface Model

A *digital surface model* (DSM) is a simplified 3D model. We call it 2.5D instead of 3D: Each voxel in $\mathbb{R}^3$ is projected orthogonally into an orthoplane; this orthoplane has two dimensions, like a normal image, but in addition each pixel has a value, which represents the orthogonal distance of a 3D point to the orthoplane. This distance is called *depth*, and the DSM is also called *depth map*.[5] Each pixel has just one depth value. Objects, which are occluding one-another, can not be represented in such a depth map.

For creating the DSM an orthoplane needs to be defined. This can be done by selecting three 3D points at least in a LRF scan, or by estimation of an adjusting plane in a point cloud (characterized by minimization of each point distance to this adjusting plane [described detailed in Section 6.4]). Figure 5.7 shows (left) a defined orthoplane “behind” a surface of interest.

The cross product of the vectors between the corner points of the orthoplane gives the normal vector **n** which is written as a quaternion as follows:

$$\bar{q}_n = (0, n_x, n_y, n_z)^T \tag{5.5}$$

To align the normal $\bar{q}_n$ of the orthoplane to the -*y*-axis of the reference system (as quaternion: $\bar{q}_y = (0, 0, -1, 0)^T$), the transformation $\bar{q}_t$ needs to be determined by the quaternions product as follows:

$$\bar{q}_t = \bar{q}_y \circ \bar{q}_n^* \tag{5.6}$$

Each LRF point **p** of all LRF scans transformed into the reference system using Equation 4.1 [written as quaternion $\bar{r} = (0, r_x, r_y, r_z)^T = (0, \mathbf{r}_v^T)^T$] is projected into the ortho-

[5] [RK98] (and other books in computer vision) make a difference between *depth* (= distance between camera and a 3D point) and *height* (= orthogonal distance between a 3D point and the orthoplane). We only refer one orthoplane and use “depth” for this general situation.

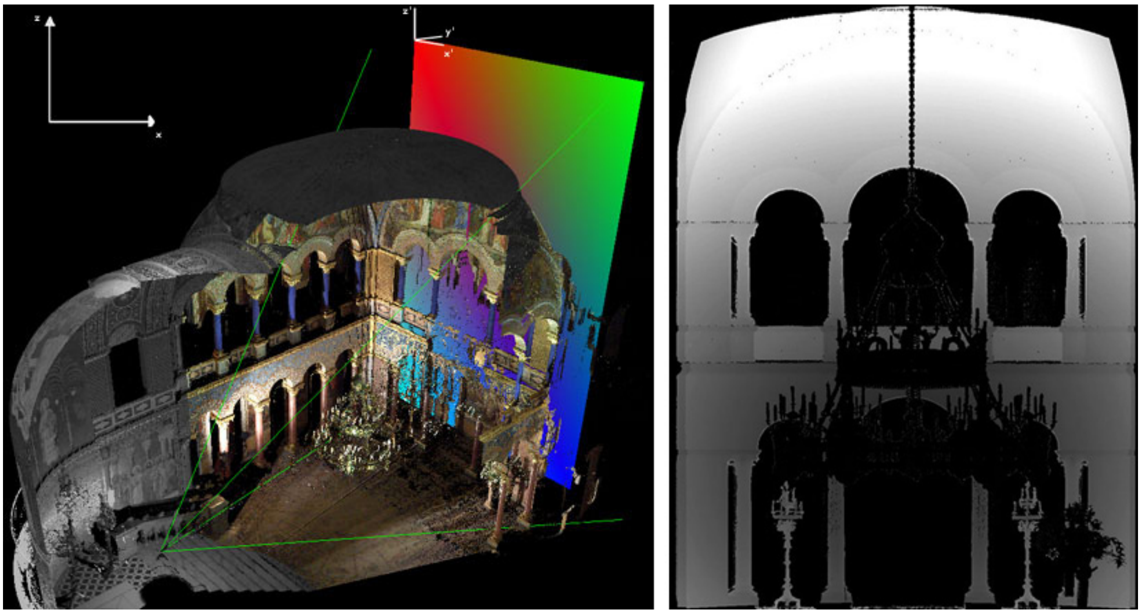


Figure 5.7: Creating a Digital Surface Model (depth map), left: A defined orthoplane "behind" the generated 3D data, and right: Gray-value encoded and orthogonally projected range data of those surface points which are in 2 meter distance to the defined plane.

plane by using following equations

$$\bar{r}\,' = \bar{q}_t \circ \bar{r} \circ \bar{q}_t{}^* \tag{5.7}$$

and

$$\text{DSM} = \begin{pmatrix} s_x & 0 & 0 & -c_x \\ 0 & s_y & 0 & c_y \\ 0 & 0 & s_z & -c_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} r'_{vx} \\ -r'_{vy} \\ r'_{vz} \\ 1 \end{pmatrix} \tag{5.8}$$

with the transformed upper left corner point **c** $= (c_x, c_y, c_z)^T$ of the aligned orthoplane for clipping the resulting image, and s_x, s_y and s_z as scaling factors to specify the resolution of the DSM. The resulting coordinates DSM$_x$, DSM$_z$ corresponds to the image coordinates i, j in this DSM-raster, and the DSM$_y$ -value is the "depth" (height=orthogonally distance of a 3D point to the defined orthoplane).

In a DSM, the data are rectified and perspective- free. This DSM is the base for further calculations (e.g., creating orthophoto). Figure 5.7 shows an example of a depth map. The image is gray-value encoded, darker values are closer to the surface. The big chandelier (in the middle of the room) is not projected into the depth map (i.e., the black values, because of using single scan), and this is an example of an occluded object. A maximum distance limit between orthoplane and surface is necessary for providing a clear depth map, without unwanted certainties (far objects are ignored this way).

Furthermore, always an object with the shortest distance to the reference plane is chosen by the algorithm (e.g., if using multiple scans for creating the DSM). This approach has the advantage that, for example, a painting on a wall is not overwritten by a second scan within an object which is near to this painting, but further away from the reference plane.

5.5 Using the Digital Surface Model for Bump Mapping

The meshing algorithm allows to set a level of detail. A planar surface with small structure (e.q., woodchip wallpaper, wrinkled surfaces) does not have to be included into the modeled geometry. In fact, if the distance of a 3D point to an approximated surface is smaller than a chosen threshold ε, the point will not be meshed. But, for example, an absolutely planar wall does not look realistically.

In 1973, Ed Catmull introduced the idea of using the parameter values of parametrically defined surfaces to index a texture definition function which scales the intensity of the reflected light. In [Bli78] the method of *bump mapping* is introduced, which simulates wrinkled surfaces. In computer graphics, bump mapping is a technique where at each pixel, a perturbation to the surface normal of the object being rendered is looked up in a bump map and applied before the illumination calculation is done. The result is a richer, more detailed surface representation that resembles more closely the details inherent in the natural world, without increasing the geometric complexity. Instead of calculating the perturbation of each pixel to its surface normal, we use the rectified DSM as a bump map. Figure 5.8 depicts a case where bump mapping is useful, and Figure 5.9 shows its result.

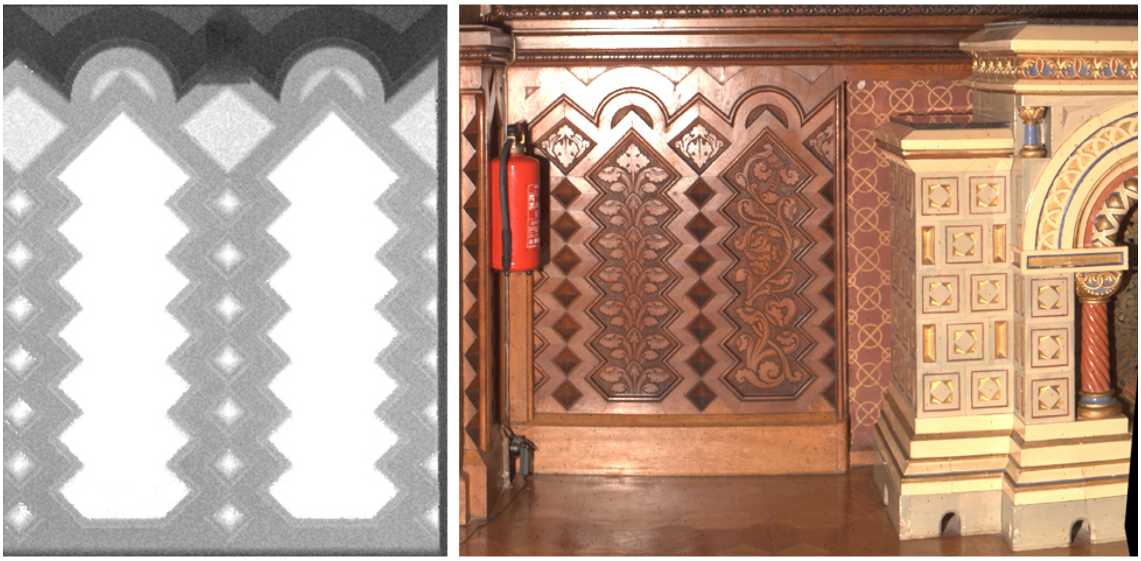


Figure 5.8: A Digital Surface Model (left) used for bump mapping for the timber panel (right) around the fire place.

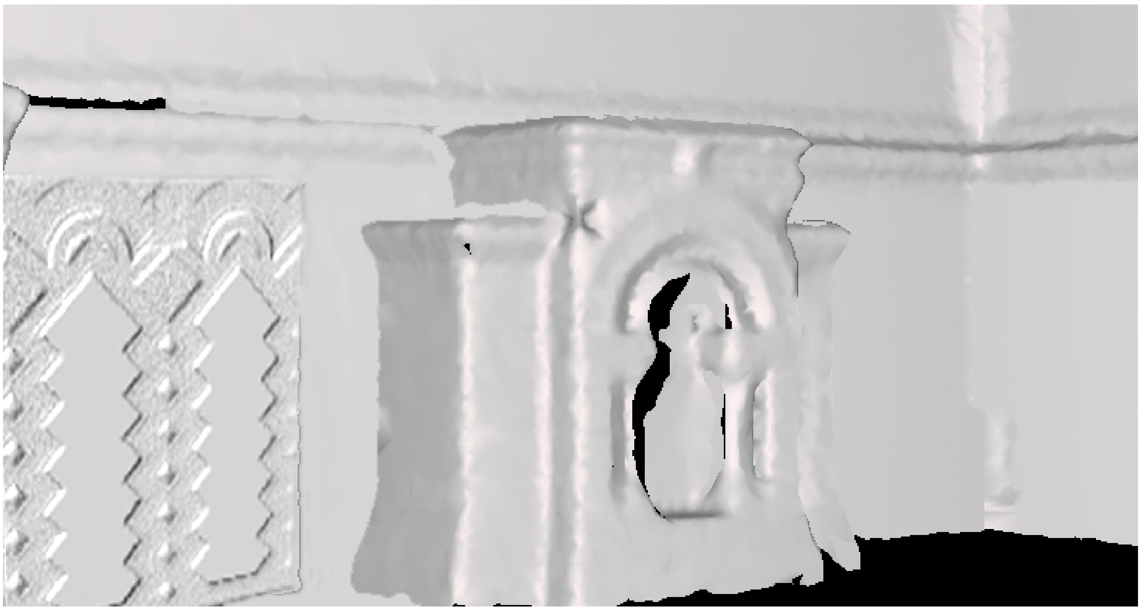


Figure 5.9: Shaded scene with bump mapping (left of fireplace) and without (right of fireplace).

6 Meshing and Optimizing

As an initial step toward generating a final composite mesh, it is first necessary to generate a dense triangular mesh. The triangulation of points in a *regular point cloud* (i.e., sorted points) poses little difficulty because of the known neighborhood relationship. The neighborhood relationship is defined by the movement of the LRF.

A calibrated LRF scan is rasterized in a 2.5D image, similar to a DSM, with dimensions $W \times H$, indexed with i, j. Unlike a DSM, the triangulated LRF scan is defined in polar coordinates. As typical in computer graphics, triangulation is determined counter-clockwise, whereby the relation is given by the image indices (i, j); $(i, j+1)$ and $(i+1, j)$ and so forth.

After triangulating all points it is necessary to separate objects from the foreground or background (or both) via two distinct filter procedures. In the first filter, the length of the triangle's sides are compared to the mesh resolution as given by the angular resolution $\Delta\varphi$ and $\Delta\vartheta$ of the LRF, and by the object's distance.

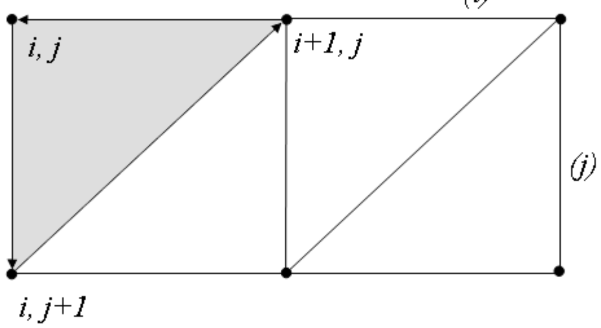


Figure 6.1: Simple triangulation of a regular point cloud.

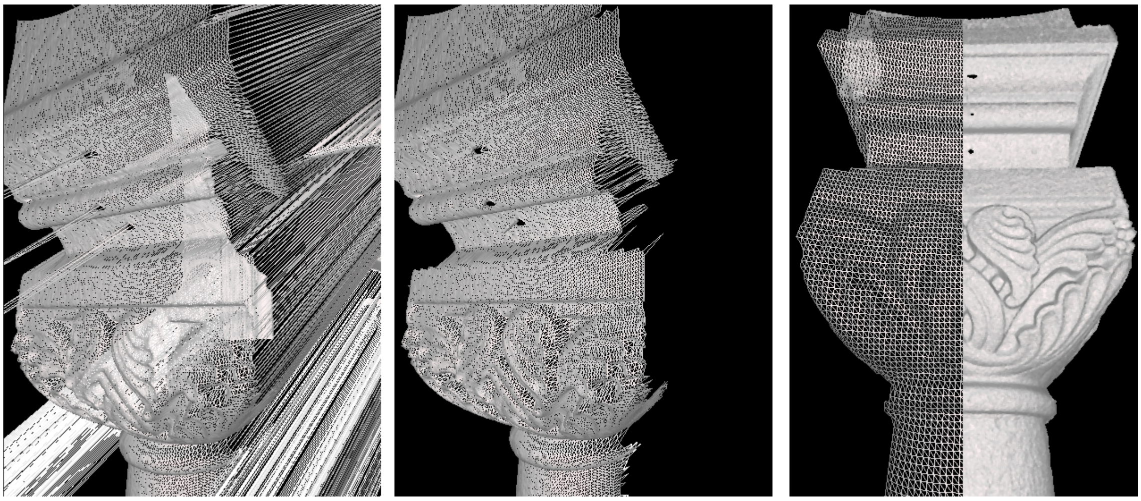


Figure 6.2: Left: triangulated regular point cloud, unfiltered; center: filtered mesh after applying first filter; right: final result as depicted half shaded and half meshed.

A second filter for removing incorrectly triangulated points is to check the side ratio; an optimal mesh should have equilateral sides. Figure 6.2 depicts these simple steps: The left image shows all triangulated points, unfiltered; the middle image the same sculpture after applying the first filter; and the right image the final mesh of the sculpture after applying both filters. Both filters are necessary to the final mesh. As depicted in the figure, the removal of points or triangles from the mesh creates holes. The detection and darning of such holes will be addressed in a separate section.

The discussion until now has addressed regular point clouds. When we combine different LRF scans to form one point cloud, however, we build an irregular point cloud. It is possible to build an irregular point cloud via a composite of multiple regular clouds, first meshing each LRF scan separately and then merging all meshes into one. This method, however, has the disadvantages first of excessive data overflow and secondly of the complex challenge merging edges and overlapping parts of the distinct meshes.

A preferable method is to build an initially dense mesh and to then insert points from different scans, point by point, into the mesh (e.g., using the algorithm proposed by Bodenmueller [BH04]). Originally, this approach was developed for online processing of unorganized data from hand-guided scanner systems (tactile sensors). Nonetheless, the method is also highly suitable for the processing of our LRF data, as it uses a sparse, dynamic data structure that can hold larger data sets. It is also able to generate a single mesh from multiple scans. The following procedure briefly lists the steps of triangulation:

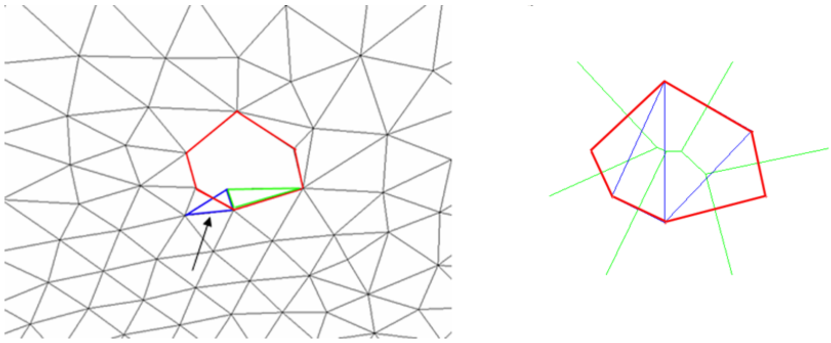


Figure 6.3: Left: triangulated point cloud (Delaunay) with a mesh hole in the middle. The arrow shows an edge which remains to be filtered (see text). Right: Delaunay triangulated hole. The green lines are the Voronoi diagram.

- thinning of 3D points (point density),
- normal approximation (local approximation of surface),
- 3D point addition (insertion of 3D points, dependent on normals and density),
- estimation of Euclidean neighborhood relations,
- neighborhood projection to tangent planes (i.e., from 3D to 2D points), and
- calculation of local Delaunay triangulations for the inserted points.

In mathematics and computational geometry, the Delaunay triangulation (as defined by B. Delaunay) for a set S of points in the Euclidean plane, the unique triangulation DT(S) of S such that no point in S is inside the circumcircle of any triangle in DT(S). DT(S) is the dual of the Voronoi diagram of S. Delaunay triangulations maximize the minimum angle of all angles in the triangulation's triangles. The triangulation was invented in 1934 by Boris Delaunay, and Voronoi diagrams are named after Georgi Woronoi. A typical Delaunay triangulation (characterized by regular triangles), applied to our LRF data, is illustrated in Figure 6.3. By virtue of estimating the surfaces, with resultant tangential projections, Delaunay triangulations are not immune to error. Such an error is depicted in the figure; the colored triangles shown here stand in three-dimensional space perpendicular to the mesh.

6.1 Basic Filters

In essence we differentiate two types of filters for optimizing the LRF data. Firstly, the noise of every 3D point in the point cloud is reduced by various filters (e.g., median filter, histogram filters, spike filters, building gaussian pyramids, etc.), before we turn to the meshing algorithm. Errors not caused by the system noise (e.g., errors caused by an unfavorable incident angle) and known object geometry are expected. They are fixed subsequent to generating the first initial dense mesh in a second filtering process. This is in fact the step of repairing the mesh.

The discussed error ε in the LRF chapter is determined on a standard dark gray surface. In practice we have specular highlights (e.g., on objects of gold metal). The error on such objects is much higher ($0 < \varepsilon < 2m$). The candlestick in Figure 6.4 is an example of such disturbances. To solve this problem we use a histogram filter, which automatically detects the first and last peak. All depth values before and after these peaks are made invalid and are then interpolated.

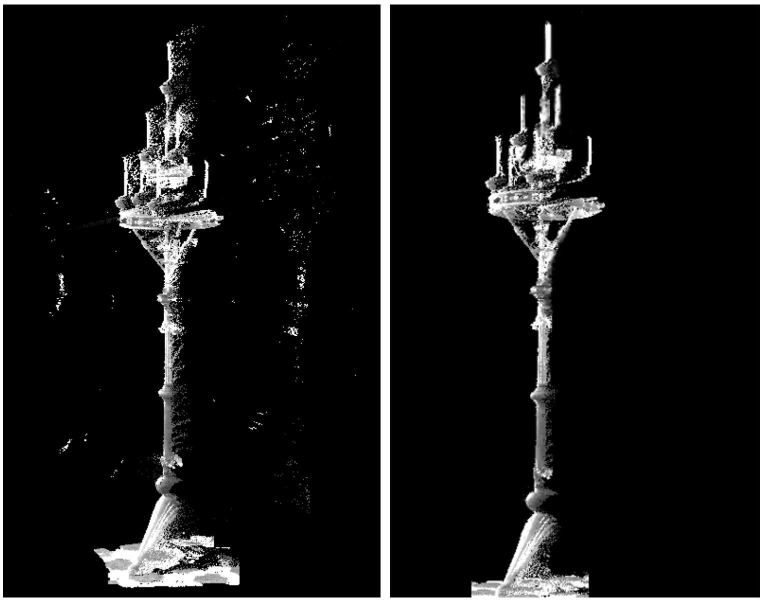
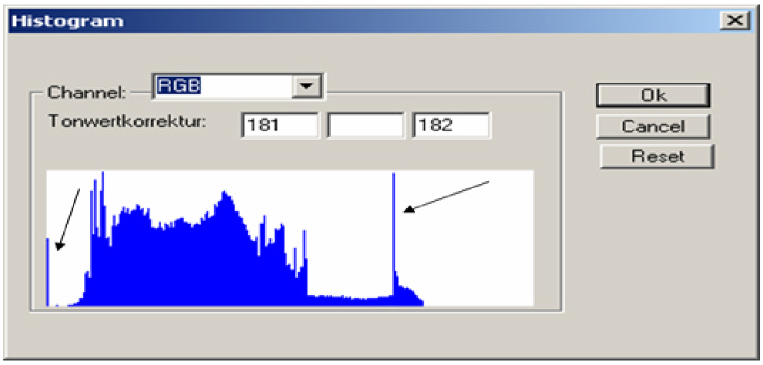


Figure 6.4: Using a histogram filter to optimize the LRF data. A candlestick with golden surface, which caused noisy LRF data (left), and the candlestick after applying a histogram filter.



Histogram

Channel: RGB

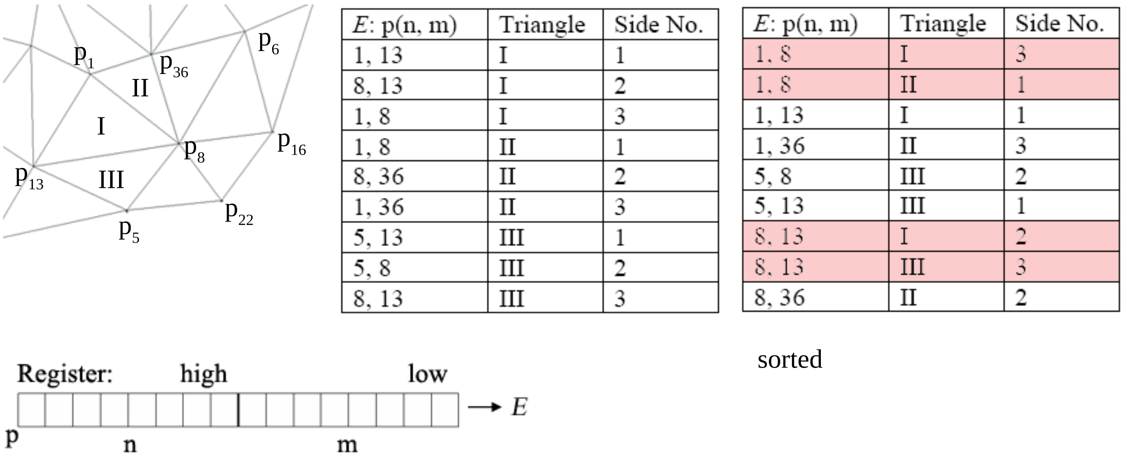
Tonwertkorrektur: 181 182

Figure 6.5: Used histogram to detect peaks for optimizing the LRF data.

6.2 Fast Connectivity

Connectivity is defined as the transitive closure of edge adjacency between polygons. If an edge E of a polygon is the same E of another polygon, then both polygons are connected to each other. An edge E is defined as two connected 3D points p_n and p_m, where n, m are the indices of these points. Each 3D point is unique in the point cloud and the meshed 3D model. Note that the point indices have no systematic order in the mesh. Each triangle has three edges; we count them counter clockwise and name them side one, side two and side three.

In computer graphics it is not necessary to improve provided algorithms for calculating



E: p(n, m)	Triangle	Side No.
1, 13	I	1
8, 13	I	2
1, 8	I	3
1, 8	II	1
8, 36	II	2
1, 36	II	3
5, 13	III	1
5, 8	III	2
8, 13	III	3

E: p(n, m)	Triangle	Side No.
1, 8	I	3
1, 8	II	1
1, 13	I	1
1, 36	II	3
5, 8	III	2
5, 13	III	1
8, 13	I	2
8, 13	III	3
8, 36	II	2

sorted

Figure 6.6: Fast connectivity calculation of triangles.

connect components, because models only have a feasible number of polygons. To check every edge of a polygon versus each edge of all the other polygons (by proper subdivision of search spaces) is straightforward but time-consuming. In our case we have many millions of polygons in just one LRF scan. The implementation of the common connectivity algorithm based on Gamasutra‘s article [Lee] leads to connected component detections, a process that lasts more than one hour.

One means of improving this processing speed is to hash the point indices to one edge index. Figure 6.6 illustrates such hashing of edges. Important is that (by the sorting of indices) the first column represents the smaller point index p_n. Let p_m be the larger index. Every pair p_n, p_m then has a unique address E by pushing the p_n value into the higher part of a register and p_m into the lower part. Now we can sort our structure (left table in Figure 6.6) by the first column E. The result is shown in the right table. One loop is sufficient to identify all dependencies. If row i and row $i + 1$ have the same E value, then the dependencies are directly given by the second and third column of our structure. For example, row one and two in Figure 6.6 must have the same E value, and the connectivity can be identified in columns two and three: triangle one, side three is connected to triangle two, side one.

Secondly, furthermore, the *point-triangle* connectivity is stored in a map within this loop. This means that the map contains for each point a list of triangles which are connected to this point. As compared to the hour or more before, using this algorithm, we needed only about 10 seconds.

Knowing these connections is an important element to most mesh based algorithms. If, for example, a polygon has only connected sides (completely connected), then it is a part inside a mesh. If more than one side of a polygon is unconnected, then it is a border polygon, which may signal a hole in the mesh. If no side of a polygon is connected, then this polygon is a single polygon, or it is just connected about one point of the polygon to the mesh. Such polygons are mostly caused by errors in the LRF scan, or by deleted points (basic filters), and they need to be filtered. We use the connectivity information to also accelerate the described shadow calculation (determination of border polygons).

Figure 6.3 is an example in which connectivity as a mesh-based filter (mesh repairing) is used. The blue and green marked triangles in this figure are invalid. The connectivity algorithm finds the marked edges in this figure by virtue of the fact that these are edges at which more than one triangle is connected. The green triangle is a simple case. Here the algorithm finds two triangles. One triangle is completely connected whereas the green triangle has one open side, wherefor the green triangle can be deleted. The blue triangle, however, connects to a triangle which also has one open side. The decision here of which triangle is the invalid one is insofar unclear.

After using a filter that deletes erroneous triangles, we then always use a second filter, which deletes all triangles that are connected on one or fewer sides. The first filter deletes the green triangle, the second the blue triangle. These filters are used iteratively and mostly more than once; this drawback we define as future work. The red marked polygon is a hole in the mesh and must be darned, to which we turn now.

6.3 Detection and Darning of Holes

In order to detect holes in the mesh we use the connectivity information. The algorithm begins with an arbitrary polygon in the mesh and checks wether the polygon is connected on all sides. If a polygon is found which does not fulfill this condition, the algorithm grows from this starting polygon to the next unconnected side from the polygon itself or the neighbor polygon (also determined by connectivity).

A hole is successfully detected when the growing algorithm reaches the start polygon. The hole size is defined by the number of sides of the triangles that describe this hole. The red polygon in Figure 6.3 therefore has a hole size of six. A threshold parameter aborts the growth of the algorithm if the hole size becomes too big, or if the polygons describe an exterior border of the mesh.

The determined edges together describe a new polygon which is projected onto a tangential plane. Therefore, the average normal vector is estimated by the normal vectors of all polygons surrounding the discovered hole. The projected polygon is then triangulated by the same Delaunay algorithm as described before.

The general concave problematic (i.e., connection of those boarder points, where its

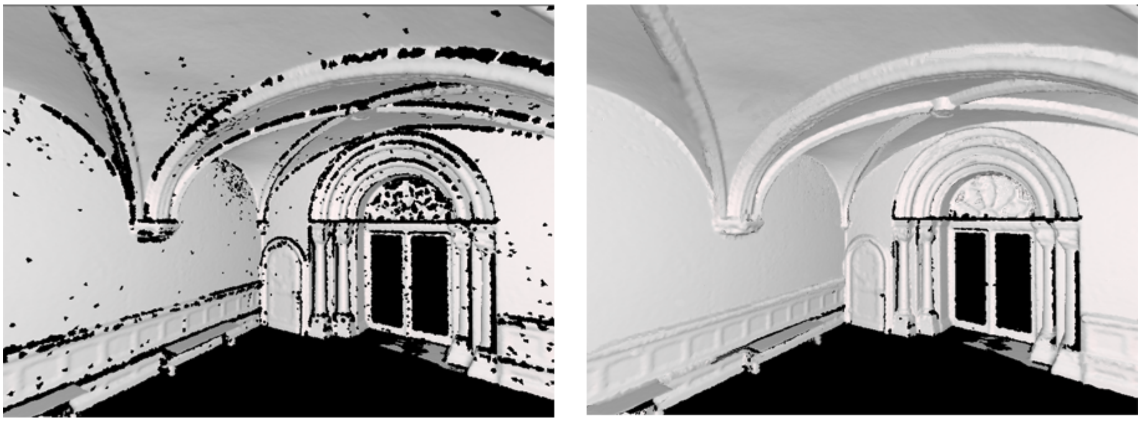


Figure 6.7: LRF scan with holes in the triangulated mesh (left), and the filled scan (right).

surface would not be inside the hole) arising through the triangulation of points is then solved by a simple trick: First all points are connected, whereby all resulting triangles are then checked for complete connectivity. Only those triangles with complete connectivity are inside of the polygon, which defines the hole. The connectivity calculation is here optimized, and considers only the new triangles from the triangulation and the triangles passed over during the hole detection procedure. Figure 6.7 shows a triangulated LRF scan with holes (left), and the filled mesh (right) using the described algorithm. Some holes are not detected, however, because the hole size is too big. In these cases, additional LRF scans should first be used to fill the larger holes. Of course, if we were to choose a bigger threshold parameter, the algorithm would fill also natural holes (e.g., windows).

6.4 Detection of Planar Surface Patches

The floor of the Thronsaal, especially in the area of the stairs, is a good example of where normal filters are not sufficient to fit planar surfaces. The reason here is that the floor is made of marble. With such a material, the LRF causes many errors as a result of the high shininess of the marble. For a case such as this, a filter that detects the average planar surface and projects each LRF point onto this approximated surface can be used. Figure 6.8 depicts such a case. The illustration depicts the profile of a floor, where $\mathbf{p}_i$ are the 3D points of the triangles. $\mathbf{s}_i$ are the main points of the triangles with its normal vectors $\mathbf{v}_i^\circ$. The average planar surface is given by the normal vector $\mathbf{n}^\circ$ and the distance

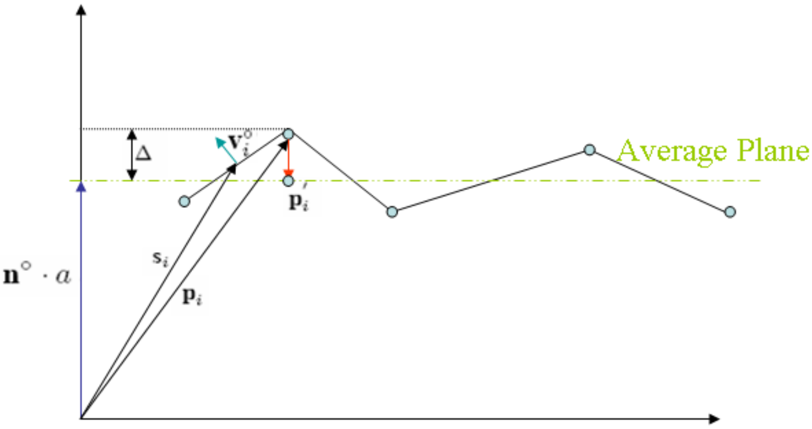


Figure 6.8: Geometric dependencies for the approximation of a planar surface.

a to the origin of the reference system. To determine $\mathbf{n}^\circ$, the average of all normal vectors $\mathbf{v}_i^\circ$ with the count of triangles k has to be calculated.

$$\mathbf{n} = \sum_{i=0}^{k-1} \mathbf{v}_i^\circ \quad \text{and} \quad \mathbf{n}^\circ = \frac{\mathbf{n}}{|\mathbf{n}|} \tag{6.1}$$

a is the average of the projections of each $\mathbf{s}_i$ onto the surface orientation $\mathbf{n}^\circ$ and is calculated by the scalar products:

$$a = \frac{1}{k} \sum_{i=0}^{k-1} (\mathbf{n}^\circ \cdot \mathbf{s}_i) \tag{6.2}$$

Finally each Δ_i, which is the distance of each 3D point $\mathbf{p}$ to the approximated average surface can be determined as follows:

$$\Delta_i = (\mathbf{p}_i \cdot \mathbf{n}^\circ) - a \tag{6.3}$$

Therefore each projected 3D point $\mathbf{p}_i^{'}$ is given with

$$\mathbf{p}_i^{'} = \mathbf{p}_i - \Delta_i \cdot \mathbf{n}^\circ \tag{6.4}$$

Note that this correction only adjusts the “z-values” along the normal vector of the approximated plane. Figure 6.9 (right) depicts the stairs after approximating average planar surfaces for each stair. The stairs are now flat and no longer have any disturbances.

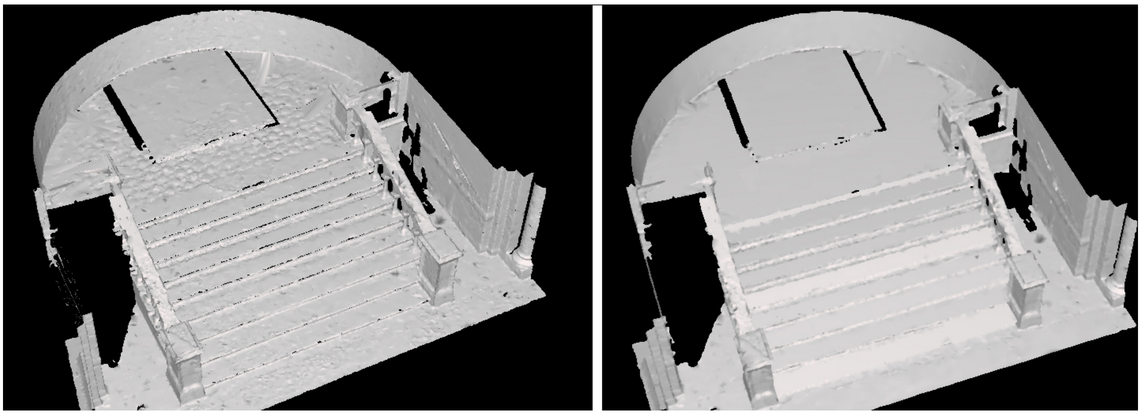


Figure 6.9: Shaded floor of the Thronsaal with errors caused by the reflection of the marble (left), and corrected floor by approximating planar average surfaces.

The pattern on the upper floor is caused by the marble structure as well. In reality the floor is absolutely flat, as seen after the correction shown in Figure 6.9 (b). In order to ensure that real objects do not become flat, a threshold parameter ϵ for a maximum distance Δ and a maximum angle $\angle(\mathbf{v}_i^\circ, \mathbf{n}^\circ)$ can be chosen.

For a correction of the edges, an orthogonal view to this plane is needed, which can be rendered from the panoramic image (see Section 6.5).

For the case we have not triangulated points, an equalization procedure by use of a covariance matrix is useful. By finding the eigenvalues and eigenvectors of the covariance matrix, we see that the eigenvectors with the smallest eigenvalues correspond to the dimensions that have the strongest correlation (smallest variance) in the point set.

$$cov_{xy} = \frac{1}{n-1}\left(\sum_{i=0}^{n-1}(x_i \cdot y_i) - n \cdot \bar{x} \cdot \bar{y}\right) \tag{6.5}$$

$$cov = \begin{pmatrix} cov_{xx} & cov_{xy} & cov_{xz} \\ cov_{yx} & cov_{yy} & cov_{yz} \\ cov_{zx} & cov_{zy} & cov_{zz} \end{pmatrix} \tag{6.6}$$

That implies that the eigenvector with the smallest eigenvalue is the normal vector of the approximated plane. Therefore the plane coefficients are given as $0 = A{\cdot}x{+}B{\cdot}y{+}C{\cdot}z{+}D$ with A, B, and C as the three components of this vector. The coefficient D is calculated as $D = -(A \cdot x + B \cdot y + C \cdot z)$.

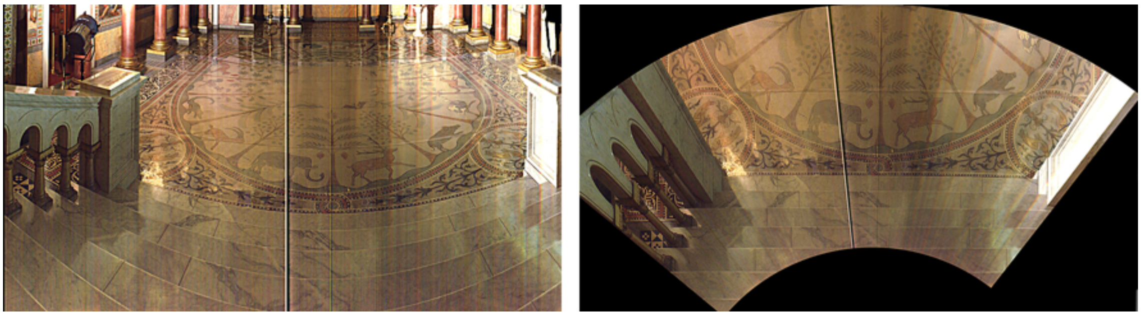


Figure 6.10: Panoramic input image with a given perspective (left), and orthogonal view of the same panoramic image.

6.5 Edge Estimation and Edge Correction

For the mechanical setup where the camera and the LRF have the same attitude, the correction of edges is straightforward. The image of the panoramic camera can be used to find edges and correct the LRF data, because both images have the same perspective.

In our case, where the camera and the LRF have different locations, the perspectives are different. For a correct co-registration of the data, however, a correct DSM has in fact already been needed. Errors that may be caused by the meshing of the LRF data are therefore already projected into the mapped panoramic image. A correct straight edge in the panoramic image will not be preserved after mapping the data if the DSM used is incorrect.

A perspective free panoramic image is therefore necessary to use for corrections. One possibility for overcoming this problem is to use approximated planar surfaces (see Section 6.4). After the estimation of such a surface, the panoramic image can be projected onto this plane. This projection is the orthogonal view of this plane. Figure 6.10 depicts this on the same marbled stairs. The left image is the original panoramic image, whereas the right image is the rectified, orthogonal image. Because all stairs are rendered onto the same plane (the floor), the upper stairs look larger than the lower stairs. In case we have more than one depth layer, we must project the panoramic image onto each layer (the approximate average plane) separately. Figure 6.11 depicts the panoramic image (blue, rendered from different layers for each stair) overlayed on the DSM. After this, each 3D point in the 3D model can be shifted along an edge.

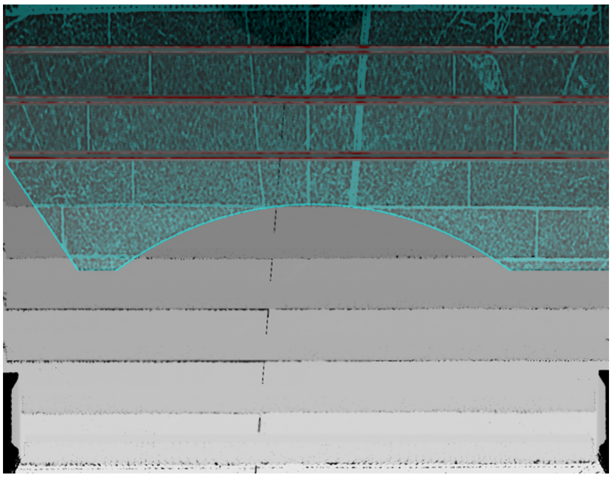


Figure 6.11: Overlayed rectified panoramic image (blue) to the DSM.

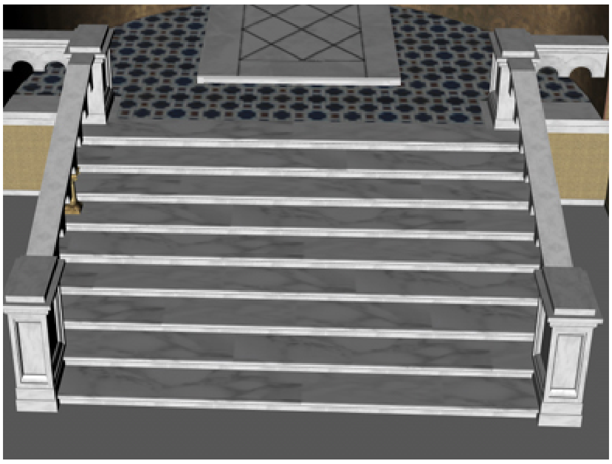


Figure 6.12: Corrected stairs.

The 3D points will only be moved in direction along the xy-plane of the used DSM. All this processing steps (i.e., calculation of the DSM, the rectified panorama related to the DSM, edge estimation) are done step by step with manually interaction. A full automatic pipeline, and other image-based algorithms we define as future work.

Figure 6.12 shows the corrected stairs in two stages, first with the adjusted z-values (by approximation of the average planar surface patches) and secondly with the corrected edges (through use of the panoramic image) in the ground plane.

7 Data Fusion

The fusion of the data sets starts with transforming coordinate systems (i.e., those of LRF and camera attitudes) into a uniform reference coordinate system (e.g., world coordinate system). For this step, the attitudes of both systems needs to be known (see Section 3.8). A transformation of LRF data into the reference coordinate system follows then Equation (4.1). The known 3D object points **r** (vertices of the triangulation) are now given by the LRF system and must be textured with color information of the panoramic image. Therefore, panoramic image coordinates related to each vertex must be determined, and this is discussed in the following section.

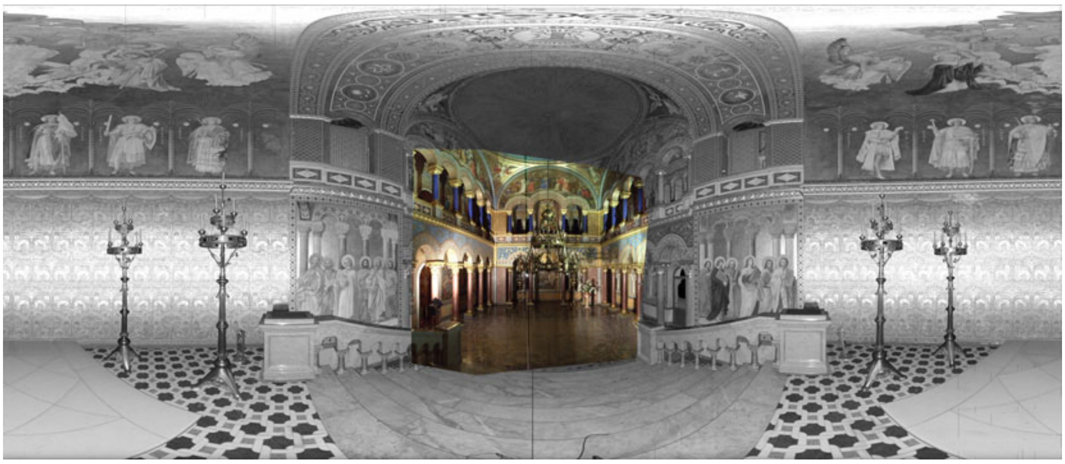


Figure 7.1: Panoramic image data have been fused in a subwindow near the center of the shown range image. (The figure shows the Thronsaal of castle Neuschwanstein.)

7.1 Determination of the Camera Image Coordinates

This section is organized into four subsections, each addressing one possible option of an “acquisition set-up”. As discussed in the introduction and when introducing the camera coordinate system, we differentiate between single-projection-center, multi-projection-center with $\omega = 0$, and multi-projection-center panoramas with $\omega \neq 0$ (typically used as a stereo set-up). The introduced camera allows to ensure such mechanic variations with sufficient accuracy. However, in the fourth subsection we also consider a general case, also allowing inaccuracies with respect to ideal assumptions.

At first we give some definitions, where we apply the determined exterior attitude parameters **R** and $\mathbf{r}_0$. This allows to define (in the reference system) a difference vector between object coordinate **r** and camera location $\mathbf{r}_0$, written as follows:

$$(\mathbf{r} - \mathbf{r}_0)^{'} = \mathbf{R}^{-1}(\mathbf{r} - \mathbf{r}_0)$$

In short, we also use the abbreviation

$$\tilde{\mathbf{r}} = (\mathbf{r} - \mathbf{r}_0)^{'}$$

By applying both rotations (i.e., **R** describing a rotation of the exterior attitude of the camera, and $\mathbf{R}_\varphi$ for the CCD line rotation) to the difference vector, we have the following:

$$(\mathbf{r} - \mathbf{r}_0)^{''} = \mathbf{R}_\varphi^{-1}\mathbf{R}^{-1}(\mathbf{r} - \mathbf{r}_0)$$

By applying the tilt parameters (i.e., $\mathbf{R}_x$ and $\mathbf{R}_\omega$ of the camera head, and the interior attitude $\mathbf{R}_i$ and Δ of the CCD line), we transform the image vector $\mathbf{v}_j = \mathbf{v}_{j0} + \mathbf{v}_{f,\Delta}$ into:

$$\mathbf{v}_{j0}^{'} = (a, b, c)^T \cdot j = \mathbf{R}_i \cdot \mathbf{v}_{j0}$$

and

$$\mathbf{v}_{f,\Delta}^{'} = (u, v, w)^T = \mathbf{R}_x \cdot \mathbf{R}_\omega \cdot \mathbf{R}_i \cdot \mathbf{v}_{f,\Delta}$$

7.1.1 Single-Projection-Center

The panoramic images are preprocessed as described in Section 3.7. Therefore, all image coordinates are in rectified ideal cylindric coordinates, and we can use the base Equation (3.7). Note that single-projection-center panoramas are rectified as accurate as supported by the calibration parameters. The use of Equation (3.7) is not just a simplification.

The requested viewing direction $\mathbf{R}_\varphi \cdot \mathbf{v}_j$ of the panoramic camera for a normal panorama (i.e., $R = 0$ and $\omega = 0$, which means a normal panorama, assuming that the camera is perfectly aligned with the rotation axis) is described by the following equation:

$$\mathbf{R}^{-1} (\mathbf{r} - \mathbf{r}_0) = \lambda \cdot \mathbf{R}_\varphi \cdot \mathbf{v}_j \tag{7.1}$$

The application of the calculated exterior attitude to the difference vector provides $(\mathbf{r} - \mathbf{r}_0)^{'}$. This allows to specify the pixel column i in the panoramic image. More detailed, we have

$$\begin{aligned} (r_x - r_{x_0})^{'} &= -\lambda \cdot \sin(i \cdot \Delta\varphi) f \\ (r_y - r_{y_0})^{'} &= \lambda \cdot \cos(i \cdot \Delta\varphi) f \end{aligned} \tag{7.2}$$

$$i = -\arctan\left(\frac{(r_x - r_{x_0})^{'}}{(r_y - r_{y_0})^{'}}\right) \frac{1}{\Delta\varphi} \tag{7.3}$$

By substituting R_φ in Equation (7.1), and due to the fact that the rotation $\mathbf{R}_\varphi$ of the CCD line corresponds to index i [i.e., $\mathbf{R}_\varphi(i)$] given by Equation (7.3), the vertical pixel row j can now be determined as follows:

$$(r_y - r_{y_0})^{''} = \lambda \cdot f \tag{7.4}$$

$$(r_z - r_{z_0})^{''} = \lambda \cdot j \cdot \delta \tag{7.5}$$

$$j = \frac{(r_z - r_{z_0})^{''}}{(r_y - r_{y_0})^{''}} \cdot \frac{f}{\delta} \tag{7.6}$$

7.1.2 Multi-Projection-Center with $\omega = 0$

Now we consider the more general case, that the camera is mounted on a lever and we have an off-axis parameter $R > 0$. (Actually $R > 0$ may also be caused by eccentricity of a set-up as described in Section 7.1.1.) The following equations are based on an ideal cylinder, which is preprocessed only for the case $R = 0$ (see Section 3.7). In fact, the following determination of the image coordinates assumes a rotation of an ideal CCD line (i.e., the CCD is not tilted). This restriction is sufficient for many applications, differences to single-projection-panoramas have been analyzed in [Hua02]. The Equation (3.8), solved for the required viewing direction, gives the following (here we assume $\omega = 0$ at first):

$$\mathbf{R}^{-1}(\mathbf{r} - \mathbf{r}_0) = \mathbf{R}_\varphi(\lambda \mathbf{v}_j + R \cdot \mathbf{e}_y) \tag{7.7}$$

In analogy to the first case, the pixel column i is determined by applying the exterior attitude $\mathbf{R}^{-1}$ to the difference vector. The known off-axis parameter, depending on the rotation $\mathbf{R}_\varphi$, and the three components of Equation (7.7) are given as follows:

$$\begin{aligned}
(r_x - r_{x0})' &= -\sin(i \cdot \Delta\varphi)(\lambda \cdot f + R) && (7.8)\\
(r_y - r_{y0})' &= \cos(i \cdot \Delta\varphi)(\lambda \cdot f + R) && (7.9)\\
(r_z - r_{z0})' &= \lambda \cdot j \cdot \delta && (7.10)
\end{aligned}$$

It can be seen that the off-axis parameter $R > 0$ has no effect on the image coordinate i, and Equation (7.3) is also valid furthermore.

The calculation of the image coordinate j (for an ideal hollow cylinder) can be determined by applying now the known image coordinate i with

$$\begin{aligned}
(r_y - r_{y0})'' &= \lambda \cdot f + R && (7.11)\\
(r_z - r_{z0})'' &= \lambda \cdot j \cdot \delta && (7.12)
\end{aligned}$$

Obviously, the panoramic image coordinate j would be given as follows:

$$j = \frac{(r_z - r_{z0})''}{(r_y - r_{y0})'' - R} \cdot \frac{f}{\delta} \tag{7.13}$$

7.1.3 Multi-Projection-Center with $\omega \neq 0$

Equations (7.7) to (7.13) describe an ideal off-axis case. The camera is moving on a circular way around a rotation center. For panoramic stereo data acquisition it is necessary to change ω (see Figure 3.1) in a second acquisition, therefore we consider $\omega \neq 0$ in this section. The image vector $\mathbf{v}_j$ has to be transformed by this angle. The Equation (7.7) changes into:

$$\mathbf{R}^{-1}\left(\mathbf{r} - \mathbf{r}_0\right) = \mathbf{R}_\varphi(\mathbf{R}_\omega \cdot \lambda \mathbf{v}_j + R \cdot \mathbf{e}_y) \tag{7.14}$$

Applying also the exterior attitude $\mathbf{R}^{-1}$ to the difference vector, and take $\mathbf{R}_\varphi$ and R to the left, we obtain the following equations in detail:

$$\begin{aligned}
\cos(i \cdot \Delta\varphi) \cdot \left(r_x - r_{x_0}\right)' + \sin(i \cdot \Delta\varphi) \cdot \left(r_y - r_{y_0}\right)' &= -\sin(\omega) \cdot \lambda \cdot f & (7.15)\\
-\sin(i \cdot \Delta\varphi) \cdot \left(r_x - r_{x_0}\right)' + \cos(i \cdot \Delta\varphi) \cdot \left(r_y - r_{y_0}\right)' - R &= \cos(\omega) \cdot \lambda \cdot f & (7.16)\\
\left(r_z - r_{z_0}\right)' &= \lambda \cdot j \cdot \delta & (7.17)
\end{aligned}$$

Therefore, ω can be determined by known R and i as follows:[1]

$$\omega = -\arctan\left(\frac{\cos(i \cdot \Delta\varphi) \cdot \left(r_x - r_{x_0}\right)' + \sin(i \cdot \Delta\varphi) \cdot \left(r_y - r_{y_0}\right)'}{-\sin(i \cdot \Delta\varphi) \cdot \left(r_x - r_{x_0}\right)' + \cos(i \cdot \Delta\varphi) \cdot \left(r_y - r_{y_0}\right)' - R}\right) \tag{7.18}$$

For determination of the required image coordinates i and j, λ needs to be determined at first. By addition of the squares of Equations (7.15) and (7.16) (with R taken to the right for this equation), the unknown $i \cdot \varphi$ is eliminated. Thus we have the following:

$$\left(r_x - r_{x_0}\right)'^2 + \left(r_y - r_{y_0}\right)'^2 = \sin^2(\omega) \cdot \lambda^2 \cdot f^2 + \cos^2(\omega) \cdot \lambda^2 \cdot f^2 + 2\cos(\omega) \cdot \lambda \cdot f \cdot R + R^2 \tag{7.19}$$

Consequently, the quadratic equation is given as follows:

$$\lambda^2 + \frac{2\cos(\omega) \cdot \lambda \cdot R}{f} + \frac{R^2 - \left(r_x - r_{x_0}\right)'^2 - \left(r_y - r_{y_0}\right)'^2}{f^2} = 0 \tag{7.20}$$

[1] Actually ω is determined by calibration. Here we assume an ideal, leveled camera rotation. i can be determined from the corresponding control point **r**.

with its solutions

$$\begin{aligned}
\lambda_{1,2} &= \frac{-R \cdot \cos(\omega)}{f} \pm \sqrt{\frac{R^2 \cdot \cos^2(\omega)}{f^2} - \frac{R^2 - (r_x - r_{x_0})'^2 - (r_y - r_{y_0})'^2}{f^2}} &(7.21)\\
\lambda_{1,2} &= \frac{1}{f}\left(-R \cdot \cos(\omega) \pm \sqrt{(r_x - r_{x_0})'^2 + (r_y - r_{y_0})'^2 - R^2 \cdot \sin^2(\omega)}\right) &(7.22)
\end{aligned}$$

A real solution is given by a non-negative radicand, therefore we assume $(r_x - r_{x0})^{'2} + (r_y - r_{y0})^{'2} \geq R^2 \cdot \sin^2(\omega)$. Secondly, it is assumed that λ can be only a positive value, because of using λ as a scaling factor without changing the viewing direction. Thus we can notice that λ_2 is not a valid solution for the case $\cos(\omega) \geq 0$. λ_1 is then a valid solution if the following equation is valid:

$$\lambda_1 \cdot f = -R \cdot \cos(\omega) + \sqrt{(r_x - r_{x_0})'^2 + (r_y - r_{y_0})'^2 - R^2 \cdot \sin^2(\omega)} > 0 \qquad (7.23)$$

transformed as follows

$$R^2 \cdot \cos^2(\omega) < (r_x - r_{x_0})^{'2} + (r_y - r_{y_0})^{'2} - R^2 \cdot \sin^2(\omega) \qquad (7.24)$$

thus λ_1 is a valid solution if $R^2 < (r_x - r_{x0})^{'2} + (r_y - r_{y0})^{'2}$ for the case $\cos(\omega) \geq 0$ (looking “outside”, away from the rotation axis).

For the case $\cos(\omega) < 0$ (looking “inside”, to the rotation axis), λ_1 is then a valid solution if the radicand is a non-negative value. λ_2 is a valid solution if also the radicand, and thus λ_2 are non-negative. This is given if the following is fulfilled:

$$R^2 \sin^2(\omega) \leq (r_x - r_{x_0})^{'2} + (r_y - r_{y_0})^{'2} \leq R^2 \qquad (7.25)$$

Such we have two valid solutions in the area of the shaded disc which is illustrated in Figure 7.2 (left). Objects which are located in this area will be seen two times (depicted as the projection centers C_1 and C_2). Furthermore, the figure shows the invisible areas (blue circles) depending on ω. The radii of these invisible circles are calculated as $R_{inv} = R \cdot |\sin(\omega)|$ for the case $\cos(\omega) < 0$, and for the case $\cos(\omega) \geq 0$ we have $R_{inv} = R$.

The requested image coordinate j is determined by using Equation (7.17) and the now known value of λ. We obtain

$$j = \frac{(r_z - r_{z_0})'}{\lambda \cdot \delta} \qquad (7.26)$$

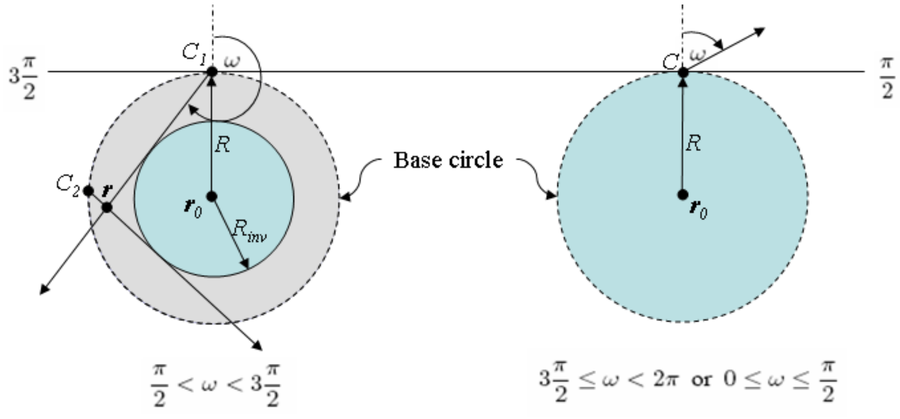

$3\frac{\pi}{2}$
C_1
ω
C_2
r
R
r_0
R_{inv}
Base circle
C
ω
$\frac{\pi}{2}$
R
r_0
$\frac{\pi}{2} < \omega < 3\frac{\pi}{2}$
$3\frac{\pi}{2} \leq \omega < 2\pi$ or $0 \leq \omega \leq \frac{\pi}{2}$

Figure 7.2: Left:“inside” looking case, and right: “outside” looking case (see text).

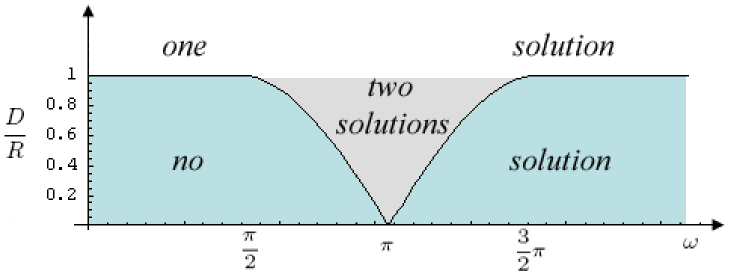


Figure 7.3: Presented solutions depending on ω, R and projected object distance in $\mathbb{R}^2$ $D = \sqrt{(r_x - r_{x0})'^2 + (r_y - r_{y0})'^2}$.

Finally, the image coordinate i is determined by dividing Equation (7.15) by Equation (7.16), after factoring out $\cos(i \cdot \Delta\varphi)$, and we obtain the following:

$$\frac{(r_x - r_{x0})' + (r_y - r_{y0})' \cdot \tan(i \cdot \Delta\varphi)}{-(r_x - r_{x0})' \cdot \tan(i \cdot \Delta\varphi) + (r_y - r_{y0})'} = \frac{-\sin(\omega) \cdot \lambda \cdot f}{\cos(\omega) \cdot \lambda \cdot f + R} \tag{7.27}$$

By solving this equation for i, the requested image coordinate is finally given as follows:

$$i = \arctan\left(\frac{-(r_x - r_{x0})' + k \cdot (r_y - r_{y0})'}{k \cdot (r_x - r_{x0})' + (r_y - r_{y0})'}\right) \cdot \frac{1}{\Delta\varphi} \tag{7.28}$$

with

$$k = \frac{-\sin(\omega) \cdot \lambda \cdot f}{\cos(\omega) \cdot \lambda \cdot f + R} \tag{7.29}$$

7.1.4 General Case

The general case is defined by solving the general camera model equation (3.12) for the required image coordinates i and j, for known object coordinates **r**. Thus, we are also able to describe multi-projection-center panoramas, but without assuming an ideal case. All determined calibration parameters are considered. Furthermore, also the option of a rotating CCD matrix sensor is covered by this general camera model (using Δ_x for describing the horizontal position of a matrix sensor).

In a first step we determine the unknown λ as a function of j. Applying also the exterior attitude $\mathbf{R}^{-1}$ to the difference vector, and take $\mathbf{R}_\varphi$ to the left, we obtain the following equations in detail:

$$\cos(i \cdot \Delta\varphi) \cdot \tilde{r}_x + \sin(i \cdot \Delta\varphi) \cdot \tilde{r}_x = \lambda \cdot (a \cdot j + u) \tag{7.30}$$

$$-\sin(i \cdot \Delta\varphi) \cdot \tilde{r}_y + \cos(i \cdot \Delta\varphi) \cdot \tilde{r}_y = \lambda \cdot (b \cdot j + v) + R \tag{7.31}$$

$$\tilde{r}_z = \lambda \cdot (c \cdot j + w) \tag{7.32}$$

To eliminate the unknown $i \cdot \Delta\varphi$, the sum of the squares of Equations (7.30) and (7.31) is computed (written in quadratic normal form) as follows:

$$0 = \lambda^2[(a \cdot j + v)^2 + (b \cdot j + w)^2] + 2 \cdot \lambda \cdot (b \cdot j + w)^2 \cdot R - \tilde{r}_x^2 - \tilde{r}_y^2 + R^2 \tag{7.33}$$

$$\lambda_{1,2} = \frac{-R \cdot V \pm \sqrt{R^2V^2 - (R^2 - r_x^2 - r_y^2)(U^2 + V^2)}}{U^2 + V^2} \tag{7.34}$$

with $U = u + a \cdot j$ and $V = v + b \cdot j$. Constituting λ into Equation (7.32) and solving for j gives the following equation:

$$j_{1,2} = M \pm \frac{\sqrt{M^2 - K \cdot L}}{K} \tag{7.35}$$

with substituted terms K, L and M defined as follows:

$$K = c^2(R^2 - \tilde{r}_x^2 - \tilde{r}_y^2) + 2b \cdot c \cdot R \cdot \tilde{r}_z \; + \tilde{r}_z^2(a^2 + b^2) \tag{7.36}$$

$$L = w^2(R^2 - \tilde{r}_x^2 - \tilde{r}_y^2) + 2v \cdot w \cdot R \cdot \tilde{r}_z + \tilde{r}_z^2(u^2 + v^2) \tag{7.37}$$

$$M = c \cdot w(R^2 - \tilde{r}_x^2 - \tilde{r}_y^2) + \tilde{r}_z(c \cdot v \cdot R - b \cdot w \cdot R) - \tilde{r}_z^2(a \cdot u + b \cdot v) \tag{7.38}$$

Because of the fact that j is a function of λ, we have the fact that for each solution of λ there exists one solution for j. Instead of a general theoretical discussion of the solutions of $\lambda_{1,2}$ and $j_{1,2}$, we provide only numeric results. These results show that the solutions satisfy the same constrains as discussed in Section 7.1.3.

7.2 Texture Mapping

Each LRF point **p** has unique coordinates in our reference coordinate system in the Euclidean space R^3. After the generation of the 3D model, all vertices have to be assigned to a corresponding pixel from our panoramic camera. The calculation of the corresponding panorama pixel (also called texture coordinate) is discussed in Section 7.1.

In our approach (different sensor attitudes), any ray (between a camera location and its determined viewing direction to a 3D point in the 3D model) can be disturbed by obstacles in the scene, and a raytracing routine has to check whether an LRF 3D point can be colored properly. Here it would be helpful using such an approach where LRF and camera setup allows to center both principal points in such a way that we are able to map any camera pixel into the reference coordinate system directly (i.e., a raytracing routine would not be necessary). However, for an efficient raytracing routine we use the idea of a stencil buffer. This means that the panoramic image (or a part of it) is loaded into the memory as a mirror of the original texture. One bit per pixel is sufficient for the mirror image (stencil buffer) for saving memory.

All 3D points **p** are sorted by distance to the camera location $\mathbf{r}_0$ of the panoramic image. The distance D can be determined easily as:

$$D^2 = (p_x - r_{x_0})^2 + (p_y - r_{y_0})^2 + (p_z - r_{z_0})^2 \qquad (7.39)$$

To save CPU time, the square root is actually not necessary. The raytracing routine starts with the nearest triangle to the current camera location. After the determination of the texture coordinates of a triangle, the same triangular area has to be masked in the stencil buffer. If the current triangle or a part of it is already masked, the triangle is not visible from this camera view. In case that the triangular area is not masked, the texture coordinates i, j are valid and placed in a texture map. Figure 7.4 illustrates the raytracing

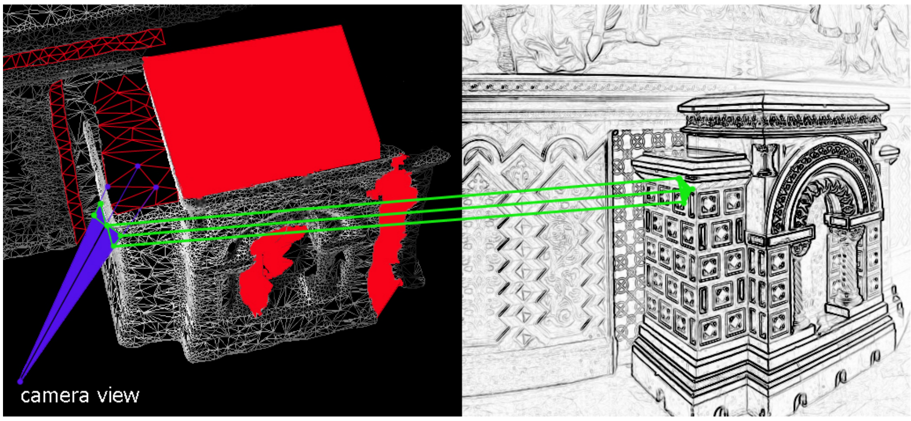


Figure 7.4: Raytracing approach with stencil buffer.

algorithm. For an better illustration, the stencil buffer image in this figure is shaded as 8 bit. This step has to iterate for each camera location (panoramic image). After all iterations, an algorithm has to decide which panoramic image, or part of this should be used as the final texture. For this decision, the following constraints are usefully.

- short distance (for an optimal resolution)
- small angle between normal vector of the triangle and viewing direction of the camera $\mathbf{v}_j$ (a zero angle would be ideal)
- no switching between selected panoramic images within a reasonably large area (to preserve radiometric uniformity)

This part of the rendering procedure is quite sensitive, and the influence of each constraint can be set by weights in the program. Figure 7.5 shows a texture mapped 3D model.

Note that the determined texture coordinates are just for the corners of each triangle. The 3D hardware accelerator of the graphic adapter maps (fills) the triangulated surface automatically, ignoring (in general) central perspective projection. The related error can be ignored as long as the triangles are small enough. Figure 7.6 illustrates such a mapping error. The red marked corner points of the triangle are rectified and calculated correctly. The mapped surface of the triangle is incorrect. However, in our meshes the triangles are actually small enough such that any displacement error is typically less than the edge length of one pixel. A possible mapping error depends on the size of the triangle and the

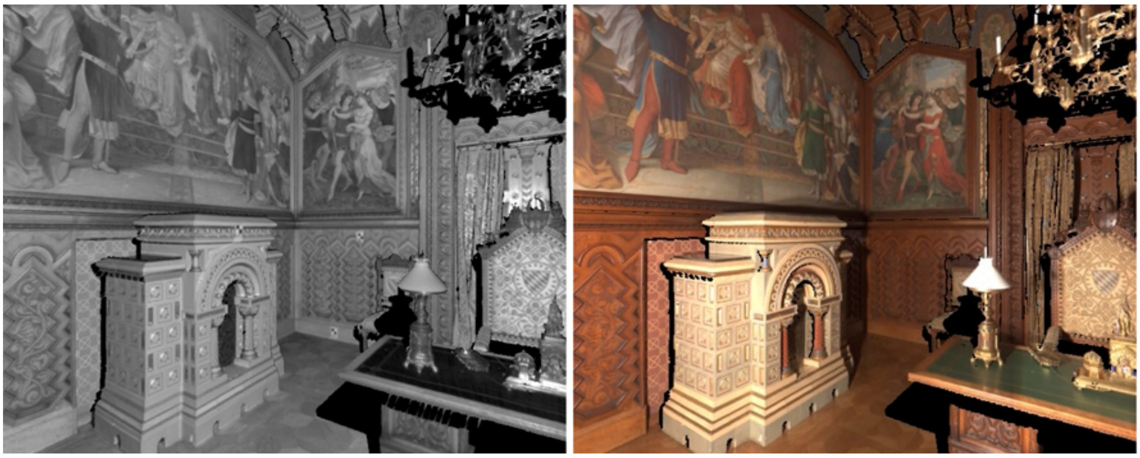


Figure 7.5: Left: 3D model with mapped texture (intensity LRF). Right: mapped texture based on panoramic images.

attitude of the normal vector of this triangle to the principle axis. The error is defined by image coordinates i (the offset angle $i \cdot \Delta\varphi$) to the theoretic principle axis, and j the vertical pixel offset to the principle axis. Figure 7.7 illustrates the mapping error depending on image offset coordinates. The principle axis in this figure is defined at $xyz = (0, 0, -1)$. The vertical offset j is set to the maximum (i.e., to 5,149 which is half of the length of the CCD line, measured in pixel).

To overcome this problem, a rectified texture has to be calculated, see Section 3.7. The calculation of the central perspective geometry of an image defines the attitude of

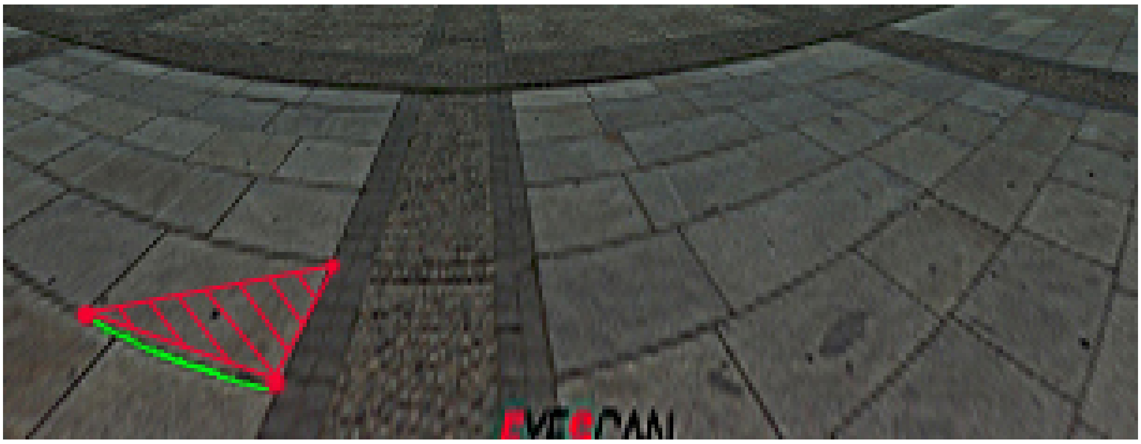


Figure 7.6: Mapping error caused by panoramic geometry.

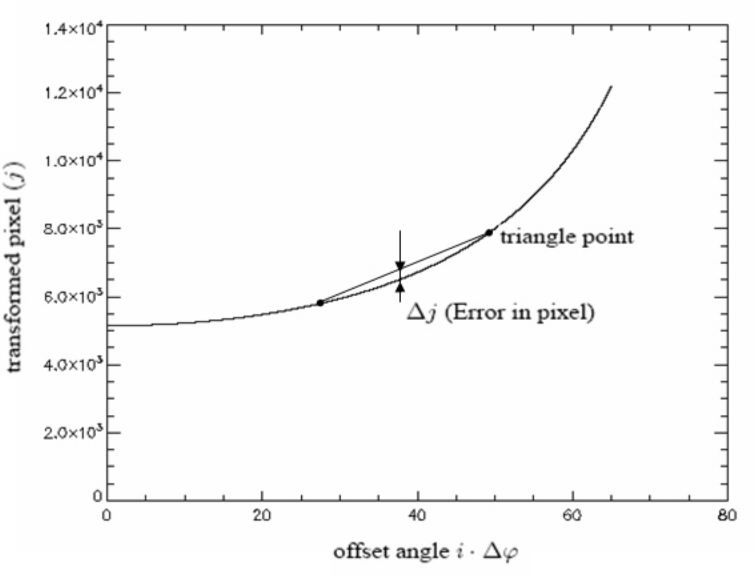


Figure 7.7: Mapping error depending on position and pixel size of the triangle. The error is calculated for a tangential plane with a cylindric panoramic image (normal panoramic distortion).

the principle axis. There might be a “visually unfavorable” perspective for such a triangle. Actually, the individual rectified texture has to be calculated for each triangle. The advantage of using a non-oriented panoramic image as texture is a dynamically specified principle axis, defined by the relevant projection center during the rotation.

Modern graphics adapter allow to replace the specification of the texture mapping routine. This has the advantage of a hardware accelerated rectifying micro code. However, for a correct mapping, each pixel inside the triangulated surface has to be calculated. Exact geometric images are, for example, calculated in case of high resolution orthophotos (see next section).

7.3 High Resolution Orthophotos

For computing orthophotos, a DSM is required (see Section 5.4). In fact, the DSM itself is the rectified orthophoto because it is already orthogonally projected to a given plane (see Figure 5.7). Instead of depth values, the original gray value has to be stored in

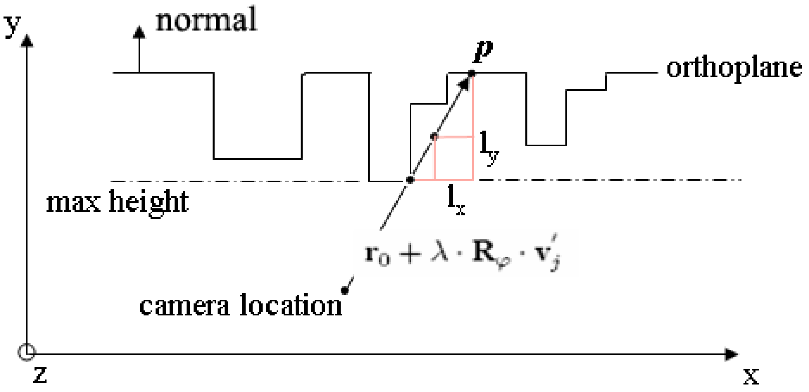

y
normal
p
orthoplane
l_y
max height
l_x
$\mathbf{r}_0 + \lambda \cdot \mathbf{R}_\varphi \cdot \mathbf{v}_j^{'}$
camera location
z
x

Figure 7.8: Ray tracing for 2.5D digitized images.

the image. A common demand is that high-resolution orthophotos (as the final product) are stored in an common file format independent from the resolution of the viewport of OpenGL. Secondly, each pixel of the orthophoto is calculated. That means, each pixel is shaded (colorized) einzeln and can not be influenced by mapping errors, as described in Section 7.2.

To compute colored orthophotos from sensors with different attitudes, the intersection between the camera ray $\mathbf{r}_0 + \lambda \cdot \mathbf{R}_\varphi \mathbf{v}_j$ and each DSM point $\mathbf{p}$ has to be determined. A disadvantage is that a fast raytracing routine cannot be used because of working with single pixels. However, raytracing in 2.5D images is easier to perform than volume raytracing. Here, a standard approach (using backward-raytracing) is used which is illustrated in Figure 7.8 and briefly described by the following steps (i and j address a raster element in the DSM):

- transform camera location $\mathbf{r}_0$ and image vector $\mathbf{v}_j$ into $\mathbf{r}_0^{'}$ and $\mathbf{v}_j^{'}$ related to the attitude of the DSM in a reference system (aligning of the orthoplane normal to the y-axis);
- determine corresponding image coordinates i', j' of the panoramic image related to a DSM-point $\mathbf{p}(i,j)$ (by using the equations of Section 7.1);
- pass backwards through the DSM raster along the camera ray, between $\mathbf{p}(i,j)$ and the transformed camera location $\mathbf{r}_0^{'}$;
- determine $l_x(i,j)$, $l_y(i,j)$, and $l_z(i,j)$ (Figure 7.8) depending on the step size given

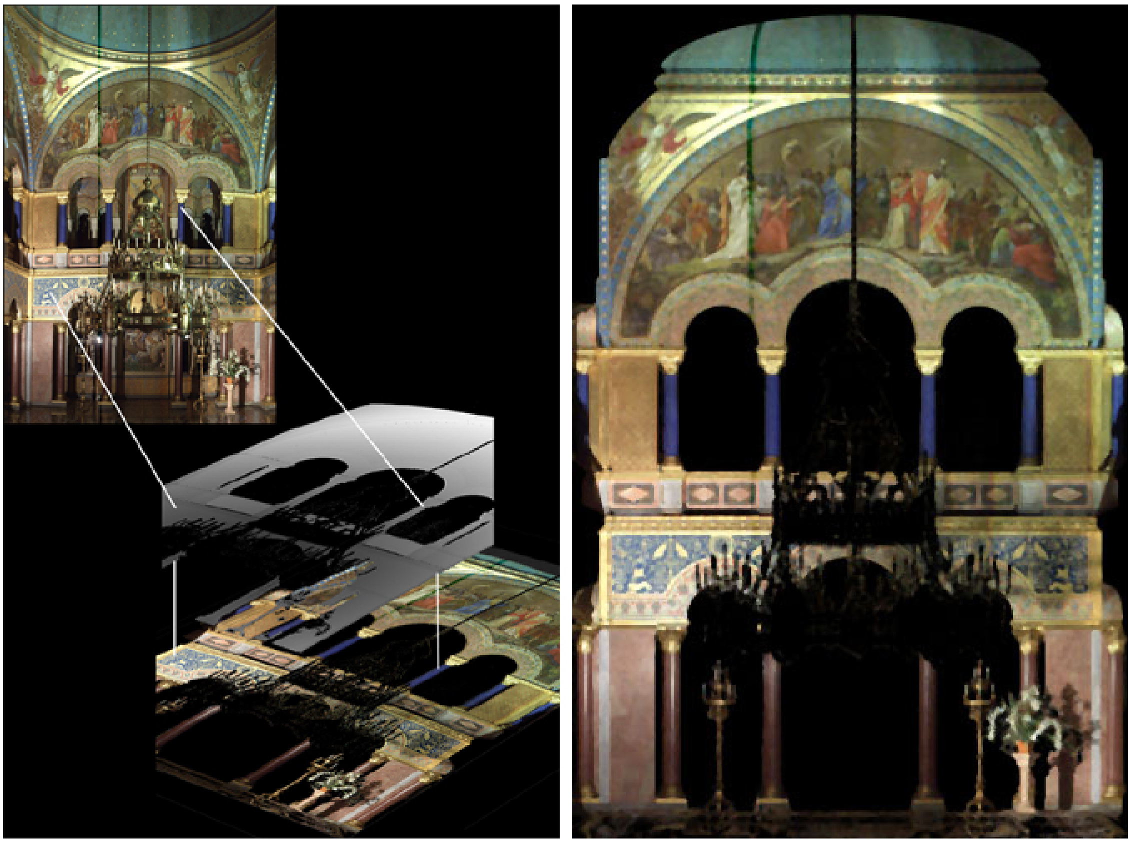


Figure 7.9: Left: projected panoramic image into a defined orthoplane by using a DSM, and result of the projected orthoimage (right).

by the resolution of the DSM;

- abort the depth check if $l_y(i,j) >$ max height;
- map the camera pixel (i',j') to the DSM-point $\mathbf{p}(i,j)$ if no depth value (DSM) on the determined ray is $> l_y(i_{0\ldots n}, j_{0\ldots n})$.

Figure 7.9 illustrates spatial relations between the panoramic image and the DSM, and depicts the rendered orthophoto.

7.4 Data Fusion of Panoramic Images and Airborne Data

This section introduces into a very recent possibility of using terrestrial panoramic images for airborne photogrammetry. Basically this is a fusion of panoramic images (e.g., a side view to a building) with topview images taken from an airplane. The goal is to replace unfavorable bird perspectives of facades of buildings by orthogonal views, where possible. Furthermore, this can also be extended into a fusion of several sensors, for example merging indoor and outdoor scene models together into one unified model. Actually, the data fusion of the panoramic image and any other achieved 3D-data is straight forward using approaches as proposed or discussed in this thesis. In a first step, the attitude of the panorama camera in the coordinates of the 3D-data needs to determined, then the requested texture coordinates of each 3D point can be determined, followed by raytracing checks.

For the generation of 2.5D DSM data from airborne photos, we had the opportunity to process data captured by the *High Resolution Stereo Camera* (HRSC)[WSG00, SGW02, SG04]. The HRSC is also based on the functional principle of CCD line panoramic cameras, but instead of a rotation of the CCD line, the image is now captured during a translative forward motion of an airplane (or satellite). The exterior attitude **R** of the camera is equivalent to the attitude of the airplane (in aerial photogrammetry denoted by angles roll, pitch and yaw). The position of the airplane corresponds to the vector $\mathbf{r}_0$. All six parameters are changing during a flight, and will be gathered by an IMU. Figure 7.10 shows a processed orthophoto based on HRSC data (by [HSH05]). The camera model corresponds to the panoramic camera model (of course, without the additional rotation $\mathbf{R}_\varphi$), and is given as follows:

$$\mathbf{r} = \mathbf{r}_0 + \mathbf{R} \cdot \lambda \cdot \mathbf{v}_j \tag{7.40}$$

The unknown scale factor λ is determined by stereo processing of the data [HSH05]. The result is a top-viewed DSM (in airborne photogrammetry also called *digital elevation model* DEM).

Because of its 2.5D representation it is necessary to extrude each point about the ground level. Consequently, the 2.5D data set is transformed into a 3D model. Points which are above the same grid element (i, j) are overwritten by the highest point (i.e., the biggest z-value). Bays, as shown schematically in Figure 7.11, cannot be presented in a 2.5D topview.

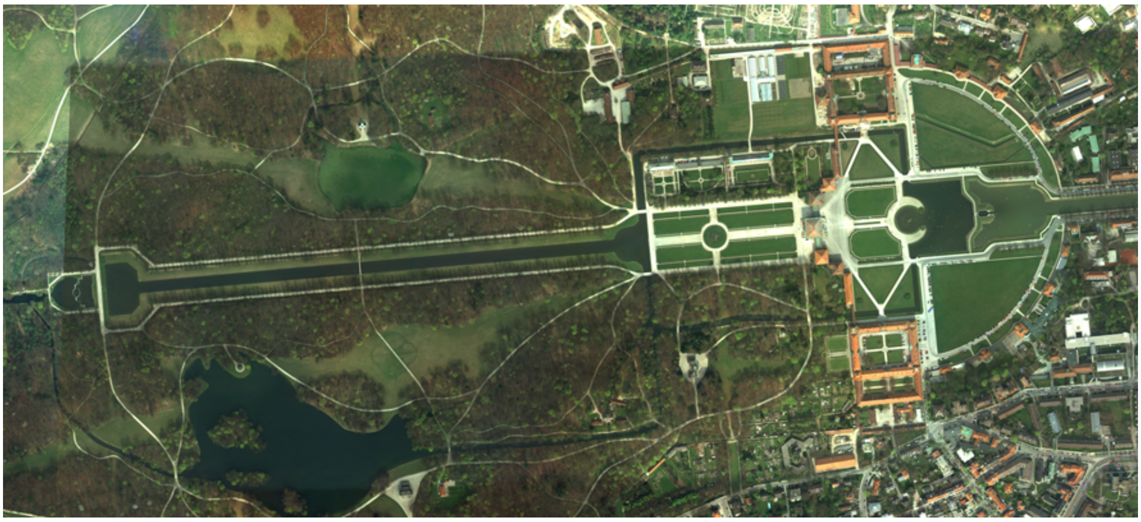


Figure 7.10: Orthophoto from HRSC data (Munich-Nymphenburg, Germany).

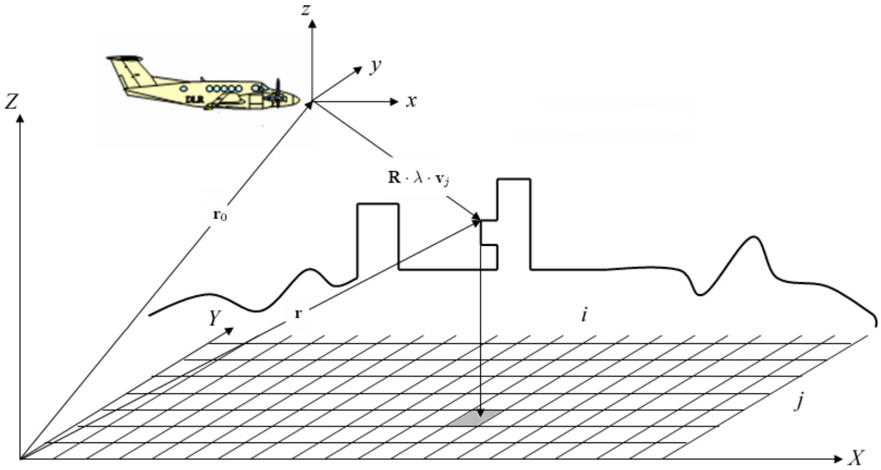


Figure 7.11: Function principle of the HRSC, and creation of a DEM.

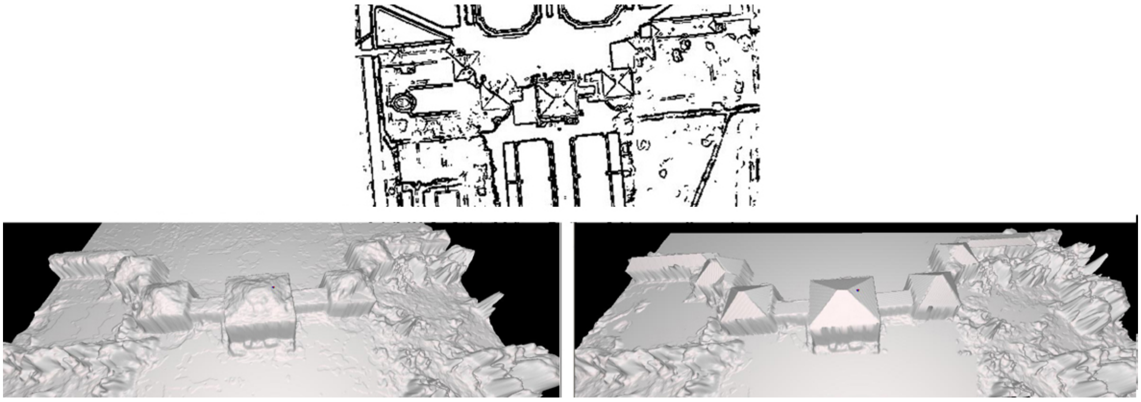


Figure 7.12: Left: visualization of a 3D model created from HRSC input DEM. Right: the same, but now based on filtered HRSC data using planar surface patches, also using an orthophoto (as shown on the top) for semi-automatic surface detection.

For the evaluation of the panoramic images, a sideview DSM should be calculated as well. With such DSMs and their extrapolated surfaces, it would be possible to texture them by the same procedure as described in Section. 7.2. Resolution and accuracy depend on the altitude of the airplane, sensor properties, and the quality of the IMU. It is complicated to merge the captured 3D data due to different accuracies of LRF and HRSC data. The sideview DSMs could be supplemented with LRF data. A calculated sideview DSM, only from HRSC data, does not have the needed resolution for a correct rectification of our panoramic images.

Nevertheless, for using the HRSC data it is also necessary to filter the data as described in Section 6.4. The orthophoto can be used for an automatization of edge detections (e.g., separation of roofs). Figure 7.12 shows filtered HRSC data.

For a simplified mapping approach, we assume at first that all facades of buildings are planar. Note that each object which has actually a non-zero distance to this plane (e.g., bays, window indentations, or balconies), is projected incorrectly. Facades will be targeted about two points (e.g., corner points of facades) in the ground plane (bottom-left and bottom-right) and then extruded by the corresponding ground level (z-value). This specify the projecting plane (4 points: bottom-left, bottom-right, upper-left and upper-right). Single textures are calculated for color mapping which are conform to a central perspective view. Therefore, the panoramic images must be aligned to the HRSC data (see Section. 3.7). The attitude is just approximated because control points of the HRSC data do not have (in general) the necessary accuracy. The remaining error (e.g., a small tilt)

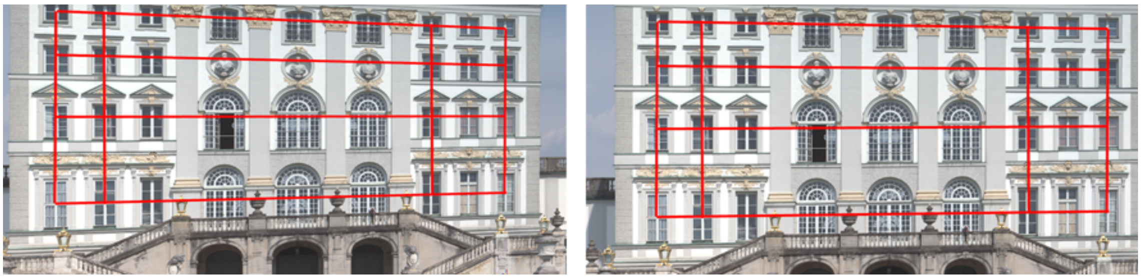


Figure 7.13: Rectification of a facade.

will be removed by a perspective transformation without knowledge about the attitude. A priori information (e.g., right angles formed by windows) is used for this transformation. The transformation is specified by the following equations:

$$i' = a_1 \cdot i + b_1 \cdot j + c_1 \cdot i \cdot j + d_1 \tag{7.41}$$

and, analogously, for j'

$$j' = a_2 \cdot i + b_2 \cdot j + c_2 \cdot i \cdot j + d_2 \tag{7.42}$$

Coefficients in these equations are determined by using four control points at least (e.g., corners on facades forming a rectangle). An over-determination is also possible using least-squares opimization. The rectification of a facade is shown in Figure 7.13.

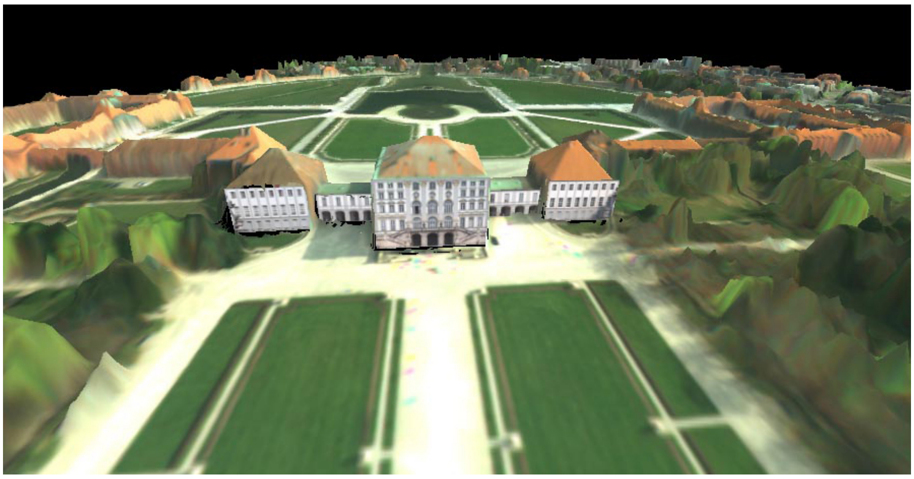


Figure 7.14: Extracted 3D model of the DEM data (HRSC) with (partially) mapped textures captured by a terrestrial panoramic camera.

8 Conclusions

The first chapter of this thesis provided a brief introduction to the subject of photogrammetry and panoramic photography. The chapter also listed common approaches for 3D scene capturing in computer vision and related work. Attempting a more specified approach, we discussed laser rangefinder applications and recent camera technologies. The combination of LRF data and images, its related work (also within this PhD project) and possible approaches, along with an evaluation of its advantages and disadvantages, were then discussed in the introduction.

Definitions and assumptions were specified in Chapter 2.

The evaluation of images (captured with a frame camera) is discussed in many photogrammetry books. The advantage of using such a camera is the static image geometry of a single photo (frame). This thesis deals with the photogrammetric interpretation of rotating CCD line cameras. Here, the image ”frame” consists of only three lines (i.e., red, green and blue), however with the camera rotating about a rotation axis. Depending on the attitude of the principle axis to the rotation axis and the off-axis distance R, various image geometries are attainable. In the area of remote sensing (e.g., airborne photogrammetry), the use of CCD lines is common practice. A related space application to our ”inside”-looking camera model, for example, is a moving satellite around the earth with a rotation axis of the Earth’s epicenter. Instead of using a formalized camera model, each attitude of a CCD line is measured during flight. With these known positions and view directions of the camera (CCD lines), mapping the captured data to a given 3D model is then rather straight forward.

In Chapter 3 the various image geometries of a rotating CCD line camera are discussed and described in terms of formalized camera models. A least-square minimization approach is presented as a new method to calibrate rotating CCD line cameras. The camera model is used therein to determine each physical movement of the CCD line. Data pre-processing and basic transformations are also explained in this chapter.

Future work is foreseen with respect to the physical improvement of the panoramic camera. Our experiences have shown that work with CCD lines is less feasible under bad light conditions. Time delay integration (TDI), as used in military applications, would improve the capturing speed. A project at DLR has been initiated to design a panoramic camera based on TDI. The idea of TDI utilizes several CCD lines and involves transporting charged electrons to the next CCD line, charging this again, and so forth. Difficulties remain in synchronizing the TDI clock with the movement of the camera.

In classical photogrammetry, the generation of 3D models is derived by evaluating at least two images. This thesis assumes a given 3D model, for example, as captured by an LRF. Modern LRFs have reasonable accuracies and scan the surrounding scene in passable time. The advantage is a real-time given 3D point cloud. An expensive computing time for evaluating stereo images is not necessary. Chapter 4 briefly reviewed the advantages and drawbacks of available LRFs. While LRFs are not the main focus of this thesis, the calibration process, error analysis and coordinate system of the LRF used are explained here in greater detail. Future work would also entail improving the LRF system, such as edge quality. The laser spot must be much better focused.

Chapter 5 began with a short introduction in 3D visualization and continued with a discussion of possible methods of visualization (i.e., data projections concerning our LRF points). Possible applications include the generation of orthophotos, interactive 3D animations (e.g., for virtual tours), and so forth. This chapter listed (with brief commentary) those techniques used for our data. The creation of digital surface models based on our LRF data is described and shown to be suitable for bump mapping.

Chapter 6 discussed the subject meshing, the basis of 3D visualization. The LRF points are thereby triangulated to an initial dense mesh. The main challenge within 3D model generation is data reduction, namely triangulating a given cloud of points by a reduced number of triangles or triangle stripes while not losing geometric resolution. The chapter also demonstrated some optimizing techniques related to our LRF data and their resulting mesh. Advanced approaches to general point clouds and their problematic nature (e.g., holes and connectivity problems) are showed in this chapter. We also discussed particular problems occurring in the process of merging range data and color texture, such as achieving straight edges or planarity in the scene, where supported by range and color data.

This chapter suggested the bulk of future work. Some of the algorithms introduced produced reasonable results by selecting a mere part of the 3D model and then choosing the filter appropriate to the occurring problem. A general definition of problems in 3D point clouds or triangulated meshes seems to be quite difficult, even when using threshold parameters and so forth. Certainly the explained means of achieving straight edges and planarity can be extended to super-linear shape complexity, but a full automatic processing of our data should also be a focus of future work. In summary it can be said that a high quality mesh is the key to a correct color mapping in our dimension with respect to accuracy and resolution.

Chapter 7 formalized the relation between object coordinates and image coordinates based on the sensor model. The calculation of the requested image coordinates by given object coordinates is the basis for each data fusion. The described general camera model allows the estimation of image coordinates for rotating CCD line cameras or rotating frame cameras for a given 3D point. The parameters of the interior and exterior as well as the spatial attitude of the principle axis are thereby considered. The chapter also incorporates the data fusion of our panoramic data with the meshed 3D model. In this texture mapping process the ray tracing routine is explained. Secondly, as an example of data fusion, the processing of high resolution orthophotos is shown. Here each pixel is mapped separately to a given DSM or to an extracted surface model by an LRF. As an alternative approach, the fusion of our panoramic images with ascertained 3D data of stereo processed airborne data is discussed at the end of this chapter. Future work is also foreseen in improving the mesh quality or in merging developed 3D models (e.g., form HRSC and an LRF).

This work was conducted in support of the Project Neuschwanstein. Within the course of the project more than 1800 LRF scans and 50 panoramic images were processed. Figure 8.1 illustrates all combined LRF scans. In connection with this project, a software tool was developed. The centrepiece of the developed software is a program for the visual-

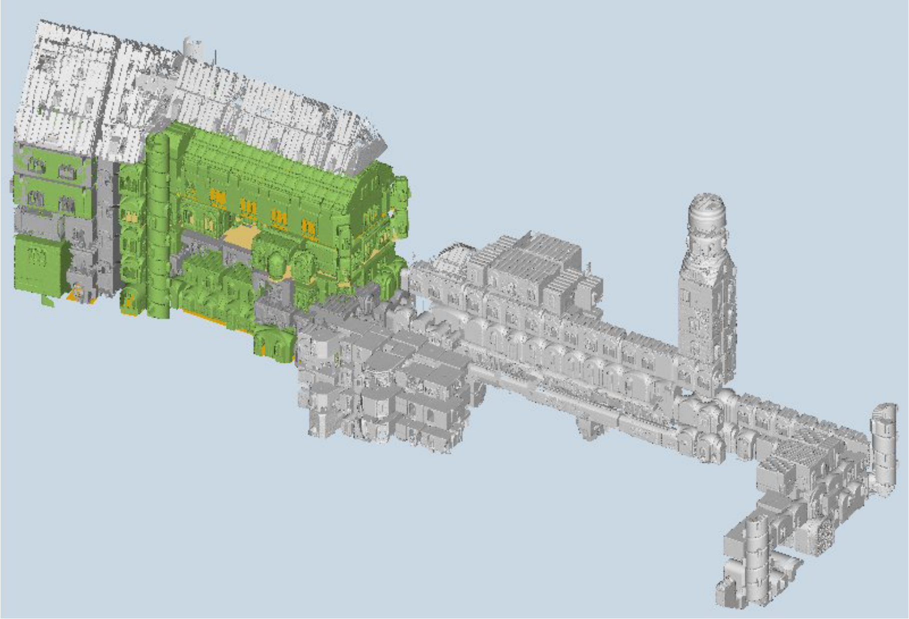


Figure 8.1: 3D model of castle Neuschwanstein, based on 1800 LRF scans.

ization and processing of large image- and 3D data. The program includes the calibration of the data and data merging. Improvement of the 3D data is presently implemented via use of filters, which must currently be individually selected and then successively applied. The aim of future work is the development of a fully-automated processing chain to produce the color 3D model.

Bibliography

[AK89] J. Albertz and W. Kreiling. *Photogrammertic Guide*. Wichmann, Karlsruhe, fourth edition edition, 1989.

[AS97] S. Avidan and A. Shashua. Novel view synthesis in tensor space. In *IEEE Conf. Computer Vision Pattern Recognition*, pages 1034–1040, 1997.

[AW96] J. Albertz and A. Wiedemann. From analogue to digital close-range photogrammetry. In O. Altan and L. Gründig, editors, *First Turkish-German Joint Geodetic Days.*, pages 245–253, Istanbul Technical University, Turkey, 1996.

[B̈85] H.P. Bähr. *Digitale Bildverarbeitung*. Herbert Wichman Verlag, Karlsruhe, 1985.

[BH04] T. Bodenmueller and G. Hirzinger. Online surface reconstruction from unorganized 3d-points for the dlr hand-guided scanner system. In *2nd Symposium on 3D Data Processing*, Thessaloniki, Greece, 2004.

[BK01] R. Benosman and S.B. Kang, editors. *Panoramic Vision: Sensors, Theory, and Applications*. Springer, Berlin, 2001.

[BKM$^+$00] A. Börner, W. Kirchhofer, B. Michalevich, R. Reulke, M. Scheele, K. Scheibe, and U. Templemann. Test resluts obtained with the lh system ads40 airborne digital sensor. In *ISPRS*, volume XXXIII, Part B4, pages 871–, Amsterdam, Netherlands, 2000.

[Bli78] J. Blinn. Simulation of wrinkled surfaces. In *Proceedings of the 5th annual conference on Computer graphics and interactive techniques, SIGGRAPH-ACM*, volume 12 (3), pages 286–292. ACM Press New York, NY, USA, 1978.

[BN99] S. Baker and S. K. Nayar. A theory of single-viewpoint catadioptric image formation. In *Int. J. Computer Vision*, pages 1–22, 1999.

[Bou] P. Bourke. http://astronomy.swin.edu.au/ pbourke/stereographics/stereorender/.

[BVM03] W. Boehler, M. B. Vincent, and A. Marbs. Investigating laser scanner accuracy. In *New Perspectives to Save Cultural Heritage, CIPA XIXth International Symposium*, pages 696–701, Anatalya, Turkey, 2003.

[CH85] B. Carrihill and R. Hummel. Experiments with the intensity ratio depth sensor. In *Computer Vision, Graphics and Image Processing*, volume 32, pages 337–358, 1985. Academic Press.

[Che95] S. E. Chen. Quicktimevr - an image-based approach to virtual environment navigation. In *SIGGRAPH*, pages 29–38, 1995.

[Cro77] F.C. Crow. Shadow algorithms for computer graphics, parts 1 and 2. In *Proc. SIGGRAPH*, volume 11-2, pages 242–248 and 442–448, 1977.

[DK06] K. Daniilides and R. Klette, editors. *Computer Vision Beyond the Pinhole Camera*, Amsterdam, 2006. Springer.

[DP04] K. Daniilides and N. Papanikolopoulos, editors. *Panoramic Robots*, volume 11 of *IEEE Robotics Automation Magazine*, December 2004.

[DS02] F. Däumlich and R. Steiger, editors. *Instrumentenkunde der Vermessungstechnik*. H. Wichmann, Heidelberg, 2002.

[DSTT00] F. Dellaert, S.M. Seitz, C.E. Thorpe, and S. Thrun. Structure from motion without correspondence. In *IEEE Conference on Computer Vision and Pattern Recognition*, page 557564, 2000.

[Eck02] A. Eckardt. *Design und Verifikation der ersten digitalen kommerziellen Luftbildkamera*. PhD thesis, TU-Berlin, 2002.

[ER01] A. Eckardt and R. Reulke. Hardware concept of the first commercial airborne digital sensor (ads). In *SPIE*, volume 4444, San Diego, USA, 2001.

[Fau93] O. Faugeras. Three-dimensional computer vision: A geometric viewpoint. *The MIT Press, London, England*, 1993.

[FHF01] I. Heinz F. Haertl and C. Fröhlich. Semi - automatic 3d cad model generation of as - built conditions of real environments using a visual laser radar. In *10th IEEE International Workshop on Robot-Human Interactive Communication (ROMAN '01)*, page 400 406, Paris, 2001.

[FL01] O. Faugeras and Q.-T. Luong. The geometry of multiple images. *The MIT Press, London, England*, 2001.

[Ö01] Jens Gühring. Dense 3d surface acquisition by structured light using offtheshelf components. In *Videometrics and Optical Methods for 3D Shape Measurement*, pages 220–231, 2001.

[GBS$^+$04] D. Griessbach, A. Börner, M. Scheele, K. Scheibe, and S. Sujew. Line scanner in combination with inertial measurement unit. In *Panoramic Photogrammetry Workshop*, volume XXXIV, pages 19–22, Dresden, Germany, 2004. ISPRS working group V/16.

[GD01] C. Geyer and K. Daniilidis. Catadioptric projective geometry. In *International Journal of Computer Vision*, volume 43, pages 223–243, 2001.

[Ger82] Helmut Gernsheim. *The Origins of Photography*. Thames and Hudson, 1982.

[GH01] A. Gruen and T.S. Huang, editors. *Calibration and Orientation of Cameras in Computer Vision*, volume 34 of *Informations Sciences*. Springer, Berlin, Germany, 2001.

[GNT98] J. Gluckman, S. K. Nayar, and K. J. Thorek. Real-time panoramic stereo. In *DARPA*, pages 299–303, 1998.

[Gou71] H. Gouraud. Illumination for computer generated pictures. In *Comm. ACM*, volume 18, pages 311–317, 1971.

[Gra06] Silicon Graphics. www.opengl.org. web, 2006.

[Har92] R. I. Hartley. Estimation of relative camera positions for uncalibrated cameras. In *European Conf. Computer Vision*, pages 579–587, 1992.

[Hey97] A. Heyden. Reconstruction from image sequences by means of relative depths. In *Int. J. Computer Vision*, pages 155–161, 1997.

[HH98] H. C. Huang and Y. P. Hung. Panoramic stereo imaging system with automatic disparity warping and seaming. In *Graph. Models Image Processing*, pages 196–208, 1998.

[Hic] R.A. Hicks. The page of catadioptric sensor design. http://www.cs.drexel.edu/ ahicks/design/design.html.

[Hir05] H. Hirschmüller. Accurate and efficient stereo processing by semi-global matching and mutual information. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 807–814, San Diego, USA, 2005.

[HK99] E. Horn and N. Kiryati. Toward optimal structured light patterns. In *Image and Vision Computing*, volume 17(2), pages 87–97, 1999.

[HNE82] O. Hofmann, P. Nav, and H. Ebner. Dps a digital photogrammetric system for producing digital elevation models and orthophotos by means of linear array scanner imagery. In *Int. Arch. of Photogrammetry.*, volume 24 Part B3, pages 216–227, Helsinki, Finland., 1982.

[Hon91] J. Hong. Image based homing. In *IEEE Int. Conf. Robotics Automation*, pages 620–625, 1991.

[HSH05] H. Hirschmüller, F. Scholten, and G. Hirzinger. Stereo vision based reconstruction of huge urban areas from an airborne pushbroom camera (hrsc),. In *Pattern Recognition of the 27th DAGM Symposium*, volume 3663 of *Lecture Notes in Computer Science*, pages 58–66, Vienna, Austria, 2005.

[Hua94] T. Huang. Motion and structure from feature correspondences: a rewiew. In *IEEE*, volume 82, pages 252–268, 1994.

[Hua02] F. Huang. *Epipolar Geometry and Camera Calibration of Cylindrical Panoramas*. PhD thesis, University of Auckland, 2002.

[HWK99] F. Huang, S. K. Wei, and R. Klette. Depth recovery system using object based layers. In *Image Vision Comp. New Zealand*, pages 199–204, 1999.

[HWK01] F. Huang, S. K. Wei, and R. Klette. Geometric analysis of symmetric panorama imaging. In *Image Vision Comp. New Zealand*, pages 19–24, 2001.

[HWK02a] F. Huang, S. Wei, and R. Klette. Calibration of line-based panoramic cameras. In D. Kenwright, editor, *Image and Vision Coumputing New Zealand*, pages 107–112, 2002.

[HWK$^+$02b] F. Huang, S. Wei, R. Klette, G. Gimmelfarb, R. Reulke, M. Scheele, and K. Scheibe. Cylindrical panoramic cameras - from basic design to applications. In D. Kenwright, editor, *Image and Vision Coumputing New Zealand*, pages 101–106, 2002.

[HZ00] R. Hartley and A. Zisserman. Multiple view geometry in computer vision. *Cambridge Univ. Press, United Kingdom.*, 2000.

[ISI97] H. Ishiguro, T. Sogo, and T. Ishida. Human behavior recognition by a distributed vision system. In *DiCoMo Workshop*, pages 615–620, 1997.

[IYT92] H. Ishiguro, M. Yamamoto, and S. Tsuji. Omni-directional stereo. In *IEEE Trans. Pattern Analysis Machine Intelligence*, pages 257–262, 1992.

[Kan93] K. Kanatani. Geometric computation for machine vision. *Oxford Univ. Press, New York*, 1993.

[KD97] S. B. Kang and P. K. Desikan. Virtual navigation of complex scenes using clusters of cylindrical panoramic images. Technical Report CRL 97/5, Digital Equipment Corporation, Cambridge Research Lab, 1997.

[Ker01] F. Kern. Supplementing laserscanner geometric data with photogrammetric images for modelling. In J. Albertz, editor, *Int. Symposium CIPA*, pages 454–461, 2001.

[KGW$^+$03] R. Klette, G. Gimel'farb, S. Wei, F. Huang, K. Scheibe, M. Scheele,

A. Börner, and R. Reulke. On design and applications of cylindrical panoramas. In N. Petkov and M. Westenberg, editors, *Proc. CAIP*. Springer, Berlin, 2003.

[Koc94] R. Koch. Model-based 3-d scene analysis from stereoscopic image sequences. *ISPRS Journal of Photogrammetry and Remote Sensing*, 49(5):23–30, 1994.

[Koc99] R. Koch. *Handbook of Computer Vision and Applications.*, chapter 3-D Modeling of Objects from Image Sequences. Academic Press, 1999.

[KR04] R. Klette and A. Rosenfeld. *Digital Geometry. Geometric Methods for Digital Picture Analysis*. Morgan Kaufmann, San Francisco, 2004.

[KR05] R. Klette and R. Reulke. Modeling 3d scenes: Paradigm shifts in photogrammetry, remote sensing and computer vision. Technical Report CITR-TR-155, Dept. of Computer Science, The University of Auckland, Auckland, 2005.

[Kra97] K. Kraus. *Photogrammetrie*. Dümmler, Bonn, sixth edition edition, 1997.

[KS97] S. B. Kang and R. Szeliski. 3-d scene data recovery using omnidirectional multibaseline stereo. In *Int. J. Computer Vision*, pages 167–183, 1997.

[KS05] R. Klette and K. Scheibe. Combinations of range data and panoramic images - new opportunities in 3d scene modeling. In *Computer Graphics Imaging Vision: New Trends*, pages 3–10, 2005.

[Lee] A. Lee. gamasutra.com/features/20000908/lee_01.htm.

[LKR05] G. Liu, R. Klette, and B. Rosnhahn. Structure from motion in the presence of noise. Technical report, The University of Auckland, CITR - Department of Computer Science, 2005.

[LRKI06] T. Luhmann, S. Roboson, S. Kyle, and I.Harley. *Close Range Photogrammetry, Principles, Techniques and Applications*. Whittles Publishing, 2006.

[Luh03] T. Luhmann. *Nahbereichsphotogrammetrie*. Wichmann, Heidelberg, 2003.

[LV94] Q. T. Luong and T. Vieville. Canonic representations for the geometries of multiple projective views. In *European Conf. Computer Vision*, pages 589–599, 1994.

[MB95] L. McMillan and G. Bishop. Plenoptic modeling: an image-based rendering system. In *SIGGRAPH*, pages 39–46, 1995.

[McB] B. McBride. A timeline of panoramic cameras. Iternet: www.panoramicphoto.com/timeline.htm.

[Mur95] D. W. Murray. Recovering range using virtual multicamera stereo. In *Com-*

puter Vision Image Understanding, pages 285–291, 1995.

[Nie95] W. Niemeier. Einsatz von Laserscannern für die Erfassung von Gebäudegeometrien. *Gebäudeinformationssystem*, 19:155–168, 1995.

[NMO97] T. Nishimura, T. Mukai, and R. Oka. Spotting recognition of gestures performed by people from a single time-varying image. In *Int. Conf. Robots Systems*, pages 967–972, 1997.

[OHS99] M. Ollis, H. Herman, and S. Singh. Analysis and design of panoramic stereo vision using equi-angular pixel cameras. Technical Report Technical Report CMU-RI-TR-99-04,, The Robotics Institute, Carnegie Mellon University, Pittsburgh, 1999.

[PA82] J. L. Posdamer and M. D. Altschuler. Surface measurement by spaceencoded projected beam systems. In *Computer Graphics and Image Processing*, volume 18(1), pages 1–17, 1982.

[PBE99] S. Peleg and M. Ben-Ezra. Stereo panorama with a single camera. In *Computer Vision Pattern Recognition*, pages 395–401, 1999.

[Pet90] G. Petrie. Developments in analytical instrumentation. In *ISPRS Journal of Photogrammetry and Remote Sensing.*, volume 45, pages 61–89, 1990.

[Pho75] B. Phong. Illumination for computer generated pictures. In *Comm. ACM*, volume 18, pages 311–317, 1975.

[RB98] R. Rademacher and G. Bishop. Multiple-center-of-projection images. In *SIGGRAPH*, pages 199–206, 1998.

[RK98] A. Koschan R. Klette, K. Schlüns. *Computer Vision Three-Dimensional Data from Images.* Springer, Singapore, 1998.

[RSS01] R. Reulke, M. Scheele, and K. Scheibe. Multi-Sensor-Ansätze in der Nahbereichsphotogrammetrie. In *Jahrestagung DGPF, Konstanz*, 2001.

[RWK+03] R. Reulke, A. Wehr, R. Klette, M. Scheele, and K. Scheibe. Panoramic mapping using ccd-line camera and laser scanner with integrated position and orientation system. In D.G. Bailey, editor, *Image and Vision Couputing New Zealand*, pages 72–77, 2003.

[RWL05] S. Richard, Jr. Wright, and B. Lipchak. *OpenGL Superbible*. Sams Publishing, third edition, 2005.

[SA96] A. Shashua and S. Avidan. The rank 4 constraint in multiple (over 3) view geometry. In *European Conf. Computer Vision*, volume II, pages 196–206, April 1996.

[Sch71] K. Schwidefsky. Albrecht meydenbauer initiator der photogrammetrie in deutschland. *Bildmessung und Luftbildwesen*, 39:183–189, 1971.

[SG04] F. Scholten and G. Gwinner. Operational parallel processing in digital photogrammetry - strategy and results using different multi-line cameras. In *20th International Congress for Photogrammetry and Remote Sensing*, pages 408–413, Istanbul, Turkey, 2004.

[Sgr05] M. Sgrenzaroli. Cultural heritage 3d reconstruction using high resolution laser scanner: New frontiers data processing. In *CIPA 2005 XX International Symposium*, Torino, Italy, 2005.

[SGW02] F. Scholten, K. Gwinner, and F. Wewel. Angewandte digitale photogrammetrie mit der hrsc. In *Photogrammetrie Fernerkundung Geoinformation (PFG)*, volume 5, pages 317–332, 2002.

[SH99] H. Y. Shum and L. W. He. Rendering with concentric mosaics. In *SIGGRAPH*, pages 299–306, 1999.

[SI04] T. SCHULZ and H. INGENSAND. Influencing variables, precision and accuracy of terrestrial laser scanners. In *Proceedings of 'Ingeo 2004'*, Bratislava, 2004.

[SJS04] M. Scheele, H. Jahn, and R. Schuster. Geometrische kalibration von ccd kameras mittels beugungsbilder. In *Publikationen der DGPF*, volume 13, pages 513–, 2004.

[SK95] R. Szeliski and S. Kang. Direct methods for visual scene reconstruction. In *IEEE Works. Representation Visual Scenes*, pages 26–33, 1995.

[SKR$^+$01] K. Scheibe, H. Korsitzky, R. Reulke, M. Scheele, and M. Solbrig. Eyescan - a high resolution digital panoramic camera. In *Robot Vision - Lecture Notes in Computer Science*, pages 77–83, New Zealand, 2001. Springer, Berlin.

[SM04] D. Schneider and H.-G. Maas. Development and application of an extended geometrical model for high resolution panoramic cameras. In *Geo-Imagery Bridging Continents*, pages 366–371, Istanbul, Turkey, 2004. XXth ISPRS Congress,.

[SPS02] K. Scheibe, M. Pospich, and M. Scheele. Multi-sensor-approach in close range photogrammetry. In *Image and Vision Computing New Zealand*, pages 77– 82, 2002.

[SS97] R. Szeliski and H. Y. Shum. Creating full view panoramic image mosaics and environment maps. In *SIGGRAPH*, 1997.

[SS99] H. Y. Shum and R. Szeliski. Stereo reconstruction from multiperspective panoramas. In *Int. Conf. Computer Vision*, pages 14–21, 1999.

[Svo99] T. Svoboda. *Central Panoramic Cameras Design, Geometry, Egomotion*. PhD thesis, Czech Technical University, Prague, Czech Republic, 1999.

[SW95] A. Shashua and M. Werman. Trilinearity of three perspective views and its associated tensor. In *Int. Conf. Computer Vision*, pages 920–925, 1995.

[Tho03] G. Thomas. Real-time panospheric image dewarping and presentation for remote mobile robot control. In *J. Advanced Robotics*, pages 359–368, 2003.

[Wat02] A. Watt. *3D-Computergrafik*. Pearson Studium, Munich, Germany, 2002.

[WHK98a] S. K. Wei, F. Huang, and R. Klette. 3-d environment visualization by panoramic color anaglyphs. In *Image Vision Comp. New Zealand*, pages 280–285, 1998.

[WHK98b] S. K. Wei, F. Huang, and R. Klette. Color anaglyphs for panorama visualizations. technical report, citr-tr-19. Technical report, Dept. of Computer Science, The University of Auckland, Auckland, 1998.

[WHK99] S. K. Wei, F. Huang, and R. Klette. Visibility determination in depth layered cylinder scene representations. In *Image Vision Comp. New Zealand*, pages 175–180, 1999.

[Wie01] A. Wiedemann. Kombination von Laserscanner-Systemen und photogrammetrischen Methoden im Nahbereich. *Photogrammetrie Fernerkundung Geoinformation*, pages 261–270, 2001.

[WIK] Wikipedia-homepage. Internet Cyclopedia.

[WSG00] F. Wewel, F. Scholten, and K. Gwinner. High resolution stereo camera (hrsc) - multispectral 3d-data acquisition and photogrammetric data processing. In *Canadian Journal of Remote Sensing*, volume 26(5), pages 466–474, 2000.

[Yag99] Y. Yagi. Omnidirectional sensing and its applications. In *IEICE Trans. Information Systems*, pages 568–579, 1999.

[YK90] Y. Yagi and S. Kawato. Panoramic scene analysis with conic projection. In *IEEE/RSJ Int. Conf. Intelligent Robots Systems*, pages 181–187, 1990.

[YNY95] Y. Yagi, Y. Nishizawa, and M. Yachida. Map-based navigation for a mobile robot with omnidirectional image sensor copis. In *IEEE Trans. Robotics Automation*, pages 634–648, 1995.

[YYY95] K. Yamazawa, Y. Yagi, and M. Yachida. Obstacle detection with omnidirectional image sensor hyperomni vision. In *IEEE Int. Conf. Robotics Automation*, pages 1062–1067, 1995.

[ZCS02] L. Zhang, B. Curless, and S. M. Seitz. Rapid shape acquisition using color structured light and multipass dynamic programming. In *Int. Symposium on 3D Data Processing Visualization and Transmission*, Padova, Italy, June 2002.

[ZT92] J. Y. Zheng and S. Tsuji. Panoramic representation for route recognition by a mobile robot. In *Int. J. Computer Vision*, pages 55–76, 1992.

Acknowledgements

The thesis is about my work as a research assistant at the German Aerospace Center (DLR), Berlin. I thank Dr. Martin Scheele, who was my tutor and supervisor at DLR. He gave me valuable lessons in mathematics on subjects as needed for this thesis.

I wish to thank my colleagues at DLR, especially Professor Herbert Jahn, Professor Ralf Reulke, Dr. Anko Börner, David Krutz and Denis Griessbach for many fruitful, sometimes controversial, but always constructive discussions. I wish to thank Professor Gerd Hirzinger and his team from DLR institute for Robotic and Mechatronic for the cooperation and exchange of experiences. Thanks to Tim Bodenmüller for providing 3D-modeling algorithms.

I would like to give my sincerest gratitude to all the people outside of DLR, who helped me to accomplish this project, especially Bernhard Strackenbrock, who told me about rangefinders, helped me to collect the data, and efficient cooperation.

Especially I thank Professor Reinhard Klette, who was my supervisor at Auckland and during his visits at DLR Berlin. Many thanks to him and his students and colleagues at CITR for valuable feedbacks and support. Thanks also for a great time in New Zealand.

I also thank Professor Robert Schaback and Dr. Hartje Kriete at Göttingen University for their interest in the subject of this thesis, and their support. The meetings at Göttingen have been of essential importance for this PhD project, as well as the DAAD support for my stay at Auckland university.

Curriculum Vitae

Curriculum vitae text